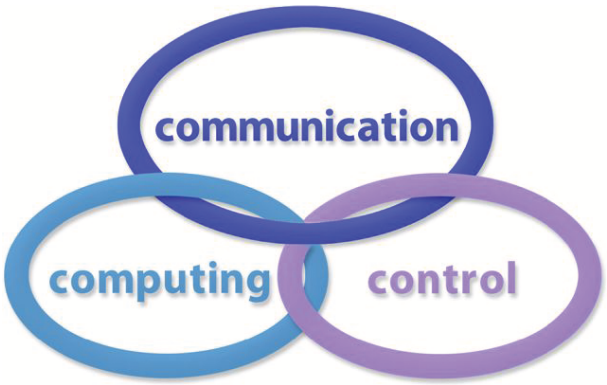
INTERNATIONAL JOURNAL

of

COMPUTERS, COMMUNICATIONS & CONTROL

With Emphasis on the Integration of Three Technologies

IJCCC



Year: 2012 Volume: 7 Number: 3 (September)

Agora University Editing House

CCC Publications

www.journal.univagora.ro

International Journal of Computers, Communications & Control

EDITORIAL ADDRESS:
R&D Agora Ltd. / S.C. Cercetare Dezvoltare Agora S.R.L.
Piaţa Tineretului 8, Oradea, jud. Bihor, Romania, Zip Code 410526
Tel./ Fax: +40 359101032
E-mail: ijccc@univagora.ro, rd.agora@univagora.ro, ccc.journal@gmail.com
Journal website: www.journal.univagora.ro

DATA FOR SUBSCRIBERS
Supplier: Cercetare Dezvoltare Agora Srl (Research & Development Agora Ltd.)
Fiscal code: RO24747462
Headquarter: Oradea, Piata Tineretului Nr.8, Bihor, Romania, Zip code 410526
Bank: MILLENNIUM BANK, Bank address: Piata Unirii, str. Primariei, 2, Oradea, Romania
IBAN Account for EURO: RO73MILB0000000000932235
SWIFT CODE (eq.BIC): MILBROBU

International Journal of Computers, Communications & Control

Mario de J. Pérez Jiménez
Dept. of CS and Artificial Intelligence
University of Seville, Sevilla,
Avda. Reina Mercedes s/n, 41012, Spain
marper@us.es

Dana Petcu
Computer Science Department
Western University of Timisoara
V.Parvan 4, 300223 Timisoara, Romania
petcu@info.uvt.ro

Radu Popescu-Zeletin
Fraunhofer Institute for Open
Communication Systems
Technical University Berlin, Germany
rpz@cs.tu-berlin.de

Imre J. Rudas
Institute of Intelligent Engineering Systems
Budapest Tech
Budapest, Bécsi út 96/B, H-1034, Hungary
rudas@bmf.hu

Yong Shi
Research Center on Fictitious Economy
& Data Science
Chinese Academy of Sciences
Beijing 100190, China
yshi@gucas.ac.cn
and
College of Information Science & Technology
University of Nebraska at Omaha
Omaha, NE 68182, USA
yshi@unomaha.edu

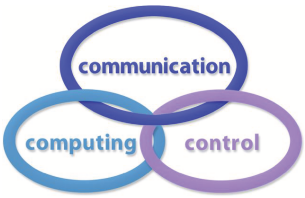
Athanasios D. Styliadis
Alexander Institute of Technology
Agiou Panteleimona 24, 551 33
Thessaloniki, Greece
styl@it.teithe.gr

Gheorghe Tecuci
Learning Agents Center
George Mason University, USA
University Drive 4440, Fairfax VA 22030-4444
tecuci@gmu.edu

Horia-Nicolai Teodorescu
Faculty of Electronics and Telecommunications
Technical University "Gh. Asachi" Iasi
Iasi, Bd. Carol I 11, 700506, Romania
hteodor@etc.tuiasi.ro

Dan Tufiş
Research Institute for Artificial Intelligence
of the Romanian Academy
Bucharest, "13 Septembrie" 13, 050711, Romania
tufis@racai.ro

Lotfi A. Zadeh
Professor,
Graduate School,
Director,
Berkeley Initiative in Soft Computing (BISC)
Computer Science Division
Department of Electrical Engineering
& Computer Sciences
University of California Berkeley,
Berkeley, CA 94720-1776, USA
zadeh@eecs.berkeley.edu



International Journal of Computers, Communications & Control

Short Description of IJCCC

Title of journal: International Journal of Computers, Communications & Control
Acronym: IJCCC
Abbreviated Journal Title: INT J COMPUT COMMUN
International Standard Serial Number: ISSN 1841-9836, E-ISSN 1841-9844
Publisher: CCC Publications - Agora University
Starting year of IJCCC: 2006
Founders of IJCCC: Ioan Dzitac, Florin Gheorghe Filip and Mişu-Jan Manolescu
Logo:

Number of issues/year: IJCCC has 4 issues/odd year (March, June, September, December) and 5 issues/even year (March, June, September, November, December). Every even year IJCCC will publish a supplementary issue with selected papers from the International Conference on Computers, Communications and Control.
Coverage:

- Beginning with Vol. 1 (2006), Supplementary issue: S, IJCCC is covered by Thomson Reuters - SCI Expanded and is indexed in ISI Web of Science.
- Journal Citation Reports/Science Edition 2010:
  - Impact factor = 0.650
- Beginning with Vol. 2 (2007), No.1, IJCCC is covered in EBSCO.
- Beginning with Vol. 3 (2008), No.1, IJCCC, is covered in Scopus.

Scope: IJCCC is directed to the international communities of scientific researchers in universities, research units and industry. IJCCC publishes original and recent scientific contributions in the following fields: Computing & Computational Mathematics; Information Technology & Communications; Computer-based Control.
Unique features distinguishing IJCCC: To differentiate from other similar journals, the editorial policy of IJCCC encourages especially the publishing of scientific papers that focus on the convergence of the 3 "C" (Computing, Communication, Control).
Policy: The articles submitted to IJCCC must be original and previously unpublished in other journals. The submissions will be revised independently by at least two reviewers and will be published only after completion of the editorial workflow.

Contents

INT J COMPUT COMMUN, ISSN 1841-9836
Vol.7 (2012), No. 3 (September), pp. 403-416

A Joint Routing and Time-Slot Assignment Algorithm for Multi-Hop Cognitive Radio Networks with Primary-User Protection

H. Chen, Q. Du, P. Ren

Hao Chen
1. Department of Information and Communication Engineering,
Xi'an Jiaotong University, Xi'an, Shaanxi,710049, China
2. State Key Laboratory of Integrated Services Networks,
Xidian University, Xi'an, Shaanxi, 710071, China
E-mail: js.sq.chenhao@stu.xjtu.edu.cn

Qinghe Du, Pinyi Ren
Department of Information and Communication Engineering,
Xi'an Jiaotong University, Xi'an, Shaanxi,710049, China
E-mail: {duqinghe, pyren}@mail.xjtu.edu.cn

Abstract:
Cognitive radio has recently emerged as a promising technology to improve the utilization efficiency of the radio spectrum. In cognitive radio networks, secondary users (SUs) must avoid causing any harmful interference to primary users (PUs) and transparently utilize the licensed spectrum bands. In this paper, we study the PU-protection issue in multi-hop cognitive radio networks. In such networks, secondary users carefully select paths and time slots to reduce the interference to PUs. We formulate the routing and time-slot assignment problem into a mixed integer linear programming (MILP). To solve the MILP which is NP-Hard in general, we propose an algorithm named RSAA (Routing and Slot Assignment Algorithm). By relaxing the integral constraints of the MILP, RSAA first solves the max flow from the source to the destination. Based on the max flow, RSAA constructs a new network topology. On the new topology, RSAA uses branch and bound method to get the near optimal assignment of time slots and paths. The theoretical analyses show that the complexity of our proposed algorithm is $O(N^4)$. Also, simulation results demonstrate that our proposed algorithm can obtain near-optimal throughputs for SUs.
Keywords: Cognitive Radio Networks; Primary-user Protection; Joint Routing and Time-slot Assignment.

1 Introduction

The rapid growth in the number of wireless applications such as WiFi, WiMAX, 3G et al. leads to a big radio spectrum shortage. Recent studies by the Federal Communications Commission (FCC) highlight that the average utilizations of some licensed spectrum bands allocated through the current static frequency spectrum assignment policies vary between 15% and 85% [1]. To make sufficient use of the spectrum resources in the environment, the notion of cognitive radio (CR) was proposed by Dr. Joseph Mitola in 1999 [2]. In cognitive radio networks, nodes are allowed to sense and explore a wide range of the frequency spectrum and identify currently underutilized spectrum blocks for data transmission. CR can transparently exploit the licensed spectrum bands and is widely considered as the technique for the next generation of wireless communication [3].

To maximize the advantages of cognitive radio networks (CRNs)ŁŹit is necessary to update the physical layer, media access control (MAC) layer and network layer of the traditional wireless communication system. After the concept of CR was proposed, many studies on spectrum sensing and MAC protocol

design have been conducted and a lot of progress has been made [4] [5] [6]. The aim of spectrum sensing is to find the spectrum holes in the CRNs and the MAC protocol is to select the one-hop optimal spectrum bands for SUs' transmitting. However, Khalife et al. indicated that MAC protocol which gave optimal solutions in a single hop configuration may become largely inefficient in a multi-hop scenario and it was of great importance to design cross-layer protocols capable of scheduling, spectrum selecting and routing [7]. Cesana et al. in [8] pointed out the challenges of routing in cognitive radio networks: any routing solutions designed for multi-hop CRNs must be highly coupled to the entire cognitive cycle of spectrum management and the routing module should be able to make fast route maintenance at the sudden appearance of PUs. The authors of [9] extended the routing solution in multi-channel multi-radio networks (MCMRNs) to CRNs and proposed a layered graph framework to address channel assignment and routing jointly. Hou et al. in [10] [11] illustrated the difference about routing in MCMRNs and CRNs. In CRNs the radios could send packets over non-contiguous frequency bands and the authors proposed a mixed integer non-linear programming (MINLP) model to minimize the required network-wide radio spectrum resources. Filippini et al. in [12] proposed a minimum maintenance cost routing for cognitive radio networks. The authors formulated the maintenance cost problem to be an integer optimization model and by carefully selecting routing metrics the authors designed a heuristic distributing algorithm.

Among all the above routing solutions the accurate information about spectrum availability sensed by the physical layer is crucial for the routing module. So these routing solutions make severe demand on the spectrum sensing module of CR nodes. To ease this demand, in [13] Chowdhury et al. proposed a routing solution which avoided harmful interference to PU receivers by designing proper routing metrics. In that solution, each time SUs chose the path that passed through regions having minimum overlap with PUs' transmission coverage areas. In [14] Ding et al. proposed a distributed algorithm to maximize the capacity of links without generating harmful interference to other users by performing joint routing, dynamic spectrum allocation and scheduling. In [15] Xie et al. proposed a geometric approach for relay selections which avoided causing harmful interference to PUs in CRNs. Each time the approach selected the best channels available to transmit data to the nearest neighbors or the farthest ones greedily. In [16] the authors Zhou et al. gave a mathematical model aiming at minimizing the interference to PUs. By relaxing the model's constraint conditions, the authors transformed the original optimization problem to a linear programming model and gave a joint channel assignment and path selection algorithm.

In spectrum sharing multi-hop CRNs, to guarantee the PUs' priority on the licensed spectrum bands SUs must not generate harmful interference to PUs. In this paper, to make sufficient use of spectrums we design the routing module of SUs through joint routing and time-slot assignments. Firstly, we exploit the Protocol Model [17] which describes the conditions of successful transmission between two nodes and abstract the routing and time-slot assignment problem to be a kind of mixed integer linear programming (MILP) model. In the MILP model, our objective is to maximize the throughput of SUs and our constraints are avoiding harmful interference to PUs and eliminating SUs' conflicts caused by concurrent transmissions. Then, to get an approximate solution of the MILP which is a NP-Hard problem we propose a near optimal joint routing and time-slot assignment algorithm named RSAA. The theoretical analysis shows that the complexity of RSAA is $O(N^4)$. The simulation results demonstrate that RSAA can obtain near optimal throughput. What is more, through the simulation results we analyze the effect of node density and slot periods on the throughput of multi-hop CRNs.

2 Network Model and Problem Formulation

2.1 Network Model

In 2004, the FCC proposed to allow unlicensed wireless devices to utilize television channel frequencies under the precondition of causing no harmful interference to PUs. Under this proposal, SUs could use the underutilized broadcasting TV spectrum bands for multi-hop communications. In multi-hop CRNs, TDMA is very necessary for the avoidance of conflicts among SUs and the reduction of interference to PUs. In the current WLANs, nodes use CSMA/CA to avoid access conflicts. However, CSMA/CA cannot assure that SUs cause no harmful interference to PUs. Hence it may be not suitable for spectrum sharing CRNs.

In this paper, we consider a cognitive Ad Hoc network consisting of P PUs, N SUs and a Cognitive Scheduling Center (CSC). The CSC is able to access the data base of PUs [8] and gathers the information about the PUs' locations and interference thresholds. The interference threshold is defined as the highest interference power which a PU can tolerate. Also the CSC is designed to be able to collect the information about SUs' locations and the transmitting power via existing communication networks like GSM. After gathering all these information, the CSC computes the optimal routing and time-slot assignments and schedule SUs' access to licensed spectrum by delivering these messages to the corresponding nodes. The network architecture is shown in Fig. 1.

As is shown in Fig. 1, the TV receivers act as PUs and have priority to use the spectrum f.The CSC coordinates and schedules SUs to utilize the band f transparently. For example, the CSC assigns the path and slot solution $s \xrightarrow{1} a \xrightarrow{2} b \xrightarrow{3} c \xrightarrow{1} e \xrightarrow{2} d$ for source node s and destination node d. The symbol "$a \xrightarrow{1} b$" represents that node a transmits to node b in time slot 1. Although the path $s \to a \to g \to e \to d$ is of less hops and much more throughput, the CSC does not choose it because node g causes harmful interference to PUs.This paper focuses on finding the optimal routing and time-slot assignment which maximizes the SUs' throughput while avoiding harmful interference to PUs.

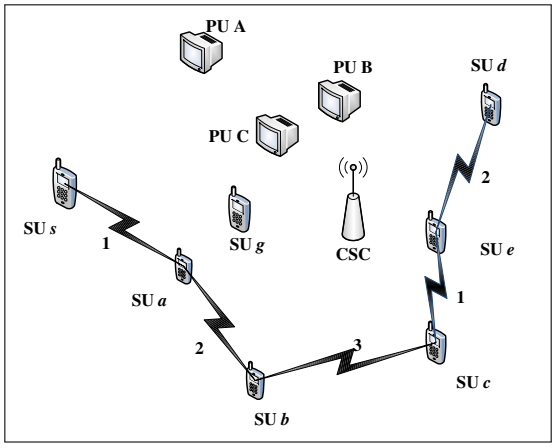


Figure 1: system model

2.2 Problem Formulation

To describe the network mathematically, we first model the successful transmitting conditions among nodes and introduce binary integral variables x_{ijt}, where x_{ijt}=1 indicates that SU node i sends packets to SU node j in slot t; otherwise x_{ijt}=0. That is,

$$x_{ijt} = \begin{cases} 1, & SU\ i \text{ send packets to } SU\ j \text{ in slot } t \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

We assume that all the SUs' radios use the same transmitting power Q and the successful transmitting range among SUs are R_T. Let A_i denote the neighbor set of node i. So we have

$$A_i = \left\{ j | \, d_{ij} < R_T \right\} \tag{2}$$

where d_{ij} denotes the Euclid distance between node i and node j.

Let I_i denote the interfered node set of node i, and R_I denote the interference range among SUs. So we have

$$I_i = \left\{ j | \, d_{ij} < R_I \right\} \tag{3}$$

Note that node i cannot transmit to multiple nodes at the same time. We have

$$\sum_{q \in A_i} x_{iqt} \leqslant 1 \tag{4}$$

Due to potential interference among nodes in the network, if node i uses slot t for transmitting data to node $j \in A_i$, then any other nodes which conflict with node j should not use this slot. So we have

$$x_{ijt} + x_{pqt} \leqslant 1, \ p \in I_j, p \neq i, q \in A_p \tag{5}$$

This is the difference between our model with the model in [9][10], where the authors state the conflict constraint as

$$x_{ijt} + \sum_{p \in I_j, p \neq i, q \neq j} x_{pqt} \leqslant 1 \tag{6}$$

It is important to understand that in the conflict constraint (5), two links which interfere with x_{ijt} but do not interfere with each other can access the channel at the same time slot. Constraint (4) is the node's successful sending condition and constraint (5) is the node's successful receiving condition. Under the constraint (4) and (5), nodes in the network can transmit data without conflicting with each other.

To guarantee that two nodes transmit data without bit error, the flow rates on each link must not exceed the link's capacity. Let f_{ijt} denote the flow rate between node i and node j at time slot t and g_{ij} denote the channel gain between node i and node j. We have

$$f_{ijt} \leqslant x_{ijt} B \log_2 (1 + \frac{g_{ij} Q}{\eta}) \tag{7}$$

where B denotes the bandwidth of spectrum band f and η denotes the noise power in the environment. Note that the denominator inside the log function contains only η. This is because the interference constraint (5) assures that the interference power received by the SU's receiver is negligible and this helps to simplify our model to be a linear model.

What's more, to make sure that no drop of packet happens at the intermediate nodes, the aggregate data rates in the slot period should meet the flow conservation constraint. i.e.

$$\sum_{t=1}^{T} \sum_{j \in A_i} f_{ijt} = \sum_{t=1}^{T} \sum_{i \in A_p} f_{pit} \tag{8}$$

where T denotes the period of scheduling slots of the network.

To make sure that all the transmitting power of SUs detected at PUs does not exceed PUs' threshold In_T, we have the following constraints

$$\sum_{i=1}^{N} x_{ijt} g_{ik} Q \leqslant In_T, j \in A_i, k \in P, \tag{9}$$

where P denotes the set of PUs. In practice the value of In_T depends on the sensitivity of PU receivers as well as the noise power in the environment.

Note the average value of SUs' throughput S is the ratio of aggregate amount of data to the time slots, so we have

$$S = \frac{1}{T}\sum_{t=1}^{T}\sum_{i\in A_s} f_{sit} \tag{10}$$

where s denotes the source node.

When there are a number of source-destination pairs in the network, we can introduce one more virtual source-destination pair and simplify the network to be a single source-destination pair network. In such networks, our aim is to find the optimal routing and time-slot assignments which maximize the throughput of SUs. Mathematically, we have the following optimization problem.

$$\max\left\{\frac{1}{T}\sum_{t=1}^{T}\sum_{i\in A_s} f_{sit}\right\} \tag{11}$$

$$s.t.\begin{cases} \sum\limits_{q\in A_i} x_{iqt} \leqslant 1 \\ x_{ijt} + x_{pqt} \leqslant 1,\ p \in I_j, p \neq i, q \in A_p \\ f_{ijt} \leqslant x_{ijt} B\log_2(1+\frac{g_{ij}Q}{\eta}) \\ \sum\limits_{t=1}^{T}\sum\limits_{j\in A_i} f_{ijt} = \sum\limits_{t=1}^{T}\sum\limits_{i\in A_p} f_{pit} \\ \sum\limits_{i=1}^{N} x_{ijt} g_{ik} Q \leqslant In_T, j \in A_i, k \in P, 0 \leqslant t \leqslant T \\ x_{ijt} = 0\ or\ 1 \\ f_{ijt} \geqslant 0 \end{cases} \tag{12}$$

Note that the above optimization problem has both integral and continuous variables and the constraint conditions are all linear. So it is a kind of MILP problem. In the above MILP problem, the optimization variables consists of continuous variables f_{ijt} and binary variables x_{ijt} , while B, T, Q, η and In_T are all constant.

3 Routing and Time-Slot Assignment Algorithm

The above optimization problem is a kind of MILP problem, which is NP-Hard in general [18]. To solve the MILP problem, Yuan et al. in [16] proposed a greedy algorithm which relaxed the integral constraints and then simplified the MILP to be a linear programming (LP) problem. After solving the LP problem, the algorithm fixed the binary variables in the descending order of their relaxed values one by one. Although the complexity of this algorithm is equal to solving just one LP problem, the algorithm cannot guarantee that the solution is a feasible flow and that the result meets the interference constraints (4) (5) absolutely. Hou et al. in [10] proposed the Sequential Fixing (SF) algorithm which solved $O(N)$ LP problems iteratively to fix the binary variables. Another method to solve MILP is to replace the integral constraint to be the following constraint

$$x_{ijt}(x_{ijt} - 1) = 0. \tag{13}$$

Based on constraint (13) we can transform MILP to be quadratic programming (QP). But we still cannot get optimal solution under polynomial complexity because constraint (13) is non-convex.

Sherali et al. in [18] pointed out that to solve the MILP problem one should exploit the problem's inherent special structures in the process of model formulations and in algorithmic developments. Taking

a closer look at our MILP problem we can find that if we relax the integral constraint (1) to be continuous constraint

$$0 \leqslant x_{ijt} \leqslant 1, \tag{14}$$

and neglect the interference constraint (4)(5). Then the MILP problem will reduce to a kind of max flow problem. So the optimal solutions of MILP are likely to be the subsets of the max flow. Based on this idea, we develop our algorithm RSAA which can solve the MILP problem efficiently.

Let E denote the number of edges and N denote the number of SU nodes in the CRNs. Since the original MILP consists of ET binary variables, to get the optimal solution of the MILP problem we should enumerate 2^{ET} combinations of x_{ijt}. Once the binary variables are fixed, the original MILP problem is reduced to be the following LP problem.

$$\max\left\{\frac{1}{T}\sum_{t=1}^{T}\sum_{i\in A_s} f_{sit}\right\} \tag{15}$$

$$s.t.\begin{cases} f_{ijt} \leqslant x_{ijt} B\log_2(1+\frac{g_{ij}Q}{\eta}) \\ \sum\limits_{t=1}^{T}\sum\limits_{j\in A_i} f_{ijt} = \sum\limits_{t=1}^{T}\sum\limits_{i\in A_p} f_{pit} \\ \sum\limits_{i=1}^{N} x_{ijt} g_{ik} Q \leqslant In_T, j \in A_i, k \in P, 0 \leqslant t \leqslant T \\ f_{ijt} \geqslant 0 \end{cases} \tag{16}$$

We denote the new LP problem as LP1 and it consists of ET continuous variables.

In RSAA algorithm, we first neglect the constraint conditions (4)(5)(7) and solve the max flow problem from the source to the destination. The max flow problem can be formulated to be the following LP problem

$$\max\left\{\frac{1}{T}\sum_{t=1}^{T}\sum_{i\in A_s} f_{sit}\right\} \tag{17}$$

$$s.t.\begin{cases} f_{ijt} \leqslant B\log_2(1+\frac{g_{ij}Q}{\eta}) \\ \sum\limits_{t=1}^{T}\sum\limits_{j\in A_i} f_{ijt} = \sum\limits_{t=1}^{T}\sum\limits_{i\in A_p} f_{pit} \\ f_{ijt} \geqslant 0 \end{cases} \tag{18}$$

We denote the above LP problem as LP2 and use Push-Pull Flow Algorithm [20] to obtain the max flow $\Phi = \{f_{ijt}\}$. After getting Φ, we construct a new set of binary variables by

$$X_T = \{x_{ijt} | f_{ijt} > 0, f_{ijt} \in \Phi\} \tag{19}$$

And X_T is the new enumeration set of RSAA. We denote K as the number of variables of X_T, i.e. $K = |X_T|$. When we enumerate the new variables in X_T, we use constraints (4)(5) as the branch and bound conditions. The complete RSAA algorithm is given in Table 1.

In the procedure of RSAA algorithm, we first construct new searching variables or edges by the solution of the max flow problem and then cut off all the nodes and edges which have nothing to do with the max flow in the network. To solve the MILP in newly constructed network, we take the condition (4)(5) as bound conditions and simplify the MILP problem to be a number of LP1 problems. Also when we solve LP1, we take it as a kind of special max flow problem, and use Dinic algorithm to solve it. As we can see from the algorithm's process, the solution of RSAA always meets all the constraints of the original MILP problem and so we can conclude that the solution of RSAA is a feasible solution to the original MILP problem.

Table 1: RSAA algorithm

Steps	Contents
Step 1:	Let the CSC update the PUs' and SUs' locations and compute the SUs' link capacity and their interference power to PUs. Introduce a virtual source-destination pair and simplify the network to be a signal source-destination pair network.
Step 2:	Set up and solve LP2 and obtain its solution Φ. Use the equation (19) and Φ to construct new binary variable set X_T. Sort the new binary variables in the ascending order by slot index.
Step 3:	Initialize the SUs' throughput $S = 0$; set the current optimal flow solution as $\Phi^* = \emptyset$, $temp\_i = 0$.
Step 4:	If $temp\_i > 2^K$, then the whole algorithm ends, output the optimal throughput S and the optimal flow solution set Φ^*; else transform $temp\_i$ into $\|X_T\|$ bit binary digits, each digit represents the assignment of corresponding link and slot. If each digit of $temp\_i$ meets the interference condition (4)(5),then go to Step 5;else $temp\_i = temp\_i + 2^{b\_last-1}$, where b_last is the smallest digit index in all $temp\_i$'s transformed digits which violate the interference condition (4)(5). Go back to Step 4.
Step 5:	After getting one combination of $\{x_{ijt}\}$,use Dinic algorithm to solve LP1 and get the max flow value $temp\_fval$ as well as the corresponding flow rates $F_T = \{f_{ijt}\}$. In the steps of Dinic algorithm, when we search the augmenting flows in the layered networks, we select the augmenting flow according to the ascending order of their interference to PUs. If the adding of augmenting flow with the minimum interference exceeds In_T, then augmenting step of Dinic algorithm ends; else continue to find the other augmenting flows. When Dinic algorithm ends, if $S < temp\_fval$, then $S = temp\_fval$,$\Phi^* = F_T$; else go back to Step 4.

4 Performance Analysis of RSAA

4.1 Complexity of RSAA

We now analyze the complexity of RSAA using the random network theory. In the original MILP problem, the complexity of obtaining optimal solution is exponential and we should solve $O(2^{|E|})$ linear programming LP1. By reducing the searching variables, the complexity of RSAA algorithm is reduced to be polynomial. In fact, we have the following theorem.

Theorem 1. *In the network where the average degree of each node is constant, the complexity of RSAA is* $O(N^4)$.

Proof: Let constant D denote the average degree of each node in the network and in practice the value of D is determined by the node density and the node's transmitting range. By the ER model in random network theory [21], the average number of edges in the network is

$$E = \frac{DN}{2} \tag{20}$$

Let L denote the average path length from the source node to the destination node, and then the number of nodes in the network has the following expression [21],

$$N \propto D^L \tag{21}$$

So the average route length can be written as

$$L = \alpha \frac{\log_2 N}{\log_2 D}, \tag{22}$$

where α is a constant. Note that the capacity of each link in random networks is i.i.d. So the maximum throughput we obtain from LP2 are DC, where C denotes the average capacity of links. And then we can conclude that the maximum number of paths in the max flow for source to destination is D. Hence, the number of newly constructed binary variables is

$$K = |X_T| = DTL = \alpha TD \frac{\log_2 N}{\log_2 D}. \tag{23}$$

So RSAA need to solve $O(2^K) = O(N)$ linear programming problems LP1. By Dinic algorithm the complexity of solving LP1 is $O(N^3)$. Hence, the total complexity of RSAA algorithm is $O(N \bullet N^3) = O(N^4)$. □

From Theorem 1 we can see that both RSAA and SF need to solve $O(N)$ linear programming problems LP1. The difference between them is that RSAA exploits the flow structures of the network to obtain better performance.

4.2 Optimal Approximation

Although RSAA restricts the searching space and this restriction may reduce the throughput of SUs, we find that this kind of reduction is almost negligible. In fact we have the following theorem.

Theorem 2. *If* Φ^* *is the optimal solution to the MILP problem and* P_T *is the optimal routing and time-slot assignment, then the intersection set between* P_T *and the RSAA's new constructed searching set* X_T *is not null.*

Proof: We prove this theorem by constructing contradictions. Suppose the intersection set between X_T and the new constructed searching set X_T is null, i.e. $P_T \cap X_T = \emptyset$. We denote the optimal flow rate of the MILP as Φ^* and denote the max flow of LP2 as Φ. Because Φ^* is the solution of the MILP, and so Φ^* satisfies all the constraint flow conditions in the MILP. Then Φ^* satisfies all the constraint conditions of LP2. So Φ^* is a feasible augmenting flow to LP2. Because $P_T \cap X_T = \emptyset$, and this means P_T is an independent augmenting path on which we can augment Φ^* to the original max flow. So the optimal solution of LP2 is $\Phi^* + \Phi$ and this is contradictory to the fact that Φ is the optimal solution. So the supposition that the intersection set between P_T and the RSAA's new constructed searching set X_T is null is false and our theorem is proven. □

However, Theorem 2 cannot guarantee that the solution of RSAA is optimal. Only when P_T is the subset of X_T, can we say that the solutions of RSAA are optimal. In fact, we can conclude that P_T belongs to the subset of X_T with high probability according to Theorem 2. Especially, in the small networks where the max flow consists of only one path, we can conclude that the solution of RSAA will be optimal according to Theorem 2 and the flow conservation condition(8).

4.3 The Effect of Time-Slots and Conflicts

In our MILP model, there is a constant T which represents the slot scheduling period of the CRNs. We can see that the existing of T increases the complexity of our algorithm, and this is because T decides the number of variables. In the network layer, the routing module should fix the best slot period according to the network parameters such as the node density and PU's threshold. Note that the minimum slot period should be enough to avoid the conflicts among SUs and the interferences to PUs. So we can get the average minimum value of T,

$$T_{\min} = \max\{LQg^*/In_T, L/c\} \tag{24}$$

In the above equation, c is the average number of mutual interference edges and L is the average route length from the source to destination, and g^* denotes the average channel gain between two neighbor nodes. In equation (24), the first item means that to connect the source and the destination at least LQg^*/In_T slot periods are needed to guarantee no harmful interference to PUs. The second item means that to avoid the conflicts among SUs at least L/c slot periods are needed. And the equation (14) means that only the slot periods are long enough to avoid harmful interference to PUs and eliminate the conflicts among SUs, the source and destination pair can set up a successful route.

In fact, it is necessary to assure that the slot period is greater than the minimum one, or the slot period will become the bottleneck constraint of SU's throughput. And when the slot periods are smaller than the minimum one the increase of slot period will dramatically increase SUs' throughput. But if the slot periods are greater than the minimum one, the increase of slot period will not necessarily lead to the increase of SUs' throughput. And in this scenario SUs' throughput in CRNs are the tradeoff between the amount of data and the delay as is shown in equation (10). However, if we divide one fixed length of time into a number of slots, we can find that the more slots we divide the fixed time into, the more throughput SUs can obtain. In the best case if we divide the fixed time into infinite slots, and then the solution of MILP will approximate the solution of the relaxed LP where the integral constraint is reduced to be constraint (14).

To describe the effect of conflict on SUs' throughput, we introduce R denoting the ratio of node's interference distance to transmitting distance. i.e.

$$R = R_I/R_T \tag{25}$$

Note that as the ratio R increases the number of conflicting edges in the network will increase. And this means the number of conflict constraint inequality (5) will increase and so the solutions of the MILP will

decrease. Intuitively, the increase of the ratio R will decrease the concurrent transmissions and cut down SUs' throughput. However, as the ratio R increases, the number of paths from the source to destination will decrease because of the concurrent conflicts. So according to Theorem 2 we can conclude that the solution of RSAA will become closer to the optimal results as the ratio R increases.

From the analysis above, we can find that RSAA algorithm can obtain the near optimal throughput of SUs at the complexity of $O(N^4)$. In the next section we will verify the performance analysis through simulations.

5 Simulation Results

In this section, we present simulation results for the proposed RSAA algorithm and compare it to SF algorithm and the enumeration algorithm. Since the enumeration algorithm can obtain optimal solutions, we denote the results of enumeration algorithm as optimal solutions in the following figures. Our simulation scenario is set at the rural and mountainous areas where the TV broadcast spectrum is underutilized and the SUs can transparently use these spectrums without generating harmful interference to PUs. The simulation parameters are shown in Table 2.

Table 2: Simulation Parameters

Simulation Parameters	Values
The topology area	1000x1000 m^2
The distribution of SUs' location	uniform distribution
Channel Propagation model	two way ground reflection model
The transmitting range of SUs	250m
The interfering range of SUs	300m
The band width of PUs	1MHz
The power of noise	-140dBW
The transmitting power of SUs	2W
The number of PU	1
The location of PU	$(0, 0)$
The length of each slot period	1s
Simulation times	200

5.1 The SUs' throughput vs PU's outage probability

Fig. 2 shows the outage probability of PUs in the condition that SUs do not take the PU's threshold into consideration. We count a time of outage of PU when PU detects that SUs' interference Power exceeds In_T. In Fig. 2 the slot period is set to be 3 and the ratio of interference to transmitting is set to be 6/5. We use software CPLEX to get the optimal solution of MILP.

From Fig. 2 we can find that as the PU's threshold increases the outage probability of PU decreases. Also the outage probability decreases as the node density Nm in the network decreases. This is because the interference power received by PU is the sum of all SUs' transmitting power and more nodes in the network means more interference. As we can see from Fig. 2, the outage probability is very high and unbearable if SUs do not carefully select paths and slots. So in the networks with high node density, it is very necessary to protect PU from SUs' interference at network layer.

Fig. 3 compares the SUs' throughput in two different scenarios: in one scenario SUs avoid harmful interference to PUs and in the other scenario SUs neglect PU's threshold. From Fig. 3 we can see that SUs' throughput are very sensitive to PU's threshold when SUs take the interference to PU into

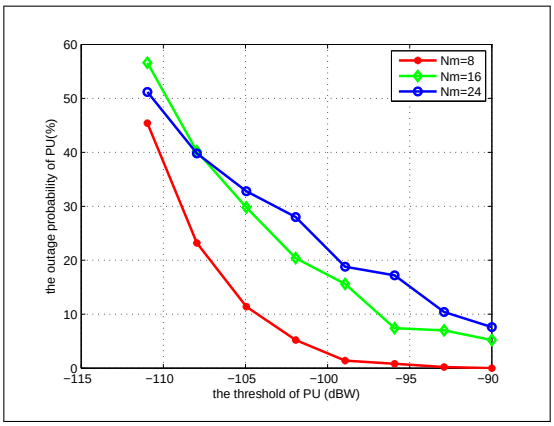


Figure 2: the outage probability of PU as the increase of PU's threshold

consideration. And the throughput obtained from neglecting PU's threshold are the upper bound of those of considering PU's threshold. What's more, Fig. 3 shows that when PU's threshold is low enough, PU's threshold becomes the bottleneck of SUs' throughput. Fig. 2 and 3 demonstrate the performance tradeoff between PUs and SUs in spectrum sharing CRNs. And so our model can offer valuable reference for the design of multi-hop CRNs.

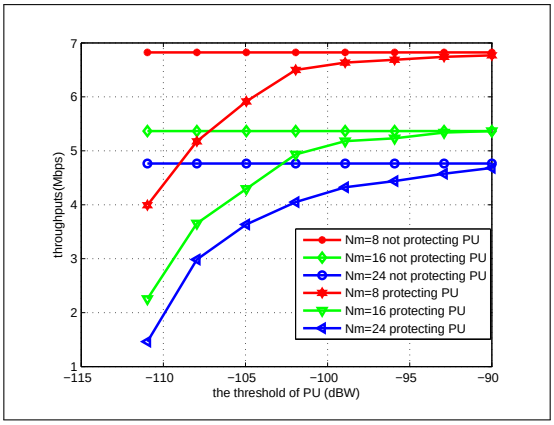


Figure 3: SUs' throughput as the increase of PU's threshold

5.2 The Effect of Node Density

Fig. 4 compares SUs' throughput of RSAA algorithm, SF algorithm and the enumerate algorithm. In the simulation, the slot period is 3 and the PU's threshold is -90dBW. The ratio of interference to transmitting is 6/5. From Fig. 4 we can find that the results of RSAA outperform the results of SF and can obtain 97% of the optimal throughput for SUs averagely. Especially when the node density is low, RSAA can obtain 99% of the optimal throughput, while SF just gets 55% optimal throughput for SUs. This is because when the node density is low, the optimal path will almost surely locate in the searching set X_T as is shown in Theorem 2. What's more, Fig. 4 shows that SUs' throughput decrease as the node density increases which is the same with the results in [16]. The reason is that when the node density increases, SUs need more slots which mean more delay to avoid the conflicts among SUs and the interference to PU.

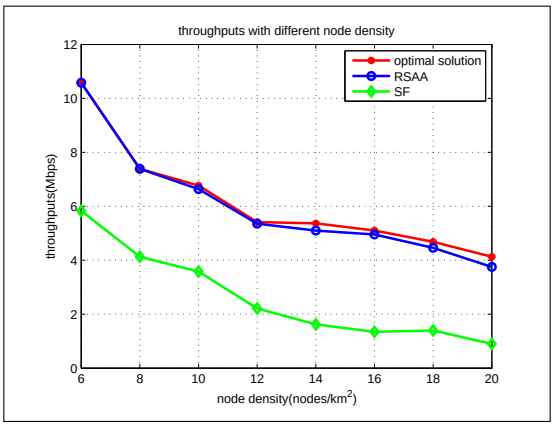


Figure 4: SUs' throughput on different node density.

5.3 The Effect of Time-Slots and Conflict

Fig. 5 compares the throughput of SUs got from the three algorithms as the slot periods in the network increase. In this simulation the node density is set as 13 nodes per square kilometer. The PU's threshold In_T and the ratio R are the same with those in Fig. 4. From Fig. 5 we can see that the RSAA's approximation to the optimal solution is not affected by slots and in any slot number condition RSAA can get 98% of the optimal throughput for SUs. Also we can find that when we fix the length of each slot as 1 second and increase the slot numbers in the network, the SUs' throughput increase dramatically when the number of slot periods is small. If we increase the slot periods from 3 to 5, SUs' throughput decrease because the increase of data cannot offset the increase of delay. But when the slot period is 6 the throughput of SUs increase as the increase of data outweighs the increase of delay. The fluctuation in Fig. 5 shows the complex relationship between relay and throughput in multi-hop wireless networks.

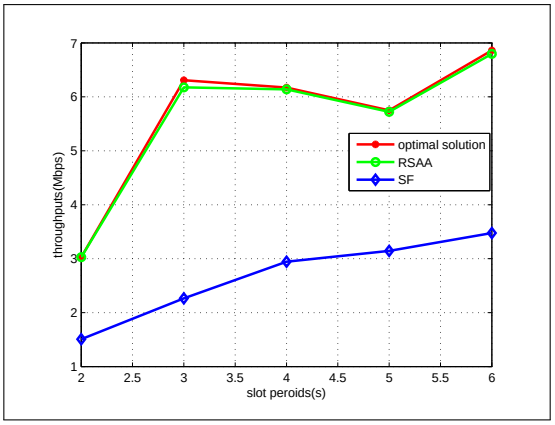


Figure 5: SUs' throughput on different slot periods.

Fig. 6 shows the throughput of SUs vary as the ratio of interference distance to transmitting distance increases under the three algorithms. In this simulation the PU's node density is set as 16 nodes per square kilometer and the PU's threshold is the same as that in Fig. 4. The slot period is 4. From Fig. 6 we can also find that RSAA can get 99% of the optimal throughput in average compared to SF's 34%. What is more, Fig. 6 shows that as the ratio increases the solutions of RSAA become much closer to the optimal results and when the ratio exceeds 2, the two solutions nearly overlap with each other which is identical with our above performance analysis.

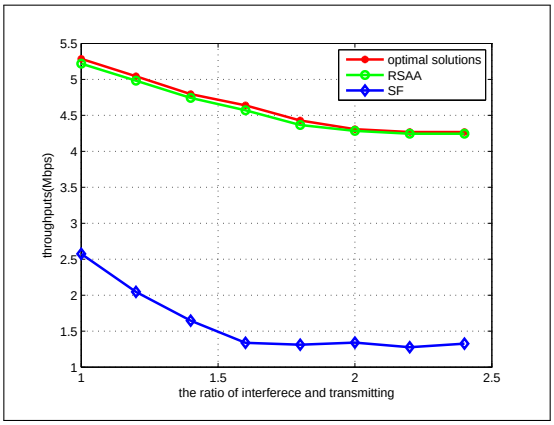


Figure 6: SUs' throughput on different ratio of interference to transmitting.

6 Conclusions

In spectrum sharing multi-hop CRNs, by carefully selecting paths and slots SUs can utilize the licensed spectrum band transparently. In this paper, we first formulate a MILP model to describe the joint routing and time-slot assignment issue and then develop RSAA algorithm to solve this NP-Hard problem. Theoretical analyses and simulation results demonstrate that RSAA algorithm can obtain near-optimal throughput with a polynomial complexity and so it can be widely used in CRNs.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China (60832007), National Hi-Tech Research and Development Plan of China (2009AA011801), and the National Science and Technology Major Project(2010ZX03005-003).

Bibliography

[1] FCC, ET Docket No 03-222 Notice of proposed rule makingand order, December 2003.

[2] J. Mitra III and G. Q. Maguire JR., "Cognitive radio: making software radios more personal," IEEE Personal Commun., pp. 13-18, Aug. 1999.

[3] I. Akyildiz, W. Lee, M. Vuran, and S. Mohanty. "NeXt Generation / Dynamic Spectrum Access/Cognitive Radio Wireless Networks: A Survey." Compo Netw. Jour. (Elsevier), Vol.50, no.13, pp. 2127-2159, Sept. 2006.

[4] T. Yucek, H. Arslan, "A survey of spectrum sensing algorithms for cognitive radio applications," Communications Surveys & Tutorials, IEEE , vol.11, no.1, pp.116-130, First Quarter 2009.

[5] H. Wang, H. Qin, L. Zhu, "A Survey on MAC Protocols for Opportunistic Spectrum Access in Cognitive Radio Networks," Computer Science and Software Engineering, 2008 International Conference on , vol.1, no., pp.214-218, 12-14 Dec. 2008

[6] Y. Wang, P. Ren, and G. Wu, "A throughput-aimed MAC protocol with QoS provision for cognitive ad hoc networks," IEICE Trans. Commun., vol. E93-B, no. 6, pp. 1426-1429, Jun. 2010.

[7] H. Khalife, N. Malouch, S. Fdida, "Multihop cognitive radio networks: to route or not to route," Network, IEEE , vol.23, no.4, pp.20-25, July-August 2009

[8] M. Ceasna, F. Cuomo, E. Ekici, "Routing in cognitive radio networks: Challenges and solutions", Ad Hoc Netw., Vol. 9, no. 3, pp.228-248, May 2011

[9] X. Zhou, L. Lin, J. Wang and X. Zhang, "Cross-layer routing design in cognitive radio networks by colored multigraph model," Wireless Personal Communications, vol. 49, no.1, pp. 123-131, April 2009

[10] Y.T. Hou, Y. Shi, H.D. Sherali, "Optimal Spectrum Sharing for Multi-Hop Software Defined Radio Networks," INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE , vol., no., pp.1-9, 6-12, May 2007

[11] Y.T. Hou, Y. Shi, H.D. Sherali, "Spectrum Sharing for Multi-Hop Networking with Cognitive Radios," Selected Areas in Communications, IEEE Journal on , vol.26, no.1, pp.146-155, Jan. 2008

[12] I. Filippini, E. Ekici, M. Cesana, "Minimum maintenance cost routing in Cognitive Radio Networks," Mobile Adhoc and Sensor Systems, 2009. MASS '09. IEEE 6th International Conference on , vol., no., pp.284-293, 12-15 Oct. 2009

[13] K.R. Chowdhury, I.F. Akyildiz, "CRP: A Routing Protocol for Cognitive Radio Ad Hoc Networks," Selected Areas in Communications, IEEE Journal on , vol.29, no.4, pp.794-804, April 2011

[14] L. Ding, T. Melodia, S.N. Batalama, J.D. Matyjas, M.J. Medley, "Cross-Layer Routing and Dynamic Spectrum Allocation in Cognitive Radio Ad Hoc Networks," Vehicular Technology, IEEE Transactions on , vol.59, no.4, pp.1969-1979, May 2010

[15] M. Xie, W. Zhang, K.K. Wong, "A geometric approach to improve spectrum efficiency for cognitive relay networks," Wireless Communications, IEEE Transactions on , vol.9, no.1, pp.268-281, January 2010

[16] Z. Yuan, J. B. Song, Z. Han, "Interference Minimization Routing and Scheduling in Cognitive Radio Wireless Mesh Networks," Wireless Communications and Networking Conference (WCNC), 2010 IEEE , vol., no., pp.1-6, 18-21 April 2010

[17] P. Gupta, P.R. Kumar, "The capacity of wireless networks," InformationTheory, IEEE Transactions on, vol.46, no.2, pp.388-404, Mar 2000.

[18] K. Jain, J. Padhye, V.N. Padmanabhan, L. Qiu, "Impact of Interference on Multi-Hop Wireless Network Performance", Wireless Networks, Vol 11, no 4, pp. 471-487, July 2005

[19] H.D. Sherali, W.P. Adams, P.J. Driscoll, "Exploiting Special Structures in Constructing a Hierarchy of Relaxations for 0-1 Mixed Integer Problems", Operations Research, Vol.46, no.3, pp.396-405, May 1998

[20] S. Gao. Graph Theory and Network Flow Theory, 1rd ed., BeiJing: Higher Education Press, 2009, pp.307-314

[21] P. Erdfs and A. Renyi, "On the evolution of random graphs", PubI. Math. Inst. Hung. Acad. Sci. 5, 1960, pp.17-61.

INT J COMPUT COMMUN, ISSN 1841-9836
Vol.7 (2012), No. 3 (September), pp. 417-431

GASANT: An ant-inspired least-cost QoS multicast routing approach based on genetic and simulated annealing algorithms

M. Damanafshan, E. Khosrowshahi-Asl, M. Abbaspour

Morteza Damanafshan, Ehsan Khosrowshahi-Asl
Department of Computer and Network,
Institute for Research in Fundamental Sciences (IPM),
Tehran, Iran.
E-mail: damanafshan@iranet.ir

Maghsoud Abbaspour
Department of Computer Engineering,
Faculty of Electrical and Computer Engineering,
Shahid Beheshti University, G.C., Tehran, Iran.
E-mail: maghsoud@sbu.ac.ir

Abstract:
Computing least-cost multicast routing tree while satisfying QoS constraints has become a key issue especially by growing communication networks. To solve this problem, a triplex algorithm called GASANT which is based on Ant Colony Optimization (ACO), Genetic Algorithm (GA), and Simulated Annealing (SA) has been proposed in this paper. Through ACO, we have both provided improved initial population to feed GA and reduced search process. Besides, SA has been deployed to refrain GA from getting stuck into local optimum solutions. Simulation results assert that GASANT not only has high speed convergence time, but also generates least-cost multicast routing trees of high QoS.
Keywords: Multicast Routing; Quality of Service (QoS); Ant Colony Optimization (ACO); Genetic Algorithm (GA); Simulated Annealing (SA)

1 Introduction

Multicasting service is a technique in which the same information is sent concurrenly from a source node to a subset of all possible destinations (multicast group) in a computer network. The current approach to provide such a service is to establish a multicast tree. This tree includes a route node (sender), some internal nodes (intermediate routers) and some leaf nodes (recipients). To carry large numbers of multicast sessions, a network must minimize the sessions' resource consumption [1]. Therefore, it is important for a multicast session to adopt a multicast tree whose network cost is minimal. By network cost we mean the accumulation of the costs of resource usages of all the links constructing the multicast tree. This problem immediately is reduced to finding a Steiner Tree [2] which is one of the Karp's 21 NP-complete problems [3]. The tree cost should be minimized to the most possible extent. This is due to the fact that after the multicast tree is built, all the network traffic flows along the links of the tree, especially in real-time applications which are intrinsically connection-oriented. The less the tree cost, the less the valuable resources are used during the whole connection time. Finding minimal multicast tree gets more problematic when some Quality-of-Service (QoS) constraints such as delay (end-to-end delay), and bandwidth constraints are also to be considered at the same time. Finding the multicast tree or the Steiner tree under any of the aforementioned QoS constraints converts the problem into finding a constrained Steiner tree (QoS multicast routing) problem, which is NP-Complete itself [4].

Several methods [4–9] have applied heuristic to solve QoS multicast routing problem. KPP [4], BSMA [5], and some others [6–8] are notable heuristic works in computing multicast trees. However,

a comprehensive research in [9] has shown that most of the heuristic algorithms are notorious either for working too slowly or failing in computing of an optimized solution or both.

Some works [1, 10–13, 15–20] have mainly focused on applying GA to find constrained multicast routing trees. In [1] a bandwidth delay constrained least-cost multicast routing algorithm based on conventional GA has been proposed. It has used tree structure coding for chromosome representation and penalty functions for those candidate solutions that violate predefined thresholds. Besides, [10, 11] have solely emphasized on conventional GA. However, all these methods [1, 10, 11] suffer from lack of local search and also problem of premature convergence. Also, all these approaches generate their initial population mainly based on a randomized depth-first search algorithm [12, 13].This method suffers from applying uninformed search which most of the time performs worse than a good heuristic based informed search [14].

Some others [16, 17] apply Shimamoto's approach [18] for coding routing tables and multicast trees. In this method for each pairs of (source, multicast-destination) several paths are stored, and the final multicast tree is yielded through combining these paths. As the network size grows, maintaining these paths can itself be a problem.

The closest work to ours is [20] which presents a method namely NGSA for least-cost QoS multicast routing based on both GA and Simulated Annealing (SA) algorithm. This paper ([20]) adopts a rather new population initialization method mostly the same way as [12,13] which has two steps: trunk-creating and limb-appending. In trunk creating phase, a path is found from a source node to one of the multicast destinations. Then, in the limb-appending phase, other multicast destination nodes are appended to the trunk through randomly discovered paths. [20] also uses SA to escape from premature convergence, one of the eminent shortcomings of GA. However, finding paths in both phases is done through random uninformed selection of neighbors which suffers from deficiencies inherent in uninformed search methods mentioned before.

Considering the fact that multicast tree creation and maintenance time is crucial and meanwhile GA's evolution time toward better solution can be unpredictable [21], it is important to improve convergence speed. Generating improved initial population [22], and reducing search process are two effective approaches to achieve this goal.

[22] has shown that improving generation of initial population and being meticulous about it can significantly improve convergence time. All the aforementioned GA based researches applied random selection approaches in their initial population generation. Adopting such a randomized behavior may cause the algorithm to go astray in establishing a multicast tree at least for a while. Therefore, GA must compensate its improper selections by making further attempts, and this prolongs the convergence time. Therefore, instead of passing the buck to next generations in GA and expecting the next generations to compensate the primitive generations' probable fault, it is reasonable to make the first decision more scrupulously. This becomes more important when we cope with large networks.

Reducing search process can also be considered as another method to improve convergence time. Actually, reducing search process hinders GA algorithm-at least for a great extent-from blundering and moving back and forth and revisiting the same links for an excessive number of times in hope of finding a solution. Applying a randomed paradigm automatically causes GA to undesirably adopt try and error behevior to acheive a satisfactory solution.

In this paper, we have proposed and impelmented a new algorithm called GASANT which takes the aforementioned two improvements on GA into consideration. To put it in a nutshell, we have both provided improved initial population and reduced search process by deploying Ant Colony Optimization (ACO). By improved initial population, we mean that links constructing initial population are more likely to appear in optimal multicast tree. By reducing search process, we mean that for finding a satisfactory solution, GASANT visits edges of the network graph for a small number of times rather than excessive number of times. Besides, SA has been used to escape from getting stuck into local optimum solutions.

ACO is based on distributed society of autonomous agents called ants. Ants provides a valuable

approach of exploring the network and collecting information about link statuses like link-state protocols, but in an efficient and deliberate way. Ants can provide better initial population for GA, since they traverse through network and discover (near) optimal paths among nodes. The produced least-cost multicast tree more likely contains a subset of edges comprising these optimal paths. Thus, considering such paths while we want to establish a multicast trees can result in a solution in higher speed. Consequently, the search process is reduced. The experiment results proves this claim.

Also, note that ants are small packets and put low load burden on network nodes; their lightweight approach can be very effective for gathering information for generating QoS-aware initial population [23]. The experiments certify this claim, too.

The rest of the paper is organized as follows: Section 2 explains the problem formulation and modeling. Section 3 describes the proposed multicast routing algorithm (GASANT) in detail. In section 4, the convergence of GASANT is investigated. We evaluate GASANT comprehensively in section 5, and finally section 6 concludes the paper.

2 Problem Formulation and Modeling

In this paper, the network is expressed as an undirected weighted graph namely $G = (V, E)$, where V is the set of network nodes and E is the set of links connecting network nodes to each other. The link $e \in E$ with source node m and destination node n is denoted by (m, n). Multicast tree which is denoted as $MT(s, M)$ consists of two main parts: $s \in V$ as the source node of multicasting, and $M \subseteq V - \{s\}$ as the multicast destination nodes. Each link e is characterized by a QoS 3-tuple $(B(e), D(e), C(e))$ representing bandwidth, delay and cost associated with it respectively. Here, $B(e) > 0$, $C(e) \geq 0$ and $D(e) \geq 0$. This paper tries to discover a multicast tree with the minimum cost subject to two QoS constraints namely bandwidth, and delay constraints. If $p(s, d_i)$ is a path in the tree MT starting from the source node s and ending at a multicast destination node d_i, then bandwidth and delay constraints and cost function are defined as follows:

- *Bandwidth constraint*
  It is required that the minimum value of the link bandwidth in the multicast tree MT, along the path (B_{path}) originating at the source node s and ending at any multicast destination node $d_i \in M$ be greater than or equal to the predefined required bandwidth Ω_b.

- *Delay Constraint*
  It is required that the end-to-end delay in the multicast tree MT, along the path (D_{path}) originating at the source node s and ending at any multicast destination node $d_i \in M$ be smaller than or equal to the predefined maximum end-to-end delay Ω_d.

- *Cost Function*
  We define the cost of the (multicast) tree as the sum of the costs of all the links of the tree. Formula 1 formalizes it:

$$C_{tree}(MT) = \sum_{e \in MT} C(e) \tag{1}$$

  $C(e)$ typically represents the cost of the link monetarily or any administratively interested cost. The proposed algorithm of this paper aims at constructing a least-cost multicast tree subject to delay and bandwidth constraints defined above. Formalizing this problem as a constrained optimization problem, we have

$$if \quad C_{tree}(MT) = \sum_{e \in MT} C(e) \quad Then \quad Minimize(C_{tree}(MT))$$

$$subject \quad to \quad B_{path}(p(s,d_i)) \geq \Omega_b \quad and \quad D_{path}(p(s,d_i)) \leq \Omega_d$$

3 The proposed multicast routing algorithm

This section explains how GASANT utilizes the cooperation between GA and ACO to produce multicast trees. As shown in Fig. 1, GASANT consists of both reactive and proactive components. The proactive component itself has an ACO Module within. The reactive component is composed of two main modules namely Path Setup (PS) Module and Genetic and Simulated Annealing (GSA) Module.

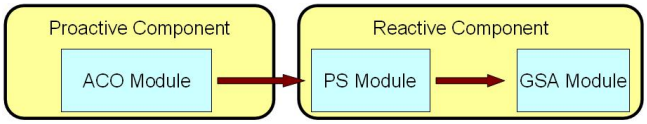


Figure 1: GASANT Architecture

In this paper, a modified version of AntNet [23] has been implemented in ACO Module. This modified version increases AntNet's performance and adapts it to multicast nature of routing. ACO Module proactively sends ants through network in order to find feasible paths to different destinations and keep nodes aware of network dynamism. This process continues periodically during the whole network uptime.

Beside this, the reactive component is responsible for multicast tree construction process. The process is carried out through two modules called PS Module and GSA Module. When a multicast tree creation request with a certain set of QoS criteria is issued by an application, PS Module starts to find paths between the multicast source node and each destination node. This process is done through propagating control messages throughout the network. These messages try to discover feasible paths considering the QoS metrics by fetching the information provided by ACO Module stored in intermediate nodes. Each of these paths is called trunk which is in fact a simple path that connects the source node to a multicast destination node. The result of PS Module is in fact a set of highly crowded paths which were frequently traversed by ants. The discovered paths are fed into the GSA Module. The GSA Module uses these paths as raw data to construct its initial population. It then tries to find a multicast tree through running crossover and mutation operators in iterations. If a multicast tree with required QoS metrics is found, then the mission is complete. Otherwise, GASANT commences a negotiation with the application that triggered the strict request in order to ask it to relax its QoS requirements. In the following sub-sections, we have explained these three modules in more detail.

3.1 ACO Module

At regular intervals, from every network node, ACO sends forward ants toward a destination node in order to discover feasible and satisfactory paths. Multicast destination nodes have higher chance of being selected as forward ant's destination nodes. Analogously to AntNet, ants will travel toward destination node and if successful, then they will return back to source node. While returning back to the source node, ants update routing table of the intermediate nodes en route. This update is based on a reinforcement value r (refer to section 4 of [23] for more detailed information) which is a function of the goodness of the path that the ant has just traversed. In GASANT, the value of r is calculated based on ant's traversed path trip time ($TripTime$), bandwidth ($Bandwidth$) and cost ($Cost$) through Formula 2:

$$r = c_1 \times (\frac{TripTime_{best}}{TripTime}) + c_2 \times (\frac{Bandwidth}{Bandwidth_{best}}) + c_3 \times (\frac{Cost_{best}}{Cost}) \quad (2)$$

Here, $TripTime_{best}$ is the best trip time experienced by the ants, and $Bandwidth_{best}$ and $Cost_{best}$ are the best bandwidth and the best cost discovered while travelling toward same destination over the last observation window (in this paper we considered last 20 samples), respectively. The coefficients c_1, c_2 and c_3 weigh the importance of each term. In our implementation, we have set each of these constants euqally to 0.33. In this way, all three QoS parameters are treated equally. Ants can easily keep track of bandwidth and cost of paths by saving minimum link bandwidth and summing the cost of individual links they are traversing.

3.2 PS Module

The multicast route discovery process triggers upon receiving a multicast tree setup request from a source to a number of destinations. The purpose of this reactive process is to find a set of trunks according to QoS metrics. This is accomplished by actually sending Multicast Tree Discovery (MTD) messages through network. These messages find links with high pheromones which can provide better trunks. In fact, this procedure mainly uses the information provided by ACO Module to find the best single paths to destinations. As soon as a multicast tree setup request is received, the source node starts propagating MTD messages to find feasible and shortest paths to multicast destinations. The source node broadcasts a limited number of MTD messages to its connected neighbors. This number is set through *branchFactor* variable which is the cardinality of a subset of the node's neighbors. Using this variable, the number of packets being broadcast throughout the network can be limited. Therefore, the traffic overhead caused by MTD messages can be controlled. This subsetting is carried out through choosing the neighbors which are connected via links with highest pheromones to the source node.

Each MTD message is composed of seven fields: 1.*requestID*, 2.*sourcenodeID*, 3.*cost*, 4.*bandwidth*, 5.*delay*, 6.*path*, 7.*branchFactor*. A MTD message stores the traversed path and its corresponding accumulated cost into its *path* and *cost* fields, respectively. The minimum bandwidth and total delay of the path are being kept track in *bandwidth* and *delay* fields. Each MTD message is associated with a *uniquerequestID* field; together with the *sourcenodeID* field, a network node can uniquely identify a MTD message. These two IDs together are considered as the $Identifier(ID)$ pair. When an intermediate node receives an MTD message, the node inspects the message's *ID* pair to check whether the message is a duplicate. In GASANT, the intermediate node is allowed to forward at most a limited number of duplicated MTD messages (messages with same *ID* pair) defined by *allowedDupNumber*. It is critical to somehow limit the number of packets being generated and forwarded through network or the network will get inundated with extraneous traffic.

Our experiments show that even with the *branchFactor* of 2 for a network of size 100 nodes, when the number of passing messages is not limited, more than millions of MTD messages of the same ID are created and passes the same nodes for more than tens of thousands times. Also, our experiments show that values around 50 for the *allowedDupNumber* for networks with more than 100 nodes can significantly reduce the message overhead while still providing vast amount of available trunks from source to destination nodes. After forwarding *allowedDupNumber* number of MTD messages of the same identifier pair, any subsequent MTD messages of this identifier pair is discarded by that intermediate node. To keep track of the number of times a particular MTD message has been forwarded, each node maintains a table of counters for each of the MTD messages the node has forwarded. If the counter of the received MTD message does not violate the *allowedDupNumber*, the node updates some fields of the message before forwarding. The *cost* field is increased by the cost of the link the message has just traversed. Also the current node's address is appended to the *path* list. The *bandwidth* field is set to minimum of the current bandwidth field value and the bandwidth of the traversed link. Once the update is complete, like the source node, this node forwards the MTD message to selected *branchFactor* number of its neighbors. The selection is based on the pheromone level entries of the probabilistic routing table [23]. The higher values have the higher chance of being selected. The value of *branchFactor* can significantly affect the

amount of messages overhead generated in the network. Higher *branchFactor* values will make messages explore more parts of the network and consequently gather more possible paths to destinations in expense of exponentially increasing message overhead.

In this paper, the *branchFactor* is initially set to a constant large value compared to average degree of the network graph nodes and then it is reduced on next hops. This way most likely all neighbors of the source node will receive the MTD message. As a result, search area is widened and more parts of the network get explored while still having reasonable load. Two logarithmic and linear methods have been tested for reducing the *branchFactor* along the path. In the logarithmic approach, the reduction is fast and *branchFactor* converges to one just after passing a few hops. When *branchFactor* value is one, the node forwards one copy of the received MTD message to its best candidate neighbor. In this way, the *branchFactor* and *allowedDupLimit* together can control the amount of MTD message overhead. A constant *branchFactor* through the whole network is also investigated. However, our experiments show that the logarithmic approach surpasses the other two approaches in controlling the overhead while still producing nearly the same trunks as the others (constant and linear reduction approaches) produce.

When a MTD message reaches its destination node, a reply message containing the same data as the MTD message is generated and is sent back to source node. The reply message traverses the same path, but will be queued in high priority queues. When the first message reaches back to the source node, the source node triggers a timer to collect as many routes as possible from different destinations. As the timeout expires, a set of trunks are extracted from the *path* field of the collected reply messages. These trunks are then sent to the GSA Module.

3.3 GSA Module

In the following subsections, the specifications and operators of the GSA Module are explained in detail.

Coding

The tree structure coding has been chosen in this paper. In this method, every chromosome represents a multicast tree. As a result, the coding space is greatly reduced and coding-decoding operation is omitted and the meaning of genetic operations becomes more visual and the time of conversion between encoding and solution spaces is saved [1].

Pruning

In this phase, those links with a bandwidth less than the predefined bandwidth threshold are deleted. It is probable that the pruned graph gets decomposed into several smaller connected subgraphs. If all the multicasting nodes including the source node are all in the same subgraph, this means that the pruned network satisfies the bandwidth threshold. Otherwise, the source node should start a new round of negotiations with the applicant program in order to reach a mutual satisfactory agreement with more relaxed threshold.

Initial population formation

Initial population is performed in two main stages: trunk-selection and limb-appending.

Trunk-selection: In this stage, one of the trunks created by the ACO Module is selected randomly. A trunk is a simple path that connects the source node to a multicast destination node (as in section 3). This trunk is supposed as the current multicast tree.

Limb-appending: In this stage, a multicast destination node which is not part of the current multicast tree (obtained from previous stage) is selected as the current node. Then, one of the links leaving this current node is selected. The more pheromone deposited on the link, the more probable the link is selected. After this, the other end of the link is selected as the current node, and the same action is done about it. This procedure continues until the current node becomes one of the nodes of the current multicast tree. At this time, all of these nodes which in turn were as current nodes and also the current multicast tree nodes together are set to the new current multicast tree. On the condition that the entire multicast destination nodes are in the current multicast tree, the mission is complete; otherwise the whole limb-appending procedure is repeated for the rest of the multicast destination nodes isolated from the current multicast tree.

Applying selections based on high pheromone values is more efficient than that of [20] which uses a random paradigm. This seems reasonable since the trunk-selection and limb-appending methods use previously gathered global information in making their decisions. The experiments in section 5 also prove this claim.

Fitness Function

Basically, penalty functions are used to handle the constraints [24]. Through these functions the constrained problems are transformed to unconstrained problem. In fact, these functions are used to penalize the individuals based on their constraint violation. The penalty imposed on infeasible individuals can range from completely rejecting the individual to decreasing its fitness based on the degree of violation [16]. In this paper, we have adopted the same fitness function formulae as in [20]. We have utilized penalty function in determining the fitness of each individual. Here, the fitness of the multicast tree MT is defined as Formula 3:

$$Fitness(MT) = e^{\frac{-Penalty(MT)}{T}} \tag{3}$$

Where, T is the temperature in which this fitness is calculated and *Penalty* is the amount of penalty that has been considered for MT. The *Penalty* itself is calculated via Formula 4:

$$Penalty(MT) = k_c \times Cost_{tree}(MT) + k_b \times BV(MT) + k_d \times DV(MT) \tag{4}$$

Here, $Cost_{tree}(MT)$ is calculated via formula 1; $BV(MT)$, and $DV(MT)$ are determined via Formulae 5 and 6; k_c , k_b, and k_d are the constants to weigh the importance of each of the cost, and the violations occurred from bandwidth and delay constraints, respectively.

$$BV(MT) = \sum_{d_j \in MT} maximum(\Omega_b - B_{path}(s, d_j), 0) \tag{5}$$

$$DV(MT) = \sum_{d_j \in MT} maximum(D_{path}(s, d_j) - \Omega_d, 0) \tag{6}$$

As it can be understood from all above, the less violation an MT makes (Formulae 5,6), the fewer penalties are considered for it (Formula 4), and consequently, such an MT is fitter to the delay and bandwidth constraints (Formula 3).

Selection Method

In this paper, fitness proportionate selection (roulette wheel selection) has been applied. In this method, the probability of selecting a chromosome C_i as a parent is defined as follows (Formula 7):

$$SelectionProbability = \frac{Fitness(Fitness(C_i))}{\sum_{j=1}^{PopulationSize} Fitness(C_j)} \tag{7}$$

Where $Fitness(C_i)$ is the fitness of chromosome i. In fact, more elitist chromosomes have higher chances to be selected in comparison with their counterparts.

Adaptive Crossover Method

Crossover is a genetic operator that combines two chromosomes called parents to produce a new chromosome namely offspring. Since experiments in [25] indicated that adaptive crossover performed as well or better than a traditional crossover, in this paper, adaptive crossover probability is applied. Here, the crossover probability is calculated through Formula 8:

$$CrossoverProbability = \begin{cases} c_1 + c_2 \times \frac{fit_{max} - fit\prime}{fit_{max} - fit_{avg}} & fit\prime \geq fit_{avg} \\ c_3 & fit\prime < fit_{avg} \end{cases} \tag{8}$$

Here, fit_{avg} is the average of the fitness of the population in the current generation; fit_{max} is the biggest fitness existent in the current generation; fit' is the maximum of the fitnesses associated with the two parents to be crossovered; c_1. c_2, c_3 are constants and $c_3 = c_2 + c_1$.

In this paper, for crossovering two parent trees, somewhat the same method as in [12] has been adopted. For more clarity, the crossover and mutation (next section) processes for a hypothetical network has been illustrated in Fig. 2. Suppose that the network graph is the same as in Fig. 2.a in which node 0 (green node) is the source node and the nodes 2 and 4 are the multicast destination nodes (yellow nodes). Also, assume that Fig. 2.b.1 are the two possible multicast trees. Now, for crossovering, first of all, the common edges of the two parent trees (Fig. 2.b.1) are extracted and are inserted to the offspring tree as the first set of edges (Fig. 2.b.2). The produced offspring is not necessarily a single connected tree and may consist of several disconnected components. In this offspring, it is likely that some of the multicast destination nodes fall into graph components (orphan components) other than the component which the source node is in (main component). Therefore, somehow these orphan components must be connected to the main component. To this end, at first, an orphan component is randomly selected. Then, it is tried to connect this orphan component to the main component through highly pheromone bridge links (links existent in network graph that can connect two different components to each other). While doing this procedure, these selected bridge links may also connect other orphan components to each other. If there may still be orphan components that are remained disconnected, the same procedure should be performed about them. The produced multicast tree for our example is now the same as Fig. 2.b.3. Of course, some extra nodes may appear as the leaf nodes of offspring tree. In fact, these nodes are not member of the multicast tree destinations (node 5 in Fig. 2.b.3); thus, such nodes and their entering links should be removed after crossover (Fig. 2.b.4).

Adaptive Mutation Method

Mutation is a genetic operator that alters one or more gene values in a parent and produces a new offspring. This operator can prevent the population from stagnating at any local optima. In this paper, a simulated annealing technique has been adopted for mutation operator. Regarding this, each new offspring produced by the mutation operator is considered as a neighbor of the initial parent. This new offspring replaces its parent according to a probability calculated in Formula 9:

$$MutationProbability = \begin{cases} e^{\frac{-(fit - fit\prime)}{T}} & fit > fit\prime \\ 1 & fit \leq fit\prime \end{cases} \tag{9}$$

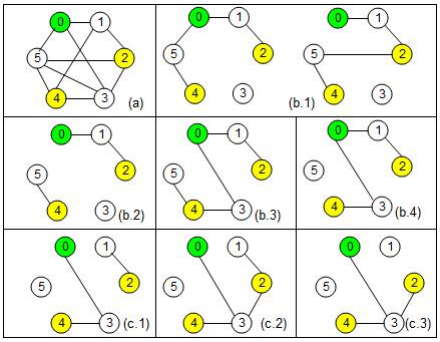


Figure 2: Crossover and Mutation Operations. (a) The hypothetical network graph. (b.1-4) Stages of crossover operator. (c.1-3) Stages of mutation operator.

Here, fit and fit' are the fitnesses associated with the parent and offspring, respectively, and T is the temperature. As can be inferred from Formula 9, if the offspring is better than the parent, it replaces the parent in the next generation, otherwise this replacement is done through an exponential probability. The temperature decreases gradually from a high initial degree. As this decreasing continues, the probability of the replacement of worse offspring decreases accordingly. Continuing our example, suppose that we want to apply mutation operator for Fig. 2.b.4. Mutation for a parent tree (Fig. 2.b.4) which leads to finding a neighbor offspring tree is done through the following procedure: a random edge is deleted from the parent tree (edge (0,1) from Fig. 2.b.4 is deleted). Then, it is tried to reconnect these two subtrees (Fig. 2.c.1) with the same procedure explained in section 3.3. This new connected tree (Fig. 2.c.2) is considered as the offspring (neighbor) of the parent tree. This offspring replaces its parent according to a probability calculated in formula 9. The same as the crossover, some extra links may appear as the leaf nodes of offspring tree. In fact, these nodes are not member of the multicast tree destinations (node 1 in Fig. 2.c.2); thus, as mentioned in crossover, they should not be included in the final multicast tree, and as a result Fig. 2.c.3 is the real neighbor of the Fig. 2.b.4.

4 Analysis of Convergence

In Theorem 2.7 of [26], Guoliang et al. has shown that applying GA can finally lead the problem to converge to a global optimized solution. Achieving an optimal solution for the aforementioned multicast QoS routing can sometimes take up a great deal of time. This is due to the NP-completeness property of the problem. Nevertheless, most of the times, achieving a solution is possible by well-adjusting various parameters in the proposed algorithm.

5 Experiments

GASANT has been implemented in C++. The ACO Module of GASANT has been implemented by extending and modifying the AntNet package provided in [27]. The whole program is simulated in $OMNeT++$ environment on a Pentium IV 2.4 GHz CPU, 2 GB RAM. To make the results independent of the networks under simulation and also in order to run experiments on a reasonable amount of graphs, we have used random graph generator introduced by Waxman [28]. In this method, the network nodes are randomly scattered in a rectangular environment. The probability of existence of an edge between two nodes u, v is calculated via Formula (10):

$$P(u,v) = \alpha e^{\frac{-d(u,v)}{\beta L}} \tag{10}$$

Here, $d(u, v)$ is the distance between two nodes u and v; L is the maximum distance between any two nodes in the generated graph. $\alpha \in (0, 1]$ is responsible for controlling the average degree of the random graph. The greater α results in a denser graph; a graph with more links. In the same way but rather different, $\beta \in (0, 1]$ effects the number of longer edges. The larger values of β increase the number of longer edges. Also, the randomly generated edges of the graph accepts their cost from [1,100], delay from [0.01,0.1] ms, and bandwidth values from range [10,50] Mbps. GASANT was run on 20, 40, 60, 80, and 100-node network graphs. Also, three different percentages of network nodes were selected to be multicast destinations in the runs: 10%, 20%, and 30%. We call these percentages as multicast percentages, or briefly *mp*. For the sake of convenience, the delay and bandwidth bounds are supposed to be the same for all multicast destinations. Besides, GSA module of GASANT iterated for 15 generations, and the population size of each of these generations was 30. All the experiments were run until we reach a confidence interval of less than 5%, using 95% confidence level. Before delving into the experiements, we need to define routing request *Success Ratio* [29] which is defined as Formula 11:

$$SuccessRatio = N_{ack}/N_{req} \tag{11}$$

where the N_{req} is the number of multicast tree requests issued by the application, and N_{ack} is the number of these requests that are successfully answered. In the next subsections, we compare GASANT and NGSA through a comprehensive set of experiments.

5.1 Success Ratio Analysis

In this paper, we have compared GASANT to NGSA [20] which is one of the recent works in QoS multicast routing literature. Table 1 illustrates the *Success Ratio* of GASANT to NGSA as a function of different parameters.

Table 1 displays the *Success Ratio* of GASANT to NGSA as a function of network size and multicast percentage (*mp*). Increasing multicast percentage and network size usually results in higher $SuccessRatios$ of GASANT to NGSA. This is due to the fact that as network size or multicast percentage increases, applying random paradigm used in trunk creation and limb appending is more likely to adopt improper links toward multicast destination nodes. This figure can be regarded as a scalability performance, too. Table 1 and Table 1 illustrate the *Success Ratios* of GASANT to NGSA as functions of minimum possible bandwidth and maximum possible end-to-end delay constrained on the problem, respectively. As it can be inferred from these figures, independent from these constraints, the *Success Ratio* of GASANT is most often higher than NGSA's one.

Table 1: (a) *Success Ratio* of GASANT To NGSA as a function of network size and multicast percentage. (b) *Success Ratio* of GASANT To NGSA as a function of minimum possible bandwidth.(c) *Success Ratio* of GASANT To NGSA as a function of maximum possible end-to-end delay.

Net. Size	mp=0.1	mp=0.2	mp=0.3
20	0.83	1	1
40	0.78	1.05	1.2
60	0.83	1.15	1.25
80	0.96	1.05	1.08
100	1	1.25	1.66

Min. BW.	Success R.
10	1.1
15	1.2
20	1.23
25	1.22
30	1.17
35	1.02
40	1.07

Max. Delay	Success R.
0.10	1.4
0.20	1.0
0.30	1.5
0.40	1.8
0.50	1.22
0.60	0.81
0.70	1.04

Table 2: (Part 1) Average time ratio of GASANT to NGSA for detecting the first satisfactory multicast tree. (Part 2) Average cost ratio of GASANT to NGSA for the first detected satisfacroty multicast tree. (Part 3) Average maximum end-to-end delay ratio of GASANT to NGSA for the first detected satisfacroty multicast tree. (Part 4) Average minimum bandwidth ratio of GASANT to NGSA for the first detected satisfacroty multicast tree.

	Part	1			2			3			4		
Network Size	mp	0.1	0.2	0.3	0.1	0.2	0.3	0.1	0.2	0.3	0.1	0.2	0.3
20		0.11	0.16	0.49	0.78	0.91	0.89	0.57	0.80	0.85	1.12	0.98	1.04
40		0.14	0.43	0.01	0.80	0.97	0.72	0.67	0.93	0.78	0.99	0.98	1.00
60		0.08	0.01	0.54	0.78	0.80	0.85	0.72	0.81	0.93	1.02	1.07	1.00
80		0.26	0.01	0.13	0.83	0.89	0.82	0.85	0.82	0.97	0.98	1.02	1.01
100		0.74	0.67	0.06	0.87	0.92	0.66	0.96	0.76	0.91	1.08	1.01	0.95

5.2 Comparison of Execution Times, Quality of Solutions and Generations

Part 1 of the Table 2 illustrates the average time ratio of GASANT to NGSA for detecting the first satisfactory multicast tree. As can be seen, the average required time for achieving the first solution by GASANT is almost less than half of the time required by NGSA. The average cost and average maximum end-to-end delay associated with the first solution discovered by GASANT, as shown in Parts 2 and 3 of the Table 2, are always less than that of NGSA's. Also, Part 4 of the Table 2 demonstrates that the average minimum existent bandwidth is nearly the same for both multicast trees discovered by GASANT and NGSA. Part 1 of Table Table 3 illustrates the average cost ratio of GASANT to NGSA for every fifth generation (yellow rows) and the average cost ratio of GASANT to NGSA for the least cost multicast tree existent in that generation (white rows) as a function of network size and multicast percentage.

As it is obvious, the cost of the trees in each generation associated with GASANT is less than that of NGSA's. Also, the least cost trees existent in each generation of GASANT has smaller cost in comparison with NGSA's. As can be inferred from this part of the table, ACO Module helps GSA start the production of generations with smaller cost trees and this continues along the rest of the generations till end.

Part 2 of Table Table 3 illustrates the average maximum end-to-end delay ratio of GASANT to NGSA for each generation (yellow rows) and the average maximum end-to-end delay ratio of GASANT to NGSA associated with the least cost multicast tree existent in that generation (white rows) as a function of network size and multicast percentage. Part 3 of Table Table 3 illustrates the average minimum bandwidth ratio of GASANT to NGSA for each generation (yellow rows) and the average minimum bandwidth ratio of GASANT to NGSA associated with the least cost multicast tree existent in that generation (white rows) as a function of network size and multicast percentage.

All the parts of the Table Table 3 illustrate that the QoS of multicast trees discovered by GASANT is superior to the trees discovered by NGSA by having smaller cost, end-to-end delay and nearly the same bandwidth. It can be inferred from this table that ACO module donates a more global view of the network to GASANT in comparison with the NGSA which only relies on bare GA and SA.

Table 3: (Part 1)Yellow rows: Average cost ratio of GASANT to NGSA for every fifth generation as a function of network size and multicast percentage. White rows: Average cost ratio of the least cost multicast tree for every fifth generation as a function of network size and multicast percentage. (Part 2) Yellow rows: Average maximum end-to-end delay ratio of GASANT to NGSA for every fifth generation as a function of network size and multicast percentage. White rows: Average maximum end-to-end delay ratio of the least cost multicast tree for every fifth generation as a function of network size and multicast percentage. (Part 3) Yellow rows: Average minimum bandwidth ratio of GASANT to NGSA for every fifth generation as a function of network size and multicast percentage. White rows: Average minimum bandwidth ratio of the least cost multicast tree for every fifth generation as a function of network size and multicast percentage.

Network Size	mp%	Part 1 - Generations				Part 2 - Generations				Part 3 - Generations			
		1	5	10	15	1	5	10	15	1	5	10	15
20	10	0.63	0.75	0.81	0.9	0.63	0.67	0.72	0.87	1.01	1.09	1.08	1.03
20	10	0.77	0.97	1	1	0.56	0.98	1	1	1.12	1.02	1	1
20	20	0.81	0.86	0.88	0.95	0.77	0.83	0.84	0.9	1.03	1.03	1.03	1.03
20	20	0.9	0.94	0.97	0.98	0.78	0.86	0.97	0.97	1	1.05	1.02	1.01
20	30	0.86	0.99	0.94	0.83	0.8	0.81	0.88	0.86	1.02	1.03	0.99	0.99
20	30	0.89	0.88	0.92	0.93	0.85	0.73	0.75	0.83	1.04	0.99	1	1
40	10	0.68	0.75	0.86	0.99	0.69	0.69	0.79	0.85	1.02	1	1	0.98
40	10	0.68	0.87	0.94	0.96	0.69	0.69	0.79	0.85	0.99	1.01	0.95	0.93
40	20	0.82	0.88	0.91	0.81	0.8	0.86	0.92	0.85	1	0.99	0.99	1
40	20	0.91	0.84	0.86	0.88	0.84	0.83	0.87	0.89	0.99	1.02	1.02	0.99
40	30	0.7	0.81	0.73	0.85	0.64	0.74	0.72	0.65	1.03	1	0.99	0.98
40	30	0.65	0.82	0.78	0.8	0.67	0.73	0.61	0.68	0.98	1.02	1.06	0.93
60	10	0.65	0.87	0.91	0.77	0.7	0.81	0.79	0.67	1.02	1.02	1.01	1.03
60	10	0.67	0.86	0.87	0.93	0.63	0.71	0.68	0.73	1.04	1.04	1.02	1.02
60	20	0.76	0.87	.87	0.87	0.71	0.79	0.81	0.78	1	1.04	1.06	1.01
60	20	0.68	0.78	0.9	0.85	0.7	0.71	0.8	0.69	1.05	1.08	1.06	1.04
60	30	0.8	0.89	0.85	0.89	0.78	0.89	0.8	0.81	1	1.01	1.01	1.01
60	30	0.85	0.84	0.84	0.86	0.91	0.87	0.86	0.83	1	1.01	1.01	1
80	10	0.75	0.83	0.8	0.74	0.81	0.87	0.8	0.77	1	1	1.02	0.99
80	10	0.63	0.77	0.77	0.83	0.68	0.7	0.76	0.75	0.98	1	1	1
80	20	0.74	0.93	0.91	0.83	0.69	0.87	0.89	0.79	1.01	1.01	1.01	1.04
80	20	0.77	1	0.87	0.87	0.74	0.9	0.81	0.84	1.02	1	1.02	1.02
80	30	0.72	0.94	0.91	0.97	0.71	0.96	0.9	0.88	1	1	1	1
80	30	0.77	0.88	0.92	0.97	0.82	0.91	0.82	0.85	1.01	1.01	1	1
100	10	0.4	0.85	0.84	0.84	0.42	0.86	0.85	1	1.05	1.03	1.07	1.09
100	10	0.49	0.77	0.78	0.8	0.59	0.86	0.81	0.8	1.12	1.09	1.07	1.12
100	20	0.67	0.74	0.77	0.7	0.62	0.64	0.75	0.61	1.02	1.01	1	1
100	20	0.79	0.93	0.91	0.98	0.7	0.81	0.9	0.77	1	1	0.99	0.97
100	30	0.65	0.56	0.4	0.29	0.66	0.53	0.43	0.38	0.99	0.96	0.99	0.99
100	30	0.63	0.63	0.66	0.66	0.78	0.78	0.91	0.91	0.95	0.95	0.95	0.95

Table 3: (Part 1) Edge hit ratio of GASANT to NGSA for detecting the first satisfactory multicast tree. (Part 2) Average Overhead Analysis of ACO Module.

	Part 1			Part 2
Network Size	mp=0.1	mp=0.2	mp=0.3	Total Consumed Bandwidth (Kbps)
20	0.04	0.12	0.26	0.12
40	0.14	0.28	0.01	0.28
60	0.08	0.16	0.67	0.45
80	0.23	0.24	0.07	0.62
100	0.48	0.33	0.09	0.88

5.3 Analysis of Reduction of Search Process

Part 1 of Table 3 illustrates edge hit frequency ratio of GASANT to NGSA during the detection of the first satisfactory multicast tree. Edge hit frequency is the number of times that the edge is visited by the multicast tree discovery algorithm during trunk creation or limb appending phases. As can be seen, edge hit ratio of GASANT is considerably smaller than NGSA's. This implies that GASANT visits the edges of the network graph sufficiently not excessively and redundantly. This redundancy can be a symptom of high try and error nature of NGSA before achieving a satisfactory multicast tree. Whereas, GASANT has a global view of the network specific properties. Thus, with a smaller effort, it gains a solution even with higher QoS. Therefore, GASANT discovers a satisfactory solution with a smaller search process. GASANT owes its success to non-stoping efforts of diligent ants of the proactive ACO Module.

5.4 Overhead Analysis of ACO Module

Part 2 of Table 3 illustates the total bandwidth consumed by control messages of ACO module as a function of network size. As can be seen, by increasing network size the amount of overhead increases almost linearly. This implies that the proactive ACO module is scalable as the network size grows.

6 Conclusion

In this paper, we have proposed GASANT to solve the NP-Complete problem of finding a QoS constrained least-cost multicast tree for a given communication network. To this end, we have combined ACO, GA and SA together to utilize the full advantages of them. Here, ACO is responsible for both improved initial population which are fed into GA, and also reduced search process. By improved initial population, we mean that links constructing initial population are more likely to appear in optimal multicast tree. By reduced search process, we mean that the GA visits the network edges for a small number of times rather than moving back and forth on the same and previously-visited edges for many times. For fleeing from standing still around a local optimal solution and also extending search space SA has been deployed. Experiments show that ants in ACO provides a valuable approach of exploring the network and collecting information about states of the links in an efficient and deliberate way. In this way, GA is fed by a better initial population in its first step. Also, it considers the links that are highly recommended by ants to be found more likely in the final multicast tree and consequently reaches to a satisfactory solution sooner. The experiments in this paper ensure the aforementioned claims, and prove that GASANT outperforms its close counterpart NGSA.

Bibliography

[1] W. Zhengying, S. Bingxin, Z. Erdun, Bandwidth-delay-constrained least-cost multicast routing based on heuristic genetic algorithm Computer Communications, Volume 24, Issues 7-8, April 2001.

[2] F.K. Hwang, D.S. Richards, P. Winter, The Steiner Tree Problem, Annals of Discrete Mathematics, vol. 53, 1992.

[3] R. M. Karp, Reducibility Among Combinatorial Problems, Complexity of Computer Computations, 1972.

[4] V. P. Kompella, J. Pasquale, G. C. Polyzos, Multicast routing for multimedia communication IEEE/ACM Transactions on Networking. 1(3) (1993) 286-292.

[5] M. Parsa, Q. Zhu, J.J. Garcia-Luna-Aceves, An iterative algorithm for delay-constrained minimum-cost multicasting, IEEE/ACM Transactions on Networking 6 (4) 1998.

[6] R. Widyono, The design and evaluation of routing algorithms for realtime channels, Technical Reports TR-94-024, Tenet Group, Department of EECS, University of California at Berkeley, 1994.

[7] A.G. Waters, A new heuristic for ATM multicast routing, 2nd IFIP Workshop on Performance Modeling and Evaluation of ATM networks, 1994.

[8] Q. Sun, H. Langendorfer, An efficient delay-constrained multicast routing algorithm, Journal of High-Speed Networks 7(1) 1998 43-55.

[9] H.F. Salama, D.S. Reeves, Y. Viniotis, Evaluation of multicast routing algorithms for real-time communication on high-speed networks, IEEE Journal on Selected Areas in Communications 15(3) (1997) 332-345.

[10] F. Xiang, L. Junzhou, W. Jieyi, G. Guanqun, QoS routing based on genetic algorithm, Computer Communications 22(15-16) (1999) 1392-1399.

[11] A.T. Haghighat, K. Faez, M. Dehghan, A. Mowlaei, Y. Ghahremani, GA-Based Heuristic Algorithms for QoS Based Multicast Routing, knowledge-based systems 16 (2003) 305-312.

[12] C.P. Ravikumar, R. Bajpai, Source-based delay-bounded multicasting in multimedia networks, Computer Communications 21 (1998) 126-132.

[13] Z. Wang, B. Shi, E. Zhao, Bandwidth-delay-constrainted least-cost multicast routing based on heuristic genetic algorithm, Computer Communications 24 (2001) 685-692.

[14] S. Russell and P. Norvig, Artificial Intelligence: A Modern Approach, Prentice-Hall, 2003.

[15] Q. Zhang, Y.W. Leung, An orthogonal genetic algorithm for multimedia multicast routing, IEEE Trans. Evol. Comput. 3 (1) (1999) 53-62.

[16] K. Vijayalakshmi, and S. Radhakrishnan, Dynamic Routing to Multiple Destinations in IP Networks Using Hybrid Genetic Algorithm (DRHGA), , International Journal of Information Technology. 4(1) 2008.

[17] Vijayalakshmi ,S. Radhakrishnan, Artificial immune based hybrid GA for QoS based multicast routing in large scale networks (AISMR), Computer communications, 2008.

[18] N. Shimamoto, A. Hiramatsu, K. Yamasaki, A dynamic routing control based on a GA, In proceedings of the IEEE International Conference on Neural Network (1993) pp. 1123-1128.

[19] J.J. Wu, R.H. Hwang, H.I. Lu, Multicast routing with multiple QoS constraints in ATM networks, Information Sciences 124 (2000) 29-57.

[20] Li Zhang, Lian-bo Cai, Meng Li, Fa-hui Wang, A method for least-cost QoS least-Cost multicast routing based on genetic simulated annealing algorithm, Computer Communications, 31 (2008) 3984-3994.

[21] Sushil J. Louis Gregory J. E. Rawlins, Predicting Convergence Time for Genetic Algorithms, Technical Report, Indiana University, 1992.

[22] R. R. Hill, Ch. Hiremath, Improving genetic algorithm convergence using problem structure and domain knowledge in multidimensional knapsack problems, International Journal of Operational Research 2005 - Vol. 1, No.1/2 pp. 145 - 159.

[23] G. Di Caro, M. Dorigo, AntNet: Distributed Stigmergetic Control for Communications Networks, Journal of Artificial Intelligence Research 9 (1998) 317-365.

[24] Özgür Yeniay, Penalty Function Methods for Constrained Optimization with Genetic Algorithms, Journal of Mathematical and Computation Applications, 10(1) (2005) pp. 45-56.

[25] J. D. Schaffer, A. Morishima, An Adaptive Crossover Distribution Mechanism for Genetic Algorithms, ICGA 1987.

[26] C. Guoliang, W. Xufa, Z. Zhenquan, Genetic Algorithm and its Application, People's Posts and Telecommunications Press, 1996.

[27] M. Farooq, Implementation of AntNet in OMNeT++, http://www.omnetpp.org/component/content/article/9-software/3559, November 2009.

[28] B.M.Waxman, Routing of multipoint connections. IEEE J. Select. Areas Commun. 6(9) (1998) 1617-1622.

[29] S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi, Optimization by Simulated Annealing, Science, New Series, Vol. 220, No. 4598 (1983) pp. 671-680.

INT J COMPUT COMMUN, ISSN 1841-9836
Vol.7 (2012), No. 3 (September), pp. 432-449

Performing MapReduce on Data Centers with Hierarchical Structures

Z. Ding, D. Guo, X. Chen, X. Luo

Zeliu Ding, Deke Guo, Xueshan Luo
Key Lab of Information System Engineering,
School of Information Systems and Management,
National University of Defense Technology,
Changsha 410073, China
E-mail: zeliuding@nudt.edu.cn, guodeke@gmail.com
xsluo@nudt.edu.cn

Xi Chen
School of Computer Science, McGill University,
Montreal H3A 2A7, Canada
E-mail: chenxiwarm@gmail.com

Abstract:
Data centers are created as distributed information systems for massive data storage and processing. The structure of a data center determines the way that its inner servers, links and switches are interconnected. Several hierarchical structures have been proposed to improve the topological performance of data centers. By using recursively defined topologies, these novel structures can well support general applications and services with high scalability and reliability. However, these structures ignore the details of some specific applications running on data centers, such as MapReduce, a well-known distributed data processing application. The communication and control mechanisms for performing MapReduce on the traditional structure cannot be employed on the hierarchical structures.
In this paper, we propose a methodology for performing MapReduce on data centers with hierarchical structures. Our methodology is based on the distributed hash table (DHT), an efficient data retrieval approach on distributed systems. We utilize the advantages of DHT, including decentralization, fault tolerance and scalability, to address the main problems that face hierarchical data centers in supporting MapReduce. Comprehensive evaluation demonstrates the feasibility and excellent performance of our methodology.
Keywords: MapReduce; Data Center; distributed hash table (DHT).

1 Introduction

In the recent years, the data centers have emerged as distributed information systems for massive data storage and processing. A data center provides many online applications [8] [2] and infrastructure services [4] [13] through its large number of servers, which are interconnected via high-speed links and switches. These devices construct the networking infrastructure of a data center, named data center network [9]. The structure of a data center network determines the way these devices are organized. To design suitable structures and make them match well with the data storage and processing applications are fundamental challenges.

MapReduce, proposed by Google, is a well-known and widely used data processing mechanism on data centers [6]. MapReduce works by separating a complex computation into Map tasks and Reduce tasks, which are performed in parallel on hundreds or thousands of servers in a data center. MapReduce provides a good control and execution mode for distributed computing and cloud computing [18]. Users

without any experience of distributed programming can easily process terabytes of data on data centers with the help of MapReduce.

Nowadays, increasingly diverse applications and services call for an improvement of data centers in the topological performance, including scalability, reliability, etc. Especially, users' various requirements for processing large amount of data results in an exponentially increasing number of servers. The traditional structure, however, can hardly sustain the incremental expanding of data centers [10]. Several novel data center structures, such as DCell [9], FiConn [15], and BCube [11], have been proposed to optimize the topological performance of data centers. These structures are all recursively defined to construct data center networks, interconnecting the servers by a hierarchical way. We represent them as hierarchical structures. These structures mainly focus on the scalability and reliability for data centers. However, the details of some specific applications running on the data centers with these structures are ignored. For example, none of these hierarchical structures treat servers as masters and workers, although this requirement is the basis of many distributed data processing applications, especially MapReduce. The communication and control mechanisms for performing MapReduce on the traditional structure can hardly be operated on these hierarchical structures.

To solve this problem, this paper presents a methodology for performing MapReduce on the data centers with hierarchical structures, represented as hierarchical data centers for short. Our methodology is based on the distributed hash table (DHT) [20] [21], an efficient data retrieval approach on distributed systems. DHT works by assigning each server a hash table that records the range of keys handled by all adjacent servers. Responsibility for hashing data to keys is distributed among the servers. This approach possesses several advantages. First, DHT makes all servers freely communicate with each other without any central coordination. This provides a control mechanism for MapReduce on hierarchical data centers without designating the masters or workers. Second, DHT ensures that the whole system can tolerate any single node failure. Since the information of a server is held by its adjacent servers, a failed server can affect only its neighbors, which causes a minimal amount of disruption [22]. Finally, DHT can flexibly deal with a large amount of nodes joining or leaving the system. This matches well with the scalability of hierarchical data centers. These advantages bring a feasibility to perform MapReduce on hierarchical data centers.

In this paper, we address the main problems that face hierarchical data centers in supporting MapReduce. Our methodology utilizes the above advantages of DHT to execute the procedure of MapReduce on hierarchical data centers. Comprehensive evaluation shows that our methodology is effective and possesses excellent performance. The main contributions of this paper are as follows.

- First, we propose the schemes for designating master servers and worker servers, and storing data files on hierarchical data centers, so as to facilitate the execution of MapReduce.

- Second, we present a specific DHT architecture and a corresponding routing scheme for assigning Map and Reduce tasks and delivering intermediate data on hierarchical data centers. Comprehensive evaluation demonstrates that our scheme can evenly distribute the workload and well support throughput-hungry MapReduce applications.

- Third, we deal with server and switch failures by proposing suitable fault-tolerant approaches for performing MapReduce on hierarchical data centers. Experimental results prove that our methodology is a reasonable solution even considering node failures.

The remainder of this paper is organized as follows: Section 2 introduces the background, related work and our motivation. Section 3 proposes the schemes for executing the basic procedure of MapReduce on hierarchical data centers. Section 4 presents the DHT architecture and routing scheme. Section 5 looks into the fault-tolerant routing and issues for performing MapReduce on hierarchical data centers. Section 6 evaluates the performance of the proposed methodology. Section 7 concludes this paper.

2 Preliminaries

2.1 Background

With a simple and practical processing procedure, MapReduce provides a standard mechanism for distributed data processing. A basic MapReduce procedure consists of a Map phase and a Reduce phase [5]. Each phase includes multiple parallel Map or Reduce tasks, respectively. A MapReduce procedure can process terabytes of data through numbers of Map and Reduce tasks.

Map tasks are applied for data classification and preparing intermediate data for Reduce tasks. By means of predefined Map programs, Map tasks transform the input data into intermediate data, which are organized as key/value pairs, and then deliver those intermediate data with the same key to corresponding Reduce tasks. The keys represent the types of intermediate data. The values represent the content of intermediate data.

Reduce tasks are responsible for merging those intermediate data and producing output files. After retrieving intermediate data from Map tasks, Reduce tasks integrate the intermediate values associated with the same key by means of predefined Reduce programs, and therefore generate output values.

In a data center, MapReduce lets a master server control many worker servers in executing Map and Reduce tasks [7]. The master assigns each Map or Reduce task to a worker, and each Map task is assigned to the worker that stores the input data for the Map task. The workers executing Map and Reduce tasks are called Mappers and Reducers, respectively. In the Map phase, Mappers execute corresponding Map tasks simultaneously. When Mappers accomplish Map tasks, they store derived intermediate data on local disks, and then send the location information of intermediate data to the Master. In the Reduce phase, the master distributes the location information of intermediate data to Reducers. Then Reducers read corresponding intermediate data from Mappers and execute their respective Reduce tasks simultaneously. Fig.1 shows the basic process of a MapReduce procedure.

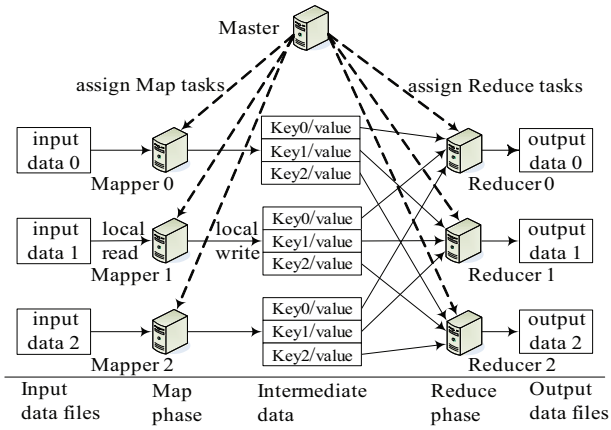


Figure 1: The basic process of a MapReduce procedure

2.2 Related Work

Many existing data centers adopt the traditional tree structure. Namely, all servers are located at leaf nodes. Aggregation switches and core switches are placed at inner nodes and root nodes, respectively. The servers are connected by the aggregation switches, which are linked by the core switches. Using these expensive and high-speed switches, the servers are fully interconnected. There is a path between each pair of servers without passing through any other server. The traditional tree structure is simple and easy to build, but it does not scale well. Expanding such a structure needs to add more inner nodes and root nodes, using more expensive and higher-speed switches. Actually, these aggregation switches

and core switches easily lead to bottlenecks. A core switch failure can break down hundreds or even thousands of servers.

Fat-tree [1] is an improved structure of the traditional tree structure. Every inner node in a Fat-tree has more than one father node. This improvement increases the number of links between the aggregation switches and core switches. The network connectivity is increased, making Fat-tree a relatively reliable structure. However, like the traditional tree structure, it still does not scale well.

Hierarchical structures constructed recursively are believed to be scalable and reliable. To construct a hierarchical structure, a high level structure utilizes a lower level structure as a unit and connects many such units by means of a given recursive rule. As the level of a structure increases, more and more servers can be added into a hierarchical DCN without destroying the existing structure.

DCell [9], FiConn [15], and BCube [11] are three typical hierarchical structures. They use the same smallest recursive unit, in which a switch interconnects several servers. However, they are different in their recursive rules. DCell employs a complete graph as its recursive rule. There is a link between any two units of the same level. As a result, DCell possesses the advantage of the complete graph. In order to obtain high connectivity, each server in DCell should be equipped with multiple network ports. Although FiConn and DCell employ similar design principles for constructing high level compound graphs recursively, they have fundamental differences. Each server in FiConn is equipped with only two network ports, and the recursive units in FiConn are connected with only a half of the idle network ports in each unit. BCube employs the generalized hypercube as its recursive rule. The neighboring servers, which are connected to the same switch, differ in only one digit in their address arrays. Thus, BCube holds the advantages of the generalized hypercube, such as high connectivity and reliability.

2.3 Motivation

The above hierarchical structures efficiently solve the problems of scalability and reliability in different ways by using recursively defined topologies. Nevertheless, they ignore the particular approaches for performing MapReduce on the data centers with these structures. In a hierarchical structure, servers are not fully interconnected, and each server only connects with several adjacent servers. To support MapReduce, the hierarchical structure cannot utilize the approaches proposed by Google for performing MapReduce on the traditional tree structure [6].

First, hierarchical structures do not treat servers as masters and workers. In the procedure of Google's MapReduce, the servers controlling the execution of MapReduce procedures are called masters. The servers executing Map and Reduce tasks are called workers. In the traditional tree structure, servers are partitioned into masters and workers for MapReduce. A master can directly communicate with its workers without intermediate server. In a hierarchical structure, however, the communication among servers is multi-hop. Assume the servers are partitioned into masters and workers, there will be lots of control information transmitted through the servers shared by different paths. It is difficult to designate servers as masters or workers in hierarchical data centers.

Second, hierarchical structures can hardly support the data maintenance mechanism used by the traditional tree structure. In the traditional tree structure, each data file is divided into 64 megabytes blocks, and each block has several copies which are stored on different workers to keep data locality [3]. When a worker updates its data files, it sends related maintaining information to a distributed file system [12], which runs on some particular servers. Since the number of servers in a data center can be up to several thousands or even more, if the same method is used on a hierarchical data center whose servers are not fully connected, the amount of maintaining information transmitted on the data center will generate huge traffic load.

Third, the approaches for transmitting intermediate data on the traditional tree structure may be inefficient on hierarchical structures. In a traditional tree structure, after accomplishing Map tasks, Mappers store all intermediate data on local disks and send corresponding messages to the master. Then the mas-

ter forwards the information about intermediate data to Reducers. After that, Reducers send massages to Mappers asking for intermediate data. Finally, Mappers distribute all intermediate data to Reducers concurrently. In a hierarchical data center with non-fully connected network, this process becomes much more complex. Since Mappers are not directly connected with Reducers, the intermediate data may be transmitted through several servers shared by different paths. Sometimes a Mapper can generate megabytes of intermediate data. When Mappers are executing Map tasks, communication channels might be idle. However, when the Map tasks are accomplished, some Mappers may have to wait to deliver intermediate data. Therefore, delivering all intermediate data concurrently through multi-hop reduces network resource utilization, takes too much bandwidth and may result in network congestion.

To perform MapReduce on hierarchical data centers, we would like to propose a methodology for addressing all the above problems with high fault-tolerance. Details about the proposed methodology are introduced in the rest of the paper.

3 MapReduce on Hierarchical Data Centers

In this section, we study the methodology for performing MapReduce on hierarchical data centers. Our methodology is mainly based on the distributed hash table (DHT), and can efficiently solve the above problems in terms of storing data files, assigning Map and Reduce tasks, and delivering intermediate data.

3.1 Roles of Servers

In a data center, servers control and execute MapReduce procedures. In our research, each server in a hierarchical data center can work as a master or a worker.

If a server receives a MapReduce request, it will be regarded as a master for the current MapReduce procedure. Different from a traditional data center, this master is only responsible for assigning Map and Reduce tasks to workers. It is not responsible for controlling the transmission of intermediate data.

If a server receives a Map or Reduce task, it will be regarded as a worker for the current MapReduce procedure. Moreover, the worker receiving a Map task is regarded as a Mapper, and the worker receiving a Reduce task is regarded as a Reducer. A worker executes received tasks according to the rule of FCFS (first come, first served). After accomplishing Map tasks, Mappers directly send derived intermediate data to Reducers without masters' control.

3.2 Scheme for Storing Data Files

We would like to design a scheme for storing data files, so that the traffic load due to transmission of the maintaining information can be reduced and the Map tasks can be easily assigned. According to the rule of DHT [20] [17], the scheme for storing data files on hierarchical data centers can be summarized as three steps. The first step is to define a suitable file key space for all servers, and assign a set of sequential file keys to each server. A file key refers to a fixed-length number or string used for denoting a data file block. The number of file keys assigned to a server depends on the disk capacity of the server. A server with more disk capacity can get more file keys. The second step is to build a file key table on each server, which records the range of file keys of every adjacent server. The third step is to define a suitable function to hash the name of each data file block to a file key, and store the data file block on a server which holds that file key.

Each server is responsible for maintaining its data file blocks, file keys and the file key table. If a server updates its data file blocks, there is no need to inform other servers. If a server updates its file keys, it sends maintaining information only to its adjacent servers to update their file key tables. Each server has only a finite number of adjacent servers. Consequently, storing data files according to the

above scheme can reduce the amount of maintaining information transmitted on the whole data center network.

3.3 Scheme for Assigning Map and Reduce Tasks

In a traditional tree structure, a master directly connects with all workers. Therefore, a master can easily send a Map or Reduce task to a worker. In a hierarchical structure, however, a master usually sends a Map or Reduce task to a worker through a number of other servers.

(1) Assigning Map Tasks.

Based on the scheme for storing data files and the rule of DHT [20], we propose the scheme for assigning Map tasks on hierarchical data centers as follows. When a server receives a MapReduce request, it first determines the input data file block processed by each Map task. This server, namely the master, then hashes the name of the input data file block to a file key for the Map task. After that, the master chooses a server from all its adjacent servers to send the Map task, and the selected server has the range of file keys *closest* to the derived file key. This server will further choose another server from its neighbors to forward the Map task according to the same rule. This process is iteratively performed until a server, which receives the Map task, holds the corresponding file key. That implies the server stores the input data for the Map task. Here the *closest* is measured by a function, which is specified according to the definition of file keys.

(2) Assigning Reduce Tasks.

We propose the scheme for assigning Reduce tasks on hierarchical data centers as following steps. Similar to the scheme for storing data files, the first step is to define a suitable Reduce key space for all servers, and assign a set of sequential Reduce keys to each server. A Reduce key refers to a fixed-length number or string. The number of Reduce keys assigned to a server depends on the computing ability of the server. A server with faster computing capacity can get more Reduce keys. The second step is to build a Reduce key table on each server, which records the range of Reduce keys of every adjacent server. The third step is to define a suitable function used for hashing the keys of intermediate data to Reduce keys. Finally, when assigning Reduce tasks, the master hashes the key of intermediate data processed by each Reduce task to a Reduce key. The master chooses a server from all adjacent servers to send the Reduce task, and the selected server has the range of Reduce keys *closest* to the derived Reduce key. This process is iteratively performed until a server, which receives the Reduce task, holds the corresponding Reduce key.

Since the master can send Map and Reduce tasks immediately without searching data files node by node, in a hierarchical data center, assigning Map and Reduce tasks through DHT can reduce the execution time of MapReduce procedures. More than that, since there is no particular path from the master to a worker, the transmission of the Map or Reduce task cannot be interrupted by node failures. The communication becomes more reliable.

3.4 Scheme for Delivering Intermediate Data

Based on the scheme for assigning Reduce tasks, it is easy to delivering intermediate data. Our scheme for delivering intermediate data on hierarchical data centers are as follows. When a Mapper is executing a Map task, it hashes the key of a derived intermediate key/value pair to a Reduce key in the same way as that of assigning Reduce tasks. Then, according to its Reduce key table, it directly sends the intermediate key/value pair to an adjacent server, which has the range of Reduce keys *closest* to the Reduce key hashed from the intermediate key among all adjacent servers. In a similar way to the scheme for assigning Map and Reduce tasks, this intermediate key/value pair will be delivered node by node to a server which holds the corresponding Reduce key. According to the scheme for assigning Reduce tasks, this server has received the Reduce task that is responsible for processing the intermediate data.

Delivering intermediate data with above scheme can avoid the unnecessary controlling process performed by masters, which is complicated and can delay executing Reduce tasks in a non-fully connected data center network. Since each intermediate key/value pair is delivered immediately after being generated, this scheme can increase network resource utilization, and facilitate intermediate data transmission. Like the scheme for assigning Map tasks, delivering intermediate data in this way can avoid the impact of node failures, and therefore can increase the reliability of the transmission.

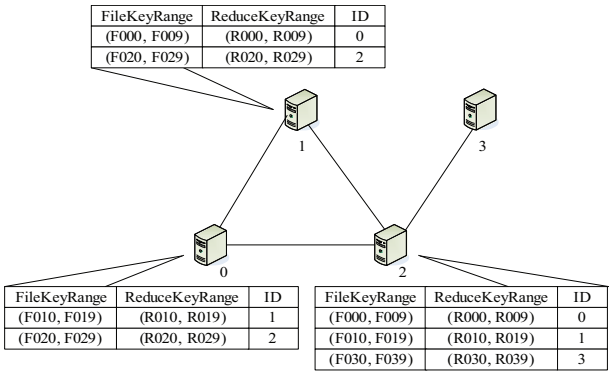


Figure 2: An example of our DHT

4 DHT Architecture and Routing Scheme

This section introduces the specific DHT architecture and corresponding routing scheme for executing MapReduce on hierarchical data centers.

Since the servers in a hierarchical data center are homogeneous and work in the same manner, we assign the file or Reduce keys to the servers in the order of their identifiers. The keys held by different servers are logically arranged in a line or a circle, like Chord [20]. This dose not means that the longest path to retrieve a key is the whole line. Adjacent servers in a hierarchical data center can belong to different recursive units and do not have sequential identifiers. The range of keys held by adjacent servers may not be sequential. Only the adjacent servers in the same smallest recursive unit hold sequential identifiers and keys. Consequently, for a hierarchical data center, the retrieval efficiency depends on its physical structure instead of the logical form in which the keys are arranged.

For ease of maintenance, we integrate the aforementioned file key table and Reduce key table into one hash table. This table consists of three attributes, including the range of file keys, the range of Reduce keys and the identifier of the adjacent server that holds those file keys and Reduce keys. In a hierarchical data center, each server stores such a table for recording the range of file keys and Reduce keys of all adjacent servers. In our work, $Item_i$=($FileKeyRange$, $ReduceKeyRange$, ID) denotes a record of the hash table, where *FileKeyRange, ReduceKeyRange*, and ID represent the three attributes, respectively. When a server is added to or removed from the data center, only its adjacent servers need to renew the records of their hash tables, so all other severs will not be affected. Fig.2 shows an example of our DHT architecture for executing MapReduce on hierarchical data centers.

Based on the hash table stored on each server, it is easy to implement the routing scheme for assigning Map and Reduce tasks and sending intermediate data. When a server receives a Map or Reduce task or an intermediate key/value pair, it first determines whether the corresponding file key or Reduce key is in its charge. If the server holds that file key or Reduce key, it performs the corresponding task according to reference [6]. Otherwise, it forwards the Map or Reduce task or intermediate data to an adjacent server in the way indicated by Algorithm 1.

In Algorithm 1, $TransObject$ denotes a Map or Reduce task or an intermediate key/value pair, which

needs to be delivered by any server in a hierarchical data center. $TransObject.key$ denotes the file key or Reduce key derived by corresponding hash functions. Function $d(,)$ is used for calculating the distance between $TransObject.key$ and the range of keys held by an adjacent server. Algorithm 1 sets a variable, denoted by $Server$, for recording the record item of the hash table on current server, which represents the next hop to send $TransObject$. It initializes $Server$ with the first record item, and then determines the type of $TransObject$. After that, Algorithm 1 iterates over the hash table to find a record whose range of file or Reduce keys is the closest to $TransObject.key$ according to function $d(,)$, and assigns the obtained record item to $Server$. Finally, it sends $TransObject$ to the adjacent server whose identifier is denoted by $Server.ID$. In this way, $TransObject$ will be delivered node by node to the server responsible for executing corresponding Map or Reduce task.

Algorithm 1 Deliver1 (object $TransObject$)

```
1:  object Server = Item_0;
2:  if TransObject is a Map task then
3:    for i = 1; i < I − 1; i++ do
4:      if d(TransObject.key, Item_i.FileKeyRange)
        < d(TransObject.key, Server.FileKeyRange) then
5:        Server = Item_i;
6:      end if
7:    end for
8:  else
9:    for i = 1; i < I − 1; i++ do
10:     if d(TransObject.key, Item_i.ReduceKeyRange)
        < d(TransObject.key, Server.ReduceKeyRange) then
11:       Server = Item_i;
12:     end if
13:   end for
14: end if
15: send TransObject to Server.ID;
```

5 Fault Tolerance

A hierarchical data center consists of much more servers, switches and links than a traditional tree structure data center, so it has more tendency to machine or link failures. A link failure leads to the disconnection of the two machines interconnected through the link, and the two machines can be regarded as failed to each other. Hence we only focus on server and switch failures in this paper.

5.1 Fault-tolerant Routing Against Failures of Servers and Switches

A MapReduce procedure cannot utilize a failed server or switch to assign tasks or transmitting intermediate data. To address this problem, we propose the fault-tolerant routing for MapReduce in hierarchical data centers.

In a traditional tree structure data center, a switch failure can break down all the servers connecting to it. Since a hierarchical data center employs the redundant structure, a switch failure may not affect the servers connecting to it. However, these servers cannot communicate to each other directly. In this work, we treat a switch failure as several disconnected servers.

To ensure that our routing scheme can forward all tasks and intermediate data to available servers, we employ the following approach. Each server sends the number of tasks in its service queue as its

state information to all adjacent servers periodically, and therefore each server knows the running state of all its adjacent servers. If a server cannot update its state information to its adjacent servers, it will be regarded as a failed server and its corresponding records in the hash tables of adjacent servers will be denoted as unavailable. Based on this failure notification mechanism among adjacent servers, we modify Algorithm 1 in order to achieve Algorithm 2 as the fault-tolerant routing scheme. Algorithm 2 assigns the first available record item to $Server$, and iterates over the hash table from that record to update $Server$ with an available record whose range of file or Reduce keys is closer to $TransObject.key$. In such a way, $Server.ID$ finally gives the identifier of the available next hop to deliver $TransObject$.

Algorithm 2 Deliver2 (object $TransObject$)

```
1: object Server = Null; int j = 0;
2: for i = 0; i < I − 1; i + + do
3:    if Item_i.available == true then
4:       Server = Item_i;
5:       j = i;
6:       break;
7:    end if
8: end for
9: if TransObject is a Map task then
10:   for i = j; i < I − 1; i + + do
11:      if Item_i.available == true then
12:         if d(TransObject.key, Item_i.FileKeyRange)
            < d(TransObject.key, Server.FileKeyRange) then
13:            Server = Item_i;
14:         end if
15:      end if
16:   end for
17: else
18:   for i = j; i < I − 1; i + + do
19:      if Item_i.available == true then
20:         if d(TransObject.key, Item_i.ReduceKeyRange)
            < d(TransObject.key, Server.ReduceKeyRange) then
21:            Server = Item_i;
22:         end if
23:      end if
24:   end for
25: end if
26: send TransObject to Server.ID;
```

5.2 Fault-tolerant Approaches to Address Failures of Masters and Workers

In a hierarchical data center, a running MapReduce procedure can be interrupted by a server failure, no matter a master failure or a worker failure. To address this problem, we propose the following approaches.

1) Addressing the failure of a master server:

- As soon as a master receives a MapReduce request, it sends the request to an adjacent server as a replica. When assigning Map and Reduce tasks, the master concurrently sends a confirmation

message for each task to that adjacent server. If that server cannot receive any confirmation message of a task within a threshold time, the master will be regarded as failed, and that server will take over the current MapReduce request and reassign the corresponding Map or Reduce task.

2) Addressing the failure of a worker server:

- If a worker server receives a Map task and its service queue has achieved a predefined threshold length, it will discard the Map task and send a message to the Master for reassigning the Map task to another server which stores a replica of the corresponding input data file.
- According to the aforementioned failure notification mechanism, each server keeps the running state of all adjacent servers. If a worker server receives a Reduce task and its service queue has achieved a predefined threshold length, it will forward the Reduce task to an available adjacent server. Moreover, it will forward all the intermediate data to that adjacent server.
- When a worker server accepts a Map or Reduce task, it sends corresponding information of the task to an adjacent server. As soon as the worker server accomplishes that task, it sends a confirmation massage of the task to that adjacent server. If that adjacent server cannot receive the confirmation massage within a threshold time, it will send the task information to the master for reassigning the task to another available server. In this case, the worker server will be regarded as a straggler [16].

6 Evaluation

Since BCube is a representative hierarchical structure [11], in this section we conduct a comprehensive evaluation based on BCube, to demonstrate that our method is feasible for executing MapReduce on hierarchical data centers.

In the following evaluation, we employ Equation 1 as the hash function for transferring a task name or an intermediate key into a file key or a Reduce key.

$$TransObject.key = f(TransObject) \text{ MOD } K \tag{1}$$

Function $f(TransObject)$ calculates the decimal ASCII number of $TransObject$. For example, suppose that $TransObject$ denotes a character string "abc", then $f(TransObject)$ equals 979899. K is a prime number less than the total number of file keys or Reduce keys. Here the value of $TransObject.key$ is an integer, hence the aforementioned function $d(,)$ can be defined for calculating the absolute value of the difference between $TransObject.key$ and the range of keys held by a server.

6.1 Load Balance

According to aforementioned schemes, Map tasks are assigned to the servers that hold corresponding data files, so the distribution of input data file blocks determines the load balance of Map tasks. Since researchers have studied how to allocate date to servers evenly with consistent hashing [14] [23], we assume that all input data file blocks are well distributed in a hierarchical data center. Therefore, we can also assume that Map tasks can be evenly assigned to different servers. Here we mainly study the load balance of Reduce tasks, which is much more uncertain than that of Map tasks.

We perform the simulation for evaluating load balance as follows. We simulate the structure of BCube and corresponding communication among its servers. Let N denote the number of servers in a level 0 BCube, and H denote the number of levels. The number of servers in BCube varies from 4 to 625 when N varies from 2 to 5 and H varies from 1 to 3. We assign a unique identifier to each server, and calculate the identifiers of its adjacent servers. In BCube, two servers that connect to the same switch can be regarded as adjacent servers. In reality, different MapReduce applications generate different kinds

of Reduce tasks, whose arrival to a data center is a stochastic process. For ease of evaluation, in our simulation, we assume that there is only one Reduce task arriving to the BCube data center within a short period of time. The execution time of a Reduce task is random and is longer than the arrival period. We consider two cases about the execution time. The first case is that the execution time of a Reduce task varies randomly from 10 to 100 times the length of arrival period. The second case is that the execution time of a Reduce task varies randomly from 100 to 500 times the length of arrival period. According to Hadoop [24], we define that each server can execute two Reduce tasks simultaneously. When a server is busy in executing two Reduce tasks, it will reject the newly arrived Reduce task and foreword it to an adjacent server. If a Reduce task is rejected for three times, we regard that this Reduce task is dropped. We consider a workload with 2×10^3 Reduce tasks, which are assigned according to the schemes studied in Section 3.

Since a server can only execute limited number of Reduce tasks simultaneously, some Reduce tasks may get dropped in a data center with skewed load. We repeat the simulation 30 times for each number of servers to calculate the mean percentage of dropped Reduce tasks. Fig.3 plots the results in the two cases about the execution time. As shown in Fig.3, the percentage of dropped Reduce tasks decreases rapidly with the increase of the number of servers. In the first case, the percentage of dropped Reduce tasks decreases to 0 when the number of servers achieves 64. In the second case, the percentage of dropped Reduce tasks decreases to 0 when the number of servers achieves 256. When we further increase the number of arrival Reduce tasks to 10^4, the results of the two cases remain the same. This implies that running MapReduce on such a data center according to our approaches can hardly drop any tasks.

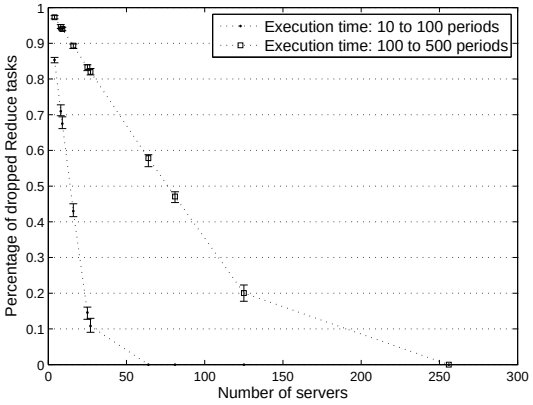


Figure 3: Variation of the percentage of dropped Reduce tasks along with the number of servers

To evaluate the workload of each server, based on the above simulation, we calculate the Reduce tasks executed by each server in the two cases. Fig.4 plots the variation of the mean percentage of Reduce tasks that per server executes along with the number of servers, which varies from 4 to 625. As shown in Fig.4, when the number of servers is small, the mean workload of a server in the first case is higher than that in the second case, as the execution time of the first case is much lower than that of the second case. In the first case, the mean workload of a server keeps on a relatively high level until the number of servers achieves 27. In the second case, the mean workload of a server dose not decrease until the number of servers achieves 125. The reason of these variations is, when the number of servers is not enough to sustain continually arrival Reduce tasks, all servers work at full capacity and many tasks are dropped. While the number of servers achieves 256, the mean workload of a server decreases to the same low level in both cases. Fig.5 shows the percentage of Reduce tasks executed by each server when there are 256 servers in all. We can derive from Fig.5 that the difference between the workload of any two servers is less than 1.6×10^{-3}, a vary small value. Therefore, the workload can be evenly distributed.

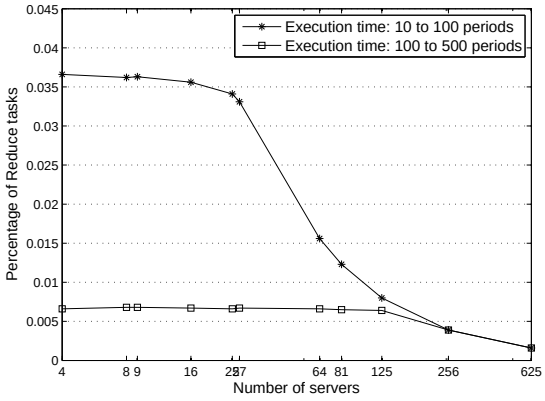


Figure 4: Variation of the mean percentage of Reduce tasks that per server executes along with the number of servers

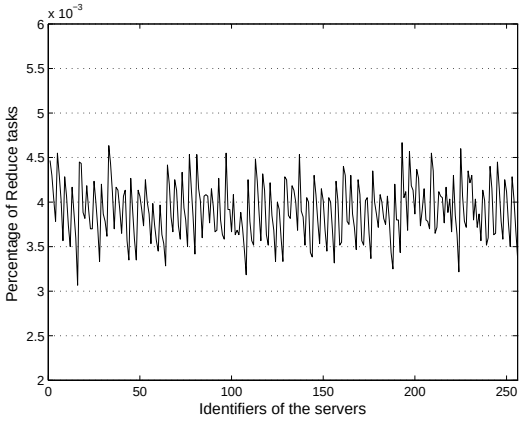


Figure 5: Percentage of Reduce tasks executed by each server when there are 256 servers

6.2 Data Forwarding Performance

A data center network mainly consists of servers, switches and links. In a hierarchical data center, servers are not only used for executing Map and Reduce tasks, but also used for forwarding intermediate data. However, since servers are not network devices, they have lower data forwarding capacity than switches and links. In this work, we assume that switches and links can provide sufficient bandwidth and only consider the servers.

Based on the former simulation, we perform the simulation for evaluating data forwarding performance, including data forwarding throughput and bandwidth between servers, as follows. We assume that our routing scheme held by each server supports receiving and sending only one data packet, namely a key/value pair, within an extremely short period of time. According to literature [11], in a BCube data center, a server forwards at 750Mb/s in all-to-all traffic pattern. MTU is usually set to be 1.5KB for a commodity computer. Thus, on this condition, we can easily calculate that the short period is 1.6×10^{-5} seconds. In such a period, a server with a packet to transmit chooses an adjacent server as the destination server, according to our routing scheme. If this destination server is available for receiving a packet,

the former server will forward that packet successfully. Then any other packets to the same destination server should wait to be transmitted until the server is available in another period. We vary the number of servers that simultaneously generate packets to calculate the best possible data forwarding performance. This procedure is recursively performed in our simulation so as to evaluate data forwarding performance on a steady working condition.

When the number of servers varies from 4 to 625, we repeat the simulation 30 times for each number of servers to obtain the mean and range of the maximum data forwarding throughput, as shown in Fig.6. We find that the maximum data forwarding throughput does not closely track the increase of the total number of servers. The throughput of 27 servers is lower than that of 25 servers, and the throughput of 81 servers is lower than that of 64 servers. The reason for that is, throughput depends not only on the number of servers, but also on the number of hops for data forwarding. If the data are forwarded through more hops, the throughput will be lower. For a hierarchical data center, the maximum number of hops is determined by the number of levels, and more levels bring more hops. In our simulation, the BCube data centers with 27 servers and 81 servers have one level more than the BCube data centers with 25 servers and 64 servers, respectively. Given a fixed number of servers in a recursively hierarchical data center, it is crucial to make a tradeoff between the number of levels and the number of servers in the smallest recursive unit to achieve desired performance. As shown in Fig.6, when the number of servers is larger than 125, data forwarding throughput increases rapidly. Therefore, our approaches can well support throughput-hungry MapReduce applications on hierarchical data centers.

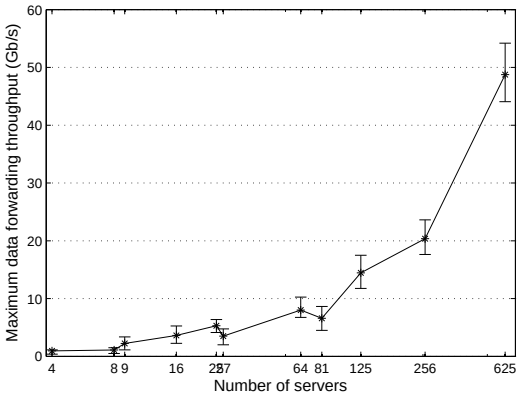


Figure 6: Variation of the maximum data forwarding throughput along with the number of servers

To evaluate the bandwidth between servers, we first keep the number of servers that simultaneously generate packets in a certain value, which ensures that data forwarding throughput remains at the maximum value. On this condition, we then calculate the mean of the maximum bandwidth that each server can achieve for sending data to another server through maximum number of hops. Fig.7 illustrates the result when there are 256 servers. We can derive that most values of the bandwidth are larger than 0.3Gb/s and less than 0.57Gb/s. This variance, which is less than 0.3Gb/s, is acceptable for MapReduce applications. In practice, different servers store different types of data, and internet service providers may store popular data on certain servers to save power [19]. Hence some of the Map or Reduce tasks in a MapReduce procedure process and generate more data than other tasks. Overall, the result of our simulation implies that the bandwidth can be evenly distributed according to our approaches.

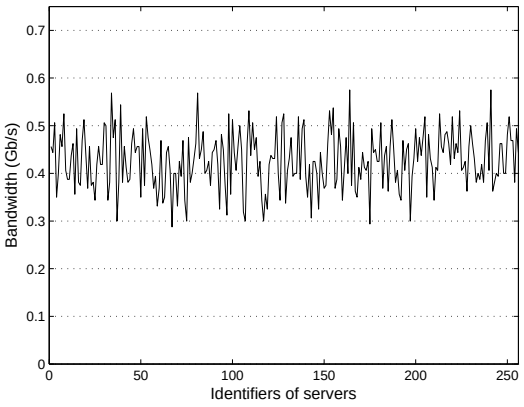


Figure 7: Mean of the maximum bandwidth that each server can achieve for sending data to another server through maximum number of hops, when there are 256 servers and data forwarding throughput remains at maximum value

6.3 Fault Tolerance

We evaluate the load balance and data forwarding performance under given failure rates of links and nodes to further validate the fault-tolerant capability of our methodology.

For a hierarchical data center, let P_1 and P_2 denote the expected probabilities that a server or a switch fails when they are executing a MapReduce procedure, respectively. Here we omit link failures which can be regarded as adjacent nodes failures. Then we can calculate the probability that a server keeps on working according to Theorem 1.

Theorem 1. *For a hierarchical data center, let P_3 denote the probability that a server keeps on working in a MapReduce procedure. Let I and J denote the number of servers and switches directly connecting with this server, respectively. P_3 is given by*

$$P_3 = (1 - P_1) \times (1 - \sum_{i=1}^{I}(\binom{I}{i} \times P_1^i)) \times (1 - \sum_{j=1}^{J}(\binom{J}{j} \times P_2^j)) \tag{2}$$

Proof: The probability that this server does not fail equals $1-P_1$. The probability that i of the I servers fail is $\binom{I}{i}\times P_1^i$. Then the probability that these I servers keep on working equals $1-\sum_{i=1}^{I}(\binom{I}{i}\times P_1^i)$. Similarly, the probability that the J switches keep on working equals $1-\sum_{j=1}^{J}(\binom{J}{j}\times P_2^j)$. This server can keep on working only if all the I servers and J switches can keep on working and itself does not fail. Therefore, the probability that this server keeps on working equals $(1-P_1)\times(1-\sum_{i=1}^{I}(\binom{I}{i}\times P_1^i))\times(1-\sum_{j=1}^{J}(\binom{J}{j}\times P_2^j))$. Theorem 1 is proved.

Based on Theorem 1, we modify the aforementioned simulations to evaluate the fault-tolerant load balance and data forwarding performance. In the corresponding period, a server with a Reduce task (or a packet) for transmitting chooses an available adjacent server in working order as the destination server, according to the fault-tolerant routing. If there is a third server having a Reduce task (or a packet) to the same destination server in the same period, it has to wait until the destination server is available in anther period.

In BCube, each server can only directly connect to switches, so P_3 equals $(1-P_1)\times(1-\sum_{j=1}^{J}(\binom{J}{j}\times P_2^j))$. We obtain different values of P_3 by varying P_1 and P_2. Then we calculate the variation of the mean percentage of Reduce tasks that per server executes along with the total number of servers, when

P_3 equals 0.98, 0.90 and 0.80, respectively. Fig.8 illustrates the results in the aforementioned two cases. In the first case, many tasks are dropped when the number of servers is small, so there is no significant difference among the workload of each server for the three values of P_3. In the second case, since the workload of each server keeps very low, the difference is not notable when the number of servers is less than 125. While the number of servers are enough to sustain the continually arrival Reduce tasks, our method has good fault tolerance, so the difference is also small in both cases. Fig.9 plots the percentage of Reduce tasks executed by each server when there are 256 servers and P_3=0.90. The mean percentage of Reduce tasks that per server executes is only 0.5×10^{-3} lower than that shown in Fig.5. Moreover, the variance is less than 1.5×10^{-3}. Thus, the workload can be evenly distributed under high failure rates of links and nodes.

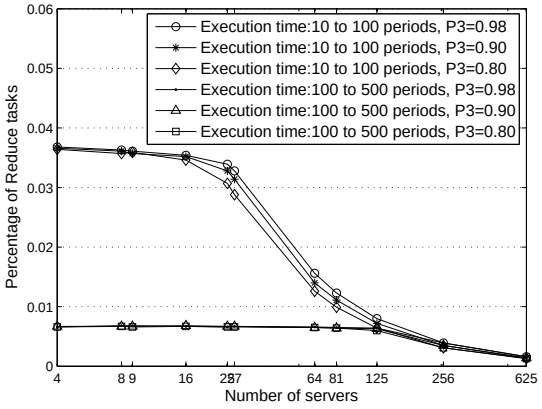


Figure 8: Variation of the mean percentage of Reduce tasks that per server executes along with the number of servers, when P_3=0.98, P_3=0.90 and P_3=0.80

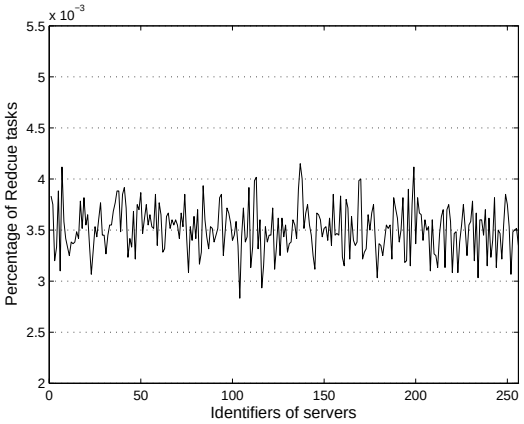


Figure 9: Percentage of Reduce tasks executed by each server, when there are 256 servers and P_3=0.90

Due to the failures of links and nodes, we calculate the maximum data forwarding throughput when P_3 equals 0.98, 0.90 and 0.80, respectively. For each value of P_3, we repeat the modified simulation 30 times to obtain the mean value. Fig.10 illustrates the result. When the number of servers is small, the throughput of the network is low, so there is no obvious difference among the throughput for the three values of P_3. When the number of servers is larger than 125, the throughput increases rapidly for all the three values of P_3. Thus, with enough servers, our method can provide satisfactory data forwarding throughput against failures of links and nodes. Considering link and node failures, we

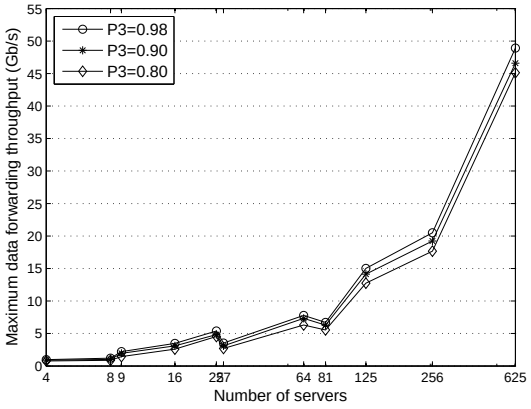


Figure 10: Variation of the maximum data forwarding throughput along with the number of servers, when P_3=0.98, P_3=0.90 and P_3=0.80

recalculate the mean of the maximum bandwidth that each server can achieve for sending data to another server through maximum number of hops. Fig.11 illustrates the result when there are 256 servers and P_3=0.90. Although the probability that a server cannot keep on working is 0.10, which is really high in practice, the bandwidth for every server is only 0.05Gb/s lower than that shown in Fig.7. Hence the bandwidth between servers is abundant against failures of links and nodes. Moreover, the range of variance, as shown in Fig.11, is less than 0.25Gb/s. Therefore, the bandwidth between servers can be evenly distributed under high failure rates of links and nodes.

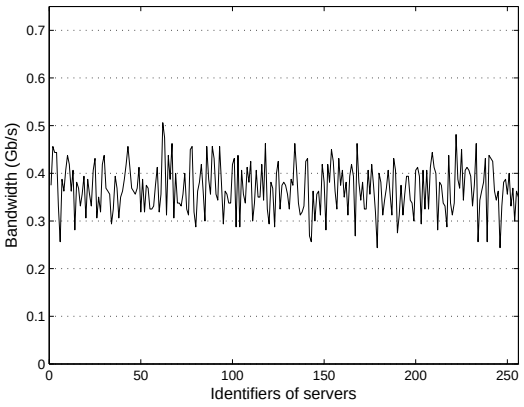


Figure 11: Mean of the maximum bandwidth that each server can achieve for sending data to another server through maximum number of hops, when there are 256 servers and P_3=0.90

7 Conclusion

Several hierarchical structures have been proposed to improve the topological properties of data centers. However, the communication and control mechanisms proposed by Google for performing MapReduce on the traditional structure can hardly be operated on these hierarchical structures. This paper presents a methodology for performing MapReduce on data centers with hierarchical structures. Comprehensive analysis and simulations show that our methodology can evenly distribute the workload

and well support throughput-hungry MapReduce applications. It is also proved that our methodology is competent for MapReduce even under node failures. The mismatch problem between hierarchical data centers and Mapreduce is effectively solved in this paper.

Acknowledgment

We would like to thank the anonymous reviewers for their constructive comments. Our work is supported in part by the NSF China under Grants No.60903206, No.60972166, No.61170284, No.71031007, No.71071160 and No.71171197, the China Postdoctoral Science Foundation under grant No.201104439, and the Preliminary Research Foundation of National University of Defense Technology under grant No.JC10-05-01.

Bibliography

[1] M. Al-Fares, A. Loukissas, and A. Vahdat. A Scalable, Commodity Data Center Network Architecture. *Proc. ACM SIGCOMM*, pp.63-74, Aug. 2008.

[2] D. Borthakur. The Hadoop Distributed File System: Architecture and Design. http://hadoop.apache.org/core/docs/current/hdfsdesign.pdf

[3] C. Bastoul and P. Feautrier. Improving Data Locality by Chunking. *Springer Lecture Notes in Computer Science*, vol.2622, pp.320-334, 2003.

[4] F. Chang, J. Dean, S. Ghemawat, W.C. Hsieh, D.A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R.E.Gruber. Bigtable: A Distributed Storage System for Structured Data. *Proc. 7th Symposium on Operating Systems Design and Implementation (OSDI)*, pp.205-218, Nov. 2006.

[5] J. Cohen. Graph Twiddling in a MapReduce world. *Computing in Science and Engineering, IEEE Educational Activities Department*, vol.2, no.4, pp.29-41, 2009.

[6] J. Dean and S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. *Proc. 6th Symposium on Operating System Design and Implementation (OSDI)*, pp.137-150, Dec. 2004.

[7] J. Dean, and S. Ghemawat. MapReduce: A Flexible Data Processing Tool. *Communications of the ACM*, vol.53, no.1, pp.72-77, 2010.

[8] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel. The Cost of a Cloud: Research Problems in Data Center Networks. *ACM SIGCOMM computer communication review*, vol.39, no.1, pp.68-73, Jan. 2009.

[9] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, and S. Lu. DCell: A Scalable and Fault-Tolerant Network Structure for Data Centers. *Proc. ACM SIGCOMM*, pp.75-86, Aug. 2008.

[10] A. Greenberg, J.R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D.A. Maltz, P. Patel, and S. Sengupta. VL2: A Scalable and Flexible Data Center Network. *ACM SIGCOMM Computer Communication Review*, vol.39, no.4, pp.51-62, Aug. 2009.

[11] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, and S. Lu. BCube: A High Performance, Server-centric Network Architecture for Modular Data Centers. *Proc. ACM SIGCOMM*, pp.63-74, Aug. 2009.

[12] S. Ghemawat, H. Gobioff, and S.T. Leung. The Google File System. *Proc. 19th ACM Symposium on Operating Systems Principles*, pp.29-43, Dec. 2003.

[13] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly. Dryad: Distributed Data-parallel programs from Sequential Building Blocks. *Proc. 2nd ACM SIGOPS/EuroSys European Conference on Computer Systems*, pp.59-72, Jun. 2007.

[14] W. Jun. A Methodology for the Deployment of Consistent Hashing *Proc. 2nd IEEE International Conference on Future Networks*, Jan. 2010.

[15] D. Li, C. Guo, H. Wu, K. Tan, Y. Zhang, and S. Lu. FiConn: Using Backup Port for Server Interconnection in Data Centers. *Proc. IEEE INFOCOM*, pp.2276-2285, Apr. 2009.

[16] J. Lin. The Curse of Zipf and Limits to Parallelization: A Look at the Stragglers Problem in MapReduce *Workshop on Large-Scale Distributed Systems for Information Retrieval*, Jul. 2009.

[17] J. Pang, P.B. Gibbons, M. Kaminsky, S. Seshan, and H. Yu. Defragmenting DHT-based Distributed File Systems *Proc. 27th IEEE International Conference on Distributed Computing Systems*, Jun. 2007.

[18] T. Redkar. Introducing Cloud Services. *Windows Azure Platform, Apress*, pp.1-51, 2009.

[19] L. Rao, X. Liu, L. Xie, and W. Liu. Minimizing Electricity Cost: Optimization of Distributed Internet Data Centers in a Multi-Electricity-Market Environment *Proc. IEEE INFOCOM*, Mar. 2010.

[20] I. Stoica, R. Morris, D. Karger, M.F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peertopeer Lookup Service for Internet Applications *Proc. ACM SIGCOMM*, pp.1-12, Aug. 2001.

[21] D. Talia and P. Trunfio. Enabling Dynamic Querying over Distributed Hash Tables. *Elsevier Journal of Parallel and Distributed Computing*, vol.70, no.12, pp.1254-1265, 2010.

[22] G. Urdaneta, G. Pierre and M.V. Steen. A Survey of DHT Security Techniques. *Journal of ACM Computing Surveys*, vol.43, no.2, pp.1-49, 2011.

[23] X. Wang and D. Loguinov. Load-balancing performance of consistent hashing: asymptotic analysis of random node join *IEEE/ACM Transactions on Networking*, vol.15, no.4, pp.892-905, 2007.

[24] http://hadoop.apache.org.

INT J COMPUT COMMUN, ISSN 1841-9836
Vol.7 (2012), No. 3 (September), pp. 450-458

Data Consistency in Emergency Management

D. Ergu, G. Kou, Y. Peng, F. Li, Y. Shi

Daji Ergu
College of Electrical and Information Engineering
Southwest University for Nationalities
Chengdu, China, 610041
E-mail: ergudaji@163.com

Gang Kou, Yi Peng , Feixiong Li
School of Management and Economics
University of Electronic Science and Technology of China
Chengdu, China, 610054
E-mail: kougang@yahoo.com; pengyicd@gmail.com;
lifx@uestc.edu.cn

Yong Shi
1. Research Center on Fictitious Economy and Data Sciences
Chinese Academy of Sciences, Beijing 100190, China, and
2. College of Information Science & Technology
University of Nebraska at Omaha, Omaha, NE 68182, USA
E-mail: yshi@gucas.ac.cn

Abstract:
Timely response is extremely important in emergency management. However, cardinal inconsistent data may exist in a judgment matrix because of the limited expertise, preference conflict as well as the complexity nature of the decision problems. The existing inconsistent data processing models for positive reciprocal matrix either are complicated or dependent on the priority weights, which will delay the decision making process in emergency. In this paper, a geometric mean induced bias matrix (GMIBM), which is only based on the original matrix A, is proposed to quickly identify the most inconsistent data in the judgment matrix. The correctness and effectiveness of the proposed model are proved mathematically and illustrated by two numerical examples. The results show that the proposed model not only preserves most of the original information in matrix A, but also is faster than existing methods.
Keywords: cardinal inconsistency, positive reciprocal matrix, geometric mean induced bias matrix (GMIBM), inconsistency identification

1 Introduction

Emergency management is an interdisciplinary field, and is in essence a complex multi-objective optimization problem [1]. Multi-criteria decision making (MCDM) methods have therefore been extensively employed to study emergency management, for instance, the Ordered Weighted Averaging (OWA) [2], Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) [3], Preference Ranking Organization Method for Enrichment Evaluations (PROMETHEE) [4], Analytic Hierarchy Process (AHP) ([5], [6],and [7]), Analytic Network Process (ANP) ([8], [9]), Decision Making Trial and Evaluation Laboratory (DEMATEL) [10], Fusion Approach of MCDM (FAMCDM) methods [11] etc. Among these MCDM methods, AHP and ANP are two of the most popular methods for studying emergency management, and usually used to assess the emergency management performance, select the best emergency response alternatives or emergency recovery alternatives and allocate reasonable relief resources etc.

In the AHP/ANP, the tangible and intangible attributes or criteria are always measured by numerical data through pairwise comparisons, and displayed in a judgment matrix whose numerical data are positive and reciprocal. The data of judgment matrix are usually provided by experts and collected through questionnaire survey [12]. Therefore, the data may be inconsistent because of the limited expertise, preference conflict as well as the complexity nature of the decision problems etc ([13], [14]). For instance, assume there are three emergency response alternatives, A, B, and C, the i^{th} expert thinks that alternative A is preferred to B 2 times, and B is preferred to C 4 times, but alternative A is preferred to C 3 times in stead of 8 times. This is called cardinal inconsistency, and for an inconsistent judgment, the inconsistent data should be identified and adjusted before it is used to make a valid decision. Therefore, the inconsistent data processing issue in a judgment matrix has become a hot topic since the introduction of the AHP/ANP. Currently, there are many methods for identifying and adjusting the inconsistent data, for example, auto-adaptive algorithms in ([14], [16]), absolute differences methods in ([17], [18]), perturbation matrix method in [19] etc. However, the existing methods are either too complicated to delay the speed of inconsistent data analysis or are difficult to preserve most of the original information in the judgment matrix, or are extremely dependent on the priority weights. Therefore, it is necessary to propose a cardinal inconsistent data analysis model to effectively and simply identify the inconsistent data in order to make a valid decision, which is independent to the priority weights while preserving most of the original information in a judgment matrix. In an attempt to establish such models, the absolute differences of geometric mean matrix in [20], the induced bias matrix model (IBMM) in [21], were proposed to identify the possible inconsistent data in a judgment matrix.

In this paper, a geometric mean induced bias matrix (GMIBM), which is only based on the original matrix A, is established to identify the most inconsistent data in a judgment matrix. Through observing and adjusting the largest bias data in the induced bias matrix, the consistency ratio of the judgment matrix can be quickly improved to make a fast decision for emergence management. Besides, a general estimating formula of the mined cardinal inconsistent data of GMIBM is provided.

The remaining parts of this paper are organized as follows. The next section briefly describes the cardinal inconsistency in a judgment matrix. The theorems, corollaries and identifying processes of GMIBM are further proposed and presented in Section 3. Two numerical examples introduced in [21] are used to test the proposed model in Section 4. Section 5 concludes the paper.

2 Cardinal Inconsistency

Let the judgment matrix be $A = [a_{ij}]_{n\times n}$, where $a_{ij}>0$ and $a_{ij} = 1/a_{ji}$ for all i, j,and k, if $a_{ij} = a_{ik}a_{kj}$ holds for all i, $j$and$k$, then matrix A is said to be perfectly cardinal consistency, otherwise, it is called cardinal inconsistency. In practice, it is unrealistic to obtain a perfectly cardinal consistency of matrix A, therefore AHP allows a certain level of cardinal inconsistency of the judgment matrix. To measure the consistency of a judgment matrix and determine a certain acceptable level of inconsistency, Saaty proposed a consistency index (CI), denoted as:

$$CI = \frac{\lambda_{\max} - n}{n - 1} \tag{1}$$

where λ_{max} is the maximum eigenvalue of matrix A, and n is the order of matrix A.

To define a unique consistency test index that does not rely on the order of judgment matrices, the consistency index (CI) was extended and the consistency ratio (CR) method was further proposed by Saaty [17],

$$CR = \frac{CI}{RI} \tag{2}$$

where CI is the consistency index shown in equation (1) while RI is the average random index based on Matrix Size, as shown in Table 1.

Table 1 The Average Random Index

n	1	2	3	4	5	6	7	8	9	10
RI	0	0	0.52	0.89	1.11	1.25	1.35	1.4	1.45	1.49

If the *CR* of a judgment matrix is less than or equal to 0.1 (*CR≤0.1*), indicating the inconsistency is relatively small, the judgment matrix is said to be of acceptable inconsistency. If *CR* is greater than 0.1 (*CR>0.1*), the judgment matrix is of unacceptable cardinal inconsistency, and the decision makers are asked to revise their judgments. To effectively identify the cardinal inconsistent data and improve the consistency ratio of a judgment matrix, a geometric mean induced bias matrix (GMIBM) by Hadarmad product is proposed in the following sections.

3 Geometric Mean Induced Bias Matrix (GMIBM)

3.1 Theorem of GMIBM

To identify the inconsistent data, a geometric mean induced bias matrix (for short GMIBM, hereinafter), which is only based on the original judgment matrix and independent to the priority weights, is established to amplify the most inconsistent data in this paper. Then, the most inconsistent data can be identified by observing the largest data in the induced bias matrix. The related theorems and corollaries are presented in this section.

Theorem 1. *The geometric mean induced bias matrix (GMIBM) C should be a U matrix if judgment matrix A is perfectly consistent, that is,*

$$C = \bar{A} \circ A^T = \left(c_{ij}\right) = (\sqrt[n]{\prod_{k=1}^{n} a_{ik}a_{kj}} \cdot a_{ji}) = U \quad if \quad a_{ik}a_{kj} = a_{ij} \tag{3}$$

where $\bar{A} = \left(\bar{a}_{ij}\right)_{n\times n} = \left(\sqrt[n]{\prod_{k=1}^{n} a_{ik}a_{kj}}\right)_{n\times n}$ *represents an n-by-n geometric mean matrix composed of all geometric mean of* $a_{ik}a_{kj}$ *for all i, j and k,* $U = \begin{bmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{bmatrix}$*, n denotes the order of A,* A^T *represents the transpose of matrix A. Symbol ' ∘ ' denotes Hadamard product (e.g.* $C = A \circ B$ *means* $c_{ij} = a_{ij}b_{ij}$ *for all i and j).*

Proof: If the judgment matrix satisfies the perfect consistency condition, namely, $a_{ik}a_{kj} = a_{ij}$ holds for all i, j and k. Since $a_{ij} = 1/a_{ji}$, we have

$$c_{ij} = \sqrt[n]{\prod_{k=1}^{n} a_{ik}a_{kj}} \cdot a_{ji} = \sqrt[n]{\prod_{k=1}^{n} a_{ij}} \cdot a_{ji} = \sqrt[n]{a_{ij}{}^{n}} \cdot a_{ji} = a_{ij}a_{ji} = 1 \tag{4}$$

Therefore, all entries in matrix C are ones if the matrix is perfectly consistent, and matrix C is a U matrix, whose entries are ones. □

To simply compute the GMIBM and easily understand the theorem of GMIBM, the *Theorem 1* is transformed to following theorem.

Theorem 2. *The geometric mean induced bias matrix (GMIBM) C should be a U matrix if judgment matrix A is perfectly consistent, that is,*

$$C = L \times R \circ A^T = \left(c_{ij}\right) = \left(\sqrt[n]{\prod_{k=1}^{n} a_{ik}} \cdot \sqrt[n]{\prod_{k=1}^{n} a_{kj}} \cdot a_{ji}\right) = U \quad if \quad a_{ik}a_{kj} = a_{ij} \tag{5}$$

where $L = \left(\sqrt[n]{\prod_{k=1}^{n} a_{ik}}\right)_{n\times 1}$ *represents an n-by-one column matrix composed of geometric mean of rows in matrix A, while* $R = \left(\sqrt[n]{\prod_{k=1}^{n} a_{kj}}\right)_{1\times n}$ *denotes an one-by-n row matrix composed of geometric mean of columns in matrix A.*

Corollary 3. *The geometric mean induced bias matrix (GMIBM) C should be as close as possible to a U matrix if judgment matrix A is approximately consistent.*

Corollary 4. *There must be some inconsistent data in the geometric mean induced bias matrix (GMIBM) C deviating far away from one if the judgment matrix is inconsistent.*

Based on *Corollary 4*, we can identify the most inconsistent data in matrix *A* by observing the largest data deviating from 1 in the geometric mean induced bias matrix (GMIBM) *C*. Details of inconsistency identification processes are presented below.

3.2 Inconsistency Identification and Adjustment Processes of GMIBM

To propose the inconsistency identification and adjustment processes of GMIBM based on above *Theorems* and *Corollaries*, the aforementioned *n-by-n* judgment matrix $A = [a_{ij}]_{n\times n}$ is used in the following. The processes of inconsistent data analysis and adjustment of GMIBM include two major steps, inconsistency identification and inconsistency adjustment:

Step I: Inconsistency Identification

Step 1: Compute a column matrix *L* and a row matrix *R*, which are composed of geometric means of rows and columns respectively.

$$L =$$

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1i} & \cdots & a_{1j} & \cdots & a_{1n} \\ & \vdots & & \vdots & & \vdots & \\ a_{i1} & \cdots & a_{ii} & \cdots & a_{ij} & \cdots & a_{in} \\ & \vdots & & & & \vdots & \\ a_{j1} & \cdots & a_{ji} & & a_{jj} & \vdots & a_{jn} \\ & & & & & & \\ a_{n1} & \cdots & a_{ni} & \cdots & a_{nj} & \cdots & a_{nn} \end{pmatrix} \begin{matrix} \sqrt[n]{\prod_{k=1}^{n} a_{1k}} \\ \sqrt[n]{\prod_{k=1}^{n} a_{ik}} \\ \sqrt[n]{\prod_{k=1}^{n} a_{jk}} \\ \sqrt[n]{\prod_{k=1}^{n} a_{nk}} \end{matrix} .$$

$$R = \sqrt[n]{\prod_{k=1}^{n} a_{k1}}, \sqrt[n]{\prod_{k=1}^{n} a_{ki}}, \sqrt[n]{\prod_{k=1}^{n} a_{kj}}, \sqrt[n]{\prod_{k=1}^{n} a_{kn}}$$

where

$$\begin{cases} L = \left(\sqrt[n]{\prod_{k=1}^{n} a_{1k}}, \quad \cdots \quad \sqrt[n]{\prod_{k=1}^{n} a_{ik}}, \quad \cdots \quad \sqrt[n]{\prod_{k=1}^{n} a_{jk}}, \quad \cdots \quad \sqrt[n]{\prod_{k=1}^{n} a_{nk}} \right)^T \\ R = \left(\sqrt[n]{\prod_{k=1}^{n} a_{k1}}, \quad \cdots \quad \sqrt[n]{\prod_{k=1}^{n} a_{ki}}, \quad \cdots \quad \sqrt[n]{\prod_{k=1}^{n} a_{kj}}, \quad \cdots \quad \sqrt[n]{\prod_{k=1}^{n} a_{kn}} \right) \end{cases} \tag{6}$$

Step 2: Compute geometric mean matrix by formula

$$\bar{A} = L \times R \tag{7}$$

We can obtain the geometric mean matrix $\bar{A}$. The computing processes are shown below:

$$\bar{A} = L \times R = \begin{pmatrix} \sqrt[n]{\prod_{k=1}^{n} a_{1k}} \\ \vdots \\ \sqrt[n]{\prod_{k=1}^{n} a_{ik}} \\ \vdots \\ \sqrt[n]{\prod_{k=1}^{n} a_{jk}} \\ \vdots \\ \sqrt[n]{\prod_{k=1}^{n} a_{nk}} \end{pmatrix}_{n\times 1} \times \left(\sqrt[n]{\prod_{k=1}^{n} a_{k1}}, \quad \cdots \quad \sqrt[n]{\prod_{k=1}^{n} a_{ki}}, \quad \cdots \quad \sqrt[n]{\prod_{k=1}^{n} a_{kj}}, \quad \cdots \quad \sqrt[n]{\prod_{k=1}^{n} a_{nk}} \right)_{1\times n}$$

$$= \begin{pmatrix} \sqrt[n]{\prod_{k=1}^{n} a_{1k}a_{k1}} & \cdots & \sqrt[n]{\prod_{k=1}^{n} a_{1k}a_{ki}} & \cdots & \sqrt[n]{\prod_{k=1}^{n} a_{1k}a_{kj}} & \cdots & \sqrt[n]{\prod_{k=1}^{n} a_{1k}a_{kn}} \\ \vdots & & \cdots & & \vdots & & \vdots \\ \sqrt[n]{\prod_{k=1}^{n} a_{ik}a_{k1}} & \vdots & \sqrt[n]{\prod_{k=1}^{n} a_{ik}a_{ki}} & \cdots & \sqrt[n]{\prod_{k=1}^{n} a_{ik}a_{kj}} & \cdots & \sqrt[n]{\prod_{k=1}^{n} a_{ik}a_{kn}} \\ \vdots & & \vdots & & \vdots & & \vdots \\ \sqrt[n]{\prod_{k=1}^{n} a_{jk}a_{k1}} & \cdots & \sqrt[n]{\prod_{k=1}^{n} a_{jk}a_{ki}} & \cdots & \sqrt[n]{\prod_{k=1}^{n} a_{jk}a_{kj}} & \cdots & \sqrt[n]{\prod_{k=1}^{n} a_{jk}a_{kn}} \\ \vdots & & \vdots & & \vdots & & \vdots \\ \sqrt[n]{\prod_{k=1}^{n} a_{nk}a_{k1}} & \cdots & \sqrt[n]{\prod_{k=1}^{n} a_{nk}a_{ki}} & \cdots & \sqrt[n]{\prod_{k=1}^{n} a_{nk}a_{kj}} & \cdots & \sqrt[n]{\prod_{k=1}^{n} a_{nk}a_{kn}} \end{pmatrix}_{n\times n}$$

$$= \left(\sqrt[n]{\prod_{k=1}^{n} a_{ik}a_{kj}} \right)_{n\times n}$$

where L is an *n-by-one* column matrix, which is composed of all geometric means of rows while R is a *one-by-n* row matrix R composed of all geometric means of columns, as shown in formula (6) and the two edges of matrix A in Step 1. The geometric mean matrix $\bar{A}$ can be easily computed by multiplying L to R.

Step 3: Compute geometric mean induced bias matrix (GMIBM) C by formula,

$$C = \bar{A} \circ A^T = \left(c_{ij}\right) = \left(\bar{a}_{ij} \cdot a_{ji}\right) = (\sqrt[n]{\prod_{k=1}^{n} a_{ik}a_{kj}} \cdot a_{ji}) \tag{8}$$

Step 4: Identify the data with the largest value, denoted as $c_{ij}^{\max}$, deviating from 1 in matrix C, then the corresponding a_{ij} is regarded as the most inconsistent data in matrix A. If there are other data, say c_{mn}, c_{pq}, whose values are also deviating far away from 1, then their corresponding data in matrix A, a_{mn}, a_{pq}, can also be considered as the possible inconsistent elements. Once the inconsistent data are

identified, the following two steps are proposed to adjust the inconsistent data.

Step II: Inconsistency Adjustment:

Step 1: Estimate the value of identified inconsistent data by formula

$$\tilde{a}_{ij} = \sqrt[n-2]{\prod_{k=1,\neq i,j}^{n} a_{ik}a_{kj}} = \sqrt[n-2]{\frac{\bar{a}_{ij}^{n}}{a_{ij}^{2}}} = \bar{a}_{ij}\left(\frac{\bar{a}_{ij}}{a_{ij}}\right)^{\frac{2}{n-2}} \tag{9}$$

where $\tilde{a}_{ij}$ denotes the estimated value of the most inconsistent data a_{ij} while $\bar{a}_{ij}$ is the geometric mean value located at the i^{th} row and the j^{th} column of geometric mean matrix $\bar{A}$.

Step 2: Test the consistency of the revised matrix A by replacing the inconsistent data with the estimated values.

To summarize, the processes of dealing with inconsistency by GMIBM include two major steps, that is, inconsistency identification and inconsistency adjustment. The first two steps in Step I are used to show the specific procedure of computing geometric mean matrix $\bar{A}$. For simplicity, one can directly use the latter two steps as the sub-steps of Step I to identify the most inconsistent data. To verify the effectiveness and accuracy of GMIBM, in the following, two numerical examples introduced in [20] are used to illustrate the proposed model.

4 Illustrative Examples

To test the effectiveness and correctness of the proposed GMIBM, and illustrate the processes of the proposed specific inconsistency identification and adjustment by numerical examples, two numerical examples in [21] are used in this paper.

Example 1 The *Example 1* used in [21], which was firstly introduced in [20], is a 4×4 inconsistent pair-wise comparison matrix A with *CR=0.173>0.1*.

$$A = \begin{pmatrix} 1 & 1/9 & 3 & 1/5 \\ 9 & 1 & 5 & 2 \\ 1/3 & 1/5 & 1 & 1/2 \\ 5 & 1/2 & 2 & 1 \end{pmatrix}$$

Apply the GMIBM to this matrix:

Step I: Inconsistency Identification

Step 1: Compute the column matrix L and the row matrix R by formula (6),

$$L = \begin{pmatrix} 0.5081 & 3.0801 & 0.4273 & 1.4953 \end{pmatrix}^{T}, R = \begin{pmatrix} 1.968 & 0.3247 & 2.3403 & 0.6687 \end{pmatrix}$$

Step 2: Compute geometric mean matrix by formula (7),

$$\begin{aligned} \bar{A} &= L \times R \\ &= \begin{pmatrix} 1 & 0.165 & 1.1892 & 0.3398 \\ 6.0615 & 1 & 7.2084 & 2.0598 \\ 0.8409 & 0.1387 & 1 & 0.2857 \\ 2.9428 & 0.4855 & 3.4996 & 1 \end{pmatrix} \end{aligned}$$

Step 3: Compute geometric mean induced bias matrix (GMIBM) C by formula (8) ,

$$C = \bar{A}^{\circ} A^{T} = \begin{pmatrix} 1 & 1.4848 & 0.3964 & 1.6990 \\ 0.6735 & 1 & 1.4417 & 1.0299 \\ 2.5227 & 0.6936 & 1 & 0.5715 \\ 0.5886 & 0.9710 & 1.7498 & 1 \end{pmatrix}$$

Step 4: Identify the largest value $c_{ij}^{\max}$ in matrix C. Here $c_{ij}^{\max} = c_{31}^{\max} = 2.5227$, deviating from 1 in matrix C, then the corresponding element a_{31} in matrix A is regarded as the most inconsistent element,indicating that it is smaller than its average values.

Step II: Inconsistency Adjustment

Step 1: Estimate the possible proper value of a_{31} using the estimating formula (9)

$$\tilde{a}_{31} = \sqrt[4-2]{\frac{\bar{a}_{31}^4}{a_{31}^2}} = \sqrt{\frac{0.8409^4}{(1/3)^2}} = 2.1213 \approx 2$$

Step 2: Test the consistency of the revised matrix A by replacing the inconsistent elements a_{31} and a_{13} with the estimated values 2 and 1/2. The revised matrix passed with *CR=0.0028<0.1*.

The identified inconsistent data and its estimated value are the same as the ones in [19] and [20], but the proposed method is faster to find the inconsistent element and estimate the values.

Example 2 The second example in [20] is a 4×4 inconsistent pair-wise comparison matrix A with *CR*=1.0242>*0.1*.

$$A = \begin{pmatrix} 1 & 2 & 4 & \frac{1}{8} \\ \frac{1}{2} & 1 & 2 & 4 \\ \frac{1}{4} & \frac{1}{2} & 1 & 2 \\ 8 & \frac{1}{4} & \frac{1}{2} & 1 \end{pmatrix}$$

Apply the GMIBM to this matrix:

Step I: Inconsistency Identification

Step 1: Compute the column matrix R and the row matrix L by formula (6),

$$L = \begin{pmatrix} 1 & 1.4142 & 0.7071 & 1 \end{pmatrix}^{T}, \quad R = \begin{pmatrix} 1 & 0.7071 & 1.4142 & 1 \end{pmatrix}$$

Step 2: Compute geometric mean matrix by formula (7),

$$\bar{A} = L \times R = \begin{pmatrix} 1 & 0.7071 & 1.4142 & 1 \\ 1.4142 & 1 & 2 & 1.4142 \\ 0.7071 & 0.5 & 1 & 0.7071 \\ 1 & 0.7071 & 1.4142 & 1 \end{pmatrix}$$

Step 3: Compute geometric mean induced bias matrix (GMIBM) C by formula (8),

$$C = \bar{A}^{\circ} A^{T} = \begin{pmatrix} 1 & 0.3536 & 0.3536 & 8 \\ 2.2828 & 1 & 1 & 0.3536 \\ 2.2828 & 1 & 1 & 0.3536 \\ 0.125 & 2.8284 & 2.8284 & 1 \end{pmatrix}$$

Step 4: Identify the largest value $c_{ij}^{\max}$ in matrix C. Here $c_{ij}^{\max} = c_{14}^{\max} = 8$, deviating from 1 in matrix C, then the corresponding element a_{14} in matrix A is regarded as the most inconsistent element, indicating that it is smaller than its average values.

Step II: Inconsistency Adjustment

Step 1: Estimate the possible proper value of a_{14} using the estimating formula (9),

$$\tilde{a}_{14} = \bar{a}_{14}\left(\frac{\bar{a}_{14}}{a_{14}}\right)^{\frac{2}{4-2}} = 1 \cdot \left(\frac{1}{1/8}\right) = 8$$

Step 2: Test the consistency of the revised matrix A by replacing the inconsistent elements a_{14} and a_{41} with the estimated values 8 and 1/8. The revised matrix passed with CR=0<0.1.

The identified inconsistent data and the estimated value are the same as the ones in [20], but the proposed method is faster to find the inconsistent data and estimate the values.

5 Conclusions

In this paper, we proposed a geometric mean induced bias matrix (GMIBM), which is only based on the original matrix and independent to the way of deriving the priority weights, to identify the cardinal inconsistent data in the judgment matrix. The inconsistent data identification process includes two major steps, namely, inconsistency identification and inconsistency adjustment. The inconsistent data can be easily and quickly identified by observing the data with the largest value(s) deviating from 1 in the induced bias matrix C. Besides, the identified data can be estimated by the estimated formula. Two examples are used to illustrate the proposed model. The results show that the proposed model is easier and faster to identify and adjust the inconsistent data than existing models.

Acknowledgments

This research has been partially supported by grants from Academic Degree Programs Construction at Southwest University for Nationalities (#2012XWD-S1201), and grants from the National Natural Science Foundation of China (#70901015 and #70921061), the Fundamental Research Funds for the Central Universities and Program for New Century Excellent Talents in University (NCET-10-0293). No additional external funding received for this study. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Bibliography

[1] Tufekci, S., Wallace, W.A, *The emerging area of emergency management and engineering*,IEEE Transactions on Engineering Management, 1998, 45(2):103-105, 1998.

[2] Shan, L., Zhao, Z.P, *Evaluation on emergency management system in supply chain based on trapezoidal fuzzy order weighted average (FOWA) operator*, Applied mechanics and materials,34-35:1170-1174, 2010.

[3] Chen, Y., Li, K., Xu, H.Y., Xu, H.Y, *A DEA-TOPSIS method for multiple criteria decision analysis in emergency management*,Journal of Systems Science and Systems Engineering, 18(4):489-507, 2009.

[4] Zhang, K., Kluck, C., Achari, G, *A Comparative Approach for Ranking Contaminated Sites Based on the Risk Assessment Paradigm Using Fuzzy PROMETHEE*, Environmental Management, 44(5):952-967, 2009.

[5] Tie, Y.B. Tang, C, *Zhou, C.H. The application of AHP to emergency response capability assessment in urban disaster*, Journal of Geological Hazards and Environment Preservation,16(4):433-437, 2005.

[6] Ergu, D., Kou, G., Peng, Y., Shi, Y., Shi, Y, *The Analytic Hierarchy Process: Task Scheduling and Resource Allocation in Cloud Computing Environment*,Journal of Supercomputing, DOI: 10.1007/s11227-011-0625-1, 2011.

[7] Wu, W.; Kou, G.; Peng, Y.; Ergu, D., *Improved AHP-group decision making for investment strategy selection*, Technological and Economic Development of Economy, 18(2), 2012. DOI: 10.3846/20294913.2012.680520

[8] Ergu, D., Kou, G., Shi, Y., Shi, Y., *Analytic Network Process in Risk Assessment and Decision Analysis*, Computers & Operations Research, 2011. DOI: 10.1016/j.cor.2011.03.005.

[9] Levy J.K., Taji K., *Group decision support for hazards planning and emergency management: A Group Analytic Network Process (GANP) approach*, Mathematical and Computer Modelling,46(7-8): 906-917, 2007.

[10] Zhou Q., Huang W., Zhang Y., *Identifying critical success factors in emergency management using a fuzzy DEMATEL method*, Safety Science, 49(2):243-252, 2011.

[11] Peng, Y., Kou, G., Wang, G., and Shi, Y.,*FAMCDM: A Fusion Approach of MCDM Methods to Rank Multiclass Classification Algorithms*, Omega, 39(6): 677-689, 2011, DOI:10.1016/j.omega.2011.01.009

[12] Ergu, D., Kou, G., *Questionnaire Design Improvement and Missing Item Scores Estimation for Rapid and Efficient Decision Making*, Annals of Operations Research, DOI: 10.1007/s10479-011-0922-3, 2011.

[13] Filip,F.G., D.A. Donciulescu, Cr.I. Filip.,*Towards intelligent real-time Decision Support Systems for industrial milieu*,Studies in Informatics and Control , 11 (4):303-311, 2001.

[14] Filip F.G., *Decision support and control for large-scale complex systems*, Annual Reviews in Control, 32(1):61-70,2008.

[15] Xu.Z, Wei.C ., *A consistency improving method in the analytic hierarchy process*, European Journal of Operational Research, 116:443–449, 1999.

[16] Cao, D., Leung, L.C., Law, J.S., *Modifying inconsistent comparison matrix in analytic hierarchy process: A heuristic approach*, Decision Support Systems ,44:944–953, 2008.

[17] Saaty, T.L., *The Analytical Hierarchy Process*, New York: McGraw-Hill, 1980.

[18] Saaty, T.L., *How to Make a Decision: The Analytic Hierarchy Process*, Interfaces, 24:19-43, 1994.

[19] Saaty, T.L., *Decision-making with the AHP: Why is the principal eigenvector necessary*, European Journal of Operational Research, 145(1): 85-91, 2003.

[20] Yang, Y.Q., *Study on adjustment method for the inconsistency of the judgment matrix in AHP*, Operations research and management science, 8(3):12-16,1999 (in Chinese).

[21] Ergu, D., Kou, G., Peng, Y., Shi, Y., *A Simple Method to Improve the Consistency Ratio of the Pairwise Comparison Matrix in ANP*, European Journal of Operational Research, 213(1):246–259, 2011.

INT J COMPUT COMMUN, ISSN 1841-9836
Vol.7 (2012), No. 3 (September), pp. 459-472

Inverse Kinematics Solution for Robot Manipulator based on Neural Network under Joint Subspace

Y. Feng, W. Yao-nan, Y. Yi-min

Yin Feng, Wang Yao-nan, Yang Yi-min
The College of Electrical and Information Engineering
Hunan University, Changsha, Hunan Province 410082, P.R.China
E-mail: yinfeng83@126.com, yaonan@hnu.cn, yimin-yang@126.com

Abstract: Neural networks with their inherent learning ability have been widely applied to solve the robot manipulator inverse kinematics problems. However, there are still two open problems: (1) without knowing inverse kinematic expressions, these solutions have the difficulty of how to collect training sets, and (2) the gradient-based learning algorithms can cause a very slow training process, especially for a complex configuration, or a large set of training data. Unlike these traditional implementations, the proposed metho trains neural network in joint subspace which can be easily calculated with electromagnetism-like method. The kinematics equation and its inverse are one-to-one mapping within the subspace. Thus the constrained training sets can be easily collected by forward kinematics relations. For issue 2, this paper uses a novel learning algorithm called extreme learning machine (ELM) which randomly choose the input weights and analytically determines the output weights of the single hidden layer feedforward neural networks (SLFNs). In theory, this algorithm tends to provide the best generalization performance at extremely fast learning speed. The results show that the proposed approach has not only greatly reduced the computation time but also improved the precision.
Keywords: Inverse kinematics, neural network, extreme learning machine.

1 Introduction

The inverse kinematics (IK) problem for a serial-chain manipulator is to find the values of the joint positions given the position and orientation of the end-effector relative to the base. There are many solutions to solve the inverse kinematics problem [1] , such as geometric, algebraic, and numerical iterative methods. In particular, some of the most popular methods are mainly based on inversion of the mapping established between joint space and task space by the Jacobian matrix. This solution uses numerical iteration to invert the forward kinematic Jacobian matrix and does not always guarantee to produce all possible inverse kinematics solutions. The artificial neural network, which has significant flexibility and learning ability, has been used in the inverse kinematics problem. One solution followed a closed-loop control scheme where a neural network is used to directly learn the nonlinear relationship between the displacement in the workspace and control signal in the joint angle space to achieve a desired position([2] and [3]) . Other schemes used neural networks to learn a mapping function from the world space to joint space. Although there are various neural networks, the multi-layer perceptron network (MLPN) and the radial basis function network (RBFN) are the most popular neural network applied to functional approximation problems.

In [4], the effects of structural parameters, iteration steps and different numbers of training points on the performance of the inverse kinematics approximation were investigated. The results showed that a more complex MLPN configuration is likely to produce a more accurate inverse kinematics approximation. However, it also leads to the number of iterations increasing significantly to satisfy the required training goal. Similarly, the neural networkss generalization ability seems to be improved when the number of training sets is increased. However, if the numbers of hidden neurons or training sets are too large,

the training process can not even converge to an expected error goal in some cases. In [5], an MLPN with various structures of the input layer were proposed to solve the inverse kinematics problem of a 6 DOF manipulator. Three different forms representing the orientation of the end-effector with respect to the base were defined: a 33 rotation matrix, a set of 3 Euler angles and one angle and a 13 unit vector. Another solution combining an MLPN and a lookup table to solve an inverse kinematics problem of a redundant manipulator was proposed in [6]. Although the use of MLPN in the inverse kinematics problem has a greater extent, there have some significant disadvantages. For example, there is no reasonable mechanism to select a suitable network configuration relating to the system characteristics represented by training sets. In addition, training MLPN using the back-error propagation algorithm is complex and slow. For a complex MLPN structure required for a complex configuration manipulator, or a large set of training data, the training process is slow to converge to a specific goal. Therefore, trends towards using RBFN which are conceptually simpler and possess the ability to model any nonlinear function conveniently have become more popular.

In [7], a variety of network configurations based on RBFN were developed to explore the effect of various network configurations on the performance of the network. In [8], a novel architecture of RBFN with two hidden layers was developed for a inverse kinematics problem of a 3-link manipulator. A fusion approach was proposed in [9]. The proposed approach used RBFN for prediction of incremental joint angles which in turn were transformed into absolute joint angles with the assistance of forward kinematics relations. Another RBFN-based method was presented in [10]. It developed a structure of six parallel RBFN, each of which consists of six inputs which represent a location of the end-effector and one output as the joint angle. Thus, the group of six parallel RBFN (one for each joint angle) could perform an inverse kinematics approximation. In addition, some hybrid techniques made use of neural networks along with expert system [11], fuzzy logic [12] and genetic algorithm [13] for solving the inverse kinematics. Though these intelligence approaches can be applied for two or three DOF planar robots, they often demand high performance computing systems and complex computer programming for complex robotic system.

Traditionally, all the parameters (weights and biases) of the feedforward networks need to be turned. For past decades gradient descent-based methods have mainly been used in various learning algorithms. It is clear that the learning process often needs many training patterns and times to cover the entire workspace. Thus, it is not surprising to see that it may take several hours and several days to train neural networks to solve the inverse kinematics. Unlike these traditional implementations, this paper uses a novel learning algorithm called extreme learning machine (ELM) for single hidden layer feedforward neural networks (SLFN) which randomly chooses the input weights and biases, and analytically determines the output weights of SLFN [14,15]. In theory, this algorithm tends to provide the best generalization performance at extremely fast learning speed.

Another issue of concern for solving the inverse kinematics using neural networks is the training data sets. As we know, the joint space of the robot can be considered as an inverse image of the Cartesian space and vice versa. Thus, the forward kinematics can be assumed to be an inverse image of inverse kinematics and vice versa [9]. The pose P can be used as an input and the corresponding joint angle Q as the output for the neural network training data. In other words, Q$\longrightarrow$P relationship is used while generating the data whereas P$\longrightarrow$Q mapping is done while training the neural network. Usually, the inverse kinematics problems have multiple solutions. For example, the PUMA 560 robot has at most eight solutions when there are no joint limits imposed. Hence, the inverse kinematics equation is one-to-many mapping.Unfortunately,the neural network can not match the actual output with the desired output. So the learning error of neural network is hard to be calculated when training. An effective solution is that the training sets are constrained to only one solution set so that the one-to-one mapping can be achieved. For simple structure, such as two-link planar manipulator, the training sets can be collected based on the inverse kinematics equation which only consist of either the positive or negative solution. However, this solution has the difficulty of how to collect constrained data without knowing the inverse kinematic

expressions of the complex robotic system. The present work attempts to resolve this crucial issue by using a novel heuristic algorithm, called electromagnetism-like method (EM)[16,17], for determining a joint subspace which includes one and only one inverse kinematics solution for a given pose. For convenience's sake, a graphic depiction of the proposed method is illustrated by using a 2D example, as shown in figure 1.

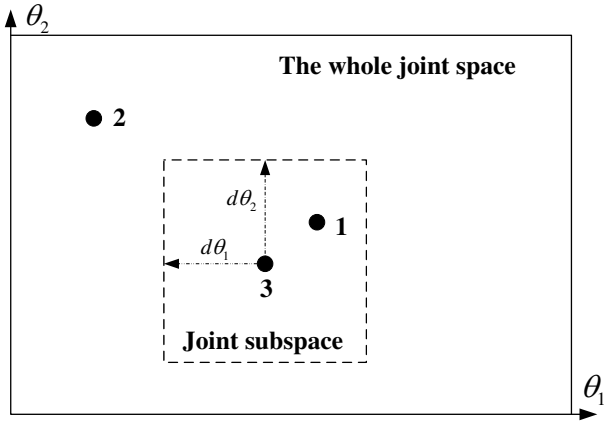


Figure 1: An illustration of the proposed algorithm. The point 1 and 2 is true solution and point 3 is an approximate solution

In Figure 1, assume that there are two inverse solutions in the whole space. One approximate solution is denoted by (θ_1^*, θ_2^*) . If we select appropriate $d\theta_1$ and $d\theta_2$ such that $\theta_1 \in [\theta_1^* - d\theta_1, \theta_1^* + d\theta_1]$ and $d\theta_2 \in [\theta_2^* - d\theta_2, \theta_2^* + d\theta_2]$, this joint subspace includes just one true solution point 1. Based on this, the data required for training of neural network is proposed to be derived from the joint subspace with the forward kinematics relations instead of deriving a set complex inverse kinematics equations from the whole joint space. Then the true solution point 1 can be approached by using the trained network. The proposed method can be summarized as follows:

1. Given a desired coordinate of position and orientation of the end-effector, making use of EM to calculate an approximate solution near to one true solution.

2. Specify appropriate value of $d\theta_k$ such that $\theta_k \in [\theta_k^* - d\theta_k, \theta_k^* + d\theta_k](k = 1, 2, \cdots, n)$, where n is the number of joint and θ_k^* is the approximate solution of the k joint variable calculated with EM in step 1. For the sake of simplicity, all $d\theta$ can be set to be the same value.

3. Collect the training sets from the joint subspace, and train the neural network with ELM.

For a new coordinate of position and orientation of the end-effector, the neural networks usually need to be retrained following the steps above. Fortunately, our results show that the training process is very fast. Thus, the retraining procedure appears to be acceptable.

2 Calculation of joint subspace with EM

2.1 Problem formulation

As shown in Figure 2, the desired position vector and orientation matrix of a manipulator end-effector are denoted by: $\boldsymbol{P_d}$ and $[\boldsymbol{R_d}] = [\boldsymbol{d_1}, d_2, \boldsymbol{d_3}]$, where $\boldsymbol{d_j}$ (j=1, 2, 3) are unit vectors along the $\mathbf{x_d}, \mathbf{y_d}, \mathbf{z_d}$ axes. $\boldsymbol{P_h}$ is the current position vector of the end-effector. The current orientation matrix is defined by : $[\boldsymbol{R_h}] = [\boldsymbol{h_1}, \boldsymbol{h_2}, \boldsymbol{h_3}]$, where $\boldsymbol{h_j}$ (j=1,2,3) are unit vectors along the $\mathbf{x_h}, \mathbf{y_h}, \mathbf{z_h}$ axes and the joint variables are denoted by the n×1 vector, $\theta = [\theta_1, \theta_2, \cdots, \theta_n]^{\mathrm{T}}$.

The error between the current and the desired locations of the end-effector can be described by the following functions [18]:

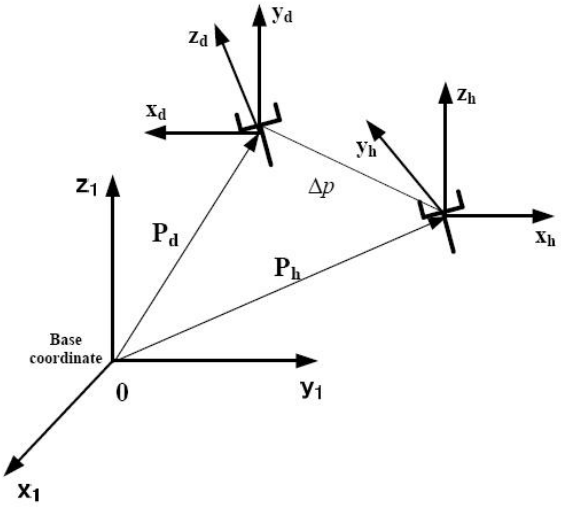


Figure 2: The current and the desired end-effector configurations

Position error:

$$\Delta p(\theta) = \| \boldsymbol{P_d} - \boldsymbol{P_h}(\theta) \| \tag{1}$$

Orientation error:

$$\Delta o(\theta) = \sum_{j=1}^{3} (\boldsymbol{d}_j \cdot \boldsymbol{h}_j(\theta) - 1)^2 \tag{2}$$

The total error:

$$e(\theta) = \Delta p(\theta) + \Delta o(\theta) \tag{3}$$

Where $(\cdot)$ denotes the vector dot product. Furthermore, the total error can be chosen to be a weighted sum of the position and orientation components:

$$e(\theta) = w_p \Delta p(\theta) + w_o \Delta o(\theta) \tag{4}$$

Where w_p and w_o are weighting factors assigned to position and orientation, respectively, such that $w_p + w_o = 1$. Now the inverse kinematics problem is to find a solution θ^*, such that $e(\theta^*) \leq \varepsilon(\varepsilon \longrightarrow 0)$.It is clear that this problem can be transformed into the following minimization problem:

$$\min e(\theta) \quad \text{s.t.} \quad \theta \in \Re^{\mathrm{n}} | \mathrm{l_k} \leq \theta_{\mathrm{k}} \leq \mathrm{u_k}, \mathrm{i} = 1, 2, \cdots, \mathrm{n} \tag{5}$$

2.2 Brief of Electromagnetism-like method (EM)

To solve the problem in (5), the general scheme of EM is given by following procedures: Initialize, local search, calculation of charge and total force vector and movement according to the total force.

Initialization

The procedure initialization is used to sample m points, $\theta^1, \cdots, \theta^m$, randomly from the feasible domain of the joint variables, where $\theta^i = [\theta_1^i, \cdots, \theta_n^i](i = 1, \cdots, m)$. The procedure uniform sampling can be determined by following

$$\theta_k^i = l_k + rand \cdot (u_k - l_k) \qquad k = 1, 2, \cdots, n \tag{6}$$

The procedure ends with m points identified, and the point that has the best function value is stored in θ^{best}.

Local search

The local search procedure is used to gather the local information and improve the current solutions. It can be applied to one or many points for local refinement per iteration. The selection of these two procedures, does not affect the convergence result.

Calculation of charge and total force vector

The charges of the points are calculated according to their objective function values, and the charge of each point is not constant and changes from iteration to iteration. The charge of the ith point, q^i, is evaluated as following

$$q^i = \mathbf{exp}[-n\frac{(e(\theta^i) - e(\theta^{best}))}{\sum_{k=1}^{m}(e(\theta^k) - e(\theta^{best}))}], i = 1, 2, \cdots, m \tag{7}$$

In this way, points that have better objective values possess higher charges. Notice that, unlike electrical charges, no signs are attached to the charge of an individual point in (7). Instead, the direction of a particular force between two points is decided after comparing their objective function values. Hence, the total force $\boldsymbol{F_i}$ exerted on point i is computed by the following equation

$$\boldsymbol{F}^i = \begin{cases} \sum_{j\neq i}^{m}(\theta^j - \theta^i)\frac{q^j q^i}{\| \theta^j - \theta^i \|^2}, & if \quad e(\theta^j) < e(\theta^i) \\ \sum_{j\neq i}^{m}(\theta^i - \theta^j)\frac{q^j q^i}{\| \theta^j - \theta^i \|^2}, & others \end{cases} \tag{8}$$

According to (8), the point that has a better objective function value attracts the other one. Contrarily, the point with worse objective function value repels the other. Since θ^{best} has the minimum objective function value, it acts as an absolute point of attraction. Then it attracts all other points in the population to better region.

Movement according to the total force

After evaluating the total force vector $\boldsymbol{F}^i$, the point i is moved in the direction of the force by a random step length in (8). Here the random step length λ is assumed to be uniformly distributed between 0 and 1.

$$\theta^i = \theta^i + \lambda\frac{\boldsymbol{F}^i}{\|\boldsymbol{F}^i\|}\boldsymbol{R}_{NG}, \quad i = 1, 2, \cdots, m \tag{9}$$

In (9), $\boldsymbol{R}_{NG}$ is a vector whose components denote the allowed feasible movement toward the upper bound u_k or the lower bound l_k of the joint variables.

After finishing the above procedures, the positions of points are updated and we have finished one iteration calculation of EM. Take the Figure 3 for example. There are three particles and their own objective values are 15, 10 and 5, respectively. Because particle 1 is worse than particle 3 while particle 2 is better than particle 3, particle 1 represents a repulsion force which is F13 and particle 2 encourages particle 3 that moves to the neighborhood region of particle 2. Consequently, particle 3 moves along with the total force F.

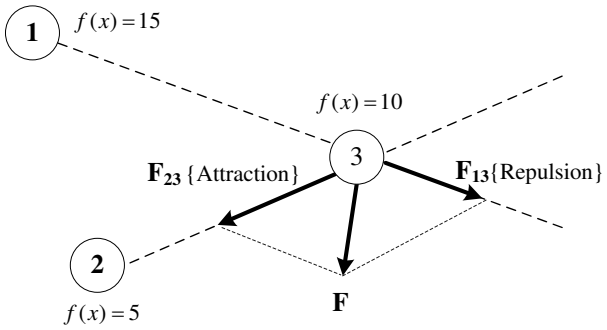


Figure 3: An example of attract-repulse effect on particle number 3

2.3 Performance evaluation of EM in solving the inverse kinematics

This example is used to examine the precision of the approximate solution calculated by EM, which directly impact the choice of the interval width $d\theta$. The robot structure for this example is based on PUMA 560. The link parameters are given in Table 1.

Table 1 The Link parameters of the PUMA 560 Robot

Joint	Link length (m)	Twist angle (degree)	Offset length (m)	Joint limitations (degree)
1	0	-90	0.6604	$[-160, 160]$
2	0.4320	0	0.2000	$[-225, 045]$
3	0	90	-0.0505	$[-045, 225]$
4	0	-90	0.4320	$[-110, 170]$
5	0	90	0.0000	$[-100, 100]$
6	0	0	0.0565	$[-266, 266]$

The desired configuration of the end-effector is given by: Pd=[0.7433, 0.3111, 0.7883] (m), and d1= [-0.6366, 0.7712, -0.0084], d2= [0.0227, 0.0296, 0.9993], d3 = [0.7709, 0.6359, -0.0364]. Note that there are multiple solutions within the joint limitations shown in table 1. For the sake of simplicity, the joint 1 and 3 limitations are rearranged into [-120,160] and [-45,120], respectively. For the given coordinates of the end-effector, it corresponds to an exact solution of θ = [10 20 30 40 50 60](degree) within the adjusted joint limitations. Then the error between an approximate solution and the true solution can be easily calculated. The stopping criterion for EM is defined by ε =0.01. In other words, stop calculation when the total error (see (3)) is less than ε. In this example, 100 trials have been conducted for EM and the maximum absolute error (absolute value) at each joint angle is shown in Figure 4.

From Figure 4, the widths of the joint limitations are set as about 20° at least, i.e.$d\theta$ = 10°, which can guarantee that the one-to-one mapping is achieved. It should be noted that EM is not suitable for high precision applications. As can be seen in figure 5, the number of evaluations drastically increases with precision. However, during the early stage of computations, EM algorithm is highly efficient. Thus, EM is suitable for providing a good initial guess.

3 Model the inverse kinematics with neural network

The architecture used for solving the inverse kinematics problems is shown in Figure 6. The single layer network consists of n outputs (joint angles) and 12 inputs [**n**, **s**, **a**, **p**] which represents a location (position and orientation) of the end-effector. As mentioned earlier, the training set have been constrained to only one solution set so that the one-to-one mapping could be achieved.

For the present work, a fast and accurate learning algorithm called as Extreme learning machine

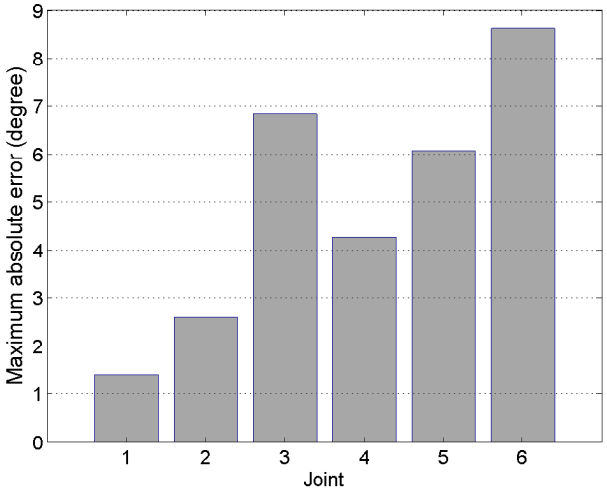


Figure 4: Maximum absolute error at each joint angle among 100 trials

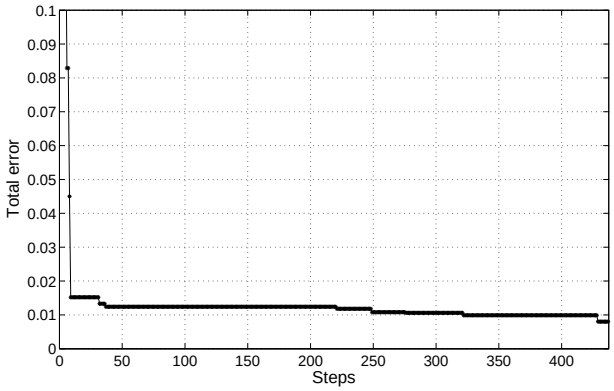


Figure 5: The performance of EM

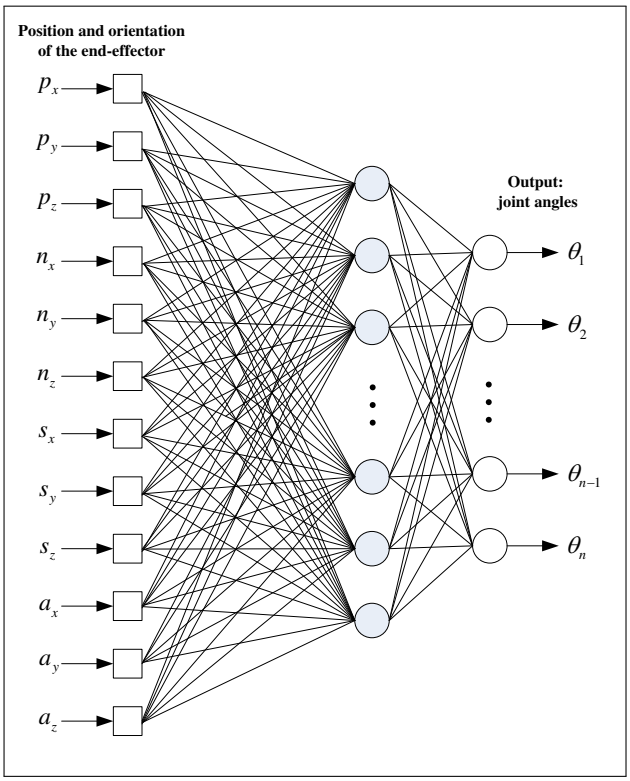


Figure 6: A general structure of the SLFN to approximate the inverse kinematics

(ELM) are used to train the neural network in modeling the inverse kinematics of robot. Test results show that the learning speed of ELM algorithm is much faster than the traditional methods. For example, the learning speed of ELM is at least 1000 and 2000 times faster than BP and SVM for solving the benchmark problem of California Housing [14]. Thus, this new training method is very suitable for solving the inverse kinematics.

3.1 Brief of Extreme learning machine (ELM)

ELM is a unified with randomly generated hidden nodes independent of the training data. The output of an SLFN with L hidden nodes can be represented by

$$f_L(\mathbf{x}) = \sum_{i=1}^{L} \beta_i G(\mathbf{a}_i, b_i, \mathbf{x}), \quad \mathbf{x} \in R^n, \mathbf{a}_i \in R^n \tag{10}$$

Where $\mathbf{a}_i$ and b_i are the learning parameters of hidden nodes and β_i is the weight connecting in ith hidden node to the output node. $G(\mathbf{a}_i, b_i, \mathbf{x})$ is the output of the ith hidden nodes with respect to the input $\mathbf{x}$. Additive and RBF hidden nodes are used often in applications.

For additive hidden node with the activation function $g(x)$ (e.g. sigmoid,threshold,sin,etc.), $G(\mathbf{a}_i, b_i, \mathbf{x})$ is given by

$$G(\mathbf{a}_i, b_i, \mathbf{x}) = g(\mathbf{a}_i \cdot \mathbf{x} + b_i), \qquad b_i \in R \tag{11}$$

Where $\mathbf{a}_i$ is the weight vector connecting the input layer to the ith hidden node and b_i is the bias of the ith hidden node. $\mathbf{a}_i \cdot \mathbf{x}$ denotes the inner product of vectors.

For RBF hidden node with activation function $g(x)$ (e.g. Gaussian), $G(\mathbf{a}_i, b_i, \mathbf{x})$ is given by

$$G(\mathbf{a}_i, b_i, \mathbf{x}) = g(b_i \parallel \mathbf{x} - \mathbf{a}_i \parallel) \qquad b_i \in R^+ \tag{12}$$

Where $\mathbf{a}_i$ and b_i are the center and impact factor of ith RBF node. R^+ indicates the set of all positive real values. The RBF network is a special case of SLFN with RBF nodes in its hidden layer. Each RBF node has its own centroid and impact factor, and its output is given by a radially symmetric function of the distance between the input and the center.

For a given set of trainning samples$(\mathbf{x}_i, \mathbf{t}_i)_{i=1}^N \subset R^n \times R^m$,if the outputs of the network are equal to the targets, we have

$$f_L(\mathbf{x}_j) = \sum_{i=1}^{L} \beta_i G(\mathbf{a}_i, b_i, \mathbf{x}_i) = t_j \qquad j = 1, 2, \cdots, N. \tag{13}$$

Above equation can be written compactly as

$$\mathbf{H}\beta = \mathbf{T} \tag{14}$$

Where

$$\mathbf{H} = \begin{bmatrix} G(\mathbf{a}_1, b_1, \mathbf{x}_1), \cdots, G(a_L, b_L, \mathbf{x}_1) \\ \vdots, \cdots, \vdots \\ G(\mathbf{a}_1, b_1, \mathbf{x}_N), \cdots, G(a_L, b_L, \mathbf{x}_N) \end{bmatrix}_{N \times L} \tag{15}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_L^T \end{bmatrix}_{L \times m} \quad \text{and} \quad T = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix}_{N \times m} \tag{16}$$

β^T is the transpose of a matrix or vector β. $\mathbf{H}$ is called the hidden layer output matrix of the network; the ith column of $\mathbf{H}$ is the ith hidden node's output vector with respect to input and the jth row of $\mathbf{H}$ is the output vector of the hidden layer with respect to input x_j.

Usually, when the number of training data is larger than the number of hidden nodes $N > L$, one can not expect an exact solution of the system (14).After the hidden nodes are randomly generated and given the training data, the hidden-layer output matrix $\mathbf{H}$ is known and need not be tuned. Thus, training SLFNs simply amounts to getting the solution of a linear system (14) of output weights β. Under the constraint of minimum norm least square, i.e., min $\|\beta\|$ and$\|\mathbf{H}\beta - \mathbf{T}\|$, a simple representation of the solution of the system (14) is given explicitly as

$$\widehat{\beta} = \mathbf{H}^{\dagger}\mathbf{T} \tag{17}$$

Where $\mathbf{H}^{\dagger}$ is the Moore-Penrose generalized inverse of the hidden-layer output matrix $\mathbf{H}$. The simple learning algorithm can be summarized as follows:

Algorithm ELM: Given a training set $\Re = (\mathbf{x}_i, \mathbf{t}_i) | \mathbf{x}_i \in \mathbf{R}^n, \mathbf{t}_i \in \mathbf{R}^m, i = 1, \cdots, N$, activation function $g(x)$,and hidden neuron number $\overline{N}$

Step 1: Assign arbitrary input weight $\mathbf{w}_i$ and bias $b_i, i = 1, \cdots, \overline{N}$;

Step 2: Calculate the hidden layer output matrix $\mathbf{H}$;

Step 3: Calculate the output wight $\beta : \beta = H^{\dagger}\mathbf{T}$

Where $\mathbf{H}$,β and $\mathbf{T}$ are defined as formula (15) and (16).

4 Performance evaluation and discussion

Example 1: This simple example demonstrates that the neural network trained by the constrained data can produce a better approximation of the inverse kinematics function. An RBFN is used to approximate the inverse kinematics function of two-link manipulator. It consists of two revolute joints and two links that have the same length of 30mm. Two coordinate values x, y describe the position of the tip of the manipulator. The forward kinematics is

$$\begin{cases} x = l_1 cos\theta_1 + l_2 cos(\theta_1 + \theta_2) \\ y = l_1 sin\theta_1 + l_2 sin(\theta_1 + \theta_2) \end{cases} \tag{18}$$

The inverse kinematics can be described by

$$\theta_2 = Atan2(\pm\sqrt{1 - (\frac{x^2 + y^2 - l_1^2 - l_2^2}{2l_1 l_2})^2}, \frac{x^2 + y^2 - l_1^2 - l_2^2}{2l_1 l_2}) \tag{19}$$

$$\theta_1 = Atan2(y, x) - Atan2(l_2 sin\theta_2, l_1 + l_2 cos\theta_2) \tag{20}$$

Given a desired configuration of the end-effector, there are usually two desired true solutions which correspond with the lower-elbow structure and the upper-elbow structure respectively. We test the neural network with three different cases.

1. The training set randomly sample from the whole joint space.

2. The training set randomly sample from the constrained joint space which only consisted of the positive solution (+sign in (19)).

3. The training set randomly sample from the sub joint space.

All the simulation are carried out in Matlab 6.5 environment running in an Intel(R) Core(TM) 2 Duo CPU 3.00GHz Pc. The training process of the neural network can be executed using Matlab code "newrb". The root mean squared (RMS) error goal is defined by 0.001, and the number of training sets is 1000.

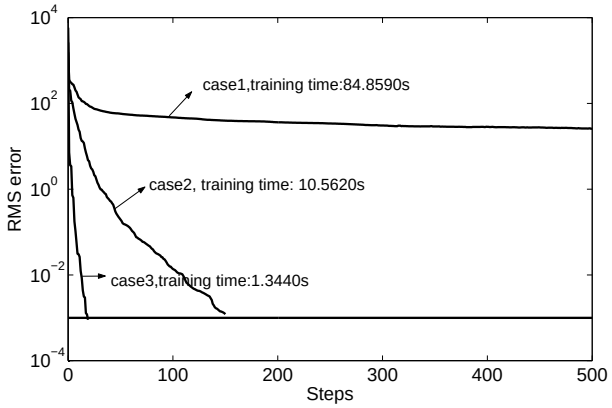


Figure 7: The training convergence performance

As can be seen in the figure 7, the same network trained by the constrained training data produces a better convergence performance.Moreover, the neural network trained within the sub joint space produces the best performance. For case 1, the training sets contain the many-to-one mapping from the joint space to the Cartesian space. It may be one reason that leads to training failure.

Example 2: In this example, the performance comparison of the new proposed ELM algorithm and the gradient-based learning algorithm has been conducted for an inverse kinematics of PUMA robot. The desired configuration of the end-effector is the same with section 2.3.

Test 1:Training the network with traditional algorithm

First, the approximate solution is calculated by EM. Set the $d\theta = 30°$, then one of the sub space is determined for each joint, as shown in table 2.

Table 2 One group of joint subspace

Joint Number	joint 1	joint 2	joint 3	joint 4	joint 5	joint 6
Subrange (degree)	[-20.1, 39.9]	[-8.7, 51.3]	[2.1, 62.1]	[2.9, 62.9]	[11.9, 71.9]	[37.6, 97.6]

Next, different training size, which is 100, 500 and 1000 respectively, sample randomly from the sub joint space. Other 500 data set is used for testing the performance of neural network. The root mean squared error goal and the maximum number of neurons are set as 10^{-8} and 500, respectively. And the spreads in three cases are experimentally selected as 3, 1.2 and 1.2 so that the RBFN can produce an appropriate performance. The training steps are repeated until the network's root mean squared error falls below goal or the maximum number of neurons are reached. Figure 8 shows the training convergence performance obtained by using different training size. The training time increases greatly with the number of the training data, as can be seen in the figure 8. Although the training error fail to reach the goal 10^{-8} using 500 and 1000 training data, all three trained networks are considered to achieve a good approximate performance. Since the training error successfully reach the 10^{-4} in three case, which is an accepted result for inverse kinematics. These conclusions can also be confirmed by the following results. Figure 9, 10 and 11 shows the testing root mean square (RMS) error at each joint angle using the corresponding networks trained above. It can be seen that the RMS error is very small. In addition, the network trained using 500 and 1000 size performs similarly. And the generalizations of both of them are better than the network trained using 100 data size. This occurs because less training data reduces the generalization of the network. However, taking into account training time, the network trained with less data size appears to be a better choice.

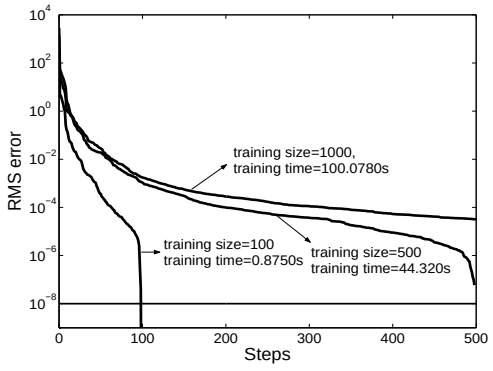


Figure 8: Network training convergence with different training size.

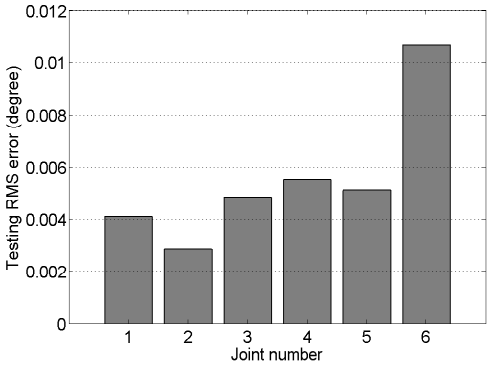


Figure 9: Absolute error at each joint angle using 100 training set

Test 2: Training the network with ELM

In this example, a single feedforward network with sigmoidal additive activation function is used. For ELM, the input weights and biases are randomly chosen from the range [-1, 1].To compare the results of ELM and gradient-based learning algorithm in test 1, two groups of tests use the same training/testing sets. Figure 12 shows the training RMS errors with different hidden nodes number in three cases. The corresponding testing RMS errors are plot in Figure 13. The average training time is 0.0014 s, 0.0056 s and 0.012 s, respectively. As observed from Figure 12 and 13, in general, the network trained using three

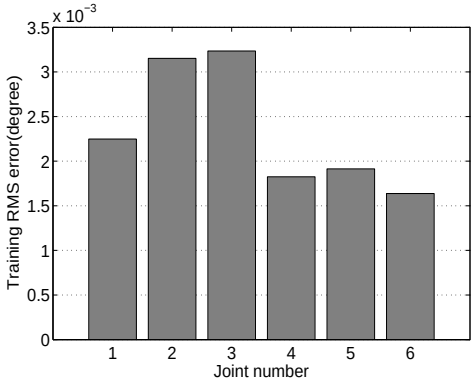


Figure 10: Absolute error at each joint angle using 500 training set.

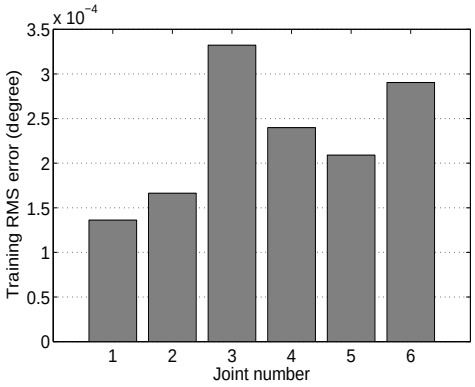


Figure 11: Absolute error at each joint angle using 1000 training set

groups of training data performs similarly. Furthermore, the lowest validation error is achieved when the numbers of hidden nodes are within the ranges [15, 50]. The results show that the generalization performance obtained by the ELM algorithm is very close to the generalization performance of gradient-based learning algorithm. However, the ELM algorithm can be simply conducted and runs much faster. According to our results, the average learning speed of ELM algorithm is at least 1000 times than the gradient-based learning algorithm.

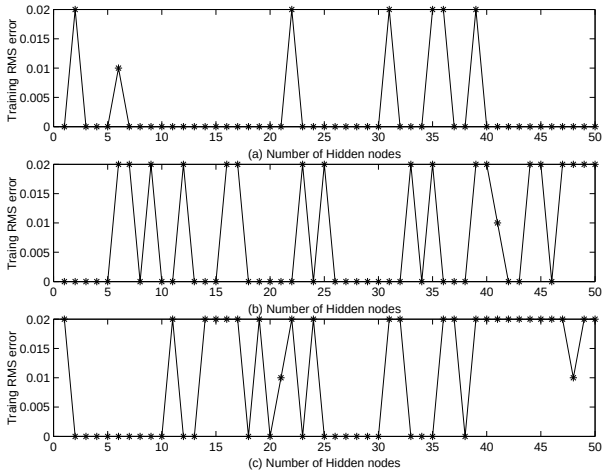


Figure 12: The training RMS error (degree) with ELM, the training size: (a) 100, (b) 500 and (c) 1000.

Example 3: This example demonstrates that the proposed method can be used for continuous joint space trajectory planning. The robot structure for this example is still based on PUMA 560 robot. The desired trajectory of the end-effector is a circle centered at (0.2, 0.05, 0.5) (m) with respect to the base coordinate frame and a radius equal to 0.2(m). The trajectory is discretized into 72 equally spaced points. To ensure the existence of solution, the joint limitations are released in this example. Moreover, noting that multiple solutions do exit, in order to prevent a sudden jump to another solution, a unique orientation is assigned and the approximate solution for each of the successive points is given by the solution of the preceding point. For example, if the calculated solution of the k point is denoted by θ_k, the joint variable limitations are set as $[\theta_k - d\theta, \theta_k + d\theta]$ for the k+1 point, instead of re-computing the approximate solution. The computed joint trajectories and the corresponding total error (sum of the position and orientation error) are plotted in Figure 14 and Figure 15, respectively. It should be noted

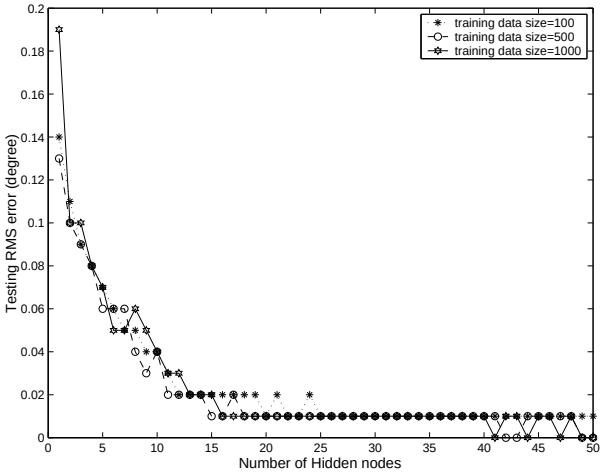


Figure 13: The testing RMS error with ELM.

that the trajectory in Figure 14 is just one of the multi-trajectory for PUMA robot.

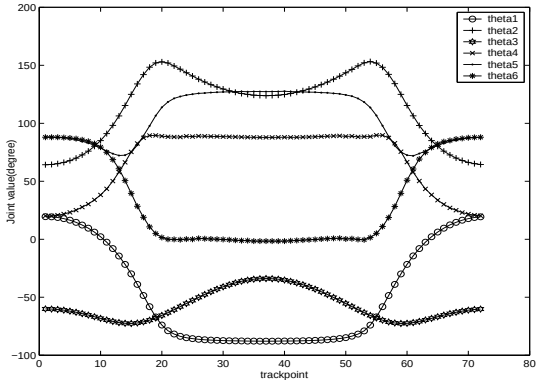


Figure 14: Computed trajectories of the joints.

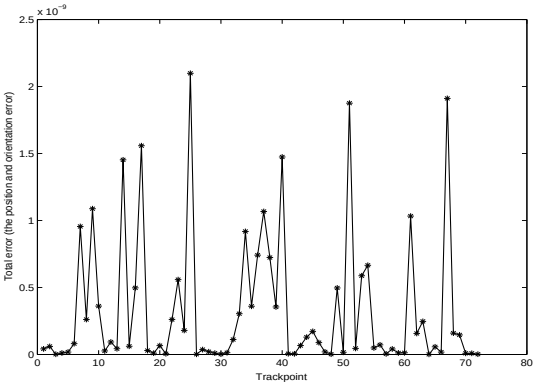


Figure 15: Total error at each track point

The results in Figure 15 show that the idea of using a neural network has produced an excellent approximation of the inverse kinematics function. Although neural network solutions are usually not suited for high precision robotic application, high precision results are achieved here. This occurs because the joint varies are limited within a small space when training network.

5 Conclusions

The proposed hybrid approach combined the electromagnetism-like method and the neural network to solve the inverse kinematics problem. Unlike the traditional neural network approaches that generate the training data from the whole joint space, the neural network in the proposed approach collects the training data from a sub joint space, in which the training set is constrained to only one solution set so that the one-to-one mapping is achieved. Another important feature of the proposed approach is to use an efficient learning algorithm, ELM, to train the neural network. The learning speed of this novel training algorithm can be thousands of times faster than traditional feedforward network learning algorithms while obtaining better generalization performance. The results show that the proposed hybrid approach has not only greatly reduced the computation time but also improved the precision.

Bibliography

[1] Bruno Siciliano, Oussama Khatib, *Springer Handbook of robotics*,Springer Press,2008.

[2] H JACK, DMA LEE, RO BUCHAL and WH ELMARAGHY, Neural networks and the inverse kinematics problem, *Journal of intelligent manufacturing*,4:43-66,2003.

[3] FL Lewis,Neural network control of robot manipulators,*IEEE Expert*,11(3):64-75,1996.

[4] BB Choi and C Lawrence,Inverse kinematics problem in robotics using neural networks,NASA Technical Memorandum-105869.

[5] Z Binggul, HM Ertunc and C Oysu, Comparison of inverse kinematics solutions using neural network for 6R robot manipulator with offset,*In Proceedings of the 2005 Congress on Computational Intelligence Method and Application*,pp:1-5.

[6] AS Morris , A Mansor,Finding the inverse kinematics of manipulator arm using artificial neural network with look-up table. *Robotica*,15:617-625,1997.

[7] JA Driscoll,Comparison of neural network architectures for the modeling of robot inverse kinematics,*In Proceedings of the 2000 IEEE*,3:44-51,2000.

[8] SS Yang, M Moghavvemi and John D Tolman,Modelling of robot inverse kinematics using two ANN paradigms, *In Proceedings of TENCON2000 Intelligent System and Technologies for the New Millennium*,3:173-177,2000.

[9] Shital S, Chiddarwar N and Ramesh Babu,Comparison of RBF and MLP neural networks to solve inverse kinematic problem for 6R serial robot by a fusion approach,*Engineering Applications of Artificial Intelligence,*, 23(7):1083-1092,2010.

[10] PY Zhang, TS Lu and LB Song, RBF neworks-based inverse kinematics of 6R manipulator,*Int. Journal of advanced manufacturing technology*,26:144-147,2004.

[11] Eimei Oyama, Arvin Agah and Karl F, A modular neural architecture for inverse kinematics model learning,*Neurocomputing*, 38(40):797-805,2001.

[12] Srinivasan Alavandar, MJ Nigam, Neuro-Fuzzy based approach for inverse kinematics solution of industrial robot manipulators, *Int. J. of computers, Communication and Control*, 3(3):224-234,2008.

[13] Karlra P, Prakash NR, A neuro-genetic algorithm approach for solving inverse kinematics of robotic manipulators, *IEEE International Conference on Systems, Man and Cybernetics*,2:1979-1984,2003.

[14] Guang-Bin Huang, Qin-Yu Zhu, Chee-Kheong Siew, Extreme learning machine: A new learning scheme of feedforward neural networks, *In Proceedings of IEEE International joint conference on Neural Networks*,2:985-990,2004.

[15] Guang-Bin Huang, Lei Chen, Enhanced random search based incremental extreme learning machine,*Neurocomputing*, 71(16-18):3460-3468,2008.

[16] Birbil SI, Fang SC, An electromagnetism-like mechanism for global optimization,*Journal of Global Optimization*, 23(3):263-282,2003.

[17] Birbil SI, Fang SC, Sheu RL, On the convergence of a population-based global optimization algorithm,*Journal of global optimization*, 30:301-318,2004.

[18] Wang LCT, Chen CC, A combined optimization method for solving the inverse kinematics problem of mechanical manipulators,*IEEE Transaction on Robotics and Automation*, 7(4):489-499,1991.

INT J COMPUT COMMUN, ISSN 1841-9836
Vol.7 (2012), No. 3 (September), pp. 473-481

Optimal Bitstream Adaptation for Scalable Video Based On Two-Dimensional Rate and Quality Models

J. Hou, S. Wan

Junhui Hou, Shuai Wan
Northwestern Polytechnical University
School of Electronics and Information
Xi'an 710129, China
E-mail: houjunhuihn@gmail.com
swan@nwpu.edu.cn

Abstract: In this paper, a two-dimensional (2D) rate model is proposed considering the joint impact of spatial (i.e., the frame size) and SNR (i.e., the quantization step) resolutions on the overall rate-distortion performance. A related 2D quality model is then proposed in terms of perceptual quality. Then the two proposed models are applied to scalable video to address the problem of optimal bitstream adaptation. Experimental results show that the proposed rate and quality models fit the actual data very well, with high coefficients of determination and small relative root mean square errors. Moreover, given the bandwidth constraint and required display resolution of the end users, the optimal combination of SNR and spatial layers that provides the highest perceptual quality can be achieved using the proposed models.
Keywords: 2D rate model, 2D perceptual quality model, scalable video, bitstream adaptation.

1 Introduction

Recent multimedia applications are featured by various resolutions designed for a variety of devices with different computational and display capabilities. These devices range from cell phones and PDA's with small screens and restricted processing power to high-end work stations with high-definition displays. The related video services or applications are connected to different types of networks with various bandwidth limitation and loss characteristics. A highly attractive approach to address the vast heterogeneity is known as scalable video, which allows for spatial, temporal, and SNR scalabilities [1]. In Scalable Video Coding (SVC), the video signal can be encoded into a Base Layer (BL) and one or more Enhancement Layers (ELs), with each enhancement layer improving the resolution (either temporally or spatially) or the fidelity of the video sequence. As a result, certain parts of the scalable bitstream can be removed for adaptation to various capabilities of end users as well as varying network conditions.

At the network proxy or gateway, a bitstream adaptor is usually employed to extract the bitstream to meet particular constraints, e.g., targeted bit-rates and/or spatial or temporal resolutions. For a given set of constraints, the solution can be varieties of resolution combinations, leading to different visual qualities. The challenging problem of bitstream adaptation is therefore how to determine the combination of the spatial resolution (i.e., the frame size (s)), temporal resolution (i.e., the frame rate (t)) and SNR (Signal-to-Noise Ratio) resolution (i.e., the quantization step (q)) to be used for bitstream extraction under a given targeted bit-rate to maximize the resulting quality.

Many efforts have been devoted to bitstream adaptation for scalable video. A basic and content independent extractor is provided in the reference software of the Joint Scalable Video Model (2) [2]. In [3], an alternative extraction method is proposed based on rate-distortion optimization. This technique utilizes the concept of quality layers and improves the performance of the JSVM basic extractor by arranging the priority of layers based on their contributions to the global improvement in quality. A more

efficient method for extraction is proposed in [4], using an accurately and efficiently estimation of the quality degradation resulting from discarding an arbitrary number of Network Abstraction Layer (NAL) units from multiple layers taking drift into account. However, the methods in [3] and [4] are executed only within a single resolution, e.g., the SNR plane. In [5], an effective method is proposed to quickly solve the problem of spatial resolution selection based on an analysis on the content information. However, the Peak-Signal-to-Noise Ratio (PSNR) is used as the distortion criterion, which does not correlate well with the perceptual quality especially with regard to spatial scalability. In [6], two-dimensional (2D) rate and perceptual quality models in terms of the frame rate and the quantization step are built, and then the two models are applied to optimal bitstream extraction in SVC. However, the spatial resolution is not considered in [6] and the parameters in the quality model are difficult to obtain. In [7], the video quality under different spatial, temporal and SNR combinations is quantitatively and perceptually assessed, based on which an efficient adaptation algorithm is proposed. However, there is lack of a rate model to estimate the related bit-rate. On the other hand, performance improvement can be achieved by resorting to network-related technologies, such as using a priority mechanism [8], or self optimization of networked communications [9] presents a model for self optimization of network communications in order to improve cluster performance by shortening the data transfer time.

In this paper, 2D rate and quality models are proposed for optimal bitstream adaptation for scalable video under given bandwidth constrains and required display resolutions at the end user. Assuming that the frame rate is determined, the two 2D models are applied to extract bitstream to achieve the optimal combination of spatial and SNR resolutions.

The rest of the paper is organized as follows: Section 2 presents the 2D rate and perceptual quality models considering the impact of spatial and SNR resolutions. Their application in constrained scalable video adaptation is introduced in Section 3. Section 4 presents the experimental results. Section 5 concludes this paper and discusses future directions.

2 Two-Dimensional Rate and Perceptual Quality Models

In this section, the impact of the spatial and SNR resolutions on the bit-rate and the perceptual quality is analyzed, based on which a 2D rate model and a 2D perceptual quality model are respectively derived.

2.1 Two-Dimensional Rate Model

Considering SNR and temporal scalabilities, we have proposed an analytical 2D rate model for H.264/SVC [10]. In this paper, this model is extended to the spatial domain where a product of a power function of the quantization step q and a power function of the spatial resolution index s are used, given as

$$R(q, s) = cq^{\alpha} s^{\gamma}, \tag{1}$$

where α and γ are both content-dependent model parameters. The values of α and γ characterize how fast the bit-rate reduces with the increase of q and how fast the bit-rate increases with the refinement of the spatial resolution, respectively. Usually a sequence with richer texture has larger absolute values of α and γ. Here s is computed through dividing the frame size of the current spatial resolution by the frame size of the lowest spatial resolution. In order to evaluate the model accuracy, sequences, with QCIF, CIF and 4CIF resolutions were encoded into 3 dyadic spatial layers using JSVM9.19.7 [11], respectively. Each spatial layer contained 5 quality layers. The base quality layer in a lower spatial layer was used to perform inter-layer prediction to avoid drifting at the decoder. The GOP (Group of Picture) size was set to 1 to avoid the effect of temporal scalability. 120 frames were encoded for each sequence. The difference of the quantization parameter (QP) between adjacent quality layers and adjacent spatial layers

were set to 5 and 6, respectively, following [12]. The QP of the base quality layer of the lowest spatial level was set to 38. The model parameters were obtained by minimizing the Root Mean Square Error (RMSE) between the actual and predicted bit-rates. The actual values and those predicted using (1) are plotted in Figure 1. It is clear that the proposed 2D rate model fits the actual data very well. Table 1 gives the used parameters and the model accuracy in terms of the RRMSE ($RRMSE = RMSE/R_{max}$, where R_{max} denotes the maximum bit-rate in the actual data) and the Coefficient of Determined (CoD), defined as:

$$CoD = 1 - \frac{\sum_i \left(X_i - \widehat{X_i}\right)^2}{\sum_i \left(X_i - \overline{X}\right)^2}, \tag{2}$$

where X_i and $\widehat{X_i}$ are the actual and the predicted values of the bit-rate, respectively, and $\overline{X}$ is the mean of all actual bit-rates shown in Figure 1. It is once again demonstrated that the proposed rate model is very accurate in prediction, where high CoD and small RRMSE values can be observed for all tested sequences. Specifically, the average CoD and RRMSE are 0.9892 and 2.81%, respectively. And as expected, the “City" sequence with the richest texture among the tested sequences has the largest absolute values of α and γ.

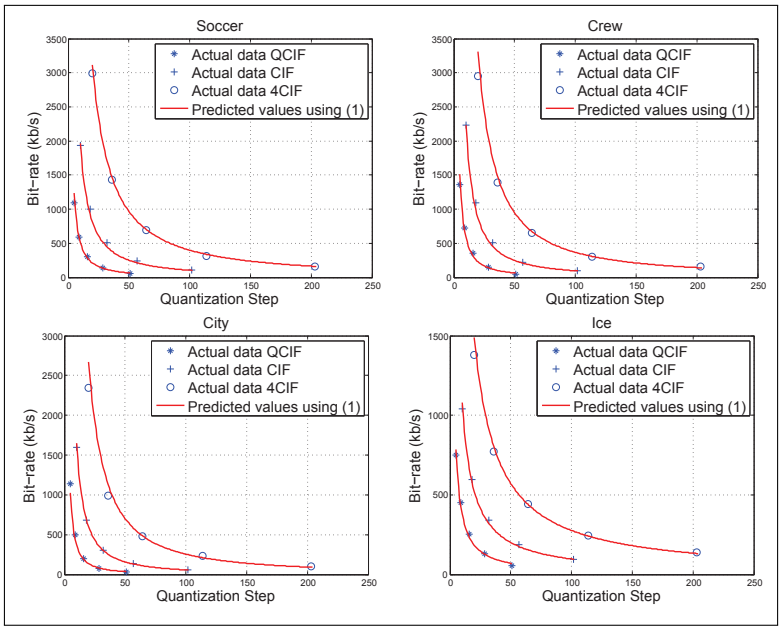


Figure 1: Actual bit-rates and predicted values using (1).

Table 1: The values of parameters in (1) and model accuracy

	Soccer	Crew	City	Ice	Ave.
$c^{\times 10^3}$	9.67	13.58	10.76	4.23	
α	-1.276	-1.365	-1.462	-1.048	
γ	0.970	0.965	1.078	0.756	
CoD	0.9958	0.9860	0.9792	0.9929	0.9892
RRMSE	1.71%	3.31%	3.97%	2.27%	2.81%

2.2 Two-Dimensional Quality Model

It has been widely acknowledged that the quality metrics of the PSNR and the Mean Square Error (MSE) do not correlate well with the perceptual quality. On the other hand, the subjective quality can be well captured by the Mean Opinion Scores (MOS) and Video Quality Metric (VQM) [13], at the cost of high complexity in testing and computations. Trading off between the complexity and the consistency with the human perception, the Structural Similarity (SSIM) [14] is used as the quality measure in this paper.

The SSIM measures the structural similarity as well as the luminance and contrast similarity between two images block by block. In this paper, the SSIM values have been measured with regard to different combinations of spatial and SNR resolutions, where the layers of lower spatial resolutions were up-sampled to 4CIF using a set of 6-taps filters provided by the JSVM. According to empirical observations, a logarithmic function in terms of the spatial resolution index and the quantization step is used to model the perceptual quality regarding different spatial and SNR resolutions, which is expressed as

$$QM_{ssim}(q, s) = a_0 + a_1 \ln q + a_2 \ln s + a_3 \ln q \ln s, \tag{3}$$

where a_0, a_1, a_2 and a_3 are all content-dependent model parameters. Here the second and third terms indicates the impact of the SNR and the spatial resolution on perceptual quality, respectively. The forth term models the joint impact of the SNR and the spatial resolution. The model parameters can be derived easily by minimizing the RMSE between the actual and predicted values. The actual and predicted qualities are shown in Figure 2. Table 2 lists the used parameters and the model accuracy in terms of the CoD and RRMSE. It can be concluded from Figure 2 and Table 2 that the proposed 2D perceptual quality model is very accurate in prediction with high CoD and small RRMSE values for all tested sequences.

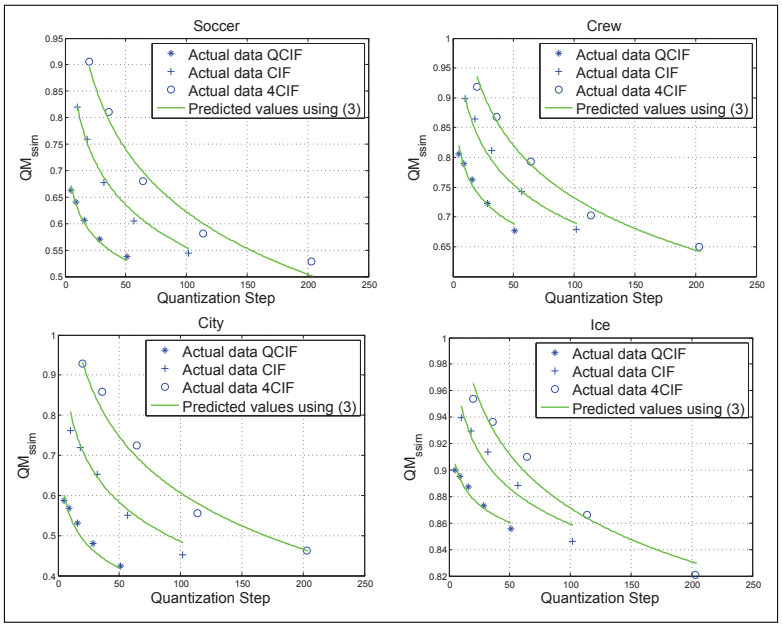


Figure 2: Actual qualities and predicted values using (3).

Table 2: The values of parameters (3) in and model accuracy

	Soccer	Crew	City	Ice	Ave.
a_0	0.7709	0.9112	0.7268	0.9395	
a_1	-0.0614	-0.0569	-0.0791	-0.0193	
a_2	0.2292	0.1454	0.2918	0.0739	
a_3	-0.0393	-0.0251	-0.0444	-0.0141	
CoD	0.9871	0.9842	0.9754	0.9561	0.9757
RRMSE	1.39%	1.11%	2.49%	0.8%	1.45%

3 Optimal Bitstream Adaptation for Scalable Video Using Proposed Models

The proposed models are applied to constrained bitstream adaptation for scalable video. Figure 3 provides a systematical view of the adaptation problem. For each video, a single full-resolution scalable bitstream is available at a server, where the bitstream will be adapted at a network proxy or gateway according to the user channel conditions and viewing preferences (i.e., the displayed spatial resolution). When a user requests the video from the server, the adaptor (at the proxy) will determine an appropriate bit-rate R_t for extraction based on the channel condition. Based on R_t and the user's settings of viewing preference (embedded in the user profile and sent to the adaptor), the adaptor determines the optimal set of spatial and SNR layers to extract, so as to provide the best perceptual quality.

For a given targeted bit-rate R_t and the required display spatial resolution, the adaptation problem can be formulated as the following constrained optimization problem:

$$\begin{aligned} \text{Determine } q, s \text{ to } & \text{ maximize } \ QM_{ssim}(q, s) \\ & \text{subject to } \ R(q, s) \le R_t \\ & \mathbf{U}(s)|s < S, \end{aligned} \tag{4}$$

where R_t and S denote the targeted bit-rate and the required display spatial resolution index, respectively. By $\mathbf{U}(s)|s < S$ it is indicated that up-sampling is executed if the extracted spatial resolution is less than the required display spatial resolution.

Assume that both the spatial resolution and the quantization step may take on any effective value. By setting $R(q, s) = R_t$, it can be obtained that

$$q = \sqrt[\alpha]{\frac{R_t}{cs^\gamma}}, \tag{5}$$

which describes the feasible q for a given s, to satisfy the rate constraint R_t. Substituting (5) into (3) yields

$$QM_{ssim}(s) = -\frac{a_3\gamma(\ln s)^2}{\alpha} + (a_3 \ln R_t/c + \alpha a_2 - a_1\gamma)\frac{\ln s}{\alpha} + a_0 + \frac{a_1 \ln R_t/c}{\alpha}. \tag{6}$$

Equation 6 is the achievable quality with different spatial resolutions under the targeted bit-rate R_t. Clearly, this function has a unique maximum, which can be derived by setting its first order derivative with respect to s to be zero. This yields

$$s = e^{(a_3 \ln (R_t/c) + \alpha a_2 - a_1\gamma)/2a_3\gamma}. \tag{7}$$

For any given R_t and S, we can solve (7) numerically to determine the optimal spatial resolution. Then using (5) and (6) the optimal quantization step can be determined, and the corresponding quality can be maximized. The parameters for the rate model, i.e., c, α and γ, can be easily derived from the bit-rates corresponding to several different (q, s)combinations using least square fitting. The quality model parameters, i.e., a_0, a_1, a_2 and a_3, can be derived using the least square fitting at the encoder, and then embedded in the header field of the video stream. Based on the simulations, only several bytes are required to represent those parameters which can be neglected compared to the actual video stream payload.

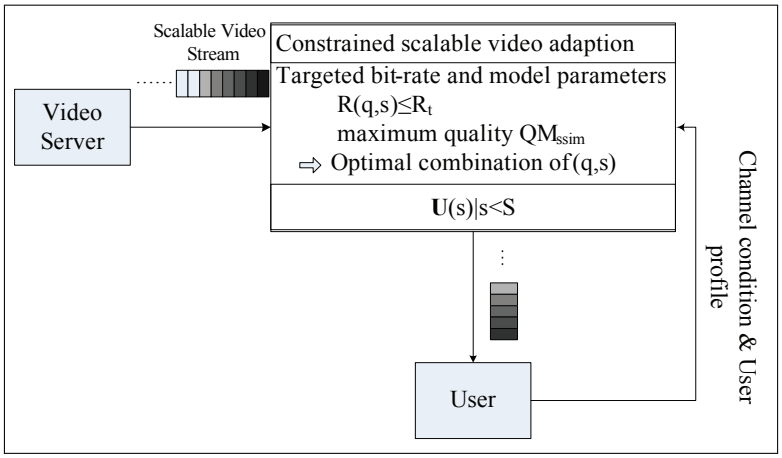


Figure 3: Constrained scalable video adaptation.

4 Experimental Results

The experimental results are presented in this section to evaluate the performance of the proposed extraction method. Firstly, assuming that the spatial resolution can be any positive values, and then the practical case where spatial resolutions to be discrete is considered.

4.1 Optimal Solutions Assuming q and s Taking Continuous Values

Assuming that both the spatial resolution and the quantization step can take continuous values. Figure 4 shows the optimal spatial resolution, quantization step and quality as functions of the targeted bit-rate R_t. As expected, as the targeted bit-rate increases, the optimal s increases while the optimal q reduces, and the achievable best quality continuously improves. Notice that the optimal s increases more rapidly for the “City" sequence than for the other sequences because of its richer texture. The up-sampling introduces more severe quality decrease than the quantization step. Therefore, under the bit-rate constraint, a larger spatial resolution with a larger quantization step is a better choice.

4.2 Optimal Solutions Under Dyadic Spatial Resolution Scalability

The H.264/SVC includes three profiles [15], i.e., the “Scalable Baseline" profile, the “Scalable High" profile, and the “Scalable High Intra" profile. While the latter two profiles support full spatial SVC scalability, the Scalable Baseline profile imposes some constraints to enable simplified application scenarios. For example, dyadic spatial scalability is provided in the baseline profile, where the scaling ratio of the width and height between adjacent spatial layers is equal to 2. From a practical point of view, it will be interesting to see the optimal combination of the spatial resolution and quantization step for different

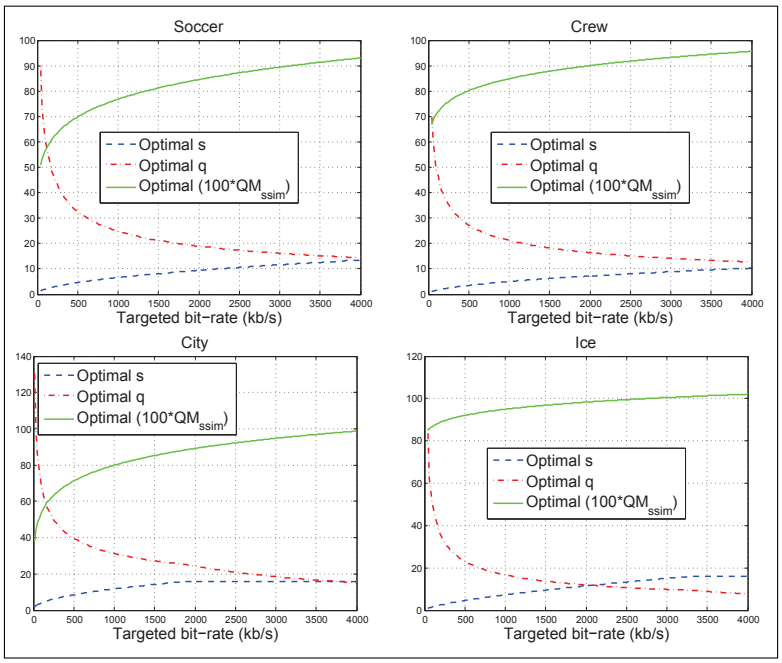


Figure 4: Optimal quantization step, spatial resolution index and the corresponding quality versus the targeted bit-rate by assuming the quantization step and the spatial resolution to be continuous.

targeted bit-rates under this SVC structure. To obtain the optimal solution for this SVC structure, we first determine the optimal s using (7), and then choose two spatial resolutions up and down around the value from the candidates. Finally, compute the quality using (6) corresponding to the two spatial resolutions and choose the spatial resolution that leads to a better quality.

The experimental results are shown in Figure 5. Because the spatial resolution can only increase in a discrete step, the optimal quantization step does not decrease monotonically with the bit-rate. Whenever the optimal s jumps to the next higher value, the optimal q first increases to meet the rate constraint, and then decreases while the optimal s is held constant, as the rate increases. Consistent with the previous results in Figure 4, for the “City" sequence with richer texture, the optimal s is 16 (corresponding to 4CIF) at a low bit-rates, whereas for other sequences, the optimal s stays 4 (corresponding to CIF) even at high bit-rates.

In practice, the SVC encoder with quality scalability does not allow the quantization step to change continuously. The finest granularity in quality scalability is a decrement of QP by 1 with each additional quality layer. This means that the quantization step reduces by a factor of $2^{-1/6}$ with each additional layer. In practice, much coarser granularity is typically used, with a decrement of QP by 3 to 6 typically [11]. When we limit the values of q to be discrete in addition to allow only dyadic spatial resolutions, a rate constraint cannot be always exactly met. However, one may still obtain the optimal q and s for any given constrains using the proposed scheme by estimating the bit-rate and quality of each combination in the finite set of feasible values for q and s.

5 Conclusions and Future Works

In this paper, a 2D rate model and a 2D quality model have been proposed, based on which a model-driven method for optimal bitstream adaptation is developed. Experimental results have demonstrated the accuracy of the two models. Using the proposed extraction method, the optimal combination of quality and spatial layers can be determined, providing the highest perceptual quality for a given bandwidth

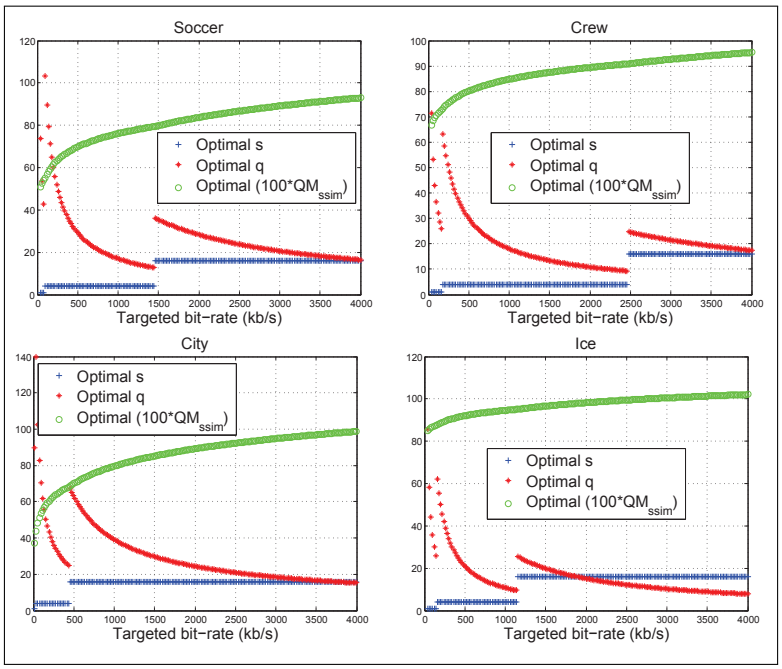


Figure 5: Optimal quantization step, spatial resolution index and the corresponding quality versus the targeted bit-rate by assuming that the q varies continuously and the spatial resolution takes QCIF/CIF/4CIF.

constraint and required display frame rate of the end user.

Future work may include an extension of the proposed models to three-dimension, taking temporal scalability into account. Moreover, the proposed models can be applied to other applications, e.g., advanced multidimensional rate control for video coding.

Acknowledgement

This work was supported by the National Science Foundation of China (60902052, 60902081), the Doctoral Fund of Ministry of Education of China (No.20096102120032), and the NPU Foundation for Fundamental Research (JC201038).

Bibliography

[1] H. Schwarz, D. Marpe, T. Wiegand, Overview of The Scalable Video Coding Extension of The H.264/AVC Standard, *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 17, No. 9, pp.1103-1120, 2007.

[2] J. Reichel, H. Schwarz, and M. Wien, Joint Scalable Video Model 11 (JSVM 11), *Joint Video Team, Doc. JVT-X202*, 2007.

[3] I. Amonou, N. Cammas, S. Kervadec, S. Pateux, Optimized Rate Distortion Extraction With Quality Layers in The Scalable Extension of H.264/AVC, *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 17, No. 9, pp.1186-1193, 2007.

[4] Ehsan Maani, Aggelos K. Katsaggelos, Optimized Bit Extraction Using Distortion Modeling in the Scalable Extension of H.264/AVC, *IEEE Trans. Image Process.*, Vol. 18, No. 9, pp.2022-2029, 2009.

[5] Yu Wang, Lap-Pui Chau, Kim-Hui Yap, Spatial Resolution Decision in Scalable Bitstream Extraction for Network and Receiver Aware Adaptation, *Proceedings of the 2008 IEEE International Conference on Multimedia and Expo*, pp.577-580, 2008.

[6] Y. Wang, Z. Ma, Y.-F. Qu, Modeling Rate and Perceptual Quality of Scalable Video as Function of Quantization and Frame Rate and Its Application in Scalable Video Adaptation, *Proceedings of the IEEE 17th Packet Video Workshop*, pp.1-9, 2009.

[7] Guangtao Zhai, Jianfei Cai, Weisi Lin, Xiaokang Yang, Wenjun Zhang, Three Dimensional Scalable Video Adaptation via User-End Perceptual Quality Assessment, *IEEE Trans. Broadcasting*, Vol.54, No.3, pp.719-728, 2008.

[8] A. Rahim, Z.S. Khan, F.B. Muhaya, M. Sher, M.K. Khan, Information Sharing in Vehicular AdHoc Network, *Int. J. of Computers, Communications and Control*, 5(5):892-899, 2010.

[9] A. Rusan, C.-M. Amarandei, A New Model for Cluster Communications Optimization, *Int. J. of Computers, Communications and Control*, 5(5):910-918, 2010.

[10] Junhui Hou, Shuai Wan, Fuzheng Yang, "Frame Rate Adaptive Rate Model for Video Rate Control," *Proceedings of the 2010 IEEE International Conference on Multimedia Communication*, pp.226-229, 2010.

[11] H.264 SVC Reference Software (JSVM 9.19.7) and Manual CVS sever, JVT, 2010 [Online]. Available: garcon.ient.rwth-aachen.de.

[12] Xiang Li, Peter Amon, Andreas Hutter, Andr Kaup, Performance Analysis of Inter-Layer Prediction in Scalable Video Coding Extension of H.264/AVC, *IEEE Trans. Broadcasting*, Vol. 57, No. 1, pp66-74, 2011.

[13] M. Pinson, S. Wolf. A New Standardized Method for Objectively Measuring Video Quality, *IEEE Transactions on Broadcasting*, Vol. 50, No.3, pp.312-322, 2004.

[14] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, Image Quality Assessment: From Error Visibility to Structural Similarity, *IEEE Trans. Image Process.*, Vol. 13, No. 4, pp.600-612, 2004.

[15] T. Wiegand, G. J. Sullivan, J. Reichel, H. Schwarz, M. Wien, Eds., Amendment 3 to ITU-T Rec. H.264 (2005) | ISO/IEC 14496-10:2005, Scalable Video Coding, 2007.

INT J COMPUT COMMUN, ISSN 1841-9836
Vol.7 (2012), No. 3 (September), pp. 482-493

A Fast and Scalable Re-routing Algorithm based on Shortest Path and Genetic Algorithms

J. Lee, J. Yang

Jungkyu Lee
Cyram Inc., 516 Seoul National University Research Park
Naksungdae-dong, Gwanak-gu, Seoul 151-919 Koera
E-mail: jklee@cyram.com

Jihoon Yang
Department of Computer Science, Sogang University
1 Sinsu-dong, Mapo-gu, Seoul 121-742 Koera
E-mail: yangjh@sogang.ac.kr

Abstract: This paper presents a fast and scalable re-routing algorithm that adapts to dynamically changing networks. The proposed algorithm, DGA, integrates Dijkstra's shortest path algorithm with the genetic algorithm. Dijkstra's algorithm is used to define the predecessor array that facilitates the initialization process of the genetic algorithm. Then the genetic algorithm keeps finding the best routes with appropriate genetic operators under dynamic traffic situations. Experimental results demonstrate that DGA produces routes with less traveling time and computational overhead than pure genetic algorithm-based approaches as well as Dijkstra's algorithm in large-scale routing problems.
Keywords: Evolutionary algorithm, routing in dynamic networks, car navigation system.

1 Introduction

The car navigation system has become a very useful tool for many drivers. When a driver turns on a car navigation system and inputs where he or she wants to go, the system searches the map and finds the best route (e.g. shortest path) to the destination. Recently, in addition to such a basic functionality, car navigation systems are equipped with real-time traffic information services like TPEG (Transport Protocol Experts Group) [1–6]. Here, the navigation system is provided with the traffic information on current road conditions, with which it re-computes the best route with minimal expected travel time. Unfortunately, such traffic information is not truly real-time but delivered from a central server at certain intervals. In addition, updating the entire map with the new information delivered from the server causes an exorbitant overhead. In this paper, we propose a novel approach to deal with these problems and to produce the best route dynamically. Our algorithm integrates Dijkstra's shortest path algorithm [7] with a genetic algorithm [8], and thus named DGA. The former is for incorporating useful prior knowledge on the network (e.g. distance between two places) and facilitating the initialization process of the genetic algorithm, and the latter is for finding the best routes. (Detailed descriptions on DGA will be given in Section 3.) DGA re-computes the routes quickly whenever new real-time traffic information is available. A car is assumed to send the traffic information (e.g. its speed) to the vehicles it meets during the trip via wireless communication. This direct and local communication among vehicles provides genuine real-time information and obviates the use of the expensive central server.

This paper is organized as follows: Section 2 briefly introduces a representative genetic algorithm-based approach to the shortest path problem proposed by Ahn [9] which will be compared with DGA. Section 3 describes DGA. Section 4 presents the results of the experiments designed to evaluate the performance of DGA. Section 5 concludes with a summary and future research directions.

2 Related Work: Genetic Algorithm for Shortest Path Problem

The genetic algorithm (GA) is one of the global search heuristics inspired from biology and has been successfully applied to a variety of optimization problems [8, 10]. A great deal of research on GA-based shortest path search has been carried out in various communication network applications [9, 11–15], among which [9, 14, 15] are related to car navigation systems. Ahn's method [9] is one of the most representative applications of GA to the shortest path routing problem. However, though Ahn's method was able to find a good solution with solid theoretical results, it worked only for moderate-sized networks. In fact, our experiments with the algorithm failed to produce solutions within a reasonable period of time for networks with more than 10,000 nodes. Considering real-world networks where there exist huge number of nodes (or places), Ahn's approach is thus far from applicable. There exists another GA-based approach for the car navigation system proposed by Kanoh [14, 15]. Kanoh's approach is similar to DGA in that it computes routes by considering dynamic road conditions and initializing the population using Dijkstra's algorithm. However, the motivation of their algorithm is quite different from ours and not fully comparable: They use GA for improving the quality of solution in terms of multi-objective criteria (e.g. traveling time, route length, number of signals, number of right turns, etc.), whereas our algorithm focuses on re-routing. In addition, Kanoh's approach was evaluated only with small networks (of less than 20,000 nodes) though the computational overhead was claimed to be low.

The main contribution of this paper is the design of an efficient algorithm that can be deployed in a car navigation system and be used frequently for re-routing in a large-scale network only with locally-transmitted traffic information. To the best of our knowledge, there does not exist a GA-based algorithm to the shortest path problem that can cope with dynamic situations in a huge network. Since DGA has characteristics common with Ahn's GA, we briefly introduce the method here. (See [9] for detailed descriptions.)

2.1 Genetic Representation

A chromosome (representing a candidate solution path) is variable-length and consists of the sequence of positive integers that represent the IDs of nodes through which a routing path passes. The gene at the first and the last loci are reserved for the source and the destination nodes, respectively.

2.2 Population Initialization

The chromosomes are initialized randomly. Starting from the source, a chromosome encodes a routing path by successively selecting the next node at random among the neighboring nodes that are linked to the current node. Note that the chance for generating a valid path is fairly slim due to the randomness, which makes the approach infeasible for large networks, as verified in Section 4.

2.3 Fitness Function

The fitness function of the ith chromosome, f_i, is defined as

$$f_i = \left(\sum_{i=1}^{m_i} C(g_i(j), g_i(j+1))\right)^{-1} \tag{1}$$

where m_i is the length of the chromosome, $g_i(j)$ represents the gene at the jth locus, and $C(g_i(j), g_i(j+1))$ is the link cost between node $g_i(j)$ and $g_i(j+1)$.

2.4 Selection

The tournament selection without replacement is used. In other words, non-overlapping random sets of 2 chromosomes are chosen from the population, and the chromosome with higher fitness was selected from each set to survive in the next generation.

2.5 Crossover

The concept of crossover is depicted in Fig. 1. First, the crossover points are determined by randomly choosing a common gene appearing in both parent chromosomes. Then the chromosomes are interchanged with respect to the crossover points and the offsprings are generated.

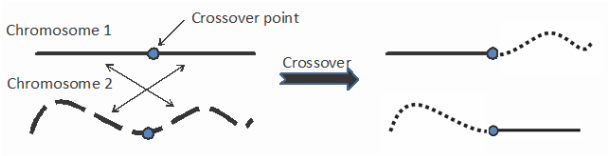


Figure 1: The concept of crossover.

2.6 Mutation

Typically, GA performs mutation by changing or flipping the genes in the candidate chromosome, thereby maintaining the genetic diversity. Here, the mutation operation attempts to maintain the diversity in the population by modifying the current path represented by a chromosome. For a chromosome, a gene is randomly selected as a mutation point. Starting from the mutation point, a sequence of neighboring nodes are randomly chosen to define a complete path (i.e. until the node chosen last is the destination). The concept of mutation is depicted in Fig. 2.

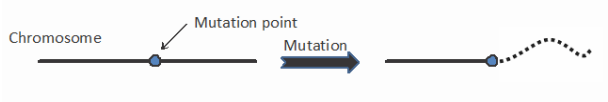


Figure 2: The concept of mutation.

2.7 Repair Function

Note that a chromosome produced by the crossover operation may contain a loop in the path it represents, which is an invalid solution. To make the path valid, the repair function was proposed. As shown in Fig. 3, the repair function eliminates a loop by finding the intersection (or repeated) node and removing the intermediate nodes in the loop.

Figure 3: The concept of repair function.

For example, assume that the following chromosome is produced:

$$1, 2, 3, 4, 5, 6, 3, 7, 8$$

Then the repair function finds (and keeps a single occurrence of) the intersection node 3 and removes the intermediate nodes 4, 5 and 6. The resulting chromosome is:

$$1, 2, 3, 7, 8$$

2.8 The overall algorithm

Now, Ahn's GA for finding the shortest path is described in Algorithm 1.

Algorithm 1. Ahn's GA
1: Initialize the population;
2: **repeat until** convergence
3: Calculate the fitness of individuals in the population;
4: Do selection;
5: Do crossover;
6: Remove loops by repair function;
7: Do mutation;
8: **end**

The condition for the convergence of the algorithm is if all chromosomes are identical.

3 DGA

We describe our algorithm, DGA, in this section. The purpose of DGA is to adapt to the dynamically changing networks and to re-route the shortest path fast. As aforementioned, DGA inherits the characteristics of both Dijkstra's shortest path algorithm and GA. The former is to initialize the population in the latter with meaningful candidate solutions (i.e. paths) instead of random ones. For instance, the chromosomes can be generated based on useful information such as the distance or average vehicle speed between two nodes. Among the various data structures used for Dijkstra's algorithm, the overflow bag introduced by Cherkassky *et al.* [16] was adopted in our work. (Cherkassky *et al.* had developed the overflow bag to reduce the memory requirement of Dijkstra's algorithm with the bucket data structure proposed by Dial [17].) Instead of Dijkstra's algorithm, any single-source shortest path algorithm (e.g. Bellman-Ford algorithm [18]) can be also used for DGA.

3.1 Population Initialization

The random population initialization of Ahn's GA does not work well for large-scale networks since the chance for generating invalid paths becomes very high as explained in Section 2. To overcome this problem, DGA makes use of Dijkstra's algorithm and produces a *predecessor array* as described in Fig. 4. First, from the start (source) node, the shortest paths to all the other nodes including the goal (destination) node are computed by Dijkstra's algorithm. Then, for the shortest path from an arbitrary node a to the goal node, all the links on the path are stored in the form of a (*direct*) predecessor array *pred* which is s sequence of nodes constituting the path in a reverse order (i.e. from the goal to a). Fig. 4(a) shows an example of *pred*. Once *pred* is constructed, the shorted path from the goal to a can be easily obtained by a call $GetPath(pred, goal, a)$ defined as follows, which makes fast initialization of the population possible:

Subroutine 1. GetPath($pred$, x, y)
// Compute the path from x to y using $pred$.
1: Set current node $s_{cur} = x$ and $path = [s_{cur}]$;
2: **repeat**

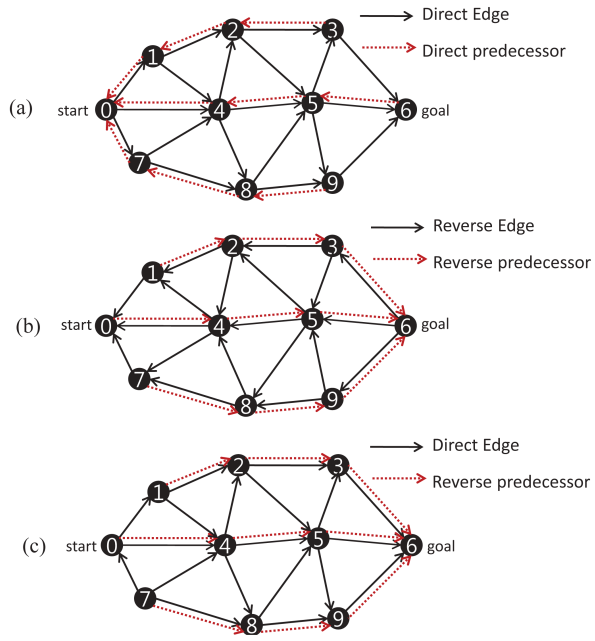


Figure 4: Example of reverse graph and predecessor array.

3: $s_{cur} = pred(s_{cur})$;
4: $path = [path\ s_{cur}]$;
5: **until** $(s_{cur} = y)$
6: **return** *path*;

Let $G(N, E)$ be a directed graph with the set of nodes N and the set of edges E. We define the reverse graph of a directed graph $G(N, E)$ as the graph $G_{rev}(N, E_{rev})$ with

$$E_{rev} = \{(u, v)|(v, u) \in E\} \tag{2}$$

For example, if we reverse all the edges of $G(N, E)$ in Fig. 4(a), we get the reverse graph $G_{rev}(N, E_{rev})$ in Fig. 4(b), with which we can compute the shortest paths from the goal node to all the other nodes by Dijkstra's algorithm. Then we can compute a *reverse-pred* for $G_{rev}(N, E_{rev})$ which is a series of nodes constituting the path from an arbitrary node to the goal node as shown in Fig. 4(b). Now, if we consider the original graph $G(N, E)$ with *reverse-pred* computed with resect to the reverse graph $G_{rev}(N, E_{rev})$ as in Fig. 4(c), we can see that *reverse-pred* contains pointers to the optimal node to travel from any node in $G(N, E)$ to reach the goal.

Suppose that an agent travels around the graph to arrive at the destination node. Even if the agent deviates from the optimal path, it can eventually reach the destination by following the next node that *reverse-pred* points. In other words, *reverse-pred* serves as a guide to the lost or wandering agents in the network. For instance, in Fig. 4(c), if an agent on node 0 moves to node 1 which is not on the optimal path, it can adapt to the situation and follow the optimal path from node 1 by consulting *reverse-pred*. Like this, if an agent deviates from the shortest path on any node, it can rectify its plan and follow the optimal path to the goal.

In the field of reinforcement learning, such a *reverse-pred* is called the optimal policy [19]. The optimal policy $\pi^* : N \mapsto N$ is the mapping from the current node to the next node that is on the optimal path. As a scheme to apply the optimal policy π^*, the ϵ-greedy method is used which picks the best move most of the times but allows a random move with a small probability of ϵ. This can be summarized as

$$\epsilon\text{-}greedy(s, \pi^*) = \begin{cases} \pi^*(s) & \text{if } \zeta < \epsilon \\ \text{random move} & \text{otherwise} \end{cases} \tag{3}$$

where $\pi^*(s)$ is equivalent to *reverse-pred(s)*, s is the current node, and ζ is a random number generated between 0 and 1 to be compared with ϵ. Now, the population can be initialized by Subroutine 2:

Subroutine 2. PopulationInit(π^*, *start*, *goal*)
1: **for** $i = 0$ **to** $PopulationSize - 1$
2: Set $s_{cur} = start$ and $chromosome[i] = [s_{cur}]$;
3: **repeat**
4: $s_{cur} = \epsilon\text{-}greedy(s_{cur}, \pi^*)$;
5: $chromosome[i] = [chromosome[i]\ s_{cur}]$;
6: **until** ($s_{cur} = goal$)
7: **end**
8: **return** *chromosome*;

There are several advantages in our population initialization method. First, the amount of data needed in a random initialization method (like Ahn's GA) is even larger than in our algorithm since the former requires the information on the overall network topology while the latter only refers to the optimal policies in the predecessor array. Therefore, in real-world situations where the size of the network is huge, DGA has significantly less overhead than Ahn's GA. Second, our initialization method increases the probability of generating valid chromosomes while the probability with random population initialization is inversely, exponentially proportional to the length of the path. This is theoretically proved in Claim 1, and experimentally verified by large networks wherein valid chromosomes could not be generated within reasonable time.

Claim1. Let x be a random variable drawn from *Bernoulli*(m) distribution defined as follows: If an agent reaches the destination node in reasonable time by selecting the next node randomly, then $x = 1$, otherwise $x = 0$. That is, the probability $P(x = 1)$ is m. The agents is assumed to makes l independent selections of next nodes. We claim that the probability $P_{rand}(l)$ of generating a valid path (chromosome) with length l in reasonable time with random population initialization is

$$P_{rand}(l) = m^l \tag{4}$$

Meanwhile, let y be a random variable drawn from *Bernoulli*(1) defined as follows: If an agent reaches the destination node in reasonable time by executing the optimal policy, then $y = 1$, otherwise $y = 0$. That is, the probability $P(y = 1)$ is 1. We now claim that the probability $P_{\epsilon\text{-}greedy}(l)$ of generating a valid path with length l in reasonable time with ϵ-*greedy* selection is

$$P_{\epsilon\text{-}greedy}(l) = m^{l(1-\epsilon)} \tag{5}$$

Proof:

$$\begin{aligned} P_{rand}(l) &= \overbrace{P(x=1) \times P(x=1) \times \cdots \times P(x=1)}^{l} \\ &= \overbrace{m \times m \times \cdots \times m}^{l} \\ &= m^l \end{aligned}$$

$$
\begin{aligned}
& P_{\epsilon\text{-}greedy}(l) \\
&= \overbrace{\underbrace{P(x=1)\cdots P(x=1)}_{l\epsilon}\underbrace{P(y=1)\cdots P(y=1)}}^{l} \\
&= \overbrace{m\cdots m\cdot\underbrace{1\cdots 1}_{l\epsilon}}^{l} \\
&= m^{l-l\epsilon} \\
&= m^{l(1-\epsilon)}
\end{aligned}
$$

□

For example, let $m = 0.995$, $l = 50$, $\epsilon = 0.5$. Then,

$$P_{rand}(l) = (0.995)^{50} = 0.7783$$
$$P_{\epsilon\text{-}greedy}(l) = (0.995)^{25} = 0.8822$$
$$P_{\epsilon\text{-}greedy}(l)/P_{rand}(l) = 1.1335$$

However, if $l = 1000$,

$$P_{rand}(l) = (0.995)^{1000} = 0.0067$$
$$P_{\epsilon\text{-}greedy}(l) = (0.995)^{500} = 0.0816$$
$$P_{\epsilon\text{-}greedy}(l)/P_{rand}(l) = 12.1791$$

3.2 Fitness Function

Since the purpose of the proposed algorithm is to re-route the shortest path considering dynamic traffic situations, the fitness of each chromosome is based on the traveling time instead of the physical distance between the source and the destination. So we redefine $C(x, y)$ in Eq. (1) with the traveling time from node x to node y, and represent the costs as a hash table. We can define the set of all edges comprising the chromosomes as

$$\Omega = \{(y_{i,j}, y_{i,j+1})|y_{i,j}\} \tag{6}$$

where $y_{i,j}$ is the jth gene in the ith chromosome in the population. Then the hash table contains the edge $(x, y) \in \Omega$ with its associated cost. This scheme provides fast access of the edge costs, and requires less communication overhead of real-time traffic information only for the edges in Ω instead of all the edges in the network.

3.3 Selection

Although the time complexity of tournament selection without replacement used in Ahn's GA is not costly ($O(|chromosomes|)$ where $|chromosomes|$ is the number of chromosomes in the population), it has a problem that good chromosomes can dropout early if they are met with chromosomes with higher fitness values in the tournament. We devised the following selection method to solve the problem.

1. The average fitness of all chromosomes in the population is calculated.
2. The chromosomes with above-average fitness survive in the next generation, and the chromosomes with below-average fitness are weeded out.
3. The deleted chromosomes are replaced by the survived ones at random.

Each step of the above selection method has time complexity of $O(|chromosomes|)$, which also makes the total complexity of $O(|chromosomes|)$. With asymptotically the same computational overhead, our selection method can overcome the early dropout problem.

3.4 Crossover

As described earlier, the crossover operator in Ahn's GA finds all genes that appear in both parent chromosomes and then chooses one of them randomly. Let α and β be the lengths of such two chromosomes, respectively. Then the time required to find the crossover point is $O(\alpha\beta)$. If α and β increase for large networks, the cost for searching the crossover point become expensive. To optimize the process of crossover point search, we use the following subroutine.

Subroutine 3. SearchCrossPoint(x_1, x_2)
// x_1, x_2 are two chromosomes to crossover.
1: s_1 = *rand* % *size*(x_1);
2: **if** (s_1 == 0) e_1 = *size*(x_1) − 1 **else** e_1 = s_1 − 1; **end**
3: s_2 = *rand* % *size*(x_2);
4: **if** (s_2 == 0) e_2 = *size*(x_2) − 1 **else** e_2 = s_2 − 1; **end**
5: **for** i = s_1 **to** e_1
6: **for** j = s_2 **to** e_2
7: **if** ($x_1[i]$ == $x_2[j]$) **return** i, j; **end**
8: **if** (j == *size*(x_2) − 1) j = 0; **end**
9: **end**
10: **if** (i == *size*(x_1) − 1) i = 0; **end**
11: **end**

Note that $SearchCrossPoint(x_1, x_2)$ determines the random crossover point of common genes starting from arbitrary positions of two chromosomes x_1 and x_2, and keeps comparing the genes in a circular way (i.e. after considering the last gene, it starts from the first gene of the chromosome). As soon as the first match occurs, the subroutine returns the genes. Otherwise, it repeats the comparisons for all possible pairs of positions. The remaining steps of the crossover operation (i.e. generation of offsprings from the crossover point and application of the repair function) remain the same as Ahn's GA.

3.5 Mutation

As described in Section 3.2, only the edge $(x, y) \in \Omega$ can appear in the chromosomes. If an edge $(x', y') \notin \Omega$ appears in the chromosomes in a new generation as a result of mutation, the traffic information on (x', y') needs to be fetched to compute the shortest path, which causes additional communication. To prevent this overhead, the mutation is omitted in our algorithm. In our preliminary experiments, there was no significant difference in performance (in terms of the path quality and the convergence speed) between two approaches where the mutation was applied or not.

3.6 The overall algorithm

DGA can be summarized as Algorithm 2.

Algorithm 2. DGA
1: Construct π^* (from *reverse-pred*) and the initial population Y (Section 3.1).
2: Remove loops in chromosomes in Y by repair function (Section 2.7).
3: Construct hash table for edges $(x, y) \in \Omega$ (Section 3.2).
4: **repeat until** convergence
5: Calculate the fitness of population Y (Section 3.2).
6: Do selection (Section 3.3), crossover (Section 3.4), and remove loops in Y (Section 2.7).
7: **end**
8: **return** Y

Table 1: Parameter settings.

Network ID	Network Size	ϵ	Population Size
#1	50	0.5	20
#2	100	0.5	20
#3	200	0.5	20
#4	400	0.5	20
#5	800	0.5	20
#6	2000	0.5	20
#7	4000	0.5	20
#8	8000	0.5	20
#9	20000	0.5	30
#10	40000	0.6	30
#11	80000	0.6	40
#12	160000	0.6	40
#13	320000	0.6	40
#14	640000	0.7	40
#15	800000	0.7	40
#16	1000000	0.7	40
#17	1200000	0.7	50

As in Ahn's GA, the condition for the convergence of the algorithm is if all chromosomes are identical.

4 Experiments

4.1 Setup

DGA is implemented in C and all the experiments were conducted on Intel Core2Quad processors (2.40GHz clock rate). We generated strongly connected random networks of size (i.e. number of nodes) ranging 50-1,200,000, and the distance $dist(i, j)$ between node i and j is assigned with a random integer in [1-9,999].

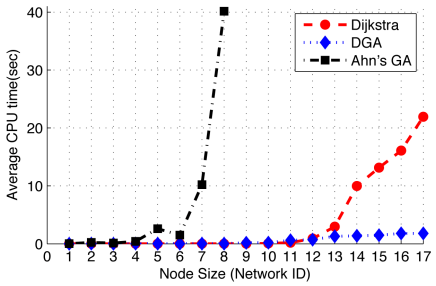
There are two parameters in DGA: ϵ (of *ϵ-greedy* strategy) and the population size, except the crossover probability which was set to 1. The lower ϵ and the higher population size we set, the greater the diversity in a population will be. We applied parameter settings for each network as illustrated in Table 1.

To evaluate the performance of DGA under real-time traffic conditions, we also constructed a traffic simulator as follows:

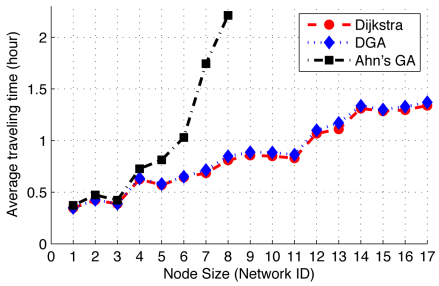
1. A vehicle travels around the networks (generated as in Table 1) to arrive at the destination node.
2. Whenever a vehicle makes a move from the current to the next node, all the edge costs of the network are changed dynamically by

$$C(i, j) = \frac{dist(i, j)}{v_{ij}}, \text{ for each node } i, j \tag{7}$$

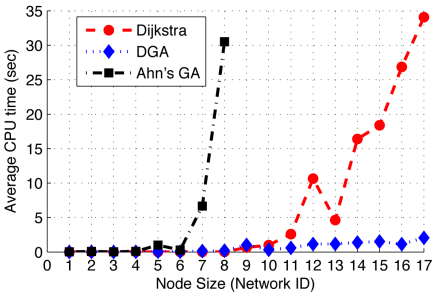
   where v_{ij} is velocity of vehicles on the link (road) between node i and j which is drawn from two normal distributions with the same mean but different standard deviations (i.e. $\mathcal{N}$(80km/s, 20km/s) and $\mathcal{N}$(80km/s, 40km/s)) to see the behavior of the algorithms in different situations.
3. A vehicle re-routes the path whenever the edge costs are changed.



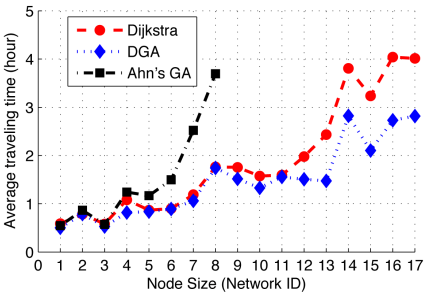
(a) A comparison of average CPU time (low variation)



(b) A comparison of average traveling time (low variation)



(c) A comparison of average CPU time (high variation)



(d) A comparison of average traveling time (high variation)

Figure 5: Simulation results.

4. There are three types of vehicles implementing three algorithms: Dijkstra's algorithm (implemented with the overflow bag structure as in [16]), Ahn's GA, and DGA.

5. For 300 randomly generated source-destination node pairs, the performance (in terms of the CPU time and the traveling time) are measured and averaged.

4.2 Results

The experimental results are shown in Fig. **??** where (a), (b) are for the networks with less drastic changes in the velocity of vehicles (i.e. standard deviation of 20km/s), and (c), (d) are with more drastic changes (i.e. standard deviation of 40km/s). The x-axis of the graphs represents the network ID of Table 1. The y-axis represents the average CPU time in (a) and (c), and the average traveling time in (b) and (d). The results of Ahn's GA for above 20,000 node-sized networks are not included due to the excessive running time.

As shown in Fig. **??**(a) and (c), it is impossible for Ahn's GA to find the path in reasonable time. For networks with less than 40,000 nodes, the average CPU time of Dijkstra's algorithm and DGA are similar. However, as the size of the network increase, DGA outperforms Dijkstra's algorithm by a large margin. This is because Dijkstra's algorithm computes a new path over the entire nodes for each traffic condition, while DGA adjusts the path based on the locally updated traffic condition with the predecessor array.

As shown in Fig. **??**(b) and (d), the quality of the path (i.e. average traveling time) of Ahn's GA is even inferior to other algorithms. Fig. **??**(b) verifies that DGA produces paths as good as the ones produced by Dijkstra's algorithm for less dynamic networks. However, DGA outperforms Dijkstra's algorithm for highly dynamic networks as shown in Fig. **??**(d). This is because Dijkstra's algorithm sticks to the current traffic conditions too much and might make inefficient changes in the path (e.g.

detours), while DGA makes local adjustments to the current path and produces stable solutions. This verifies the feasibility of DGA in real-world car navigation systems where traffic conditions are constantly and possibly drastically changing.

5 Conclusion

We have presented a fast and scalable re-routing algorithm, DGA, that adapts to dynamically changing networks. In addition to the theoretical soundness, the experimental results have also shown the outstanding performance of DGA on large networks. DGA is a good candidate for intelligent car navigation systems since it is capable of re-routing the optimal path swiftly whenever new traffic information is available. In addition, DGA has a significant merit of requiring the minimal traffic information and reducing the communication cost between the car navigation system and the central server, or among the navigation systems in each vehicle.

We have not tested DGA with real-world maps and traffic information due to the lack of required infrastructures (e.g. communication, information collection). Therefore, DGA needs to be deployed and fully evaluated when the infrastructures become available. Also, DGA can be extended to work in the unexplored environment where the agent does not have the global picture on the environment where it belongs. In such an environment, Markov decision processes (MDPs) [20] and reinforcement learning approaches [19, 21] can be useful to learn the optimal routing policy, as attempted in [22]. In addition, DGA can be also extended to consider additional criteria for navigation similar to [15, 23, 24]. Some of these research issues are currently in progress.

Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology (2009-0076594) to Jihoon Yang, the corresponding author.

Bibliography

[1] EBU BPN. 027-1 ŇTransport Protocol Experts Group (TPEG) Specifications.

[2] EBU BPN. 027-2 ŇTransport Protocol Experts Group (TPEG) Specifications.

[3] EBU BPN. 027-3 ŇTransport Protocol Experts Group (TPEG) Specifications.

[4] EBU BPN. 027-4 ŇTransport Protocol Experts Group (TPEG) Specifications.

[5] EBU BPN. 027-5 ŇTransport Protocol Experts Group (TPEG) Specifications.

[6] EBU BPN. 027-6 ŇTransport Protocol Experts Group (TPEG) Specifications.

[7] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, 1959.

[8] M. Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, 1996.

[9] C.W. Ahn and R. S. Ramakrishna. A genetic algorithm for shortest path routing problem and the sizing of populations. *IEEE Transactions on Evolutionary Computation*, 6(6):566–579, 2002.

[10] D. E. Goldberg. *Genetic Algorithms in Search and Optimization*. Addison-wesley, 1989.

[11] Q. Zhang and Y. W. Leung. An orthogonal genetic algorithm for multimedia multicast routing. *IEEE Transactions on Evolutionary Computation*, 3(1):53–62, 1999.

[12] F. Xiang, L. Junzhou, W. Jieyi, and G. Guanqun. QoS routing based on genetic algorithm* 1. *Computer Communications*, 22(15-16):1392–1399, 1999.

[13] Y. Leung, G. Li, and Z. B. Xu. A genetic algorithm for the multiple destination routing problems. *IEEE Transactions on Evolutionary Computation*, 2(4):150–161, 1998.

[14] H. Kanoh. Dynamic route planning for car navigation systems using virus genetic algorithms. *International Journal of Knowledge-based and Intelligent Engineering Systems*, 11:65–78, 2007.

[15] H. Kanoh and K. Hara. Hybrid genetic algorithm for dynamic multi-objective route planning with predicted traffic in a real-world road network. In *Proceedings of the Conference on Genetic and Evolutionary Computation*, pages 657–664. ACM, 2008.

[16] B. V. Cherkassky, A. V. Goldberg, and T. Radzik. Shortest paths algorithms: theory and experimental evaluation. *Mathematical Programming*, 73(2):129–174, 1996.

[17] R. B. Dial. Algorithm 360: Shortest-path forest with topological ordering [H]. *Communications of the ACM*, 12(11):632–633, 1969.

[18] R. Bellman. On a routing problem. *Quarterly of Applied Mathematics*, 16(1):87–90, 1958.

[19] R. S. Sutton and A. G. Barto. *Reinforcement Learning*. MIT Press, 1998.

[20] R. Bellman. A Markovian decision process. *Journal of Mathematics and Mechanics*, 6, 1957.

[21] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: a survey. *Journal of Artificial Intelligence Research*, pages 237–285, 1996.

[22] J. A. Boyan and M. L. Littman. Packet routing in dynamically changing networks: A reinforcement learning approach. *Proceedings of the Advances in Neural Information Processing Systems*, pages 671–671, 1994.

[23] M. Stanojević, M. Vujošević, and B. Stanojević. Number of Efficient Points in some Multiobjective Combinatorial Optimization Problems. *International Journal of Computers, Communications & Control*, 3(Suppl.): 497-502, 2008.

[24] I. Harbaoui Dridi, R. Kammarti, M. Ksouri, and P. Borne. Multi-Objective Optimization for the m-PDPTW: Aggregation Method With Use of Genetic Algorithm and Lower Bounds. *International Journal of Computers, Communications & Control*, 6(2): 246-257, 2011.

INT J COMPUT COMMUN, ISSN 1841-9836
Vol.7 (2012), No. 3 (September), pp. 494-508

Mining Temporal Sequential Patterns Based on Multi-granularities

N. Li, X. Yao, D. Tian

Naiqian Li, Xinhui Yao, Dongpin Tian
Department of Computer Science
Baoji University of Arts and Sciences
Baoji 721016, Shaanxi, China
E-mail: xalnq@hotmail.com, baojiyxh@163.com,
tdp211@163.com

Abstract:
Sequential pattern mining is an important data mining problem that can extract frequent subsequences from sequences. However, the times between successive items in a sequence is typically used as user-specified constraints to pre-process the input data or to prune the pattern search space. In either cases, the times cannot be used to identify item intervals of sequential patterns. In this paper, we introduce a form of multi-granularity sequence patterns, which is a sequential pattern where each transition time is annotated with multi-granularity boundary interval and average time derived from the source data rather than the user-predetermined time interval or only a typical time. Then we present a novel algorithm, MG-PrefixSpan, of multiple granularity sequential patterns based on PrefixSpan[, which discovers all such patterns. Empirical evaluation shows that MG-PrefixSpan scales up linearly as the size of database, and has a good scalability with respect to the length of sequence and the size of transaction.
Keywords: Data Mining Algorithm, Sequential Pattern Mining, Sequential Data, Time Granularity, Temporal Patterns.

1 Introduction

Among various types of data mining applications [1,2], the sequential pattern mining, which discoveries interesting sequential patterns hidden in sequence of events, is an important data mining problem with broad applications, including market analysis, decision support, the prediction of occurrences of recurrent illnesses, system performance analysis and telecommunication network analysis etc.

The problem of mining sequential patterns was first proposed by Agrawal and Strikant [3]: Given a data set of sequences, each sequence is a list of transactions, where each transaction is a set of items. The sequential pattern mining is to find all subsequences that is more frequent than a user-specified minimum support threshold while maintaining their item occurrence order.

For example, in the database of a book-club, a sequential pattern might be “5 percent of customers bought ‘Foundation’, then ‘Foundation and Empire’, and then ‘Second Foundation”’ [4]. Although the discovered sequential patterns reveal what items are frequently bought together and in what order, they cannot reveal how long time the items will be bought after the preceding items. Unfortunately, not knowing the time means that we cannot exactly predict when the next purchase will happen. In addition, some sequential patterns could occur in different periods with different time granularities. For example, “HP stock could rise within 5 days after IBM stock rose" and “HP stock could fall within 6 month after IBM stock rose", which reveal HP stock rise or fall with respect to IBM stock rise at different time granularities (day and month), are two useful different patterns. However, these traditional sequence patterns can only tell us “HP stock could rise after IBM stock rose" and “HP stock could fall after IBM stock rose". This situation means that the two patterns are useless. Another situation may be “HP stock

could rise 5 days later after IBM stock rose" and “HP stock could rise 6 months later after IBM stock rose". Although these two patterns are completely different with regard to different time granularities, these traditional sequence patterns could treat the two patterns as the same pattern “HP stock could rise after IBM stock rose", which make the extracted pattern less precise and some useful information lost.

Given the above reasons, in this paper we generalize the problem definition given in [1,8,9,13] to incorporate the maximum, minimum and average time between successive transactions, which are derived from the source data, and different time granularities in traditional sequential patterns. We present a novel algorithm of multiple granularity sequence patterns based on PrefixSpan [6] , called MG-PrefixSpan, which discovers all such patterns. Empirical evaluation shows that MG-PrefixSpan scales up linearly as the size of database, the length of sequence and the size of transaction.

The rest of this paper is organized as follows. Related work is discussed in section 2. We give a formal description of the problem of mining temporal sequence patterns based multiple granularities in section 3. In section 4, we describe MG-PrefixSpan, an algorithm for finding such patterns, and then Section 5 provides empirical evaluation of the performance of MG-PrefixSpan. Finally, we conclude the paper in section 6.

2 Related Work

Sequential pattern mining, in general, can be grouped into two categories. One category, called un-temporal sequence pattern mining or traditional sequence pattern mining, considers only the item occurrence order in a sequential pattern, but does not deal with time-related data [1, 3–6]. The other category, called temporal sequence pattern mining, consider not only the item occurrence order in a sequence pattern, but also the time between successive items in a sequential pattern, such as [8,9,11–13].

2.1 Un-temporal Sequence Pattern Mining

Agrawal and Strikant [3] introduced the notion of sequential pattern mining, and based on the property that any sub-pattern of a frequent pattern must be frequent, three Apriori-based algorithms were proposed: AprioriSome, DynamicSome and AprioriAll. Two of these algorithms were designed only to find maximal sequential patterns. The third algorithm, AprioriAll, finds all patterns. Briefly, AprioriAll is decomposed into two phases: (1) generating candidate sequences; (2) scanning the sequential database to check the support of each candidate to determine frequent sequence patterns according to minimal support threshold. Although AprioriAll is not efficient, it is the basis of many efficient algorithms developed later. SPADS [5] is an algorithm proposed to find frequent patterns using efficient lattice search technology and simple joins. It decomposes the original search space (lattice) into smaller pieces (sub-lattices), which can be processed independently in the main memory. Due to adopting a vertical id-list database format to count the number of frequent patterns, all the sequential patterns are discovered with only a few passes over the database. SPADS outperforms AprioriAll [2,3]. PrefixSpan[6] is another more efficient algorithm for mining sequential patterns comparing with the aprior-based algorithm AprioriAll and SPADS, especially in dealing with very large databases. It mainly adopts a projection-based, sequential pattern-growth method to make the database for next pass much smaller and consequently make the algorithm more speedy. Also in PrefixSpan there is no need to generate candidates, only recursive projection of database according to their prefix. Our method is based on the PrefixSpan algorithm.

Some have tried to exert constraints on the mining of sequential patterns so that only those sequential patterns interesting to users are discovered rather than the whole possible sequential patterns[2,5,6,7]. Strikant and Agrawal [4] generalized their definition of sequential patterns in [3] to integrate with time constraints, sliding time window, and user-defined taxonomy, and proposed algorithm GSP. Mannila et al. [7] proposed a method of mining frequent episodes in a sequence of events, in which episodes are essentially constraints on events in form of acyclic graphs. Garofalakis et al. [8] proposed a family of

SPIRIT algorithms of mining user-specified sequential patterns by using regular expression constraints. Pei et al. [9] developed an extended framework based on a sequential pattern growth for constraint-based sequential pattern mining. Although time between successive items is typically used as a user-specified constraint to shrink the pattern search space to make the computation more efficient, it is not used in the output frequent sequence patterns.

2.2 Temporal Sequence Pattern Mining

Yoshida et al.[10] proposed a notation of delta pattern, which is a temporal sequence pattern with temporal constraints in the form $A \overset{[0,7]}{\rightarrow} B \overset{[3,5]}{\rightarrow} C$ of bounding intervals. An example of delta pattern has the form denoting a sequential pattern $A \rightarrow B \rightarrow C$ that frequently appears in the database with transition times from A to B and from B to C that contained in [0, 7] and in [3, 5] respectively. However, Yoshida et al. only provided a heuristics for finding some frequent delta patterns, and did not investigate the problem of finding all of them. Along the same direction, Chen et al. [11] introduced a form of temporal sequence pattern by inserting pseudo items into the original sequential pattern. Pseudo items are user-defined time interval segmentations in advance. When counting the support of a sequential pattern, only the sequence, in which the pseudo item between successive items is same, supports the sequential pattern. Two algorithms, I-Apriori and I-PrefixSpan, were proposed. I-Apriori is based on Apriori algorithm [12], and I-PrefixSpan is based on PrefixSpan[6]. Hirate et al. [13] proposed generalized sequential pattern mining with item intervals. They extended sequences which are defined by inserting pseudo items based on the interval itemization function and exerting four interval constraints on items. However, as they adopted some user-predefined pseudo items or constraints on the time intervals between successive items, it is difficult for a user to specify optimal constraints related to item interval and cannot reveal the time interval between successive items in the patterns precisely, it may result in some useful sequential patterns not being found. Giannotti et al. [14] introduced another form of sequential pattern, called Temporally-Annotated Sequence, TAS in short, where each transition is annotated with temporal information representing a typical time derived from the source data. For instance, TAS $A \overset{t_1}{\rightarrow} B \overset{t_2}{\rightarrow} C$ denotes the fact that a sequential pattern $A \rightarrow B \rightarrow C$ frequently appears in the database and that the time for getting from event A to B is close to t_1 and from event B to C is close to t_2. Although this method using typical time to annotate transition between two successive events can reveal how much time the event will occur after the preceding event, they cannot distinguish completely patterns with regard to different time granularities. For example, from event A to B is close to t_1 days, other from event A to B may be close to t_2 months, these two different patterns are treated as the same. It is useful to be able to distinguish the patterns to understand not only what event will follow, but also when these events will occur.

Finally, we mention the work in [15], where event structures that have temporal constraints with multiple granularities are introduced. Event structures essentially are used as a flexible user-defined constraint specification to define the pattern discovery problem with these structures that enable users to focus on their interested sequential patterns. This method can only find the patterns that satisfy the event structures. Furthermore, an event structure consists of a number of variables representing events and temporal constraints among these variables, an efficient event structure is difficult to define beforehand even if you are a domain expert.

In our work, we introduce a new form of sequential pattern with multi-granularities, which is a sequential pattern where each transition is annotated with multi-granularity boundary interval and average time derived from the source data rather than user-predetermined time intervals [8,9] or only a typical time [14]. We also define the pattern discovery problem involving multi-granularities for these pattern and study efficient algorithm based on PrefixSpan to solve it.

3 Problem Formulation

3.1 Time granularity

In order to formally define temporal sequence pattern that involves time granularities, we first review the notion of a time granularity [15].

Definition 1. A granularity is a mapping μ from the set of the positive integers (the time ticks) $\mathcal{R}$ to $\mathcal{R}^2$ (the set of absolute time sets) such that for all positive integer i and j with $i \leq j$, the following two condition are satisfied:

(1) $\mu(i) \neq \emptyset \wedge \mu(j) \neq \emptyset$ implies that each number in $\mu(i)$ is less than all the numbers in $\mu(j)$, and

(2) $\mu(i) \neq \emptyset$ implies $\mu(j) \neq \emptyset$.

Each set $\mu(i)$, if non-empty, is called a granule of the μ. Property (1) says that granules do not overlap and the order on time ticks follow the order on the corresponding granules. Property (2) says that the subset of the time ticks corresponding to the granules forms a set of contiguous integers. The set $\mu(i)$ of reals is said to be the ith tick of μ, or tick i of μ . For example, hour, day, week, month, and year, satisfy the above definition. We can also define more complex granularities like business-week, weekend and so on.

When dealing with temporal types, it is needed to determine the tick (if any) of a temporal type μ that covers a given tick z of anther temporal type v. Formally, for each positive integer z and temporal types μ and v, if $\exists$ z' (necessarily unique) such that $v(z) \subseteq \mu(z')$ then $\lceil z \rceil_v^\mu = z\prime$, otherwise $\lceil z \rceil_v^\mu$ is undefined[15]. In this paper, all timestamps in a sequence are assumed to be in terms of a fixed granularity g_0, and abbreviate $\lceil z \rceil_v^\mu$ as $\lceil z \rceil^\mu$ if $v = g_0$.

Definition 2. A multi-granularity schema is a tuple of the form $G_m = (g_m, g_{m-1}, \cdots, g_2, g_1)$, where each g_i $(1 \leq i \leq m)$ is a granularity.

For simplicity, we require that in G_m and g_0, each unit of g_i is contained in a unit g_{i+1} $(0 \leq i \leq m-1)$.

3.2 Temporal sequence and multi-granularity sequence pattern

Let $I = \{i_1, i_2, \cdots, i_n\}$ be a set of items. An itemset s_i is a non-empty subset of items (without loss of generality, we assume that items of an itemset are sorted alphabetically) denoted as $i_1, \cdots, i_k$, where i_j $(1 \leq j \leq k)$ is an item. A traditional data sequence is an ordered list of itemsets, which is sorted by the order of priority of the transaction time and denoted as $(s_1 \rightarrow s_2 \rightarrow \cdots \rightarrow s_n)$, where s_i $(1 \leq i \leq n)$ is an itemset [1,3]. In practice, a data sequence of a customer is also formed as an ordered list of itemsets and time stamps [11], we call it a temporal sequence.

Definition 3. A temporal sequence s is represented as $\langle (s_1, t_1) \rightarrow (s_2, t_2) \rightarrow \cdots \rightarrow (s_n, t_n) \rangle$, where s_i is an itemset and t_i stands for the time at which s_i occurs, $t_i < t_j$ for $1 \leq i < j \leq n$.

When adding the itemset time information in the sequence, the time interval value between any two elements in the sequence can be computed as follows:

$$T_{ij} = T_j - T_i, \text{where } 1 \leq i < j \leq n$$

We now formally define multi-granularity schema, multi-granularity sequence patterns and its' support, then formalize our novel mining problem as the discovery of all frequent multi-granularity sequence patterns in the database D.

Definition 4. Given a multi-granularity schema $G_m = (g_m, g_{m-1}, \cdots, g_2, g_1)$, a multi-granularity sequence pattern is represented as follows:

$$\alpha = \alpha_1 \overset{[L_1, M_1, U_1]\mu_1}{\rightarrow} \alpha_2 \overset{[L_2, M_2, U_2]\mu_2}{\rightarrow} \cdots \overset{[L_{n-1}, M_{n-1}, U_{n-1}]\mu_{n-1}}{\rightarrow} \alpha_n$$

where (1) α_i $(1 \le i \le n)$ is an itemset; (2) $\mu_i \in \{g_j | j = 1, \cdots, m\}$ $(1 \le i \le n-1)$ a granularity; (3) L_i, M_i, and U_i $(1 \le i \le n-1)$ are respectively the lower bound, average time and upper bound of the time that α_{i+1} occurs after α_1, α_2, $\cdots$ and α_i with the granularity μ_i. $[L_i, M_i, U_i]\mu_i$ is called temporal annotation of α_{i+1}, and the tuple $([L_i, M_i, U_i]\mu_i, \alpha_{i+1})$ also called a temporal item $(1 \le i \le n-1)$.

Example $\alpha = \alpha_1 \overset{[1,25,5]day}{\rightarrow} \alpha_2 \overset{[2,33,4]week}{\rightarrow} \alpha_3$ is a multi-granularity sequential pattern, where $\alpha_1, \alpha_2, \alpha_3$ are itemsets or events, day and week are time granularities. The pattern indicates that α_2 occurs in 1 day to 5 days or average 2.5 days after α_1, then 2 weeks to 4 weeks or average 2.5 weeks later, α_3 occurs.

The total number of items in a multi-granularity sequence pattern is referred as the length of the pattern. A multi-granularity sequence pattern whose length is k is also represented as k-multi-granularity sequence pattern.

Definition 5. Given a temporal sequence $s = (s_1, t_1) \rightarrow (s_2, t_2) \rightarrow \cdots \rightarrow (s_n, t_n)$ and a multi-granularity schema $G_m = (g_m, g_{m-1}, \cdots, g_2, g_1)$. Let α be a multi-granularity sequence pattern $\alpha = \alpha_1 \overset{[L_1,M_1,U_1]\mu_1}{\rightarrow} \alpha_2 \overset{[L_2,M_2,U_2]\mu_2}{\rightarrow} \cdots \overset{[L_{k-1},M_{k-1},U_{k-1}]\mu_{k-1}}{\rightarrow} \alpha_k$, where $\mu_i \in g_j | j = 1, \cdots, m$ $(1 \le i \le k-1)$. s is said to support(or contain) α if and only if there exists integers $1 \le i_1 < i_2 < \cdots < i_k \le n$ such that

(1) $\forall j 1 \le j \le k$, $\alpha_i \subseteq s_{i_j}$

(2) $\forall j 1 \le j \le k-1$, if $\mu_j = g_h$ $(1 \le h \le m-1)$ then $g_h(1) \le T_{i_j i_{j+1}} < g_{h+1}(1)$, otherwise $T_{i_j i_{j+1}} \ge g_m(1)$.

The condition (2), in fact, is a constraint which divides the time between successive items into non-overlap partitions according to the sizes of different time granules. Without this constraint, the too long or too short time interval between successive items can be treated as the same in a pattern, and may result in some useful patterns not being found.

Definition 6. Let $\alpha = \alpha_1 \overset{[L_1,M_1,U_1]\mu_1}{\rightarrow} \alpha_2 \overset{[L_2,M_2,U_2]\mu_2}{\rightarrow} \cdots \overset{[L_{k-1},M_{k-1},U_{k-1}]\mu_{k-1}}{\rightarrow} \alpha_k$ be a multi-granularity sequential pattern, G_m be a multi-granularity schema, and S_l be a set of temporal sequence that support $\alpha^{l+1} = \alpha_1 \overset{[L_1,M_1,U_1]\mu_1}{\rightarrow} \alpha_2 \overset{[L_2,M_2,U_2]\mu_2}{\rightarrow} \cdots \overset{[L_l,M_l,U_l]\mu_l}{\rightarrow} \alpha_{l+1}$ $(1 \le l \le k-1)$ in database D. The annotation $[L_1, M_1, U_1]\mu_l$ of α_{l+1} is defined as follows:

$$\begin{aligned} U_l &= \max_{s \in S_l} \max_{i_1, \cdots, i_{l+1}} \lceil T_{i_l i_{l+1}} \rceil^{\mu_l} \\ L_l &= \min_{s \in S_l} \min_{i_1, \cdots, i_{l+1}} \lceil T_{i_l i_{l+1}} \rceil^{\mu_l} \\ M_l &= \frac{1}{|S_l|} \sum_{i_1, \cdots, i_{l+1}} \lceil T_{i_l i_{l+1}} \rceil^{\mu_l} \end{aligned}$$

where $i_1, i_2, \cdots, i_l, i_{l+1}$ are the indexes of the elements in $s \in S_l$ that match the elements of α^{l+1}, $\mu_l \in G_m$.

Definition 7. Let D be the database of temporal sequence, G_m be a multi-granularity schema. For a multi-granularity sequence pattern α, its support in database D is defined by:

$$Support_D(\alpha) = \frac{|\{s | s \in D \wedge s \ supports \ \alpha\}|}{|D|}$$

α is said to be frequent if its support is no less than the user-specified minimum support threshold min-sup i.e. $Support_D(\alpha) \ge min - sup$.

Given a temporal sequence database D, multi-granularity schema G_m and user-specified minimum support threshold min-sup, the task of mining multi-granularity sequence pattern is to find all frequent multi-granularity sequence patterns in the database D.

4 Algorithm for Mining Multi-Granularity Sequence Patterns

In this section, we propose a novel multi-granularity sequence pattern mining algorithm based on the well-known PrefixSpan[6], called as MG-PrefixSpan. The PrefixSpan algorithm solves the pattern mining problem by a divide-and-conquer, pattern-growth principle as follows: for each frequent item a, a projection of the initial sequence database D is created, denoted by $D|a$, i.e., which (1) contains only the suffix of sequences in D with respect to a; (2) contains only the frequent items; (3) in general is much smaller than D. Then, for each frequent item b in $D|a$, appends b to a to form a sequential pattern a' which starts with a. Finally a new, smaller projection $D|a'$ can be constructed recursively and used for finding longer patterns starting with a'.

However, although the PrefixSpan algorithm is very efficient for mining sequential pattern, it cannot be used straightforwardly to find multi-granularity sequence patterns. We extend the sequence database projection operation in PrefixSpan to be able to deal with the temporal relationship between successive items in the sequential pattern based on multi-granularities. Before introduction of the algorithm MG-PrefixSpan, we first extend definitions of prefix, suffix, projection and projected database based on PrefixSpan[6].

Definition 8. Given a temporal sequence $s = \langle (s_1, t_1) \rightarrow (s_2, t_2) \rightarrow \cdots \rightarrow (s_n, t_n) \rangle$ and a multi-granularity schema $G_m = (g_m, g_{m-1}, \cdots, g_2, g_1)$. A multi-granularity sequence pattern $\alpha = \alpha_1 \overset{[L_1,M_1,U_1]\mu_1}{\rightarrow} \alpha_2 \overset{[L_2,M_2,U_2]\mu_2}{\rightarrow} \cdots \overset{[L_{k-1},M_{k-1},U_{k-1}]\mu_{k-1}}{\rightarrow} \alpha_k$ $(k \leq n)$ is a multi-granularity sequence pattern prefix of s if and only if

(1) $\alpha_i = s_i$ for $1 \leq i \leq k-1$;

(2) $\alpha_k \subseteq s_k$ and all the items in $(s_k - \alpha_k)$ are alphabetically after those in α_k;

(3) $\forall i 1 \leq i \leq k-1$, if $\mu_i = g_h$ $(1 \leq h < m)$ then $g_h(L_i) \leq T_{ii+1} \leq g_h(M_i)$.

For example, let multi-granularity schema be G_2 = (hour, minute), g_0 = second. If there is a temporal sequence $\alpha = \langle (c, 5) \rightarrow (abc, 80) \rightarrow (ab, 320) \rightarrow (d, 5400) \rightarrow (cf, 6400) \rangle$, $\beta_1 = (c)$ is a multi-granularity sequence pattern prefix of and so is $\beta_2 \overset{[1,2.4,4]minute}{\rightarrow} (ab)$.

Definition 9. Given a temporal sequence s and a multi-granularity sequence pattern α such that s supports α. A subsequence β of s is called a projection of s with respect to α if and only if

(1) β has a multi-granularity sequence pattern prefix α, and

(2) there exists no proper super-sequence β' of β such that β' is a subsequence of s and also has a multi-granularity sequence pattern prefix α.

If a temporal sequence $\alpha = \langle (c, 5) \rightarrow (abc, 80) \rightarrow (ab, 320) \rightarrow (d, 5400) \rightarrow (cf, 6400) \rangle$ is projected with respect to $\beta_2 \overset{[1,2.4,4]minute}{\rightarrow} (ab)$, then two projection are $\langle (c, 5) \rightarrow (abc, 80) \rightarrow (ab, 320) \rightarrow (d, 5400) \rightarrow (cf, 6400) \rangle$ and $\langle (c, 5) \rightarrow (abc, 80) \rightarrow (ab, 320) \rightarrow (d, 5400) \rightarrow (cf, 6400) \rangle$. If α is projected with respect to $\beta_1 = (c)$, then three different projections are $\langle (c, 5) \rightarrow (abc, 80) \rightarrow (ab, 320) \rightarrow (d, 5400) \rightarrow (cf, 6400) \rangle$, $\langle (c, 5) \rightarrow (abc, 80) \rightarrow (ab, 320) \rightarrow (d, 5400) \rightarrow (cf, 6400) \rangle$ and $\langle (cf, 6400) \rangle$ respectively.

In the above, a temporal sequence α that is projected with respect to a prefix produces more than one projection. To differentiate these projections from the same temporal sequence, the tag [Sid, item, t] is attached to each projected sequences, where Sid is the identifier of the temporal sequence, item is last item in the multi-granularity sequence pattern and t is the time of element in temporal sequence that matches the last element of the multi-granularity sequence pattern.

Consequently, if sequence $\alpha = \langle (c, 5) \rightarrow (abc, 80) \rightarrow (ab, 320) \rightarrow (d, 5400) \rightarrow (cf, 6400) \rangle$ is projected with respect to the β_2, then different projections are presented as follows:

[Sid, b, 80]: $\langle (c, 5) \rightarrow (abc, 80) \rightarrow (ab, 320) \rightarrow (d, 5400) \rightarrow (cf, 6400) \rangle$;

[Sid, b, 320]: $\langle (c, 5) \rightarrow (abc, 80) \rightarrow (ab, 320) \rightarrow (d, 5400) \rightarrow (cf, 6400) \rangle$.

Definition 10. Given a temporal sequence s and a multi-granularity schema $G_m = (g_m, g_{m-1}, \cdots, g_2, g_1)$. Let $\alpha = \langle (s_1, t_1) \rightarrow (s_2, t_2) \rightarrow \cdots \rightarrow (s_n, t_n) \rangle$ be the projection of s with respect to a multi-granularity sequence pattern prefix

$$\alpha = \alpha_1 \overset{[L_1, M_1, U_1]\mu_1}{\rightarrow} \alpha_2 \overset{[L_2, M_2, U_2]\mu_2}{\rightarrow} \cdots \overset{[L_{k-1}, M_{k-1}, U_{k-1}]\mu_{k-1}}{\rightarrow} \alpha_k, (k \leq n)$$

Temporal sequence $\gamma = \langle (s_k'', t_k) \rightarrow (s_{k+1}, t_{k+1}) \rightarrow \cdots \rightarrow (s_n, t_n) \rangle$ is called the suffix of s with respect to α, denoted as $\gamma = s/\alpha$, where $s_k'' = s_k - s_k'$.

Note, if α is not a multi-granularity sequence pattern prefix of s, the suffix of s with respect to α is empty.

This definition indicates that the suffix of a temporal sequence can be obtained by directly removing the prefix from its projection. In the example above, if the sequence $\alpha = \langle (c, 5) \rightarrow (abc, 80) \rightarrow (ab, 320) \rightarrow (d, 5400) \rightarrow (cf, 6400) \rangle$ is projected with respect to the β_2, then three different suffix are obtained as follows:

[Sid, b, 80]: $\langle (c, 80) \rightarrow (ab, 320) \rightarrow (d, 5400) \rightarrow (cf, 6400) \rangle$;

[Sid, b, 320]: $\langle (d, 5400) \rightarrow (cf, 6400) \rangle$.

Definition 11. Let α be a multi-granularity sequence pattern in a temporal sequence database D, The α-projected databases, denoted as $D|\alpha$, is the collection of suffixes of temporal sequences in D with respect to α.

Based on the above description, the newly proposed algorithm, MG-PrefixSpan, is shown as following:

Input: D: temporal sequence database; G_m: multi-granularity schema; min-sup: minimal support threshold.

Output: The complete set of frequent multi-granularity sequence patterns.

Method:

(01) scan D once, to find all frequent items, which are denoted by L^1.
(02) for each a L^1 do begin
(03) construct candidate temporal items $\alpha = (0, a)$ and add α in L_1;
(04) end
(05) output L_1;
(06) for each $s \in D$ do begin
(07) for each $\alpha \in L_1$ do begin
(08) for each β suffix of s with respect to α do begin
(09) if $\beta \neq null$ null then add β and its tag to $D|\alpha$;
(10) end
(11) if $D|\alpha \neq null$ null then call MG-PrefixSpan $(\alpha, 1, D|\alpha)$;
(12) end
(13) end

Subroutine MG-PrefixSpan $(\alpha, 1, D|\alpha)$

Parameter: α a multi-granularity sequence pattern; k the length of α; $D|\alpha$ the α-projected database.

(20) if $L^k = null$ then return;
(21) for all $a \in L^k$ and all $\mu \in G_m$ do begin
(22) construct candidate temporal item (A, a), where A=[L, M, U]μ or 0, add it in C_k;
(23) end
(24) for each s $s \in D|\alpha$ with tag [Sid, b, t] do begin
(25) for each $(A, a) \in C_k$ do begin
(26) for each (s_i, t_i) in s where a $a \in s_i$ do begin
(27) if $t_i == t$, then add 1to the counting number of (0, a) for different sid;

(28) if $g_j(1) \leq t_i - t < g_{j+1}(1)$ or $g_m(1) \leq t_i - t$ for $1 \leq j \leq m$, then do begin
(29) increase the counting number of (A, a) for different sid;
(30) let $L = min\{L, \lceil t_i - t \rceil^{g_j}\}$, $U = max\{U, \lceil t_i - t \rceil^{g_j}\}$, $M = M + \lceil t_i - t \rceil^{g_j}$;
(31) end
(32) end
(33) end
(34) end
(35) for each $(A, a) \in C_k$ do begin
(36) if counting number of (A, a) ≥ min-sup $|D|$, then do begin
(37) let M=M/the total number of pulsing M (A≠0);
(38) append (A, a) to α to form k+1-multi-granularity sequential pattern γ, add a in L^{k+1};
(39) let γ. support=min α. support, counting number of (A, a)/$|D|$, add γ in L^{k+1};
(40) end
(41) end
(42) output L^{k+1}
(43) for each $\gamma \in L_{k+1}$ do begin
(44) for each $s \in D|\alpha$ do begin
(45) for each suffix of s with respect to γ: β do begin
(46) if $\beta \neq null$ then add β and its tag to $D|\gamma$;
(47) end
(48) end
(49) if $D|\gamma = null$ then return;
(50) call MG-PrefixSpan $(\gamma, k + 1, D|\gamma)$;
(51) end
(52) return

The overall MG-PrefixSpan algorithm is summarized in the above. The most importance difference lies in that MG-PrefixSpan algorithm is able to deal with the temporal relationship between successive items in the patterns based on multi-granularities. Steps 21-34 handle the frequent items in $D|\alpha$ and correspond annotations with multi-granularities, where steps 21-23 construct candidate temporal items, steps 24-30 derive the low bound, upper bound and average time of each candidate temporal item from the projected database $D|\gamma$. If a candidate temporal item is frequent, then it can be appended to the prefix to form k+1-multi-granularity sequence pattern γ in the steps 35-41, where when a annotation of a item is 0, it only insert the item into the last element of the prefix, otherwise appends the temporal item to the prefix as an new element. Steps 43-48 construct a new, smaller projection $D|\gamma$, recursively find temporal items in $D|\gamma$ and yield longer multi-granularity sequence patterns until all the multi-granularity sequence patterns are found.

5 Experimental Results and Performance Analysis

In this section, we provide an experimental assessment of the proposed algorithm on synthetic data sets. The purpose is to test the performance of our algorithm MG-PrefixSpan. We will analyze the effects of input parameters on execution times and compare the performances with those of the most efficient algorithm PrefixSpan [6] and I-PrefixSpan [11]. The first algorithm being the fastest algorithm for sequential pattern mining without time intervals [6] and the second algorithm the most effective algorithm in finding sequential patterns with user-defined time intervals, It is natural to compare our algorithm with its.

Table 1: PARAMETERS USED IN THE GENERATION OF DATASET

|D|	Number of customers
|C|	Average number of transactions of per customer
|T|	Average number of items per transaction
|S|	Average length of maximal potentially large sequences
|I|	Average size of itemsets in maximal potentially large sequences
$\|N_S\|$	Number of maximal potentially large sequences
$\|N_I\|$	Number of maximal potentially large itemsets
|N|	Number if items
$\|T_T\|$	Average length of time intervals

Table 2: PARAMETER SETTINGS

Name	**|C|**	**|T|**	**|S|**	**|I|**	**|D|**	**Size (Mb)**
C10T2S4I1	10	2	4	1	10K	1.733
C10T2S4I2	10	2	4	2	10K	1.743
C10T2S8I1	10	2	8	1	10K	1.746
C10T4S4I1	10	4	4	1	10K	2.509
C20T2S4I1	20	2	4	1	10K	3.317

5.1 Evaluation Environment

The two algorithms are implemented by Sun Java language and experiments were conducted on a 1.7MHz Intel Pentium IBM laptop with 512MB main memory, running Microsoft Windows XP and Borland JBiulder 9.0 as the Java execution environment. Detailed algorithm implementation of PrefixSpan and I-PrefixSpan is according to the algorithms described in [4,9], but with the pseudo projection turned off; MG-PrefixSpan is implemented as described in this paper.

5.2 Synthetic dataset generation

In this work we extended the IBM synthetic generator described in [1,10] to generate synthetic data sets. Basically, each data-sequence is a list of transactions, where each transaction is a set of items, called itemset. However, the transaction data is extended so that the items in different itemsets are assigned different time values and that those in the same itemsets are assigned the same time values. The interval time value between successive itemsets for each customer obeys a Poisson distribution with mean w. the value w is drawn repetitively from a Poisson distribution with mean T_T for this particular customer [11]. After that, we determine the time t for each transaction of this customer by summing the interval times before the transaction.

Table 1 lists the parameters used in the generation of the simulating dataset. The parameters except for the last one are the classical ones used in the previous research. The parameter T_T is a new one used to generate the time for each transaction.

We generate datasets by setting $N_S = 500$, $N_I = 2500$, $N = 1000$ and $T_T = 60$. Table 2 summarizes the dataset parameter settings.

Table 3: PART OF THE EXACTED PATTE

Extracted patterns	**Support**(%)
$< (80) >$	22.89
$< (80) \overset{[1,3,6]day}{\rightarrow} (80) >$	1.71
$< (80) \overset{[1,2,5]day}{\rightarrow} (80) >$	1.97
$< (80) \overset{[1,3,6]day}{\rightarrow} (80) \overset{[1,2,5]day}{\rightarrow} (80) >$	0.54
$< (80) \overset{[2,3,6]day}{\rightarrow} (569) >$	0.62
$< (80) \overset{[1,2,4]day}{\rightarrow} (569) >$	0.84
$< (80, 432) >$	1.19
$< (80, 432) \overset{[1,2,3]day}{\rightarrow} (944) >$	0.60

Table 4: PART OF THE EXACTED PATTERN

Extracted patterns	**Support**(%)
$< (80) >$	22.89
$< (80) \rightarrow (80) >$	3.03
$< (80) \rightarrow (80) \rightarrow (80) >$	0.71
$< (80) \rightarrow (569) >$	1.39
$< (80, 432) >$	1.19
$< (80, 432) \rightarrow (944) >$	0.62

5.3 Time Granularity Selection

In the experiment, we use multi-granularity schema: G_3 = (Week, Day, Hour), and all the timestamps in the synthetic data sets are Hour, i.e. g_0 = Hour.

In order to compare the performance of algorithm I-PrefixSpan to that of MG-PrefixSpan, all the intervals between successive items are partitioned into five intervals: $\{l_0, l_1, l_2, l_3, l_4\}$, where $l_0 : t = 0 : 0 < t < 1$, $l_2 : 1 \leq t < 24$, $l_3 : 24 \leq t < 168$, and $l_4 : 168 \leq t < \infty$.

5.4 Comparison of Extracted Sequential Pattern Quality

The first test is a comparison of the quality of extracted patterns by algorithm PrefixSpan[6], I-PrefixSpan[11] and MG-PrefixSpan. The test is designed using the data set C10T4S5I2 and all minimum support thresholds is set to 0.005(0.5%). Table 3 shows part of the extracted frequent multi-granularity sequence patterns using MG-PrefixSpan algorithm, and Tables 4 and 5 show part of the extracted frequent sequential patterns using the algorithm PrefixSpan and I-PrefixSpan, respectively.

As shown in Table 3, we can made the following observation:

(1) once item 80 occurs, then item 80 will occur again within 1 day to 6 days, or with average time 3 days, with probability $(1.71/22.89) \times 100\% = 7.4\%$; within 1 week to 5 weeks, or with average time 2 weeks, with probability $(1.97/22.89) \times 100\% = 8.6\%$.

(2) once item 80 occurs and item 80 occurs within 1day to 6 days, or with average time 3 days, then item 80 will occur again within 1 week to 5 weeks, or with average time 2 weeks, with probability $(0.54/1.19)100\% = 45.4\%$.

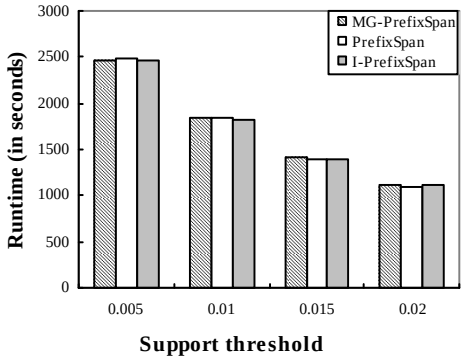


Figure 1: Performance of the of the algorithms on data set C10T2S4I1.

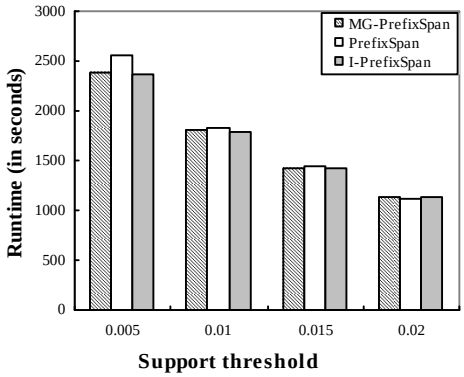


Figure 2: Performance of the algorithms on data set C10T2S4I2.

(3) once item 80 occurs, then item 569 will occur within 2 days to 6 days, or with average time 3 days, with probability $(0.62/22.89) \times 100\% = 2.7\%$; within 1 week to 4 weeks, or with average time 2 weeks, with probability $(0.84/22.89) \times 100\% = 3.7\%$.

(4) once item 80 and 432 occur, then item 94 will occur within 1 days to 3 days, or with average time 2 days, with probability $(0.60/1.19) \times 100\% = 50.4\%$;

As shown in Table 4, although the patterns are similar to those in Table 3, the time intervals between successive itemsets are not tighter than those in table 3. Thus users cannot precisely predict when items 569 and 944 occur and 80 occurs again. On the other hand, as shown in Table 5, there is no item annotation information based on multi-granularities in extracted sequences. Thus, users are not able to predict how long time item 569, 944 will occur and item 80 will occur again. Furthermore, the pattern $< (80) \rightarrow (569) >$ also makes users unable to distinguish periods of item 80 and 569 occurrence.

These results indicate that the multi-granularity sequence patterns mined by algorithm, MG-PrefixSpan, are more useful than those mined by algorithms, I-PrefixSpan, or PrefixSpan.

5.5 Comparison of the Execution Time

The second test of the three algorithms would compare the run times for different minimum supports. The comparison is on the five data sets shown in the table 2, where the minimum support threshold is varied from 0.5% to 2%. Fig.1 to Fig. 5 summarizes the results. It is clearly show that how the performance of the three algorithms changes as varying of the parameters $|\mathbf{C}|$, $|\mathbf{T}|$, $|\mathbf{S}|$ and $|\mathbf{I}|$, and the differences among the three algorithms.

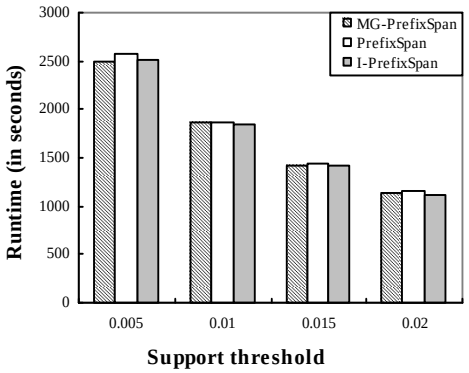


Figure 3: Performance of the algorithms on data set C10T2S8I1.

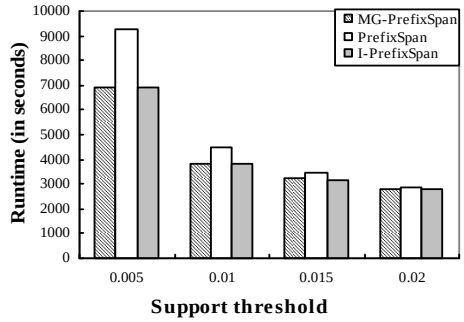


Figure 4: Performance of the algorithms on data set C10T4S4I1.

The results in the Fig. 1 Fig. 3 indicate that the processing time of algorithm MG-PrefixSpan is at different support threshold is no significant difference. The efficiency of MG-PrefixSpan is little less than that of the I-PrefixSpan. This result matches our expectation, because the MG-PrefixSpan algorithm does some more complicated computes for the bounds and average interval time in different granularities than that of the I-PrefixSpan.

On the other hand, the results in the Fig. 1, Fig. 2 and Fig. 3 show that the run time of three algorithms at different support threshold is also no significant difference. The speed of MG-PrefixSpan and I-PrefixSpan are little less than that of PrefixSpan. It is correct, because the algorithm PrefixSpan does not some complicated computes for interval time.

When we see the Fig. 3, Fig. 4 and Fig. 5 we can find that the run time of the three algorithms is increase rapidly as the minimum support threshold values vary from small to large. It is correct, because larger number of frequent sequential patterns could be found as the minimum support threshold value became smaller. However, it is worth note that the algorithm MG-PrefixSpan and I-PrefixSpan take less time than the PrefixSpan algorithm and the time difference of processing become larger as the minimum support threshold declines although the MG-PrefixSpan and I-PrefixSpan are more complicated than the PrefixSpan. In order to find the reason, Let us note the data sets. Fig. 3 test is performed on the data set C10T2S8I1, where the parameter average length of maximal potentially large sequences S increased from 4(Fig.1) to 8. On average, a potentially frequent sequential pattern consists of 8 transactions, which mean that although the average number of transactions in a potentially frequent sequential pattern is set to 8, the number of transactions in a sequence is still set to 10. This situation can not cause the number and length of frequent sequential pattern increase considerably, because the data set C10T2S8I1 (1.746MB) is as sparse as C10T2S4I1 (1.733MB); Fig.4 on the data set C10T4S4I1, where the parameter average number

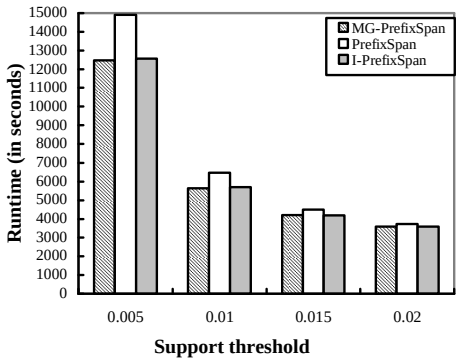


Figure 5: Performance of the algorithms on data set C20T2S4I1.

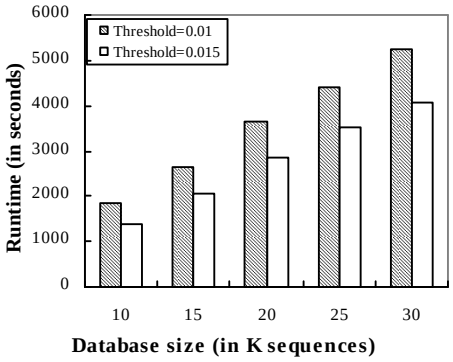


Figure 6: Scalability test of the algorithm MG-PrefixSpan on data set T2S4I1.

of items per transaction T increased from 2 (Fig.1) to 4, which mean, on average, a sequence consists of 10 transactions, each transaction composed of 4 items. It may result in the number of frequent sequential pattern increasing significantly; Fig.5 on the data set C20T2S4I1, where the parameter average number of transactions of per customer C increased from 10 (Fig.1) to 20, which mean, on average, a sequence consists of 20 transactions, each transaction composed of 2 items. It may result in the length of frequent sequential pattern increasing greatly. This both situations may cause the number and length of frequent sequential pattern increase considerably, because the data sets C10T4S4I1 and C20T2S4I1 are denser than the C10T2S8I1 and C10T2S4I1. When we use PrefixSpan to find all frequent sequential patterns, much more time would be spent in dealing with these more frequent sequential patterns although many of frequent sequential patterns may be useless. On the contrary, to the MG-PrefixSpan and I-PrefixSpan, although adding multiple time granularities or pseudo items to the traditional sequential pattern may make each of them to produce the several multiple granularities or time-interval patterns and need spend some time to does some complicated computation, many of them may not be frequent and some patterns found frequent by PrefixSpan may be infrequent by MG-PrefixSpan and I-PrefixSpan too. This reason reducing the numbers of frequent sequential patterns is why the MG-PrefixSpan and I-PrefixSpan are faster than the PrefixSpan.

5.6 Related global performances

The final test is the scalabilities of the MG-PrefixSpan algorithm. Four tests are designed using the data set C10T2S4I1 and all minimum support thresholds are set to 0.01 and 0.015. In each test, test how the runtime of the MG-PrefixSpan algorithm scales as one parameter is increased. Fig. 6 shows the result

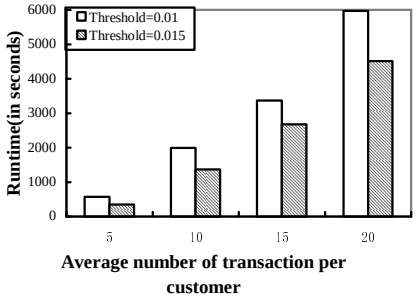


Figure 7: Performance of the algorithms on data set T2S10I1.

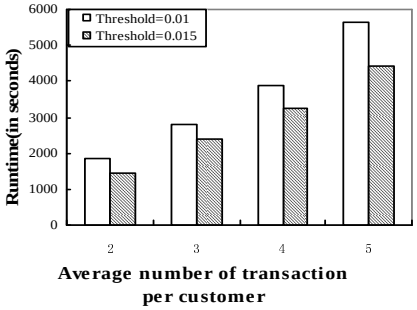


Figure 8: Performance of the algorithms on data set C10S10I1.

of scalability of the algorithm as the data set size grows from 10000 to 30000; Fig.7 shows the result as the average number of transactions of per customer varies from 5 to 20; Fig. 8 shows the result as the average number of items per transaction varies from 2 to 5. The results indicate that the MG-PrefixSpan algorithm has good scalabilities for its runtime increases linearly as each parameter varies from small to large respectively.

6 Conclusions

In this paper, we have proposed a novel approach for mining multi-granularities sequence patterns based on PrefixSpan [6], called MG-PrefixSpan. The multi-granularities sequence pattern not only reveals what items occur frequently together and in what order, but also the time interval that the next items occur after the preceding items. We use boundary interval and average time with multi-granularities, which are derived from the source data, rather than user-predetermined the time interval or only a typical time to annotate time interval between successive items in the patterns. The performance analysis shows that MG-PrefixSpan scales up linearly as the size of database, and has a good scalability with respect to length of sequence and the size of transaction.

The future work along this line of research includes several aspects as follows: (1) validation on large, real databases; (2) low-level optimizing mechanism of the algorithm.

Acknowledgment

This work was supported by the Department of Education of Shaanxi Province of China under Grant 05JK137 and the Natural Science Foundation of Shaanxi Province of China under Grant 2005F11.

Bibliography

[1] Constantinescu, Z, Marinoiu, C, Vladoiu, M, "Driving Style Analysis Using Data Mining Techniques," International Journal Of Computers Communications and Control, Vol.5, No.5, pp. 654-663, Dec 2010.

[2] Andonie, R, "Extreme Data Mining: Inference from Small Datasets," International Journal Of Computers Communications and Control, Vol.5, No.3, pp. 280-291, Sep 2010.

[3] R. Agrawal and R. Srikant, "Mining sequential patterns," Proc. of the 7th International Conference on Data Engineering (ICDE'95), pp. 3-14, March, 1995.

[4] R. Srikant and R. Agrawal, "Mining sequential patterns: Generalizations and performance improvements," Proc. of the 5th International Conference on Extending Database Technology, pp. 3-17, March, 1996.

[5] M. Zaki, "An Efficient Algorithm for Mining Frequent Sequences," Machine Learning, Vol. 40, pp. 31-60, 2000.

[6] J. Pei, J. Han and H. Pinto et al, "Mining Sequential Pattern-Growth: The PrefixSpan Approach," IEEE Transactions on Knowledge and Engineering, Vol.16, No.11, pp. 1424-1440, 2004.

[7] H. Manila, H. Toivonen, and A.I. Verkamo, "Discovery of frequent episodes in event sequences," Data Mining and Knowledge Discovery, Vol. 1, No. 3, pp. 256-289, 1997.

[8] M.N. Garofalakis, R. Rastogi and K. Shim, "SPIRIT: Sequential Pattern Mining with Regular Expression Constraints," Proc. of the 25th International Conference on Very Large Data Bases (VLDB'99), pp. 223-234, September, 1999.

[9] J. Pei, J. Han and W. Wang, "Constraint-based Sequential Pattern Mining: The Pattern-growth Methods," Journal of Intelligent Information Systems, Volume 28, Issue 2, pp. 133-160, April, 2007.

[10] M. Yoshida et al. "Mining sequential patterns including time intervals", Proc. of SPIE Conf.-DMKD, pp. 213-220, April, 2000.

[11] Y.-L. Chen, M.-C. Chiang and M.-T. Ko, "Discovering time-interval sequential patterns in sequence databases," Expert System with Applications, Volume 25, Issue3, Pp. 343-354, October, 2003.

[12] R. Algawal and R. Srikant, "Fast algorithm for mining association rules in Large Databases.," Proc. of the 20th International Conference on Very Large Data bases (VLDB'94), pp. 487-499, September 1994.

[13] Y. Hirate, H. Yamana, "Generalized Pattern Mining with Item Intervals," Journal of Computers, Vol.1, No3, pp. 51-60, June, 2006.

[14] F. Giannotti, M. Nanni, and D. Pedreschi. "Efficient Mining of Temporally Annotated Sequences," Proc. of the 6th SIAM International Conference on Data Mining, pp. 346-357, April, 2006.

[15] C. Bettini, X.S. Wang and S. Jajodia et al, "Discovering Temporal Relationships with Multiple Granularities in Time Sequences," IEEE Transactions on Knowledge and Data Engineering, Vol. 10 (2), pp. 222-237, 1998.

INT J COMPUT COMMUN, ISSN 1841-9836
Vol.7 (2012), No. 3 (September), pp. 509-517

An Entropy-based Method for Attack Detection in Large Scale Network

T. Liu, Z. Wang, H. Wang, K. Lu

Ting Liu
SKLMS Lab and MOE KLNNIS Lab,
Xi'an Jiaotong University
Xi'an, Shaanxi, 710049, P.R.China
E-mail: tingliu@mail.xjtu.edu.cn

Zhiwen Wang, Haijun Wang, Ke Lu
MOE KLNNIS Lab, Xi'an Jiaotong University
Xi'an, Shaanxi, 710049, P.R.China
E-mail: wzw@mail.xjtu.edu.cn
{hjwang,klu}@sei.xjtu.edu.cn

Abstract:
Intrusion Detection System (IDS) typically generates a huge number of alerts with high false rate, especially in the large scale network, which result in a huge challenge on the efficiency and accuracy of the network attack detection. In this paper, an entropy-based method is proposed to analyze the numerous IDS alerts and detect real network attacks. We use Shannon entropy to examine the distribution of the source IP address, destination IP address, source threat and destination threat and datagram length of IDS alerts; employ Renyi cross entropy to fuse the Shannon entropy vector to detect network attack. In the experiment, we deploy the Snort to monitor part of Xi'an Jiaotong University (XJTU) campus network including 32 C-class network (more than 4000 users), and gather more than 40,000 alerts per hour on average. The entropy-based method is employed to analyze those alerts and detect network attacks. The experiment result shows that our method can detect 96% attacks with very low false alert rate.
Keywords: Network Security, Entropy-based, IDS, Shannon Entropy, Renyi Cross Entropy.

1 Introduction

Network attacks are defined as the operations that disrupt, deny, degrade, or destroy information resident in computer networks or the networks themselves. In recent years, more and more network attacks threatened the reliability and QoS of Internet, compromised the information security and privacy of users. KSN (Kaspersky Security Network) recorded 73 million Internet browsers attacks on their users in 2009, and that number skyrocketed to 580,371,937 in 2010 [1]. Symantec reported that they recorded 3 billion attacks from their global sensor and client [2].

Intrusion Detection System (IDS) is used to monitor and capture intrusions into computer and network systems which attempt to compromise their security [3]. With the development of networks, a large number of computer intrusions occur every day and IDSs have become a necessary addition to the security infrastructure of nearly every organization. However, IDSs still suffer from two problems: 1) large amount of alerts. In fact, more than 1 million alerts are generated by Snort each day in our research; 2) high false alerts rate. Gina investigated the extent of false alerts problem in Snort using the 1999 DARPA IDS evaluation data, and found that 69% of total generated alerts are considered to be false alerts [4]. These problems result in a huge challenge on the efficiency and accuracy of the network attack detection.

Several methods have been applied to resolve the problems of large amount of alerts and high false rate. Pietraszek used the adaptive alert classifier to reduce false alerts, which is trained with lots of labeled

past alerts [5]. Whereas, it is difficult to label large volume alerts generated in large-scale network. In order to reduce the false alarms, Mina propose the extend DPCA to standardize the observations according to the estimated means [6]. Spathoulas and Katsikas propose a post-processing filter based on the statistical properties of the input alert set [7]. Cisar employ EWMA to detect attacks by analyzing the intensity of alerts [3]. In our research, 32 C-class subnets are monitored by Snort and more than 1 million alerts are generated every day. Therefore, we propose a method to spot anomalies which is more tolerable for the operator rather than reduce false alerts.

In information theory, entropy is a measure of the uncertainty associated with a random variable, which is widely used to analyze the data and detect the anomalies in information security. Lakhina et al argue that the distributions of packet features (IP addresses and ports) observed in flow traces reveal both the presence and structure of a wide range of anomalies. Using entropy as a summarization tool to analyze traffic from two backbone networks, they found that it enables highly sensitive detection of a wide range of anomalies, augmenting detections by volume-based methods [8]. Brauckhoff ind that entropy-based summarizations of packet and flow counts are affected less by sampling than volume-based method in large networks [9]. A. Wagner and B Plattner applied entropy to detect worm and anomaly in fast IP networks [10]. Relative entropy and Renyi cross entropy can be used to evaluate the similarity of different distributions. Yan et al use a traffic matrix to represent network state, and use Renyi cross entropy to analyze matrix traffic and detect anomalies rather than Shannon entropy. The results show

Renyi cross entropy based method can detect DDoS attacks at the beginning with higher detection rate and lower false rate than Shannon entropy based method [11]. Gu et al proposed an approach to detect anomalies in the network traffic using Maximum Entropy estimation and relative entropy [12]. The packet distribution of the benign traffic was estimated using Maximum Entropy framework and used as a baseline to detect the anomalies.

In this paper, an entropy-based method is proposed to detect network attack. The Shannon entropy and Renyi cross entropy are employed to analyze the distribution characteristics of alert features and detect network attack. The experimental results under actual network data show that this method can detect network attack quickly and accurately. The rest of the paper is organized as follows: the method is introduced in Section 2, and the experimental results are shown in Section 3. Section 4 is the conclusion and future work.

2 Methodology

In this paper, Snort is used to monitor the network and five statistical features of the Snort alert are selected. The Shannon entropy is used to analyze the distribution characteristics of alert that reflect the regularity of network status. When the monitored network runs in normal way, the entropy values are relatively smooth. Otherwise, the entropy value of one or more features would change. The Renyi cross entropy of these features is calculated to measure the network status and detect network attacks.

2.1 Snort Alert and Feature Selection

Each Snort alert consists of tens of attributions, such as *timestamp*, *source IP address* (*sip*), *source port*, *destination IP address* (*dip*), *destination port*, *priority*, *datagram length* and *protocol*, etc. Suppose there are n alerts generated in time interval t. The alerts set in time interval t is denoted as $Alert(t) = \{alert_1, alert_2, \ldots, alert_n\}$.

Assuming there are m distinct *sip* and k distinct *dip* in $Alert(t)$, we can generate the distinct source IP addresses set (*SIP*) and distinct destination IP addresses set (*DIP*):

$$SIP = \{sip_1, sip_2, \ldots, sip_m\},$$

$$DIP = \{dip_1, dip_2, \ldots, dip_k\}.$$

Suppose the number of alerts come from sip_i is $snum_i$, and the number of alerts send to dip_i is $dnum_i$. The alert number of each source IP ($SNUM$) and destination IP ($DNUM$) can be calculated:

$$SNUM = \{snum_1, snum_2, \ldots, snum_m\},$$

$$DNUM = \{dnum_1, dnum_2, \ldots, dnum_k\}.$$

There are 4 default priorities of Snort alert: 1, 2, 3 and 4. The threat severity gradually weakens from 1 to 4(high, medium, low, info). In order to strengthen the threat degree of high severity alerts, the threat degree of the $alert_i$ is denoted as $threat_i = 5^{(4-priority_{alert_i})}$ in present work. Suppose the threat degree sum of all alerts come from sip_i is $sthreat_i$, and the threat degree sum of all alerts send to dip_i is $dthreat_i$. The threat degree of each source IP ($STHREAT$) and destination IP ($DTHREAT$) can be calculated:

$$STHREAT = \{sthreat_1, sthreat_2, \ldots, sthreat_m\},$$

$$DTHREAT = \{dthreat_1, dthreat_2, \ldots, dthreat_k\}.$$

The datagram length is the size of the packet that breaks the alarm rules of Snort. We search the distinct datagram length of all alerts, and generate the datagram length set

$$DGMLEN = \{dgmlen_1, dgmlen_2, \ldots, dgmlen_x\},$$

where x is the number of the distinct datagram length of all alerts. Suppose the number of alerts whose datagram length equal to $dgmlen_i$ is $dgmNum_i$. The alert number with different datagram length can be calculated:

$$DGMNUM = \{dgmNum_1, dgmNum_2, \ldots, dgmNum_x\}.$$

Above 5 features ($SNUM, DNUM, STHREAT, DTHREAT, DGMNUM$) are selected to evaluate the alerts and detect attacks.

2.2 Shannon Entropy-based Feature Analysis

Shannon entropy is used as measures of information and uncertainty [13]. For a dataset $X = \{x_1, x_2, x_3, \ldots, x_n\}$, each data item x belongs to a class $x \in C_x$. The entropy of X relative to C_x is defined as

$$H(X) = -\sum_{i=1}^{n} p_i \log_2 p_i \tag{1}$$

where p_i is the probability of x_i in X.

The distribution characteristics of five features are analyzed using Shannon entropy. The entropies of $SNUM$ and $DNUM$ in time interval t can be calculated

$$H(Sip_t) = -\sum_{i=1}^{m} (snum_i/n) \log(snum_i/n) \tag{2}$$

$$H(Dip_t) = -\sum_{i=1}^{k} (dnum_i/n) \log(dnum_i/n) \tag{3}$$

The entropy of $STHREAT$ and $DTHREAT$ can be calculated:

$$H(Sthreat_t) = -\sum_{i=1}^{m} \frac{threat\_of\_sip(i)}{sum\_threat} \cdot \log\left(\frac{threat\_of\_sip(i)}{sum\_threat}\right) \tag{4}$$

$$H(Dthreat_t) = -\sum_{i=1}^{k} \frac{threat\_of\_dip(i)}{sum\_threat} \cdot \log\left(\frac{threat\_of\_dip(i)}{sum\_threat}\right) \tag{5}$$

where $threat\_of\_sip(i)$ is the threat sum of the alerts from sip_i, $threat\_of\_dip(i)$ is the threat sum of the alerts to dip_i, and sum_threat is the threat sum of all the alerts in ALERTS which can be calculated using

$$sum\_threat = \sum_{i=1}^{n} threat_i \tag{6}$$

The entropy of datagram length is

$$H(Dgmlen_t) = -\sum_{i=1}^{x} (dgmNum_i/n) \cdot \log(dgmNum_i/n) \tag{7}$$

After calculating the entropies of above features, we can use an entropy vector $V(t) = [H(Sip_t), H(Dip_t), H(Sthreat_t), H(Dthreat_t), H(Dgmlen_t)]$ to represent the network status of time interval t.

2.3 Renyi Cross Entropy-based Attack Detection

The Renyi entropy, a generalization of Shannon entropy, is a measure for quantifying the diversity, uncertainty or randomness of a system. The Renyi entropy of order α is defined as

$$H_\alpha(P) = \frac{1}{1-\alpha} \log_2 \sum_r p_r^\alpha \tag{8}$$

where $0 < \alpha < 1$, P is a discrete stochastic variable, and p_r is the distribution function of P [14]. Higher values of α, approaching 1, giving a Renyi entropy which is increasingly determined by consideration of only the highest probability events. Lower values of α, approaching zero, giving a Renyi entropy which increasingly weights all possible events more equally, regardless of their probabilities. The special case $\alpha \to 1$ gives the Shannon entropy. The Renyi cross entropy of order α is derived as

$$I_\alpha(p, q) = \frac{1}{1-\alpha} \log_2 \sum_r \frac{p_r^\alpha}{q_r^{\alpha-1}} \tag{9}$$

where p and q are two discrete variables, p_r and q_r are their distribution functions [14]. If $\alpha = 0.5$, the Renyi cross entropy is symmetric, which means $I_\alpha(p, q) = I_\alpha(q, p)$. In the rest of the paper, when referring to the cross entropy we mean the symmetric case

$$I_{0.5}(p, q) = 2 \log_2 \sum_r \sqrt{p_r q_r} \tag{10}$$

The Renyi cross entropy is used to fuse the values of different features. As mentioned above, we use an entropy vector $V(t) = [H(Sipt), H(Dipt), H(Sthreatt), H(Dthreatt), H(Dgmlent)]$ to represent the network status of time t, thus the network status can be viewed as a time series of entropy vector $V(1), V(2), \ldots, V(t)$. Before calculating Renyi cross entropy, $V(t)$ is unitized to

$$\bar{V}(t) = [\bar{H}(Sip_t), \bar{H}(Dip_t), \bar{H}(Sthreat_t), \bar{H}(Dthreat_t), \bar{H}(Dgmlen_t)] \tag{11}$$

where

$$\begin{aligned}
\bar{H}(Sip_t) &= H(Sip_t)/H_{sum} \\
\bar{H}(Sthreat_t) &= H(Sthreat_t)/H_{sum} \\
\bar{H}(Dip_t) &= H(Dip_t)/H_{sum} \\
\bar{H}(Dthreat_t) &= H(Dthreat_t)/H_{sum} \\
\bar{H}(Dgmlen_t) &= H(Dgmlen_t)/H_{sum}
\end{aligned} \tag{12}$$

and $Hsum = H(Sip_t) + H(Dip_t) + H(Sthreat_t) + H(Dthreat_t) + H(Dgmlen_t)$.

To determine if there is any change in the network at time t compare with previous time $t-1$, we use the following equation to calculate the Renyi cross entropy of $\bar{V}(t)$ and $\bar{V}(t-1)$

$$I_{0.5}(\bar{V}(t), \bar{V}(t-1)) = 2\log_2 \sum_r \sqrt{p_r(t-1)p_r(t)} \tag{13}$$

We set η as the threshold of $\left|I_{0.5}(\bar{V}(t-1), \bar{V}(t))\right|$ to test whether there is a change. The choice of threshold η is network dependent and it can be set as experience. Since our purpose is to detect network attack, it is not enough to compare network status of time t to its previous time $t-1$, unless we make sure that no attack occurs in time $t-1$. Thus, the average of the latest n normalized Shannon Entropies is employed to replace the $t-1$, called $\bar{V}(t,n)$

$$\bar{V}(t,n) = \frac{1}{n}\sum_{i=1}^{n} \bar{V}(t-i) \tag{14}$$

Then, we calculate the Renyi cross entropy of $\bar{V}(t)$ and $\bar{V}(t,n)$, and network attack is detected if its absolute is greater than η.

$$I_{0.5}(\bar{V}(t,n), \bar{V}(t)) = 2\log_2 \sum_r \sqrt{p_r(t,n)p_r(t)} \tag{15}$$

3 Experiment Results

3.1 Data Collection

In the research, we have used Snort to monitor 32 C-class subnets in the Xi'an Jiaotong University campus network for two weeks, which include more than 4,000 users. In this paper, we select the alerts gathered in 2010-12-6. There are 862,284 alerts with 65 signatures, which come from 42,473 distinct source IP addresses and send to 11,790 distinct destination IP addresses.

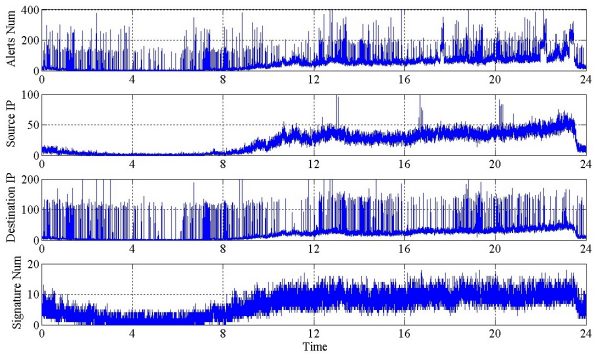


Figure 1: The statistical results of alerts (2010-12-6).

As shown in Fig.1, four statistical features of alerts display the trend as the people living customs and habits (the time interval set as 5 seconds). Few alerts are generated in the middle night; then, more alerts are detected from 8:00 to 10:00 when students get up successively; the alerts keep the same trend from 10:00 to 23:30; the alerts collapse at last 30 minutes, since network constraint due to the dormitory administrating rules.

At the same time, the statistical features change abruptly in some time intervals. In general, these abnormal upheavals are the sign of the faults or network attacks.

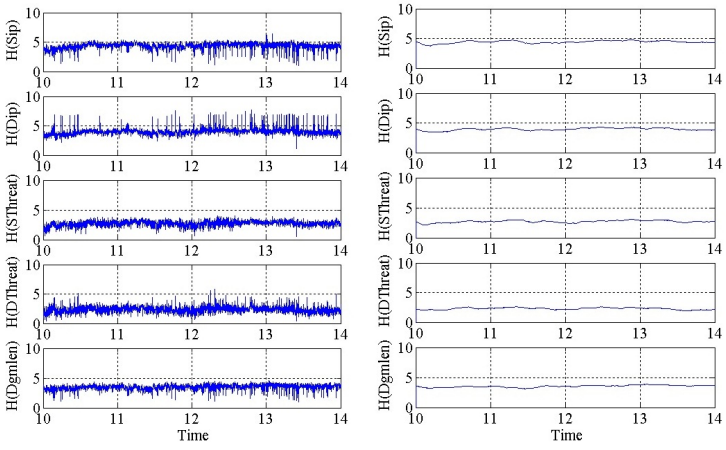
We select two alerts sets in different time period as training and test data set:

Training data set includes 170,516 alerts generated from 10:00 to 14:00. These alerts come from 13,148 IP addresses and send to 7,570 IP addresses. By analyzing these alerts manually, we identify 87 host scan attacks, 5 port scan attacks, 1 DoS attack and 1 host intrusion.

Test data set includes 578,389 alerts generated from 14:00 to 23:30. These alerts come from 29,327 IP addresses and send to 10,590 IP addresses. By analyzing these alerts manually, we identify 203 host scan attacks, 7 port scan attacks, 6 DoS attack, 3 host intrusion and 1 worm attack.

3.2 Entropy-based Attack Detection

The training data is evaluated by Shannon entropy, as shown in Fig. 2 (a). We remove the alerts associated to true attacks, which called as *Attack Alert*. The remainders are called as *Flase Alert*. We re-evaluate the Noise Alert in the training data set, as shown in Fig. 2 (b). The Shannon entropies are relatively smooth when no attack occurs; otherwise, one or some of the values would change abruptly.



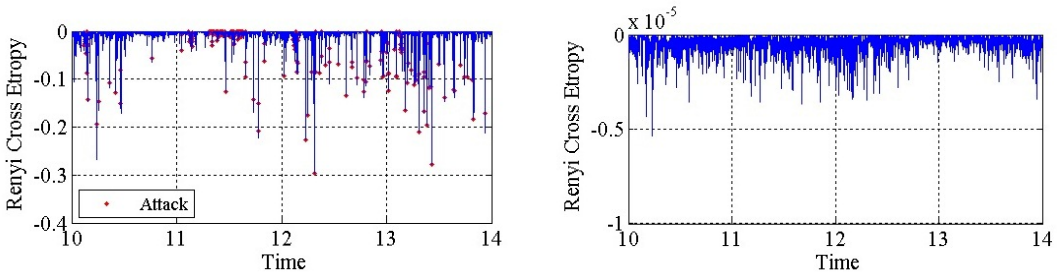
(a) All alerts (b) False alerts

Figure 2: Shannon entropy.

Although the Shannon entropies reflect the regularity of network status, it is difficult to detect attack directly by using five fixed thresholds. Because the Shannon entropy value varies with the activities of end users even the network runs in normal way. In our experiment, the Renyi cross entropy is used to fuse the Shannon entropy of five statistical features to detect attack. As shown in Fig. 3, we calculate the Renyi cross entropy of the alerts in train data set using (13). It is clearly shown that 1) the Renyi cross entropy will change sharply when the network are attacked, see Fig. 3 (a); 2) the Renyi cross entropy will be close to 0 without the large-scale network attacks and failures, see Fig. 3 (b). Thus, it is easy to detect attack using fixed threshold.

In the experiments, when $\eta_{detect} = -0.016$, 84 attacks can be detected from 94 attacks with 11 false detections. 81 host scan attacks can be detected from 87 host scans. The missed scan attacks last for a relative long time and with small scan density. 1 port scan is detected from 5 port scans. 1 host intrusion and 1 DoS attack are detected successfully.

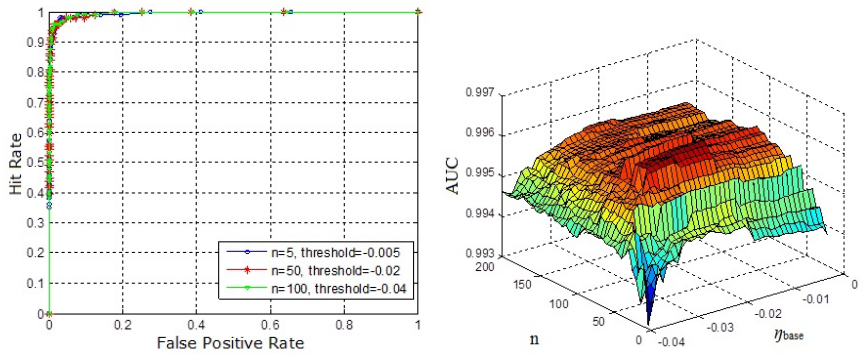
According to (14) and (15), the n and η are important for the accuracy of attack detection. In the experiments, we set $\eta_{base} = \{-0.001, -0.002, -0.003, \ldots, -0.04\}$ and $n = \{5, 10, 15, \ldots, 200\}$. For each combination of η_{base} and n, the training data is analyzed in the following method. Firstly, each $V(t)$ is unitized to $\bar{V}(t)$ using (11) and (12); Secondly, the Shannon entropy can be calculated using (14). Its unitized form is $\bar{V}(t, n)$. Finally, $\bar{V}(t)$ is compared with $\bar{V}(t, n)$ using (15) to calculate Renyi cross entropy value.



(a) All alerts (b) False alerts

Figure 3: Renyi cross entropy.

In the experiment, ROC (Receiver Operating Characteristic) is used to describe the detection results. ROC is a graphical plot of true positive rate and false positive rate [15]. Fig. 4(a) shows the ROC curve of detection results in training data, where the size of NTS n and base threshold η_{base} equals (5, 0.005), (50, 0.02) and (100, 0.04) separately. When detection threshold η_{detect} comes to 0, almost all the time intervals are detected as network attack. Thus, the detection false positive rate and hit rate are both near 100%. A detection result with high hit rate and low false rate is considered to be a good result. In this case, the ROC curve is plotted at the top left corner, and the AUC value (Area Under ROC Curve) has large value. In this paper, we use AUC value to evaluate the detection results. The best combination of n and ?base can be obtained using training data. As shown in Fig. 4(b), the AUC values of all the combinations are calculated, and the highest AUC is 0.9962 when $n = 95$ and $\eta_{base} = -0.022$.



(a) ROC (b) AUC

Figure 4: Detection result on training data set.

3.3 Testing

The test data set is analyzed to detect the attacks using entropy-based method. As shown in Fig. 5, 211 attacks can be detected from 220 attacks (detection rate is as high as 96%) with 8 false detections. 197 host scan attacks can be detected from 203 host scans. 4 port scans are detected from 7 port scans. 3 host intrusions, 1 worm attack and 6 DoS attacks are detected successfully.

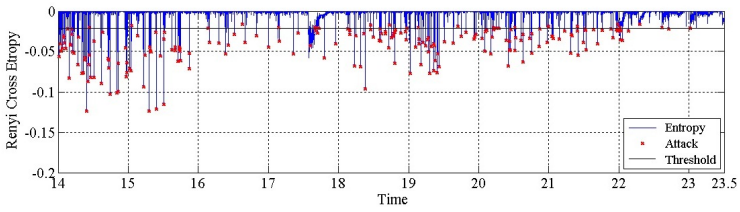


Figure 5: Attack detection results on test data set.

4 Conclusion

In this paper, a new network attack detection method based on entropy is proposed. The source IP, destination IP, alert treat and alert datagram length are selected from tens of Snort alert attributions. The Shannon entropy is used to analyze the alerts to measure the regularity of current network status. The Renyi cross entropy is employed to fuzz the Shannon entropy on different features to detect network attacks.

In the experiments, the network traffic of more than 4000 users in 32 C-class network are monitored using Snort. 748905 alerts, generated from 10:00 to 23:30 Dec. 6 2010, are selected and separated into training data set and test data set. The experiments show that the Renyi cross entropy value is near 0 when the network runs in normal, otherwise the value will change abruptly when attack occurs. The attack detection rate of entropy method is as high as 96% with only 8 false alerts.

In next step, more alerts from different time segments will be collected to test our method and an attack classification method will be considered.

Acknowledgment

This work was supported by the National Natural Science Foundation (60921003, 60970121, 91018011), National Science Fund for Distinguished Young Scholars (60825202) and the Fundamental Research Funds for the Central Universities.

Bibliography

[1] A. Gostev, "Kaspersky Security Bulletin. Malware Evolution 2010," Kaspersky, 2011.

[2] M. Fossi, G. Egan, K. Haley, E. Hohnson, T. Mack and A. Et, "Symantec Global Internet Security Threat Report Trends for 2010," Symantec, 2011.

[3] P. Cisar, S. Bosnjak and S. M. Cisar, "EWMA Algorithm in Network Practice," International Journal of Computers, Communications & Control, vol.5, pp. 160-170, 2010.

[4] G. C. Tjhai, M. Papadaki, S. M. Furnell and N. L. Clarke, in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Turin, Italy, 2008, pp. 139-150.

[5] T. Pietraszek, "Using Adaptive Alert Classification to Reduce False Positives in Intrusion Detection-Recent Advances in Intrusion Detection," vol.3224, pp. 102-124, 2004.

[6] J. Mina and C. Verde, "Fault detection for large scale systems using Dynamic Principal Components Analysis with adaptation," International Journal of Computers, Communications & Control, vol.2, pp. 185-194, 2007.

[7] G. P. Spathoulas and S. K. Katsikas, in 2009 16th International Conference on Systems, Signals and Image Processing, IWSSIP 2009, Chalkida, Greece, 2009.

[8] A. Lakhina, M. Crovella and C. Diot, in Computer Communication Review, New York, United States, 2005, pp. 217-228.

[9] D. Brauckhoff, B. Tellenbach, A. Wagner, M. May and A. Lakhina, in Proceedings of the ACM SIGCOMM Internet Measurement Conference, IMC, Rio de Janeriro, Brazil, 2006, pp. 159-164.

[10] A. Wagner and B. Plattner, in Proceedings of the Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, WET ICE, Linkoeping, Sweden, 2005, pp. 172-177.

[11] R. Yan and Q. Zheng, "Using Renyi cross entropy to analyze traffic matrix and detect DDoS attacks," Information Technology Journal, vol.8, pp. 1180-1188, 2009.

[12] Y. Gu, A. McCallum and D. Towsley, "Detecting anomalies in network traffic using maximum entropy estimation," in Proc. 2005 Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement, pp. 32.

[13] C. E. Shannon, "A mathematical theory of communication," SIGMOBILE Mob. Comput. Commun. Rev., vol.5, pp. 3-55, 2001.

[14] C. E. Pfister and W. G. Sullivan, "Renyi entropy, guesswork moments, and large deviations," IEEE Transactions on Information Theory, vol.50, pp. 2794-2800, 2004.

[15] A. P. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," Pattern Recognition, vol.30, pp. 1145-1159, 1997.

INT J COMPUT COMMUN, ISSN 1841-9836
Vol.7 (2012), No. 3 (September), pp. 518-529

An Adaptive Iterative Learning Control for Robot Manipulator in Task Space

T. Ngo, Y. Wang, T.L. Mai, J. Ge, M.H. Nguyen, S.N. Wei

ThanhQuyen Ngo, M. Hung Nguyen
HCM City University of Industry
College of Electrical Engineering
HCM City, Vietnam
E-mail: thanhquyenngo2000@yahoo.com,
nmhung289@yahoo.com

YaoNan Wang, Ji Ge, ShuNing Wei
Hunan University
College of Electrical and Information Engineering
Changsha, Hunan Province 410082, P.R.China
E-mail: yaonan@hnu.cn, gechunxi@126.com,
weishuning@sina.com

T. Long Mai
HCM City University of Industry
College of Electronic Engineering
HCM City, Vietnam
E-mail: mailongtk@gmail.com

Abstract: In this paper, adaptive iterative learning control (AILC) of uncertain robot manipulators in task space is considered for trajectory tracking in an iterative operation mode. The control scheme incluces a PD controller with a gain switching technique plus a learning feedforward term, is exploited to predict the desired actuator torque. By using Lyapunov method, an adaptive iterative learning control scheme is presented for robotic system with both structured and unstructured uncertainty, and the overall stability of the closed-loop system in the iterative domain is established. The validity of the scheme is confirmed through a numerical simulation.
Keywords: PD Control; Learning control; robot manipulator.

1 Introduction

In general, robotic manipulators have to face various uncertainties in their dynamics, such as friction, and external disturbance. It is difficult to establish exactly mathematical model for the design of a model-based control system. In order to deal with this problem, the branches of current control theories are broad include classical control, robust control, adaptive control, optimal control, nonlinear control, neural network control, fuzzy logic and intelligent control. However, many adaptive control approaches are rejected as being too computationally intensive because of the required of real-time parameter identification and control design

Owing to its simplicity and robustness to modelling uncertainties, are usually used for repetitive tasks. In this case, the reference trajectory is repeated over a definite operation time. So, iterative learning control (ILC) for robot manipulator has attracted considerable attention in the recent years [1]-[3], the ideal of learning control is to improve the tracking performance from iteration to iteration for applications of robotics in industry which a single repetitive task [4], [5].

ILC is a relatively recent but well-established area of study in control theory. ILC, which can be categorized as an intelligent control methodology, is an approach for improving the transient performance of system that operates repetitively over a fixed time interval [4]. Starting from the classical

Arimoto-type ILC algorithm, we can develop a PID-like update law can be given in [6]. So far some of robot manipulators control in the published literature [7]-[9] and etc, proposed an adaptive ILC to deal with parameter uncertainties, such as the link length, mass inertia, and friction nonlinearity, with a self-organizing capability.

In this paper, a new method is given based on a combination of the advantages of a several control methods into a hybrid one. In particular, it is further extended to the task space or the so-called Cartesian space. To apply robot manipulators to a wide class of tasks, it will be necessary to control not only the position of the end-effector, but also the force exerted by the end-effector on the object. By designing the control law in task space, force control can be easily formulated.

This paper is organized as follows. Section 2 described a dynamic model of an n-link robot manipulator in task space. Section 3 presents AILC and its features are discussed. By using Lyapunov method to prove the asymptotic convergence of proposed controller. Numerical simulation results of a two-link robot manipulator in task space under the possible occurrence of uncertainties are provided to demonstrate the tracking control performance of the proposed AILC system in section 4. Conclusions are drawn in section 5.

2 Robotic Dynamic Model In Task Space

In general, the dynamic of an n-link robot manipulator may be expressed in the Lagrange form [10] as:

$$D(q^i(t))\ddot{q}^i(t) + C(q^i(t), \dot{q}^i(t))\dot{q}^i(t) + G(q^i(t), \dot{q}^i(t)) + \tau_a(t) = \tau^i(t) \tag{1}$$

With $t \in [0, t_f]$ denotes the time and $i \in N$ denotes the iteration, $q^i(t) \in R^n$, $\dot{q}^i(t) \in R^n$, $\ddot{q}^i(t) \in R^n$ are the joint position, joint velocity, and joint acceleration variables vector, respectively. $D(q^i(t)) \in R^{n\times n}$ is the inertia matrix, $C(q^i(t), \dot{q}^i(t)) \in R^n$ is the coriolis-ccentripetal matrix, is the gravity vector plus friction force vector. Bounded unknown disturbances are denoted by $\tau_a(t) \in R^n$ and the control input torque is denoted by $\tau^i(t) \in R^n$. For convenience, a two-link robot manipulator, as shown in Fig.1, is utilized to verify dynamic properties are given in section 4.

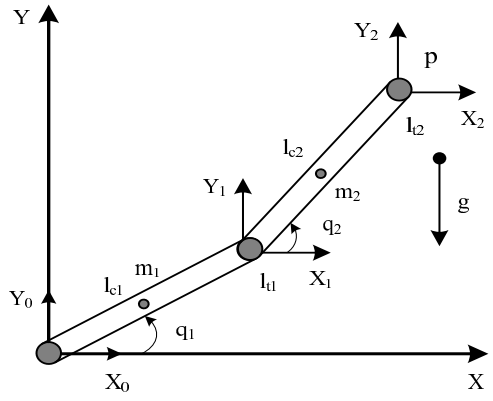


Figure 1: Architecture of two-link robot manipulator.

Usually, the manipulator task specification is given relative to the end-effector. Thus, it is natural to attempt to derive the control algorithm directly in task space, better than in the joint space. Denote the end-effector position and orientation in task space by $x \in R^n$. The task space dynamic can be rewritten as follow:

$$D_x(q^i(t))\ddot{x}^i(t) + C_x(q^i(t), \dot{q}^i(t))\dot{x}^i(t) + G_x(q^i(t), \dot{q}^i(t)) = \tau_a(t) + F_x{}^i(t) \tag{2}$$

Where

$$D_x(q^i(t)) = J^{-T}(q^i(t))D(q^i(t))J^{-1}(q^i(t))$$

$$C_x(q^i(t),\dot{q}^i(t)) = J^{-T}(q^i(t))(C(q^i(t),\dot{q}^i(t)) - D(q^i(t))J^{-1}(q^i(t))\dot{J}(q^i(t)))J^{-1}(q^i(t))$$
$$G_x(q^i(t),\dot{q}^i(t)) = J^{-T}(q^i(t))G(q^i(t),\dot{q}^i(t)), \quad F_x^i = J^{-T}(q^i(t))\tau^i(t)$$

and $J(q^i(t)) \in R^{n\times n}$ is the configuration-dependent Jacobian matrix, which is assumed as non-singular in the finite work space Ω. The above dynamic equation has the following useful structural properties [11], which can be exploited to facilitate the controller design in the next section.

Property 1: The inertia matrix $D_x(q^i(t))$ is symmetric and positive definite. It is also bounded as a function of q: $m_1 I \le D_x\left(q^i(t)\right) \le m_2 I$, where $m_1, m_2 > 0$.

Property 2: $\dot{D}_x(q^i(t)) - 2C_x(q^i(t),\dot{q}^i(t))$ is a skew symmetric matrix. Therefore, $y^T[\dot{D}_x(q^i(t)) - 2C_x(q^i(t),\dot{q}^i(t))]y = 0$, where y is a $n \times 1$ nonzero vector.

Assumption 1: The given desired joint trajectory $x_d(t)$ belongs to $C^2[0,t_f]$, where $C^2[0,t_f]$ denotes 2^{nd}-order continuously differentiable functions on $t \in [0,t_f]$.

Assumption 2: Initial condition $x^i(0) = x_d(0)$, $\dot{x}^i(0) = \dot{x}_d(0)$ and for all $i \ge 1$.

3 AILC Design

Linearizing the system (2) along the desired trajectory $x_d(t), \dot{x}_d(t), \ddot{x}_d(t)$ at the ith iterative, we obtain the following linear time-varying system according [8]:

$$D(t)\ddot{e}^i(t) + [C(t) + C_1(t)]\dot{e}^i(t) + R(t)e^i(t) + n(\ddot{e}^i,\dot{e}^i,e^i,t) - \tau_a(t) = S(t) - F^i(t) \tag{3}$$

Where:

$$D(t) = D_x(x_d(t)), \quad C(t) = C_x(x_d(t),\dot{x}_d(t))$$
$$R(t) = \left.\frac{\partial D_x}{\partial x}\right|_{x_d(t)}\ddot{x}_d(t) + \left.\frac{\partial C_x}{\partial x}\right|_{x_d(t),\dot{x}_d(t)}\dot{x}_d(t) + \left.\frac{\partial G_x}{\partial x}\right|_{x_d(t)}$$
$$S(t) = D_x(x_d(t))\ddot{x}_d(t) + C_x(x_d(t),\dot{x}_d(t))x_d(t) + G_x(x_d(t)) + \tau_a(t)$$
$$e(t) = x_d(t) - x(t)$$

The term $n(\ddot{e}^i,\dot{e}^i,e^i,t)$ contains the higher order terms $\ddot{e}^i(t)$, $\dot{e}^i(t)$ and $e^i(t)$ and it can be negligible. The control problem is to find a control law so that the end-effector position $x(t)$ can track specific commands $x_d(t)$. We construct controller as follows:

$$F^i = F_e^i + H^i \tag{4}$$

Where the first term $F_e^i = K_p^i(e^i(t)) + K_d^i(\dot{e}^i(t))$ is feedback PD control law with the following gain switching rule in [8]:

$$K_p^{i+1} = \beta(i)K_p^i, \quad K_d^{i+1} = \beta(i)K_d^i, \quad \beta(i+1) > \beta(i), \quad i = 0,1,2,...,N \tag{5}$$

With K_p^i, K_d^i are the initial proportional and derivative control gain matrices that diagonal positive definite, K_p^{i+1}, are the control gains of the ith iterative. $\beta(i) > 1$ is the gain switching factor. The gains adaptive law in (5) are used to adjust the PD gains from iterative to iterative. And H^i is the initial predicted feedforward control input to be computed at each iterative by a learning rule. According demonstrated in [8], the feedback PD control law with the gain switching factor in (5) plus the feedforward learning control law with the input force profile, the convergence of system (3) is guaranteed. However, in order to, the trajectory tracking convergence fast in some initial iterative, we cannot increase the switching factor arbitrarily large because actuator forces are limited, especially when the system has modelling errors or nonlinearity. Hence, according [12] to deal this problem, we propose feedforward control input $H^i(t)$ with a learning rule so that $H^i(t)$ converges to $R(t)$ for all $t \in [0,t_f]$ as follow:

$$H^{i+1} = H^i + \alpha F_e^i \tag{6}$$

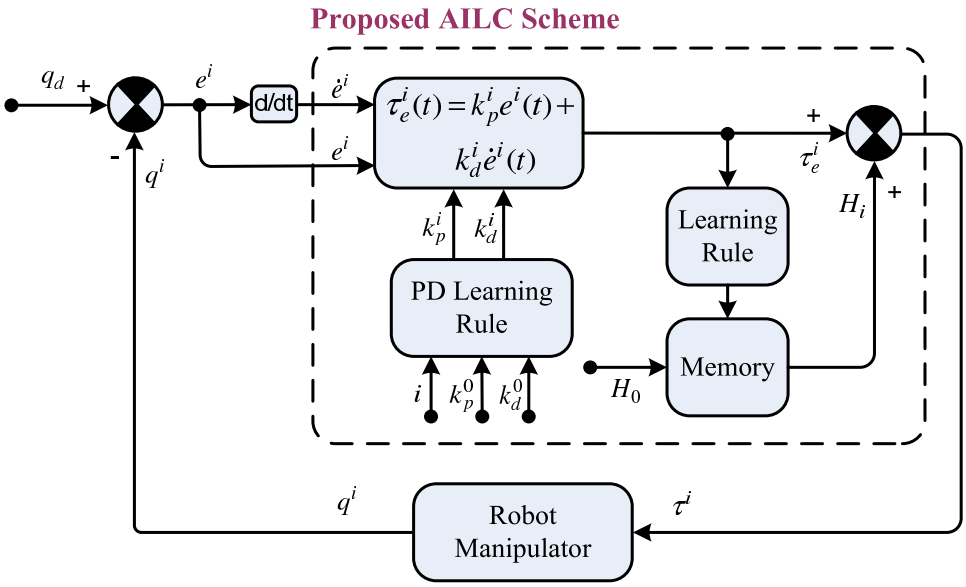


Figure 2: Schematic diagram of the learning control scheme.

At the initial stage of learning, the $H^i(t)$ are set to zero. α is a positive constant often called a training factor. Therefore, for the ith and $(i+1)$th iterations, applying the input (4), (6) to system (3), we obtain an error equation as follows:

$$D(t)\ddot{e}^i(t) + [C(t) + C_1(t)]\dot{e}^i(t) + R(t)e^i(t) = S(t) - F_e^i - H^i \tag{7}$$

$$D(t)\ddot{e}^{i+1}(t) + [C(t) + C_1(t)]\dot{e}^{i+1}(t) + R(t)e^{i+1}(t) = S(t) - F_e^{i+1} - H^i - \alpha F_e^i \tag{8}$$

To simplicity the proof of stability, let $K_p^i = aK_d^i$ for the initial iteration, and define the filter errors as follow:

$$\tilde{x}^i(t) = \dot{e}^i(t) + ae^i(t) \tag{9}$$

Also, define $\delta\tilde{x}^i = \tilde{x}^{i+1} - \tilde{x}^i$ and $\delta e^i = e^{i+1} - e^i$. Then, from (9):

$$\delta\tilde{x}^i = \delta\dot{e}^i + a\delta e^i \tag{10}$$

From (5)-(9) and (10), one can obtain the following equation:

$$D\delta\dot{\tilde{x}}^i + (C + C_1 - aD + K_d^{i+1})\delta\tilde{x}^i + (R - a(C + C_1 - aD))\delta e^i = -(K_d^{i+1} + (\alpha - 1)K_d^i)\tilde{x}^i \tag{11}$$

The following theorem can be proved.

Theorem: Suppose robot system (2) satisfies property (1, 2) and assumption (1, 2). With the control input (4), the gain switching rule (5) and learning rule (6). The following should hold for all $t \in [0, t_f]$.

$$x^i(t) \to x_d(t), \quad \dot{x}^i(t) \to \dot{x}_d(t), \quad i \to \infty$$

If the controller gains are selected so that the following relationships hold:

$$l_p = \lambda_{\min}((2 - \alpha)K_d^i + 2C_1 - 2aD) > 0 \tag{12}$$

$$l_r = \lambda_{\min}((2 - \alpha)K_d^i + 2C + 2R/a - 2\dot{C}_1/a) > 0 \tag{13}$$

$$l_p l_r \geq \|R/a - (C + C_1 - aD)\|_{\max}^2 \tag{14}$$

Where $\lambda_{\min}(A)$ is the minimum eigenvalue of matrix A, and $\|M\|_{\max} = \max \|M(t)\|$ for $t \in [0, t_f]$. Here, $\|M\|$ represents the Euclidean norm of M.

Proof: We select a performance index $V^i(t)$ as follow:

$$V^i = \int_0^t e^{-\rho\tau} (\tilde{x}^i)^T Q \tilde{x}^i d\tau \geq 0 \tag{15}$$

Thus, $\beta(i) > 1$ according (5) so we have $K_e^{i+1} > K_d^i$ and α is a positive constant, so $Q = K_d^{i+1} + (\alpha - 1)K_d^i) > 0$. From the definition of V^j, for the $(i+1)$th iteration, one can get:

$$V^{i+1} = \int_0^t e^{-\rho\tau} (\tilde{x}^{i+1})^T Q \tilde{x}^{i+1} d\tau \tag{16}$$

Let $\Delta V^i = V^{i+1} - V^i$ then from (15), (16) and (11), we obtain:

$$\begin{aligned}
\Delta V^i &= \int_0^t e^{-\rho\tau} ((\delta\tilde{x}^i)^T Q \delta\tilde{x}^i + 2(\delta\tilde{x}^i)^T Q \tilde{x}^i) d\tau \\
&= \int_0^t e^{-\rho\tau} ((\delta\tilde{x}^i)^T Q \delta\tilde{x}^i + 2(\delta\tilde{x}^i)^T Q \tilde{x}^i) d\tau \\
&= \int_0^t e^{-\rho\tau} (\delta\tilde{x}^i)^T Q \delta\tilde{x}^i d\tau - 2\int_0^t e^{-\rho\tau} (\delta\tilde{x}^i)^T D \delta\dot{\tilde{x}}^i d\tau \\
&- 2\int_0^t e^{-\rho\tau} (\delta\tilde{x}^i)^T ((C + C_1 - aD + K_d^{i+1})\delta\tilde{x}^i + (R - a(C + C_1 - aD))\delta e^i) d\tau
\end{aligned} \tag{17}$$

Applying the partial integration, from assumption 2, and property 2, we have:

$$\begin{aligned}
\int_0^t e^{-\rho\tau} (\delta\tilde{x}^i)^T D \delta\dot{\tilde{x}}^i d\tau &= e^{-\rho\tau} (\delta\tilde{x}^i)^T D (\delta\tilde{x}^i)\Big|_0^t - \int_0^t (e^{-\rho\tau} (\delta\tilde{x}^i)^T D)' \delta\tilde{x}^i d\tau \\
&= e^{-\rho\tau} (\delta\tilde{x}^i)^T(t) D(t) \delta\tilde{x}^i(t) + \rho \int_0^t e^{-\rho\tau} (\delta\tilde{x}^i)^T D \delta\tilde{x}^i d\tau \\
&- \int_0^t e^{-\rho\tau} (\delta\tilde{x}^i)^T D \delta\dot{\tilde{x}}^i d\tau - 2\int_0^t e^{-\rho\tau} (\delta\tilde{x}^i)^T C \delta\dot{\tilde{x}}^i d\tau
\end{aligned} \tag{18}$$

Substituting (18) into (17), and $Q = K_d^{i+1} + (\alpha - 1)K_d^i$ yields:

$$\begin{aligned}
\Delta V^i &= -e^{-\rho\tau} (\delta\tilde{x}^i)^T(t) D(t) \delta\tilde{x}^i(t) - \rho \int_0^t e^{-\rho\tau} (\delta\tilde{x}^i)^T D \delta\tilde{x}^i d\tau \\
&- 2\int_0^t e^{-\rho\tau} (\delta\tilde{x}^i)^T (R - a(C + C_1 - aD)) \delta e^i d\tau \\
&- \int_0^t e^{-\rho\tau} (\delta\tilde{x}^i)^T (K_d^{i+1} - (\alpha - 1)K_d^i + 2C_1 - 2aD) \delta\tilde{x}^i d\tau
\end{aligned} \tag{19}$$

From (5), we have:

$$\int_0^t e^{-\rho\tau}(\delta\tilde{x}^i)^T K_d^{i+1}\delta\tilde{x}^i d\tau = \beta(i+1)\int_0^t e^{-\rho\tau}(\delta\tilde{x}^i)^T K_d^i\delta\tilde{x}^i d\tau \quad \geq \int_0^t e^{-\rho\tau}(\delta\tilde{x}^i)^T K_d^i\delta\tilde{x}^i d\tau \tag{20}$$

Substituting (10) into (19) and noticing (20), we obtain:

$$\begin{aligned}
\Delta V^i \leq & -e^{-\rho\tau}(\delta\tilde{x}^i)^T(t)D(t)\delta\tilde{x}^i(t) - \rho\int_0^t e^{-\rho\tau}(\delta\tilde{x}^i)^T D\delta\tilde{x}^i d\tau \\
& - \int_0^t e^{-\rho\tau}(\delta\dot{e}^i)^T((2-\alpha)K_d^i + 2C_1 - 2aD)\delta\dot{e}^i d\tau \\
& - 2a\int_0^t e^{-\rho\tau}(\delta e^i)^T((2-\alpha)K_d^i + 2C_1 - 2aD)\delta\dot{e}^i d\tau \\
& - 2\int_0^t e^{-\rho\tau}(\delta\dot{e}^i)^T(R - a(C + C_1 - aD))\delta e^i d\tau \\
& - a^2\int_0^t e^{-\rho\tau}(\delta e^i)^T((2-\alpha)K_d^i + 2C_1 - 2aD)\delta e^i d\tau \\
& - 2a\int_0^t e^{-\rho\tau}(\delta e^i)^T(R - a(C + C_1 - aD))\delta e^i d\tau
\end{aligned} \tag{21}$$

Applying the partial integration again gives:

$$\begin{aligned}
\int_0^t e^{-\rho\tau}(\delta e^i)^T((2-\alpha)K_d^i + 2C_1 - 2aD)\delta\dot{e}^i d\tau = & \; e^{-\rho\tau}(\delta e^i)^T((2-\alpha)K_d^i + 2C_1 - 2aD)(\delta e^i)\Big|_0^t \\
& + \rho\int_0^t e^{-\rho\tau}(\delta e^i)^T((2-\alpha)K_d^i + 2C_1 - 2aD)\delta e^i d\tau \\
& - \int_0^t e^{-\rho\tau}(\delta\dot{e}^i)^T((2-\alpha)K_d^i + 2C_1 - 2aD)\delta e^i d\tau \\
& + 2\int_0^t e^{-\rho\tau}(\delta e^i)^T(a\dot{D} - \dot{C}_1)\delta e^i d\tau
\end{aligned} \tag{22}$$

Therefore

$$\begin{aligned}\Delta V^i \leq & -e^{-\rho\tau}(\delta\tilde{x}^i)^T D\delta\tilde{x}^i - \rho\int_0^t e^{-\rho\tau}(\delta\tilde{x}^i)^T D\delta\tilde{x}^i d\tau \\ & - ae^{-\rho\tau}(\delta e^i)^T((2-\alpha)K_d^i + 2C_1 - 2aD)\delta e^i \\ & - \rho a\int_0^t e^{-\rho\tau}(\delta e^i)^T((2-\alpha)K_d^i + 2C_1 - 2aD)\delta e^i d\tau - \int_0^t e^{-\rho\tau} w d\tau \\ \leq & -e^{-\rho\tau}(\delta\tilde{x}^i)^T D\delta\tilde{x}^i - ae^{-\rho\tau}(\delta e^i)^T l_p \delta e^i - \rho\int_0^t e^{-\rho\tau}(\delta\tilde{x}^i)^T D\delta\tilde{x}^i d\tau \\ & - \rho a\int_0^t e^{-\rho\tau}(\delta e^i)^T l_p \delta e^i d\tau - \int_0^t e^{-\rho\tau} w d\tau\end{aligned} \tag{23}$$

Where

$$\begin{aligned}w = & (\delta\dot{e}^i)^T((2-\alpha)K_d^i + 2C_1 - 2aD)\delta\dot{e}^i + 2a(\delta\dot{e}^i)^T(R/a - (C + C_1 - aD))\delta e^i \\ & + a^2(\delta e^i)^T((2-\alpha)K_d^i + 2R/a + 2C - 2\dot{C}_1/a)\delta e^i\end{aligned} \tag{24}$$

Let $P = R/a - (C + C_1 - aD)$. Then from (12) and (13), we obtain

$$w \geq l_p\|\delta\dot{e}\|^2 + 2a\delta\dot{e}^T P\delta e + a^2 l_r\|\delta e\|^2 \tag{25}$$

Applying the Cauchy-Schwartz inequality, we obtain:

$$\delta\dot{e}^T P\delta e \geq -\|\delta\dot{e}\|\,\|P\|_{\max}\,\|\delta e\| \tag{26}$$

From (12)-(14)

$$\begin{aligned}w &= l_p\|\delta\dot{e}\|^2 - 2a\,\|\delta\dot{e}\|\,\|P\|_{\max}\,\|\delta e\| + a^2 l_r\|\delta e\|^2 \\ &= l_p(\|\delta\dot{e}\| - \frac{a}{l_p}\|P\|_{\max}\,\|\delta e\|)^2 + a^2(l_p - \frac{1}{l_r}\,\|P\|_{\max}^2)\|\delta e\|^2 \\ &\geq 0\end{aligned} \tag{27}$$

According property 1 and (27), based on (23), it can be ensured that $\Delta V^i \leq 0$, therefore $V^{i+1} \leq V^i$. From the definition, K_d^i is a positive definite matrix. From the definition of V^i, $V^i \geq 0$ and V^i is bounded. As a result, $y^i(t) \to \infty$ when $i \to \infty$. Because $e^i(t)$ and $\dot{e}^i(t)$ are two independent variables, and a is a positive constant. Thus, if $i \to \infty$, $e^i(t) \to 0$ and $\dot{e}^i(t) \to 0$ for $t \in [0, t_f]$. Finally, the following conclusions hold for $t \in [0, t_f]$.

$$x^i(t) \to x_d(t), \quad \dot{x}^i(t) \to \dot{x}_d(t), \quad i \to \infty$$

From the above analysis it can be seen that the adaptive PD control method can guarantee that the tracking errors converge arbitrarily close to zero as the number of iterations increases. The following case studies based on simulation will demonstrate this conclusion.

4 Numerical Simulation

A two-link robot manipulator as shown in Fig.1 is utilized in this paper to verify the effectiveness of the proposed control scheme. The dynamic equation of the robot manipulator is adopted in [13].

$$D(q) = \begin{bmatrix} m_1 + m_2 + 2m_3\cos(q_2) & m_2 + m_3\cos(q_2) \\ m_2 + m_3\cos(q_2) & m_2 \end{bmatrix}$$

$$C(q,\dot{q}) = \begin{bmatrix} -m_3\dot{q}_2\sin(q_2) & -m_3(\dot{q}_1 + \dot{q}_2)\sin(q_2) \\ m_3\dot{q}_1\sin(q_2) & 0 \end{bmatrix}$$

$$G(q) = \begin{bmatrix} m_4 g\cos(q_1) + m_5 g\cos(q_1 + q_2) \\ m_5 g\cos(q_1 + q_2) \end{bmatrix}$$

and m_i are the parameters of interest given by $M = P + p_l L$ with

$$M = \begin{bmatrix} m_1 & m_2 & m_3 & m_4 & m_5 \end{bmatrix}^T$$

$$P = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 & p_5 \end{bmatrix}^T$$

$$L = \begin{bmatrix} l_1^2 & l_2^2 & l_1 l_2 & l_1 & l_2 \end{bmatrix}^T$$

and p_l is the payload, ł$_1 = 1(m)$ and ł$_2 = 1(m)$ are the lengths of link 1 and link 2, respectively, P is the parameter vector of the robot itself. g = 9.8 m/s2. The jacobian matrix is known as:

$$J(q) = \begin{bmatrix} -l_1\sin(q_1) - l_2\sin(q_1 + q_2) & -l_2\sin(q_1 + q_2) \\ l_1\cos(q_1) + l_2\cos(q_1 + q_2) & l_2\cos(q_1 + q_2) \end{bmatrix}$$

For the convenience of the simulation, the nominal parameters of the robotic system are given as:

$$P = \begin{bmatrix} 1.66 & 0.42 & 0.63 & 3.75 & 1.25 \end{bmatrix}^T kg.m2$$

The desired reference trajectories in the Cartesian space are:

$$x_{d1} = 1.0 + 0.2\cos(\pi t), \quad x_{d2} = 1.0 + 0.2\sin(\pi t).$$

Which represent a circle of radius $0.2m$ and its centres is located at $(x_1, x_2) = (1.0, 1.0)m$. The robot is initially rested with its end-effector positioned at the center of the circle. With initial condition are $x_1(0) = x_{d1}(0) = 1.2$, $x_2(0) = x_{d2}(0) = 1.0$, $\dot{x}_1(0) = \dot{x}_{d1}(0) = 1.0$, $\dot{x}_2(0) = \dot{x}_{d2}(0) = 1.2$ and $t = [0, 2]$.

The most important parameters that effect the control performance of the robotic system are the external disturbance $\tau_a(t)$, the friction term $f(\dot{q})$, in simulation, payload variation situation and external disturbance situation occurring at fifth the iteration are considered. The payload variation situation is that $p_l = 0(kg)$ from first the iteration to fourth the iteration, and then it was put on $p_l = 3$ after the fifth iteration. The disturbance situation is that external forces are injected into the robotic system, and their shapes are expressed as follows:

$$\tau_a(t) = \begin{bmatrix} 5\sin(5t) & 5\sin(5t) \end{bmatrix}^T \tag{28}$$

In addition, friction forces are also considered in this simulation and given as:

$$f(\dot{q}) = \begin{bmatrix} 2\dot{q}_1 + 0.8\mathrm{sgn}(\dot{q}_1) & 4\dot{q}_2 + 0.1\mathrm{sgn}(\dot{q}_2) \end{bmatrix}^T \tag{29}$$

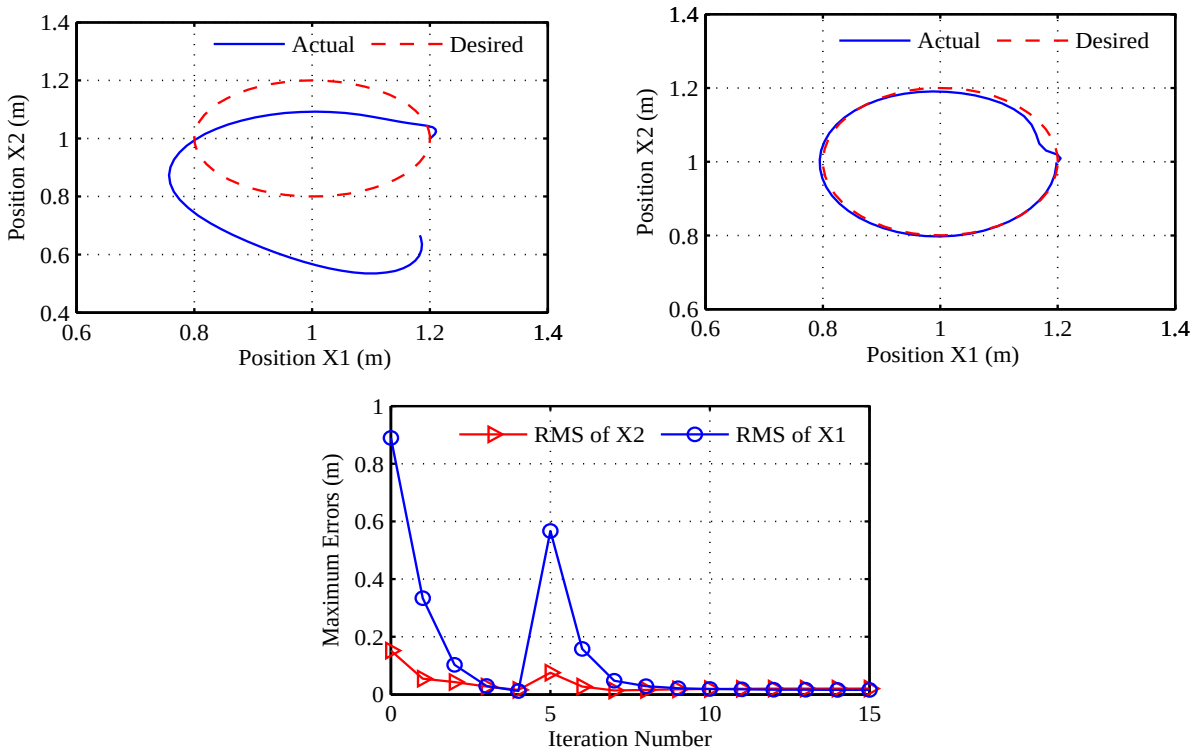


Figure 3: Reference tracking and actual trajectory for end-effector at different iterative under AILC is adopted in [11]. (a) Position tracking of end-effector at first iteration, (b) Position tracking of end-effector after fifteenth iteration and (c) profile of rms position errors payload and disturbance variation after the fifth iteration.

In order to exhibit the superior control performance of AILC system, two extra control systems including an adaptive switching learning PD control system (ASL-PD) is represented in [8], an iterative learning control (ILC) is represented in [12].

In three simulation situations, The PD control gain was set to be the same as [8]:

$$K_p^i = K_d^i = diag\{30, 30\}, \quad K_p^{i+1} = 2iK_p^i, \quad K_d^{i+1} = 2iK_d^i, \quad \beta = 0.75$$

The simulated results of ILC system, the responses of end-effector position at first and fifteenth iteration and tracking error from iteration to iteration to be depicted Fig. 3(a), (b) and (c) respectively. From the simulated results, we can be seen that the tracking performance was not acceptable because the errors were too large compared to ASL-PD and proposed controllers, especially at the fifth iteration poor tracking errors responses are resulted due to the occurrence of payload variation situation and external disturbance. In the ASL-PD system are depicted in [8]. The end-effector position responses, tracking error from iteration to iteration to be depicted in Fig. 4(a), 4(b), and 4(c), respectively. The end-effector tracking performance is obvious under the occurrence of payload variation and external disturbance. The convergence rate increased greatly compared with the ILC control method.

Now, the AILC system depicted in Fig. 2 is applied to control the robot manipulator for comparison. The simulated results of end-effector position responses and tracking error from iteration to iteration are depicted in Fig. 5(a), 5(b) and 5(c), respectively. Table 1 shows tracking performance of proposed system from the initial iteration to fifteenth are obvious. Therefore, the comparison of their method and our method demonstrated a bit fast convergence rate with the proposed control method. Specially, control

	Iterative	1	5	10	15
ILC	$\max \|e_1^i\|(m)$	0.0822	0.0773	0.0168	0.0140
	$\max \|e_2^i\|(m)$	0.1998	0.5634	0.0130	0.0105
ASL-PD	$\max \|e_1^i\|(m)$	0.0398	0.0076	0.0012	0.0010
	$\max \|e_2^i\|(m)$	0.0646	0.0593	0.0008	0.0006
AILC	$\max \|e_1^i\|(m)$	0.0377	0.0076	0.0011	0.0009
	$\max \|e_2^i\|(m)$	0.1741	0.0590	0.0008	0.0006

Table 1: Maximum tracking errors from iteration to iteration.

	Iterative	1	5	10	15
ASL-PD	$\max \|\tau_1^i\|(N.m)$	90	930	3330	7230
	$\max \|\tau_2^i\|(N.m)$	51.45	531.6	1903	3618
AILC	$\max \|\tau_1^i\|(N.m)$	82.50	772.5	2647	5647
	$\max \|\tau_2^i\|(N.m)$	47.14	441.6	1513	3228

Table 2: Maximum control forces from iteration to iteration.

torques of proposed system is much smaller than the ASL-PD controller. Detail results are depicted in table 2.

5 Conclusions

This paper has successfully implemented an AILC scheme to control the position of end-effector in task space for achieving desired position control. All the system dynamics may be unknown. A new control method is a combination of advantages some other method into a hybrid one as explained above. By using Lyapunov theorem, the asymptotic convergence of the closed-loop control system can be ensured whether or not the uncertainties occur. Simulation results of a two link robot manipulator in task space via various existing control methods including ASL-PD and ILC control were also applied in this paper to compare and display the manipulative performance of the proposed control system. According to the result as depict in Figs. 3-5 and table 1-2, the desired position tracking and tracking errors response of the AILC scheme decrease with the increase of the iteration number under wide range of payload and external disturbance.

The main of the paper is to construct a simple scheme, easy implementation, fast convergence. Especially, in this paper is applied not only to control the position of the end-effector, but also the force exerted by the end-effector on the object. By designing the control law in task space, force control can be easily formulated.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (60775047; 60835004), the National High Technology Research and Development Program of China (863 Program) (2007AA04Z244; 2008AA04Z214).

The authors would like to thank the associate editor and the reviewers for their valuable comments.

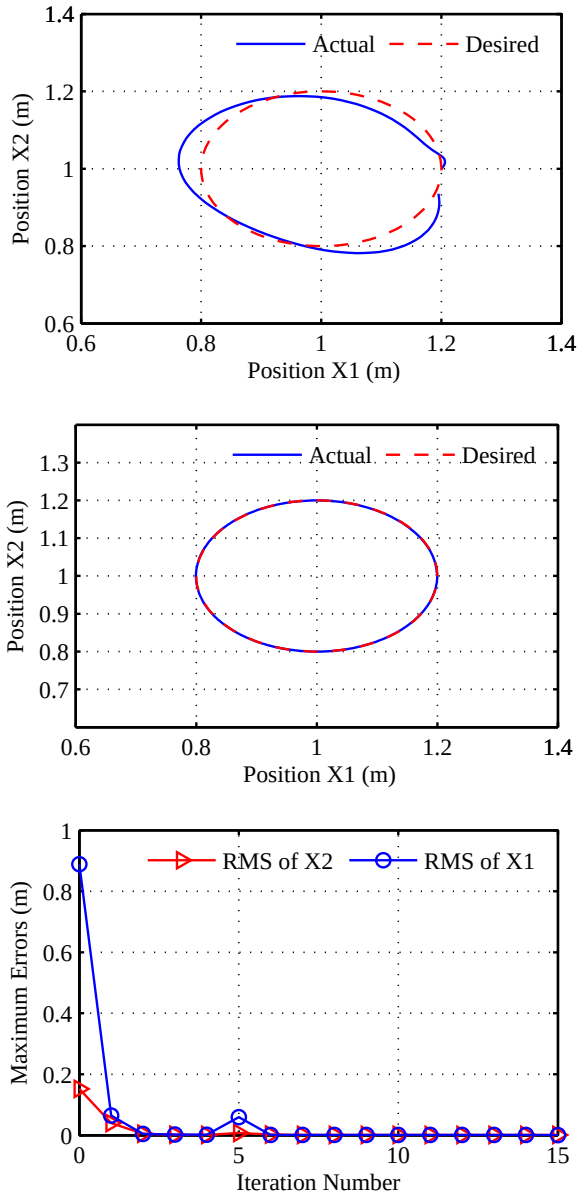


Figure 4: Reference tracking and actual trajectory for end-effector at different iterative under ASL-PD is adopted in [6]. (a) Position tracking of end-effector at first iteration, (b) Position tracking of end-effector after fifteenth iteration and (c) profile of rms position errors payload and disturbance variation after the fifth iteration.

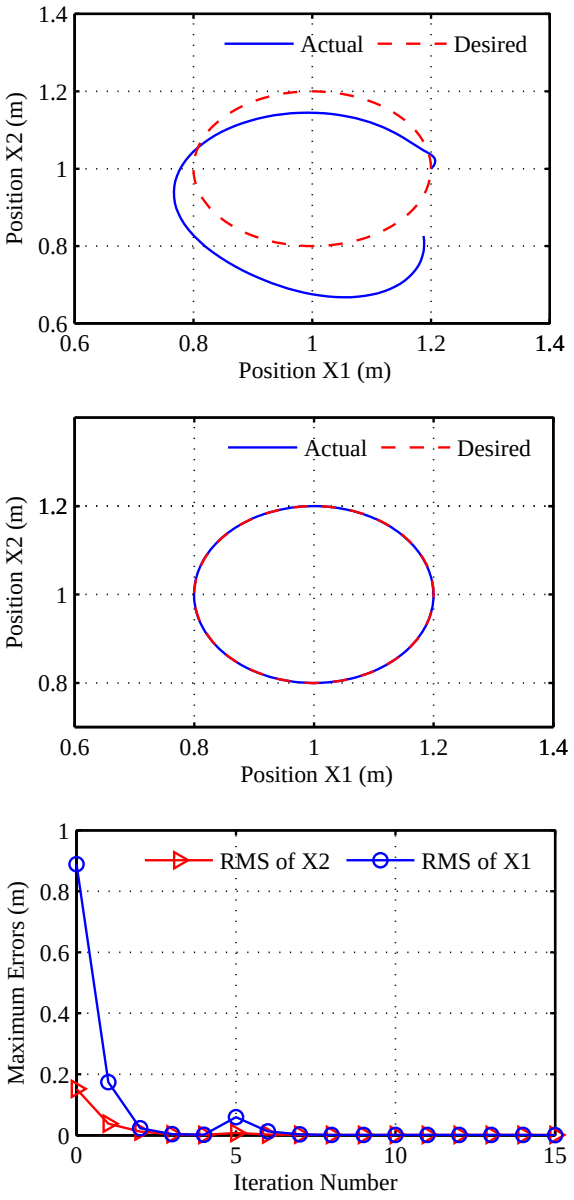


Figure 5: Reference tracking and actual trajectory for end-effector at different iterative under AILC controller. (a) Position tracking of end-effector at first iteration, (b) Position tracking of end-effector after fifteenth iteration and (c) profile of rms position errors payload and disturbance variation after the fifth iteration.

Bibliography

[1] P. Bondi, G. Cacaline and L. Gambardella, On the iterative learning Control theory for robot manipulators, *IEEE J. On Robotics and Automation*, 4:14-22, 1984.

[2] S. Arimoto, S. Kawamura and F. Miyazaki, Bettering operation of dynamic system by learning: A new control theory for servomechanism or mechatronics systems, *Proceedings of 23rd. CDC*, 1064-1069, 1984.

[3] J.J. Craig, Adaptive control of mechanical manipulators, *Addison-Wesley, New York*, 1988.

[4] Hyo-Sung Ahn, YangQuan Chen, Kevin L. Moore, Iterative learning control: Brief Survey and Categorization, *IEEE Trans. Ind. Sys*, 6:1099-1121, 2007.

[5] Dong-II Kim, Sungkwun Kim An iterative learning control method with application for CNC machine tools, *IEEE Trans. Ind. App*, 32(1):66-72, 1996.

[6] K.L. Moore, Iterative learning control for deterministic systems,*Advances Industrial Control, New York, Springer-Verlag*, 1993.

[7] Tayebi A, Adaptive iterative learning control for robot manipulators, *Automatica*, 40:1195-1203, 2004.

[8] Ouyang P R, Zhang W J, Gupta M M, An adaptive switching learning control method for trajectory tracking of robot manipulators, it Mechatronics, 16:51-61, 2006.

[9] Choi JY, Lee JS, Adaptive iterative learning control of uncertain robotic systems,*IEE Proc Contr Theory*, 147(4):217-223, 2000.

[10] B.S. Chen, H.J. Uang, and C.S. Tseng, Robust tracking enhancement of robot systems including motor dynamics: A fuzzy-based dynamic game approach,*IEEE Trans. Fuzzy syst*, 11(4):538-552,1998.

[11] Craig JJ. Introduction to robotics: mechanics and control. Reading, *MA: Addison-Wesley*, 1986.

[12] T.Y Kuc, K. Nam, J.S. Lee, An iterative learning control of robot Manipulators, *IEEE Trans Robot Automat.*, 7(6): 835-842, 1991.

[13] S.S. Ge, C.C. Hang, Adaptive neural network control of robot manipulators in task space,*IEEE Trans. Ind. Elec.*, 44(6):746-752, 1997.

INT J COMPUT COMMUN, ISSN 1841-9836
Vol.7 (2012), No. 3 (September), pp. 530-539

Brain Tumor Segmentation on MRI Brain Images with Fuzzy Clustering and GVF Snake Model

A. Rajendran, R. Dhanasekaran

Arthanari Rajendran
Professor and HoD,
Department of Electronics and Communication Engineering,
Sriguru Institute of Technology,
Coimbatore,Tamilnadu, India
E-mail:rajendranav@gmail.com

Raghavan Dhanasekaran
Professor, Director-Research,
Syed Ammal Engineering College,
Ramanathapuram, Tamilnadu, India
E-mail:rdhanashekar@yahoo.com

Abstract: Deformable or snake models are extensively used for medical image segmentation, particularly to locate tumor boundaries in brain tumor MRI images. Problems associated with initialization and poor convergence to boundary concavities, however, has limited their usefulness. As result of that they tend to be attracted towards wrong image features. In this paper, we propose a method that combine region based fuzzy clustering called Enhanced Possibilistic Fuzzy C-Means (EPFCM) and Gradient vector flow (GVF) snake model for segmenting tumor region on MRI images. Region based fuzzy clustering is used for initial segmentation of tumor then result of this is used to provide initial contour for GVF snake model, which then determines the final contour for exact tumor boundary for final segmentation. The evaluation result with tumor MRI images shows that our method is more accurate and robust for brain tumor segmentation.
Keywords: Deformable model; FCM; Segmentation; MRI image; GVF

1 Introduction

The accurate and automatic segmentation of brain tumor on MRI image is of great interest for assessing tumor growth and treatment responses, enhancing computer-assisted surgery, planning radiation therapy, and constructing tumor growth models. This is very difficult task in existing methods. The existing methods are divided into region-based and contour-based methods. Region-based methods [1-9] seek out clusters of pixels that share some measure of similarity. These methods reduce operator interaction by automating some aspects of applying the low level operations, such as threshold selection, histogram analysis, classification, etc. They can be supervised or non-supervised. In general these methods take advantage of only local information for each pixel and do not include shape and boundary information. Contour-based methods [10-14] rely on the evolution of a curve, based on internal forces and external forces, such as image gradient, to delineate the boundary of brain structure or pathology. These methods can also be supervised or non-supervised. In general these methods suffer from the problem of determining the initial contour and leakage in imprecise edges.

In this paper we propose a method that is a combination of region-based fuzzy clustering method called Enhanced Possibilistic Fuzzy C-Means (EPFCM) and Gradient vector flow (GVF) snake model to remove the problems using the capabilities of each one. For example a region-based method can solve the problem of the initialization of a contour-based method (GVF snake model) and a contour-based method is able to improve the quality of region-based segmentation at the boundary of objects.

So the proposed method has two main phases for tumor segmentation on MRI brain images namely, initial segmentation which is done by a region-based method and final segmentation that is performed by a boundary-based GVF snake model. We discuss these approaches in the following section.

2 Region-Based Enhanced Possibilistic Fuzzy C-Means (EPFCM)

In this proposed Enhanced Possibilistic Fuzzy C-Means (EPFCM) method, distance metric D_{ij} in PFCM [15] is modified in such a way that it includes membership, typicality and both local, nonlocal spatial neighbourhood information to overcome the noise effect in MRI brain medical images. This modified distance metric is incorporated into objective function of PFCM. Then resultant algorithm is called Enhanced Possibilistic Fuzzy C-means (EPFCM) is obtained for enhanced segmentation results. Therefore objective function of our proposed EPFCM is defined as follows,

$$J_m(U,V,T;X) = \sum_{i=1}^{c}\sum_{j=1}^{n}\left(a\mu_{ij}^m + bt_{ij}^\eta\right)D_{ij}^2 + \sum_{i=1}^{c}\gamma_i\sum_{j=1}^{n}\left(1-t_{ij}\right)^\eta \tag{1}$$

Where, the modified distance metric is given by

$$D^2\left(x_j,v_i\right)\& = \&D_{ij}^2 = \left(1-\lambda_j\right)d_l^2\left(x_j,v_i\right) + \lambda_j d_{nl}^2\left(x_j,v_i\right) \tag{2}$$

$$\sum_{i=1}^{c}\mu_{ij}\& = \&1\ \forall j, 0 \le \mu_{ij}, t_{ij} \le 1 \text{ and } a>0, b>0, m>1, \eta>1 \tag{3}$$

The membership function:

$$\mu_{ij} = \left[\sum_{i=1}^{c}\left(\frac{D_{ij}}{D_{kj}}\right)^{\frac{2}{m-1}}\right]^{-1} \tag{4}$$

Typicality:

$$t_{ij} = \frac{1}{1+\left(\frac{b}{\gamma_i}D_{ij}^2\right)^{\frac{1}{\eta-1}}} \tag{5}$$

Cluster centre:

$$v_i = \frac{\sum_{j=1}^{n}\left(a\mu_{ij}^m + bt_{ij}^\eta\right)x_j}{\sum_{j=1}^{n}\left(a\mu_{ij}^m + bt_{ij}^\eta\right)} \tag{6}$$

In the following equation is suggested to compute γ_i:

$$\gamma_i = \frac{K\sum_{j=1}^{n}\mu_{ij}^m D_{ij}^2}{\sum_{j=1}^{n}\mu_{ij}^m}, K>1 \tag{7}$$

2.1 Importance of modified distance metric term $\left(D_{ij}\right)$

The modified distance metric or dissimilarity measure is rewritten from Equation (2) as follows,

$$D^2\left(x_j,v_i\right)\& = \&D_{ij}^2 = \left(1-\lambda_j\right)d_l^2\left(x_j,v_i\right) + \lambda_j d_{nl}^2\left(x_j,v_i\right) \tag{8}$$

Where, d_l is the distance metric influenced by local spatial information. This added local spatial neighborhood term is similar to the one which is used in [16] to incorporate the neighborhood effects in the classic FCM.The local spatial constraint is evaluated by the feature difference between neighboring pixels in the image.

d_{nl} is the distance measurement influenced by non- local spatial information. This added non local term is obtained from the non local means (NL-means) algorithm [17] for image denoising. The non-local constraint determined by all points whose neighborhood configurations look like the neighborhood of the pixel of interest. λ_j is the weighting factor controlling the tradeoff between local and nonlocal spatial information. It varies from zero to one.

2.2 Importance of local distance metric (D_l)

Let N_j denote a chosen local neighborhood configuration of fixed size with respect to a center pixel x_j. If the value of a pixel x_k in N_j is close to the center pixel, then x_j should be influenced greatly by it, otherwise, its influence to x_j should be small. According to the above description, the distance measurement influenced by local information d_l is given by,

$$d_l^2\left(x_j, v_i\right) = \frac{\sum\limits_{x_k \in N_j} \omega_l\left(x_k, x_j\right) d^2\left(x_k, v_i\right)}{\sum\limits_{x_k \in N_j} \omega_l\left(x_k, x_j\right)} \tag{9}$$

where $d^2\left(x_k, v_i\right) = \|x_k - v_i\|^2$ is the Euclidean distance metric measure the similarity between pixel pixel x_k and cluster centroid $v_i, \omega_l(x_k, x_j)$ is the weight of each pixel x_k in N_j and is given by

$$\omega_l\left(x_k, x_j\right) = e^{\frac{|x_k - x_j|^2}{\sigma^2}} \tag{10}$$

Where, σ^2 is the variance of N_i. It specifies the steepness of the sigmoid curve.

2.3 Importance of non local distance metric (D_{nl})

The distance measurement influenced by non-local information d_{nl} is computed as a weighted average of all the pixels in the image $I, x_k \in I$

$$d_{nl}^2\left(x_j, v_i\right) = \sum_{x_k \in I} \omega_{nl}\left(x_k, x_j\right) d^2\left(x_k, v_i\right) \tag{11}$$

Where the family of weight $\omega_{nl}\left(x_k, x_j\right); x_k \in I$ depends on the similarity between the pixel x_k and x_j, and satisfies the usual conditions $0 \leq \omega_{nl}\left(x_k, x_j\right) \leq 1$ and $\sum \omega_{nl}\left(x_k, x_j\right) = 1$.

The similarity between two pixels x_k and x_j depends on the similarity of the intensity gray level vector $v(N_k)$ and $v(N_j)$, where N_k denotes a square neighborhood of fixed size and centered at a pixel x_k. This similarity is measured as a decreasing function of the weighted Euclidean distance $\left\|v\left(N_k\right) - v\left(N_j\right)\right\|_{2,a}^2$, where $a > 0$ is the standard deviation of the Gaussian kernel. The pixels with a similar gray level neighborhood to $v(N_j)$ have larger weights in the average. These weights are defined as

$$\omega_{nl}\left(x_k, x_j\right) = \frac{1}{Q\left(x_j\right)} S\left(x_k, x_j\right) \tag{12}$$

Where, $S\left(x_k, x_j\right)$ is the exponential form of the similarity and $Q\left(x_j\right)$ is the normalizing constant. These terms are defined as,

$$S\left(x_k, x_j\right) = e^{\frac{\left\|v(N_k)-v\left(N_j\right)\right\|_{2,a}^2}{h^2}} \tag{13}$$

$$Q\left(x_j\right) = \sum_{x_k \in I} e^{\frac{\left\|v(N_k)-v\left(N_j\right)\right\|_{2,a}^2}{h^2}} \tag{14}$$

The parameter h acts as a degree of filtering. It controls the decay of the exponential function and therefore the decay of the weights as a function of the Euclidean distance.

2.4 Importance of trade-off parameter (λ)

For computational purpose, the search of the similar neighborhood configuration always be restricted in a larger "search window" denoted by Ω_i. Let x_j be the pixel under consideration. For each pixel x_k in the search window of size $S \times S$, calculate its exponential similarity to x_j using Equation (13). The tradeoff parameter of x_j is then defined as

$$\lambda_j = \frac{1}{m} \sum_{i=1}^{m} S_i\left(x_k, x_j\right) \tag{15}$$

Where S_i represents the ith exponential similarity term in the search window and choose $m = S - 1$.The parameter λ_j decides the trade-off between local and non local spatial information.

3 Algorithm for Proposed EPFCM Method

Finally the algorithm for carrying out our proposed EPFCM for tumor segmentation of MRI brain images can now be stated from the following steps

1. Select the number of clusters $'C'$ and fuzziness factor $'m'$
2. Select initial class centre prototypes $v = \{v_i\}; i = 1, 2 \cdots C$, randomly and $\in, a$ very small number
3. Select the neighbourhood size and search window size
4. Calculate modified distance measurement D_{ij}^2 using the Equation (2)
5. Update membership function μ_{ij} using D_{ij}^2
6. Update $\gamma_i; i = 1, 2 \cdots C$, using Equation (7)
7. Update typicality using the Equation (5)
8. Update cluster centre using equation (6)
9. Repeat steps 4 to 8 until termination. The termination criterion is as follows,

   $\|V_{t+1} - V_t\| \leq \in$ where$'t'$ is the iteration steps, $\|.\|$ is the Euclidean distance norm.

We applied this proposed algorithm to segment tumor on MRI images. In this case, we segmented the brain image into five classes: namely, CSF (Cerebrospinal fluid), WM (White matter), GM (Gray matter), tumor and background .Due to some classification errors, there are undesired additional pixels in the tumor class. To remove these misclassified components, several binary morphological operations

are applied to the tumor class after users defined segmentation classes are obtained (number of clusters). An opening operation is first used to disconnect the components. Then we select the largest connected component, which proved to always correspond to the tumor, even if it has a small size. Here the elementary neighborhood of the morphological operations corresponds to 6-connectivity. The result of this algorithm gives segmented tumor class as shown in Figure 1(c) .This output is the initial contour for the GVF snake model.

4 Boundary-Based GVF Snake Model

The traditional deformable active contour model [18-19] is a curve $X(S) = [x(s), y(s)]$, $s \in [0, 1]$,that move within the image to minimize the energy function. The curve dynamically changes the shape of an initial contour in response to internal and external forces. The internal forces provide the smoothness of the contour. While the external forces push the curve move toward the desired features, such as boundaries. The object contour will be got when the energy function is minimized. The energy is defined as:

$$E = \int_0^1 \frac{1}{2}\left[\alpha|X'(S)|^2 + \beta|X''(S)|^2\right] + E_{ext}(X(S))\,ds \tag{16}$$

Where, $X'(S)$ and $X''(S)$ are first and second derivatives of $X(S)$ with respect to s. The parameter α controls the tension of the curve and β controls its rigidity. E_{ext} is the external energy which is calculated from the image data. To minimize the energy function, the snake must satisfy the Euler equation

$$\alpha X''(S) - \beta X''''(S) - \nabla E_{ext} = 0 \tag{17}$$

Then the snake is made dynamic by treating as the function of time t, as follows:

$$X_t(S,t) = \alpha X''(S,t) - \beta X''''(S) - \nabla E_{ext} \tag{18}$$

When the solution $X(S,t)$ stabilizes, the term $X_t(S,t)$ is zero. Then we get the solution of equation (18).The typical external energies include:

$$E_{ext}(x,y) = -|\nabla I(x,y)|^2 \tag{19}$$
$$E_{ext}(x,y) = -|\nabla [G_\sigma(x,y) * I(x,y)]|^2 \tag{20}$$
$$E_{ext}(x,y) = I(x,y) \tag{21}$$
$$E_{ext}(x,y) = G_\sigma(x,y) * I(x,y) \tag{22}$$

Where, $G_\sigma(x,y)$ is a 2-D Gaussian function with standard deviation σ and mean is zero.∇ denotes the gradient operator $*$ denotes linear convolution. These external forces have a short capture range and poor convergence to boundary concavities. To overcome these problems, Gradient vector flow snake was proposed by *Xu* and Prince [18], which uses the force balance condition as a starting point of snake. It defined a new static external force field called GVF field

$$F_{ext} = V(x,y) = [u(x,y), v(x,y)] \tag{23}$$

Where, u and v are the grey changes on x-axis and y-axis of the image respectively .F_{ext} can be got by minimizing the following energy function:

$$\in = \int\int \mu\left(u_x^2 + u_y^2 + v_x^2 + v_y^2\right) + |\nabla f|^2 |v - \nabla f|^2 dxdy \tag{24}$$

Where, u_x, u_y, v_x, v_y are derivative of x-axis and y-axis respectively. $f(x, y)$ is the edge map (using Canny edge detector) which is derived from image $I(x, y)$. μ is a regularization parameter governing the tradeoff between the first term and the second term in the formula. It should be set according to the noise of the image. The calculus of variations and numerical implementation discussed in [18] is used to obtain the solution of equation. This deformable contour is first initialized by the tumor class output of EPFCM method, which then moves towards the final tumor boundary.

5 Results and Discussion

Initially tumor MRI brain image is segmented for tumor class using EPFCM method, which then initial contour for GVF snake. Then the contour attracted towards final tumor boundary by edge map derived from the image using Canny edge detector. We set parameter $h = 500$, search window size is 7×7,neighborhood window size is $3 \times 3, m = 2, a = 5, b = 3$ and $\eta = 2$ for EPFCM method to have proper segmentation result. We set α and β value between 0.1 to 0.2 and μ value between 0.2 to 0.3 for GVF snake model to have final tumor boundary.The application of our combined method to 10 contrasts enhanced T1-weighted images and 5 FLAIR images shows better tumor segmentation. The results of four cases are as shown in Figure 1.

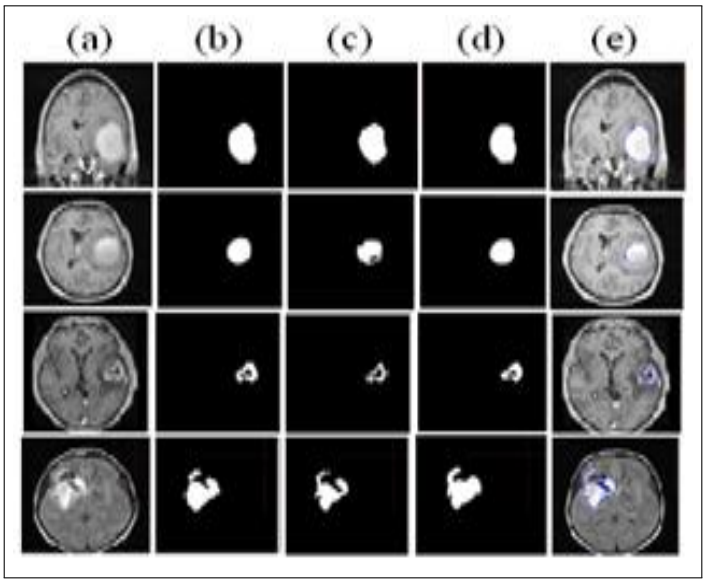


Figure 1: (a)First Column: First two images; Original CE-T1w enhanced tumors; Third image; Original CE-T1w ring enhanced tumor; Fourth image; Original non enhanced tumor FLAIR image (b) Second Column: Manual segmentation result (c) Third Column: EPFCM result showing tumor class after morphological operations (d) Fourth Column: Segmentation of tumor class using combined approach (EPFCM and GVF snake model) (e) Fifth Column: Final boundary detection (Blue curve) shows tumor region using GVF snake model.

The evaluation of segmentation performance is also carried out quantitatively by employing four volume metrics namely, the similarity index(S), false positive volume function (FPVF), false negative volume function (FNVF) and Jaccard index in our experiment. For a given image, suppose that A_i and B_i represent the sets of pixels belong to class i in manual and in automatic segmentation, respectively. $|A_i|$ denotes the number of pixels in A_i. $|B_i|$ denotes the number of pixels in B_i.

The similarity index is an intuitive and clear index to consider the matching pixel between A_i and B_i,

and defined as

$$S = \frac{2|A_i \cap B_i|}{|A_i| + |B_i|} \tag{25}$$

Similarity index $S > 70\%$ indicates an excellent similarity [20].

The false positive volume function (FPVF) represents the error due to the misclassification in class i and the false negative volume function (FNVF) represents the error due to the loss of desired pixels of class i, they are defined as follows,

$$\text{FPVF} = \frac{|B_i| - |A_i \cap B_i|}{|A_i|} \tag{26}$$

$$\text{FNVF} = \frac{|A_i| - |A_i \cap B_i|}{|A_i|} \tag{27}$$

Higher value of S, and lower value of FPVF, FNVF gives better segmentation result.

The Jaccard index between two volumes is represented as follows,

$$J_i(A, M) = \frac{|A_i \cap B_i|}{|A_i \cup B_i|} \times 100 \tag{28}$$

Table 1: Evaluation of the segmentation results of enhanced tumors and nonenhanced tumor by combined approach (EPFCM and GVF model) on a few CE-T1w and FLAIR images.(FET denotes the Full enhanced tumor, RET the ring-enhanced tumor,NET the enhanced tumor

MRI modality type	Type of tumor	Volume metric functions (%)			
CE-T1w & FLAIR	FET, RET & NET	S	FPVF	FNVF	J
CE-T1w	FET1	98.8	0.4	0.2	88.2
CE-T1w	FET2	96.3	0.7	0.4	84.5
CE-T1w	FET3	92.6	1.2	0.7	86.7
CE-T1w	FET4	95.7	0.6	0.6	89.5
CE-T1w	FET5	93.2	0.4	0.5	87.8
CE-T1w	RET1	95.8	0.1	0.2	80.2
CE-T1w	RET2	92.3	1.3	0.7	78.6
CE-T1w	RET3	97.6	0.2	0.3	76.2
CE-T1w	RET4	91.8	2	1.2	75.5
CE-T1w	RET5	96.3	0.1	0.3	77.3
FLAIR	NET1	98.8	0.9	0.6	76.9
FLAIR	NET2	91.5	2.1	1.8	83.2
FLAIR	NET3	98.6	2.9	3	80.1
FLAIR	NET4	94.4	1.8	2.2	85.2
FLAIR	NET5	95.3	1.2	2.1	81.6
Average		95.3	1.06	0.98	82.1

The result of four volume metrics for our method applied to 15 tumor cases is as shown in Table 1and plotted in Figure 2. From this table, we can see that an average similarity metrics and Jaccard index of our method is 95.3% and 82.1% that is, the overlap degree between our segmentation result and the manual segmentation is higher. The average FPVF and FNVF values are equal to 1.06% and 0.98%. It shows misclassification and loss of desired tumor pixels are reduced in great degree. These average values are obtained from 15 tumor cases as shown in Figure 3.To compare the results with other methods, there is no a good standard, however in comparison with works such as in [1,2,7] shows that our method has a better tumor segmentation performance.

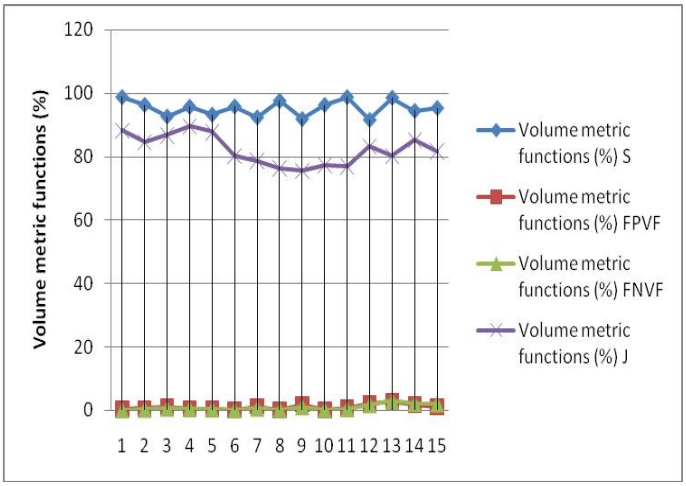


Figure 2: Graph of the quantitative comparison results of three volume metrics for 15 MRI brain tumor images.

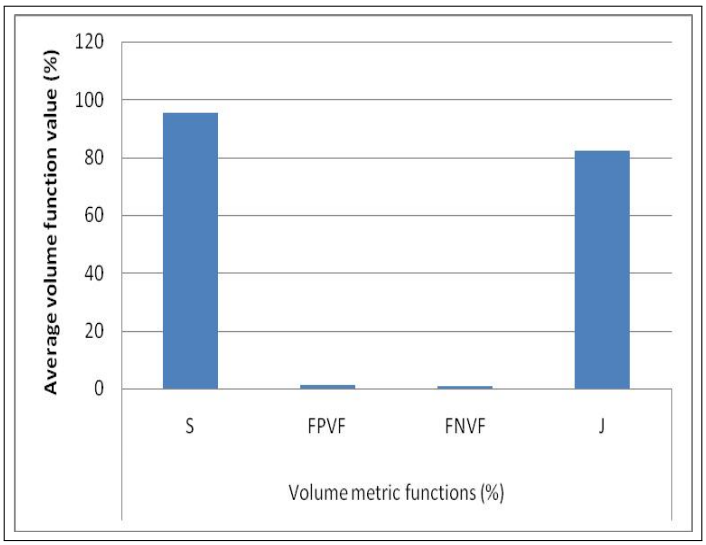


Figure 3: Graph of the average value of the three volume metrics obtained from 15 MRI brain tumor images.

6 Conclusions

We have presented in this paper a tumor segmentation method which combines both region based fuzzy clustering method called EPFCM and boundary based method called GVF snake model .We verified our method with brain tumour MRI images. The obtained results are quantitatively verified with other existing methods and they show that our combined approach provides better result.

Bibliography

[1] M. Prastawa, E. Bullitt, S. Ho, G. Gerig, *A brain tumor segmentation framework based on outlier detection*, Medical Image Analysis, 2004, **18** (3), 217-231.

[2] J.J. Corso, E. Sharon, A. Yuille, *Multilevel segmentation and integrated Bayesian model classification with an application to brain tumor segmentation*, in: MICCAI2006, Copenhagen, Denmark, Lecture Notes in Computer Science, October 2006,Vol. 4191, Springer, Berlin, pp. 790-798.

[3] M.B. Cuadra, C. Pollo, A. Bardera, O. Cuisenaire, J. Villemure, J.-P. Thiran, *Atlas-based segmentation of pathological MR brain images using a model of lesion growth*, IEEE Transactions on Medical Imaging, 2004,23 (10) ,1301-1313.

[4] J.-P. Thirion, *Image matching as a diffusion process: an analogy with Maxwells demons*, Medical Image Analysis, 1998,2 (3) ,243-260.

[5] G. Moonis, J. Liu, J.K. Udupa, D.B. Hackney, *Estimation of tumor volume with fuzzy-connectedness segmentation of MR images*, American Journal of Neuroradiology, 2002,23 ,352-363.

[6] A.S. Capelle, O. Colot, C. Fernandez-Maloigne, *Evidential segmentation scheme of multi-echo MR images for the detection of brain tumors using neighborhood information*, Information Fusion, 2004 ,5 ,203-216.

[7] W. Dou, S. Ruan, Y. Chen, D. Bloyet, J.M. Constans, *A framework of fuzzy information fusion for segmentation of brain tumor tissues on MR images*, Image and Vision Computing, 2007, 25 ,164-171.

[8] M. Schmidt, I. Levner, R. Greiner, A. Murtha, A. Bistritz, *Segmenting brain tumors using alignment-based features*, in: IEEE Internat. Conf. on Machine learning and Applications, 2005, pp. 215-220.

[9] J. Zhou, K.L. Chan, V.F.H Chong, S.M. Krishnan, *Extraction of brain tumor fromMR images using one-class support vector machine*, in: IEEE Conf. on Engineering in Medicine and Biology, 2005, pp. 6411-6414.

[10] A. Lefohn, J. Cates, R. Whitaker, *Interactive, GPU-based level sets for 3D brain tumor segmentation*, Technical Report, University of Utah, April 2003.

[11] Y. Zhu, H. Yang, *Computerized tumor boundary detection using a Hopfield neural network*, IEEE Transactions on Medical Imaging, 1997 ,16 (1) ,55-67.

[12] S. Ho, E. Bullitt, G. Gerig, *Level set evolution with region competition: automatic 3D segmentation of brain tumors*, in: ICPR, Quebec, August 2002, pp. 532-535.

[13] K. Xie, J. Yang, Z.G. Zhang, Y.M. Zhu, *Semi-automated brain tumor and edema segmentation using MRI*, European Journal of Radiology, 2005, 56 ,12-19.

[14] Wang Guoqiang, Wang Dongxue, *Segmentation of Brain MRI Image with GVF Snake*, Model,in: 2010 First International Conference on Pervasive Computing, Signal Processing and Applications,2010,pp.711-714.

[15] Pal, N. R., Pal, K., Keller, J. M., and Bezdek, J. C.A, *Possibilistic fuzzy c-means clustering algorithm*, IEEE Transactions on Fuzzy Systems, 2005, 13(4), pp.517-530.

[16] Buades A,Coll B,Morel J-M. "A non-local algorithm for image denoising",In CVPR 2005:60-5.

[17] Ma, L. and Staunton, R. C., *A modified fuzzy c-means image segmentation algorithm for use with uneven illumination patterns*, Pattern Recognition, 2007, 40(11), pp.3005-3011.

[18] C. Xu and J.L. Prince, *Snakes, shapes, and gradient vector flow*, IEEE Trans. on Image Processing, March 1998,vol. 7, pp. 359-369.

[19] Bingrong Wu, Me Xie, Guo Li, Jingjing Gao, *Medical Image Segmentation Based on GVF Snake Model* IEEE Conference on Second International Intelligent Computation Technology and Automation (ICICTA 09), IEEE Press, 2009, vol. 1,Oct., pp. 637 - 640.

[20] Zijdenbos, A. P., Dawant, B. M., Margolin, R. A., and Palmer, A. C. *Morphometric analysis of white matter lesions in MR images: Method and validation*, IEEE Transactions on Medical Imaging, 1994;13(4):716-724.

INT J COMPUT COMMUN, ISSN 1841-9836
Vol.7 (2012), No. 3 (September), pp. 540-549

Neural Network Model Predictive Control of Nonlinear Systems Using Genetic Algorithms

V. Ranković, J. Radulović, N. Grujović, D. Divac

Vesna Ranković, Jasna Radulović, Nenad Grujović
Faculty of Mechanical Engineering, University of Kragujevac
Department for Applied Mechanics and Automatic Control
Serbia, 34000 Kragujevac; Sestre Janjić 6
E-mail: vesnar@kg.ac.rs, jasna@kg.ac.rs, gruja@kg.ac.rs

Dejan Divac
Institute for Development of Water Resources "Jaroslav Černi"
Serbia, 11000 Belgrade
Jaroslava Černog St., 11226 Beli Potok
E-mail: ddivac@eunet.rs

Abstract: In this paper the synthesis of the predictive controller for control of the nonlinear object is considered. It is supposed that the object model is not known. The method is based on a digital recurrent network (DRN) model of the system to be controlled, which is used for predicting the future behavior of the output variables. The cost function which minimizes the difference between the future object outputs and the desired values of the outputs is formulated. The function *ga* of the Matlab's Genetic Algorithm Optimization Toolbox is used for obtaining the optimum values of the control signals. Controller synthesis is illustrated for plants often referred to in the literature. Results of simulations show effectiveness of the proposed control system.
Keywords: model predictive control, nonlinear system, identification, digital recurrent network, genetic algorithm.

1 Introduction

The predictive controllers are based on the mathematical model of the object, which is being controlled. Nonlinear system identification and prediction is a complex task. All the processes in nature are nonlinear. In large number of processes, the nonlinearities are not prominent, so their behavior can be described by the linear model. In the linear systems theory there exist a large number of methods that can be applied for obtaining the linear model of processes. The nonlinear model must be chosen when the nonlinearity is strongly exhibited. In the identification process, the parameters of the mathematical model are being determined as such that the difference between the system response and its mathematical model is as least as possible, both in the transient regime and in stationary state. The general model of linear processes is ARX (Auto Regressive eXogenous), while for the nonlinear ones it is NARX (Nonlinear Auto Regressive eXogenous). The NARX model structure enables application of the neural networks, the fuzzy systems and the neuro-fuzzy systems for approximation of the nonlinear function.

Neural networks have been applied to the identification of nonlinear dynamical systems. The most of the works are based on multilayer feedforward neural networks with backpropagation learning algorithm. However, the conventional back-propagation algorithm has the problems of local minima and slow rate of convergence. A novel multilayer discrete-time neural network is presented for the identification of nonlinear dynamical systems, [1]. In [2] a new scheme for on-line states and parameters estimation of a large class of nonlinear systems using radial basis function neural network has been designed. A new approach to control nonlinear discrete dynamic systems, which relies on the identification of a

discrete model of the system by a feedforward neural network with one hidden layer, is presented in [3]. Nonlinear system identification via discrete-time recurrent single layer and multilayer neural networks are studied in [4]. In [5] an identification method for nonlinear models in the form of fuzzy-neural networks is introduced. The fuzzy-neural networks combine fuzzy if-then rules with neural networks. The adaptive time delay neural network is used for the identification of nonlinear systems [6], and four architectures are proposed for identifying different classes of nonlinear systems. The identification of nonlinear systems by feedforward neural networks, radial basis function neural networks, Runge-Kutta neural networks and adaptive neuro-fuzzy inference systems is investigated in [7]. Result of simulation indicates that adaptive neuro fuzzy inference systems are a good candidate for identification purposes. However, neural networks are the simplest approaches in the sense of computational complexity. In [8] nonlinear system identification via feedforward neural network and digital recurrent network is studied.

Model predictive control (MPC) is applied to a large number of nonlinear industrial process, [9,10,11,12]. The methodology to design and implement neural predictive controllers for nonlinear system has been developed in [13,14,15].

In [13] feedforward neural networks to estimate the nonlinear process are applied. Also, for the minimization of the cost function, the Matlab⁄s Optimal Toolbox functions *fminunc* and *fmincon* were used. The design methodology for predictive control of industrial processes via recurrent fuzzy neural networks is presented in [14]. In [15] the multilayer perceptron is used to identification of the nonlinear object and genetic algorithm is applied to solve the multi-criteria optimization problem.

In this paper the control of the nonlinear object is studied and it is identified by the DRN. Recurrent networks are more powerful than nonrecurrent networks and have important uses in control and signal processing applications, [16]. They have been shown to be more efficient than feedforward neural networks in terms of the number of neurons required to model a dynamic system [17,18]. Models with recurrent networks are shown to have the capability of capturing various plant nonlinearities, [19,20].

The major objective of the study presented in this paper is to take advantages of the recurrent neural networks for modelling and genetic algorithms for optimization. The proposed method formulates a dynamic nonlinear optimization problem, where the cost function consists of two terms: the differences between the DRN model predictions and the desired output trajectory over a prediction horizon and the control energy over a control horizon. For the solution of nonlinear optimization problem, a genetic algorithm is used, which can approximate the optimum solution very fast, compared to conventional optimization techniques. In this paper, the genetic algorithm has been successfully used in combinations with digital recurrent network.

In the second section the identification of the nonlinear object by application of the DRN is explained. In section three the principle of work of the predictive controllers is analyzed. In this paper for obtaining the optimum values of the control signals, the genetic algorithm (GA) was used. Results of simulations are given in section four, while section five presents the concluding remarks.

2 Neural network for identification of nonlinear dynamic

Different methods have been developed in the literature for nonlinear system identification. These methods use a parameterized model. The parameters are updated to minimize an output identification error.

A wide class of nonlinear dynamic systems with an input and an output can be described by the model:

$$y_m(k) = f_m(\boldsymbol{\varphi}(k), \boldsymbol{\theta}), \tag{1}$$

where $y_m(k)$ is the output of the model, $\boldsymbol{\varphi}(k)$ is the regression vector and $\boldsymbol{\theta}$ is the parameter vector. Depending on the choice of the regressors in $\boldsymbol{\varphi}(k)$, different models can be derived:

• **NFIR** (Nonlinear Finite Impulse Response) model -

$$\varphi(k) = (u(k-1), u(k-2), \ldots, u(k-n_u)),$$

where n_u denotes the maximum lag of the input.

• **NARX** (Nonlinear AutoRegressive with eXogenous inputs) model -

$$\varphi(k) = (u(k-1), u(k-2), \ldots, u(k-n_u), y(k-1), y(k-2), \ldots, y(k-n_y)),$$

where n_y denotes the maximum lag of the output.

• **NARMAX** (Nonlinear AutoRegressive Moving Average with eXogenous inputs) model -

$$\varphi(k) = (u(k-1), u(k-2), \ldots, u(k-n_u), y(k-1), y(k-2), \ldots, y(k-n_y),$$

$$e(k-1), e(k-2), \ldots, e(k-n_e))$$

where $e(k)$ is the prediction error and n_e is the maximum lag of the error.

• **NOE** (Nonlinear Output Error) model -

$$\varphi(k) = (u(k-1), u(k-2), \ldots, u(k-n_u), y_m(k-1), y_m(k-2), \ldots, y_m(k-n_y)).$$

• **NBJ** (Nonlinear Box-Jenkins) model - uses all four regressor types.
The NARX and NOE are the most important representations of nonlinear systems. The block scheme of the DRN model which corresponded to the NOE model is shown in Figure 1.

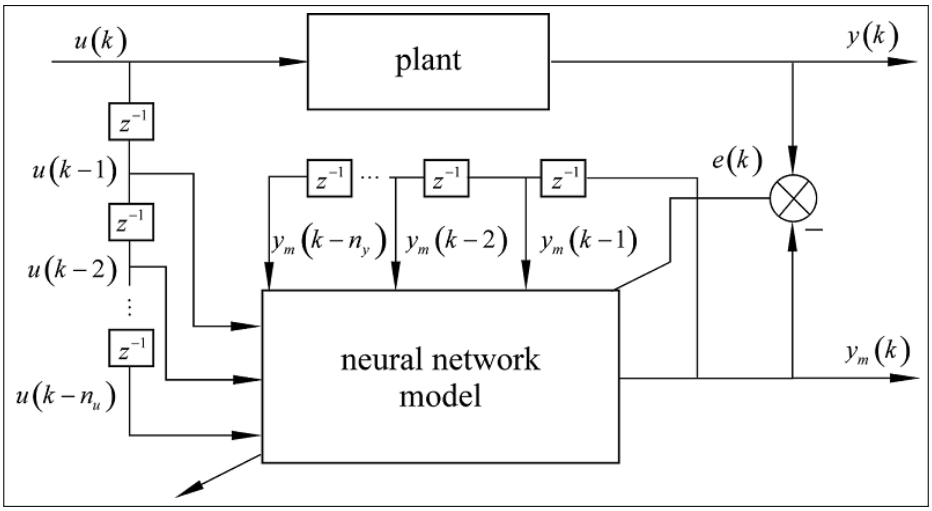


Figure 1: The block scheme of the neural network model

Figure 2 is an example of a DRN. The output of the network is feedback to its input. The output of the network is a function not only of the weights, biases, and network input, but also of the outputs of the network at previous points in time. In [16] dynamic backpropagation algorithm is used to adapt weights and biases.

DRN network is composed of a nonlinear hidden layer and a linear output layer. The inputs $u(k-1), u(k-2), \ldots, u(k-n_u)$ are multiplied by weights ω_{yij} outputs $y_m(k-1), y_m(k-2), \ldots, y_m(k-n_y)$ are

multiplied by weights ω_{yij} and summed at each hidden node. Then the summed signal at a node activates a nonlinear function. The hidden neurons activation function is the hyperbolic tangent sigmoid function. In Figure 2, ω_i represents the weight that connects the node i in the hidden layer and the output node; b_i represents the biased weight for i-th hidden neuron and is a biased weight for the output neuron.

The output of the network is:

$$y_m(k) = \sum_{i=1}^{n_H} \omega_i \nu_i + b \tag{2}$$

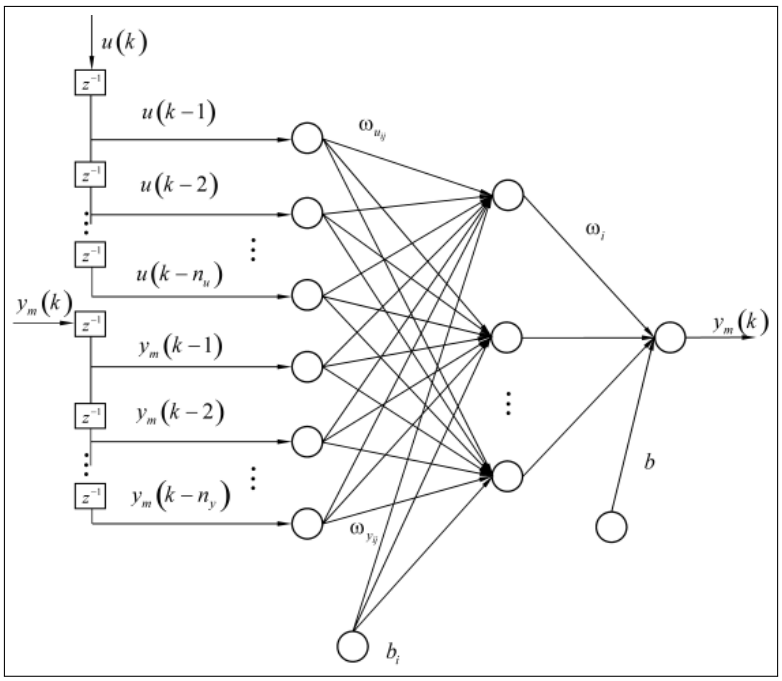


Figure 2: Digital Recurrent Network

where n_H is the number of hidden nodes and:

$$\nu_i = \frac{e^{n_i} - e^{-n_i}}{e^{n_i} + e^{-n_i}} \tag{3}$$

$$n_i = \sum_{i=1}^{n_u} u(k-j)\omega_{u_{ij}} + \sum_{j=1}^{n_y} y_m(k-j)\omega_{y_{ij}} + b_i \tag{4}$$

This error is used to adjust the weights and biases in the network via the minimization of the following function:

$$\varepsilon = \frac{1}{2}[y(k) - y_m(k)]^2 \tag{5}$$

Using the gradient decent, the weight and bias updating rules can be described as:

$$\omega_{u_{ij}}(k+1) = \omega_{u_{ij}}(k) - \eta \frac{\partial \varepsilon}{\partial \omega_{u_{ij}}} \tag{6}$$

$$\omega_{y_{ij}}(k+1) = \omega_{y_{ij}}(k) - \eta \frac{\partial \varepsilon}{\partial \omega_{y_{ij}}} \tag{7}$$

$$b_i(k+1) = b_i(k) - \eta \frac{\partial \varepsilon}{\partial b_i} \tag{8}$$

$$b(k+1) = b(k) - \eta \frac{\partial \varepsilon}{\partial b} \tag{9}$$

where:

$$\frac{\partial \varepsilon}{\partial \omega_{u_{ij}}} = \frac{\partial^e \varepsilon}{\partial y_m} \frac{\partial y_m}{\partial \omega_{u_{ij}}}; \frac{\partial \varepsilon}{\partial \omega_{y_{ij}}} = \frac{\partial^e \varepsilon}{\partial y_m} \frac{\partial y_m}{\partial \omega_{y_{ij}}}; \frac{\partial \varepsilon}{\partial b_i} = \frac{\partial^e \varepsilon}{\partial y_m} \frac{\partial y_m}{\partial b_i}; \frac{\partial \varepsilon}{\partial b} = \frac{\partial^e \varepsilon}{\partial y_m} \frac{\partial y_m}{\partial b};$$

where the superscript e indicates an explicit derivative, not accounting for indirect effects through time.

The terms $\frac{\partial y_m}{\partial \omega_{u_{ij}}}$, $\frac{\partial y_m}{\partial \omega_{y_{ij}}}$, $\frac{\partial y_m}{\partial b_i}$ and $\frac{\partial y_m}{\partial b}$ must be propagated forward through time, [16].

3 Model predictive control

Here the principle of operation of the predictive controllers will be briefly presented. Let us suppose that the mathematical model of the process is known. Based on the model, it is possible to determine the future outputs from the object $y(k+1)$, $l = 1, 2, \ldots, N_H$, where N_H is the prediction horizon. The future outputs depend on the object current states and future control signals, $u(k+l)$, $l = 1, 2, \ldots, N_c$, where N_c is the control horizon and $N_c \leq N_H$. The predictive controllers compute potential future control signals such that the future outputs will be as close as possible to the desired values $r(k+l)$, $l = 1, 2, \ldots, N_H$. Figure 3 shows the basic concept of the model predictive control.

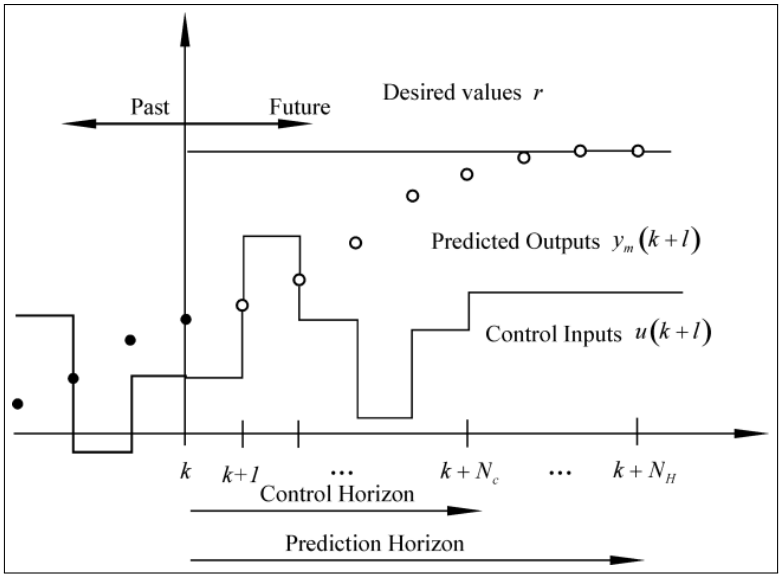


Figure 3: Basic concept of a model predictive control method

There are several types of the predictive controllers. In this work the GPC (General Predictive Control) controller is used, where the cost function is calculated as:

$$J(k) = \sum_{l=1}^{N_H} \left[r(k+l) - y_m(k+l) \right]^2 + \alpha \sum_{l=1}^{N_c} \Delta u^2(k+l-1), \tag{10}$$

where:

$$\Delta u^2(k+l-1) = u(k+l-1) - u(k+l-2),$$

and α is the weight factor of the control signal.

Real processes are subject to constraints. We consider constraints which limit the range of the control signal, the gradient of the control signal and the future model predictions:

$$u_{min} \leq u(k+l) \leq u_{max}$$

$$\mid u(k+l) - u(k+l-1) \mid \leq \Delta u_{max}$$

$$y_{min} \leq y_m(k+l) \leq y_{max}.$$

Model output $y_m(k+l)$ is calculated based on (2):

$$y_m(k+l) = \sum_{i=1}^{n_H} \omega_i \cdot \frac{1}{1+e^{-(\sum_{j=1}^{n_u} u(k+l-j)\omega_{u_{ij}} + \sum_{j=1}^{n_y} u(k+l-j)\omega_{y_{ij}} + b_i)}} + b. \tag{11}$$

Taking into account the term $\alpha \sum_{m=1}^{N_c} \Delta u^2(k+l-1)$ in the cost function (10) prevents the control signals to be too big such that the executive organs would not be able to realize.

The control signals in the k-th step: $[u(k), u(k+1), \ldots, u(k+N_c-1)]^T$, is possible to solve both numerically and analytically. Analytically it is solved in such a manner that from the system of algebraic equations:

$$\frac{\partial J}{\partial u(k)} = 0, \frac{\partial J}{\partial u(k+1)} = 0, \ldots, \frac{\partial J}{\partial u(k+N_c-1)} = 0$$

one obtains $[u(k), u(k+1), \ldots, u(k+N_c-1)]^T$.

For analytical solution, it is necessary for the model to be linear. The neural network model is nonlinear, its linearization should be performed first, or apply the numerical methods for solving the optimization problem. In this work for obtaining the optimum values of the control signals, the genetic algorithm is used (the function ga of the Matlab's Genetic Algorithm Toolbox). The genetic algorithms represent the global optimization technique. In the k-th step $N_c - 1$ control signals are obtained. The first calculated signal is being sent to the controller output.

The GA used in this study is simple genetic algorithm. The elements of the populations are encoded into bit-strings. The chromosome selection for reproduction is performed using the Roulette selection method. The multi-point crossover operator was used. The uniform mutation operator is applied in this study.

4 Simulation results

Example 1

For the simulation example 1, we consider the nonlinear plant, which is described by the following nonlinear difference equation:

$$y(k) = 0.5y(k-1) + u(k-1)\Big(1 + 0.2y^2(k-1)\Big) + u^3(k-1), \tag{12}$$

where y is the output of the plant and u is the plant input.

We assume that structure of the model is known, $n_u = 1, n_y = 1$. The inputs and output of the neural network model are $u(k-1), y_m(k-1)$ and $y_m(k)$ respectively. The data set are obtained by applying random input signal uniformly distributed in the interval [-0.5 0.5]. The plant output is bounded within region [-2.2 2.2]. In this example, 4500 training patterns are generated to train the DRN and 1500 to test the obtained DRN model.

Selection of an appropriate number of neurons in the hidden layer is very important. The optimal network size was selected from the one which resulted in maximum correlation coefficient for the training and test sets, Table 1.

Based on Table 1, it was concluded that the optimal number of hidden neurons is 12.

Table 1.Correlation coefficient for the training and test sets

DRN-structure	2-10-1	2-12-1	2-15-1	2-17-1
Training	0.9878	0.9987	0.9856	0.9879
Test	0.9845	0.9921	0.9795	0.9812

Since the object model is formed, it is necessary to define the cost function parameters, (10). These parameters are selected by the trial and error method. By increasing the variance of the control signal is decreasing, but simultaneously the difference between the set and real value of the object output is increasing. In the considered example, the satisfactory results are obtained for the following values of parameters: $N_H = 3, N_C = 3, \alpha = 0.05$.

The optimal predictive controller in the k-th step computes the control signals:

$[u(k), u(k+1)]^T$.

In Figure 4, the reference signal is shown. The difference between the reference signal and the object output is presented in Figure 5. From the Figure 5, it is obvious that the tracking error is small. The obtained solution is good for practical realization.

Example 2

The plant is given by:

$$y(k) = 0.35\left\{\frac{y(k-1)y(k-2)\Big[y(k-1)+2.5\Big]}{1+y^2(k-1)+y^2(k-2)} + u(k-1)\right\}, \tag{13}$$

where y is the output of the plant and u is the plant input. The dynamical system used in the simulation example is given in [21]. It is assumed that structure of the model is known, $n_u = 1, n_y = 2$. The inputs and output of the neural network are $u(k-1), y_m(k-1), y_m(k-2)$ and $y_m(k)$, respectively.

The input-output patterns are generated randomly. The data set included 4000 data samples. In the training process of the DRN, 3000 samples were used. The DRN model was tested using 1000 selected data. In this example, it is found that the optimal number of hidden neurons is 15 (Table 2).

Table 2.Correlation coefficient for the training and test sets

DRN-structure	3-10-1	3-12-1	3-15-1	3-17-1
Training	0.9811	0.9899	0.9945	0.9921
Test	0.9799	0.9831	0.9897	0.9895

The control and predictive horizons are chosen as respectively, $N_H = 3$, and $N_C = 3$. The choice of these values determines the complexity of the optimization problem. The values of N_H and N_C may not be too large, because the computation time would become also too large. These values should provide a good compromise between performance and computational load. It is found that the weight factor of the control signal $\alpha = 0.045$ by the trial and error method.

The optimal predictive controller in the k-th step computes the control signals:
$[u(k), u(k+1)]^T$. The reference signal is shown in Figure 6. The difference between the reference signal and the object output is presented in Figure 7.

From the Figure 7, it is obvious that the control is quite effective. The obtained solution is good for practical realization.

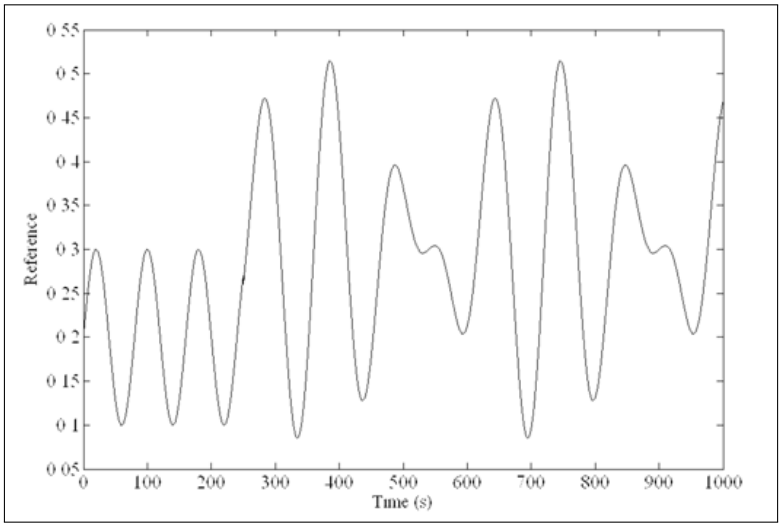


Figure 4: The reference signal

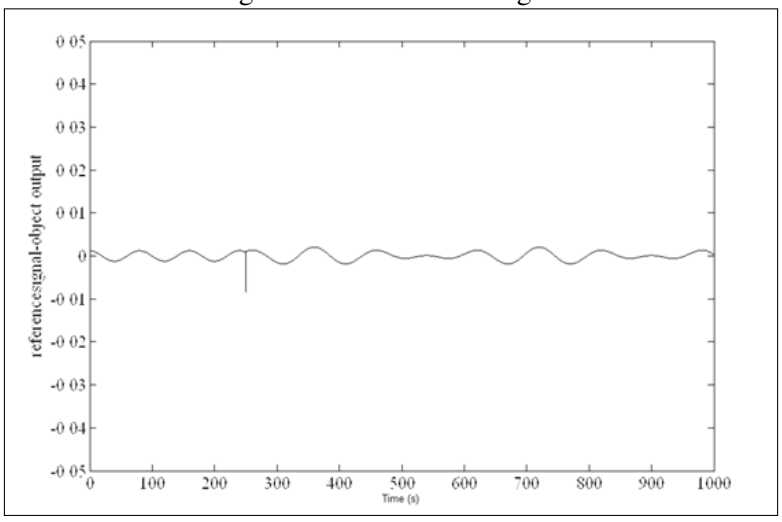


Figure 5: Difference between the reference signal and the object output

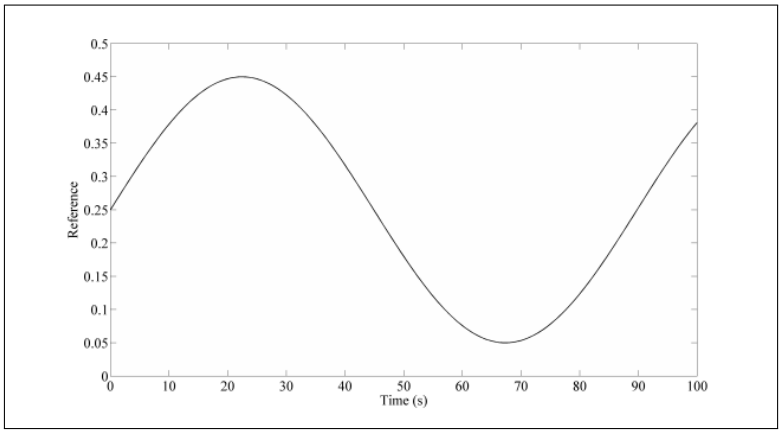


Figure 6: The reference signal

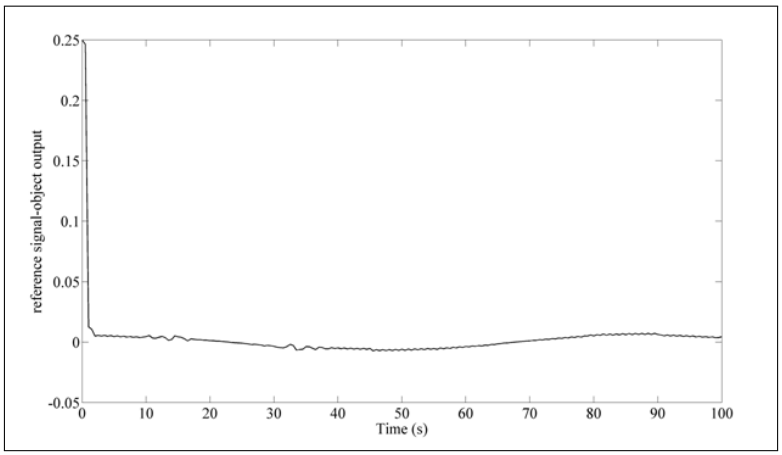


Figure 7: Difference between the reference signal and the object output

5 Conclusions

In this paper the synthesis of the predictive controller for control of the nonlinear object is considered. The object is modeled by the digital recurrent network. In the designing of neural network model, the problem is how to determine an optimal architecture of network. The determination of the values of n_u and n_y is an open question. Large time lags result in better prediction of the NN. However, large n_u and n_y also result in large number of parameters (weights and biases) that need to be adapted.

The simulation results, given in Section 4, show that the predictive controllers can successfully be applied for control of the prominently nonlinear object. The optimum values of the control signals are obtained by the genetic algorithm which represents the global optimization technique. The given trajectory tracking error is small. The proposed structure can be applied in control of the linear objects that are modeled by the neural network. Recurrent networks are more powerful than nonrecurrent networks and have important uses in control and signal processing applications.

Simulation results on the examples selected from literature provide good results and that encourages our future plan for real time control system implementation.

Acknowledgement

Parts of this research were supported by the Ministry of Sciences, Technologies and Development of Republic of Serbia.

Bibliography

[1] S. Jagannathan, F.L. Lewis, Identification of Nonlinear Dynamical Systems Using Multilayered Neural Networks, *Automatica*, 32(12):1707-1712, 1996.

[2] G. Kenne, T. Ahmed-Ali, F. Lamnabhi Lagarrigue, H. Nkwawo, Nonlinear systems parameters estimation using radial basis function network, *Control Engineering Practice*, 14(7):819-832, 2006.

[3] J.I. Canelon, L. Shieh, N.B. Karayiannis, A new approach for neural control of nonlinear discrete dynamic systems, *Information Sciences*, 174(3-4):177-196, 2005.

[4] W. Yu, Nonlinear system identification using discrete-time recurrent neural networks with stable learning algorithms, *Information Sciences*, 158(1):131-147, 2004.

[5] S.-K. Oh, W. Pedrycz, H.-S. Park, Hybrid identification in fuzzy-neural networks, *Fuzzy Sets and Systems*, 138(2):399-426, 2003.

[6] A. Yazdizadeh, K. Khorasani, Adaptive time delay neural network structures for nonlinear system identification, *Neurocomputing*, 47(1-4):207-240, 2002.

[7] M. Onder Efe, O. Kaynak, A comparative study of neural network structures in identification of nonlinear systems, *Mechatronics*, 9(8):287-300, 1999.

[8] V. Ranković, I. Nikolić, Identification of Nonlinear Models with Feedforward Neural Network and Digital Recurrent Network, *FME Transactions*, 36(2):87-92, 2008.

[9] S. Dubljevic, P. Mhaskar, N.H. El-Farra, P.D. Christofides, Predictive control of transport-reaction processes, *Computers and Chemical Engineering*, 29(11-12):2335-2345, 2005.

[10] P. Mhaskar, N.H. El-Farra, P.D. Christofides, Robust hybrid predictive control of nonlinear systems, *Automatica*, 41(2):209-217, 2005.

[11] H. Peng, T. Ozaki, Y. Toyoda, K. Oda, Exponential ARX model-based long-range predictive control strategy for power plants, *Control Engineering Practice*, 9(12):1353-1360, 2001.

[12] V. Ranković, I. Nikolić, Model Predictive Control Based on the Takagi-Sugeno Fuzzy Model, *Journal of Information*, *Control and Management Systems*, 5(1):101-110, 2007.

[13] M. Lazar, O. Pastravanu, A neural predictive controller for non-linear systems, *Mathematics and Computers in* Simulation*,* 60(3-5):315-324, 2002.

[14] C.-H. Lu, C.-C. Tsai, Generalized predictive control using recurrent fuzzy neural networks for industrial processes, *Journal of Process Control*, 17(1):83-92, 2007.

[15] K. Laabidi, F. Bouani, M. Ksouri, Multi-criteria optimization in nonlinear predictive control, *Mathematics and Computers in Simulation*, 76(5-6):363-374, 2008.

[16] M. Hagan, O.D. Jesus, R. Schultz, Training Recurrent Networks for Filtering and Control, Chapter 11 of Recurrent Neural Networks: Design and Applications, L.R. Medsker and L.C. Jain, Eds., CRC Press, 325-354, 1999.

[17] R.K. Al Seyab, Y. Cao, Nonlinear system identification for predictive control using continuous time recurrent neural networks and automatic differentiation, *Journal of Process Control*, 18(6):568-581, 2008.

[18] D.R. Hush, B.G. Horne, Progress in supervised neural networks, *IEEE Signal Processing Magazine*, 10(1):8-39, 1993.

[19] K.L. Funahashi, Y. K.L. Funahashi, Y. Nakamura, Approximation of dynamical systems by continuous time recurrent neural networks, *Neural Networks*, 6(6):183-192, 1993.

[20] L. Jin, P. Nikiforuk, M. Gupta, Approximation of discrete-time state-space trajectories using dynamic recurrent neural networks, *IEEE Transactions on Automatic Control*, 40(7):1266-1270, 1995.

[21] P.S. Sastry, G. Santharam, K.P. Unnikrishnan, Memory Neuron Networks for Identification and Control of Dynamical Systems, *IEEE Transactions on Neural Net-works*, 5(2):306-319, 1994.

INT J COMPUT COMMUN, ISSN 1841-9836
Vol.7 (2012), No. 3 (September), pp. 550-564

Improving Tracking Performance of Predictive Functional Control Using Disturbance Observer and Its Application to Table Drive Systems

T. Satoh, K. Kaneko, N. Saito

Toshiyuki Satoh, Kotaro Kaneko, Naoki Saito
Akita Prefectural University
Department of Machine Intelligence and Systems Engineering
84-4, Aza-Ebinokuchi, Tsuchiya, Yurihonjo, Akita, Japan
E-mail: tsatoh@akita-pu.ac.jp, m12a011@akita-pu.ac.jp
naoki_saito@akita-pu.ac.jp

Abstract: A practical approach for improving tracking performance of the predictive functional control (PFC) is proposed. The disturbance observer is utilized to nominalize the actual plant and to reduce the predicted output error in the PFC algorithm by canceling not only constant but also high-order disturbances. The proposed control scheme is experimentally validated on a single axis table drive system and is compared with the standard PFC and the industrial cascade control. The experimental results prove the effectiveness of the proposed disturbance observer-based PFC scheme.
Keywords: predictive functional control, disturbance observer, table drive system, model predictive control

1 Introduction

The model-based predictive control (MPC) has been successfully applied mainly in the petrochemical industry since it was put into practical use in the 1970's. In general, the quadratic optimization problem must be solved on-line to compute the optimal control sequence in the standard constrained multivariable MPC algorithm [14]. When the MPC is applied to the control of mechatronic servo systems, the time-consuming optimization may be problematic since the sampling period of such systems is usually less than a few milliseconds and may be too short to complete the optimization.

On the other hand, a simple MPC scheme called the predictive functional control (PFC) [11–13] is widely used. Unlike the standard MPC, the PFC is primarily intended to single-input-single-output systems, and no on-line optimization is required since the control input is expressed as a linear combination of time-dependent basis functions. Although the PFC algorithm is much simpler than the standard MPC algorithm, it gives a similar performance to the full MPC. Accordingly, a variety of successful industrial applications have been reported (*e.g.*, [2–5, 7, 8, 10, 20]).

The tracking performance attained by the PFC heavily depends on the accuracy of the internal model of the actual plant. So, if the predicted output error caused by disturbances or model uncertainties is significant, the tracking performance may deteriorate, and the designed response cannot be obtained on the actual system. Another weakness of the PFC is that it cannot cope with high-order disturbances, such as ramp or parabola disturbances. Assume that disturbances entering the system are the $1/s^k$-type ($k \geqslant 1$). Then, for non-integrating plants, it is known that the PFC is offset-free for stepwise (*i.e.*, $k = 1$) disturbances, which means that the stepwise disturbance is asymptotically rejected. However, this property does not hold true for high-order (*i.e.*, $k \geqslant 2$) disturbances [1]. Also, for integrating plants, even stepwise disturbances cannot be rejected in the PFC scheme.

[1] Among high-order disturbances, ramp disturbances are especially worth considering. Examples of ramp disturbances include the viscous friction under uniform accelerated motion in mechatronic systems, or flow fluctuations in petrochemical plants.

The purpose of this paper is to propose a practical way to overcome the above-mentioned drawbacks. To this end, we construct a dual loop control structure which consists of an inner loop formed by the disturbance observer (DOB) [15, 16, 19] and an outer loop formed by the PFC. The DOB is known to be an effective compensation mechanism which reduces the influence of disturbances, uncertainties and nonlinearities in the plant and enforces the nominal input/output behavior on the actual plant especially in the low frequency range where the frequency of the reference signal concentrates. Accordingly, the actual plant behaves as if it is the nominal plant by introducing the DOB. In the DOB design, we can specify the number of integrators introduced in the loop transfer function. Hence, the DOB has the ability to asymptotically reject high-order disturbances as well as stepwise disturbances.

The contributions of the paper are as described below. First, it offers a way of improving the prediction accuracy within the PFC algorithm. This leads to the improvement of the transient performance of the PFC in the presence of disturbances and/or modeling errors since the DOB nominalizes the actual plant especially at low frequencies. Second, it solves the inherent weakness of the PFC against the high-order disturbances. When the $1/s^k$-type ($k \geqslant 2$) disturbances are applied, those disturbances cannot be rejected by the standard PFC controller. However, due to the introduction of the DOB, such high-order disturbances can be asymptotically rejected since the DOB has the specified number of integral action.

We should mention here that the idea of the combination of the PFC and the disturbance observer has originated in the recent application to the control of a pneumatic artificial muscle actuator by the authors [18]. However, in-depth argument about the advantages of the proposed scheme was not provided there. The present paper states the advantages that were left unclear in [18]. In addition, the proposed method is validated on a single axis table drive system and compared with the standard PFC and the industrial cascade control which consists of the inner proportional-plus-integral control loop and the outer proportional control loop. We also examine the performance differences between two different control systems that have the same PFC controller but different DOBs.

2 Predictive Functional Control Enhanced with Disturbance Observer

2.1 Predictive Functional Control

In this subsection, we give a brief overview of the *predictive functional control* (PFC). For more details of the PFC algorithm based on the state-space model, see, for example, S. Abu el Ata-Doss *et al.* [1].

Figure 1 shows the basic concept of the PFC. Suppose that the current time is labeled as time step k. A *set-point trajectory* is defined as a command signal which the process output y_P should follow, and the value of the set-point trajectory at the current time step is denoted by $c(k)$. Also shown is a *reference trajectory* denoted by y_R. This trajectory starts at the current process output $y_P(k)$ and defines a desired trajectory along which the process output y_P should approach the set-point trajectory. On the reference trajectory, there are a few *coincidence points* on which the performance index is defined so that the process output y_P will coincide with the reference trajectory y_R. As an example, three coincidence points are drawn in Figure 1. The optimal control input trajectory is then computed on the basis of the predicted output. Once we have computed a future control input trajectory, we apply only the first element to the process. At the next time step, we repeat the whole cycle from the definition of the reference trajectory to the application of the first element of the optimal control input trajectory. We call this way of control a *receding horizon control*.

Next, we show the basic PFC algorithm which is a slightly modified formulation for handling time delay. If we set $d = 0$ in the following description, the control law accords with the one without time delay. In the following, let $\mathbb{R}^n$ and $\mathbb{Z}^n$ be the set of all n-dimensional real vectors and the set of all n-dimensional integer vectors, respectively. Now assume that the plant is stable and has the time delay of L and that the sampling period is T_s. The development of the PFC algorithm is based on the following

SISO discrete-time linear state-space model of the plant:

$$\begin{cases} x_M(k+1) = A_M x_M(k) + B_M u(k), \\ y_M(k) = C_M x_M(k) \end{cases} \tag{1}$$

where $x_M \in \mathbb{R}^n$ is the state vector, $u \in \mathbb{R}$ is the control input, $y_M \in \mathbb{R}$ is the model output, respectively. Here, the model output $y_M(k)$ is used to predict the future plant output $\hat{y}_P(k+d)$ where $d \in \mathbb{Z}$ is defined as the nearest integer of L/T_s. Assume that the following condition holds:

$$\det \begin{pmatrix} A_M - I & B_M \\ C_M & 0 \end{pmatrix} \neq 0. \tag{2}$$

Then the reference trajectory is defined as follows:

$$y_R(k+d+i) := c(k+d+i) - \alpha^i \left(c(k+d) - \hat{y}_P(k+d)\right), \ i = 0, 1, \ldots \tag{3}$$

where $\alpha \in \mathbb{R}$ is a parameter which adjusts the approaching ratio of the reference trajectory to the set-point ($0 < \alpha < 1$). For example, Dieulot *et al.* [3] have chosen the parameter α as $\alpha = e^{-3T_s/T_{\mathrm{CLTR}}}$ along with the following three coincidence points:

$$\begin{pmatrix} h_1 & h_2 & h_3 \end{pmatrix} = \begin{pmatrix} \dfrac{T_{\mathrm{CLTR}}}{3T_s} & \dfrac{T_{\mathrm{CLTR}}}{2T_s} & \dfrac{T_{\mathrm{CLTR}}}{T_s} \end{pmatrix} \tag{4}$$

where $T_{\mathrm{CLTR}} \in \mathbb{R}$ is constant and called the *desired closed-loop time response*, which is taken as the time required to reach 95% of the final value [13]. In many cases, the performance index is defined as the quadratic sum of the errors between the predicted process output $\hat{y}_P$ and the reference trajectory y_R as follows:

$$J(k) := \sum_{j=1}^{n_h} \left\{ \hat{y}_P\left(k+d+h_j\right) - y_R\left(k+d+h_j\right) \right\}^2 \tag{5}$$

where $h_j \in \mathbb{Z}$ ($j = 0, 1, \ldots, h$) and $n_h \in \mathbb{Z}$ are respectively the coincidence time point and the number of coincidence points. In the PFC, the future control input computed at each sampling instant is assumed to be the sum of weighted basis functions, and a time-dependent polynomial basis is usually employed. Then the optimal control input that minimizes the performance index (5) is given by

$$u(k) = k_0 \{c(k+d) - y_P(k)\} - \sum_{m=1}^{d_e} k_m e_m(k+d) + \tilde{v}_x^T x_M(k) + \tilde{v}_{xd}^T x_M(k-d) \tag{6}$$

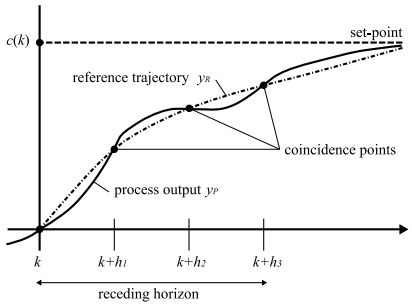


Figure 1: Concept of the predictive functional control.

where $d_e \in \mathbb{Z}$ and $e_m \in \mathbb{Z}$ respectively denote the degree of the polynomial and unknown coefficients that are used when the predicted error at the future time step $k+d+i$ is approximated by a time-dependent polynomial, and $k_0 \in \mathbb{R}$, $k_m \in \mathbb{R}$, $\tilde{\nu}_x \in \mathbb{R}^n$ and $\tilde{\nu}_{xd} \in \mathbb{R}^n$ are respectively given by

$$k_0 = \nu^T \begin{pmatrix} 1-\alpha^{h_1} \\ 1-\alpha^{h_2} \\ \vdots \\ 1-\alpha^{h_{n_h}} \end{pmatrix}, \quad k_m = \nu^T \begin{pmatrix} h_1^m \\ h_2^m \\ \vdots \\ h_{n_h}^m \end{pmatrix}, \quad \tilde{\nu}_x = -\begin{pmatrix} C_M\left(A_M^{h_1} - \alpha^{h_1} I\right) \\ C_M\left(A_M^{h_2} - \alpha^{h_2} I\right) \\ \vdots \\ C_M\left(A_M^{h_{n_h}} - \alpha^{h_{n_h}} I\right) \end{pmatrix}^T \nu, \quad \tilde{\nu}_{xd} = -\begin{pmatrix} \left(\alpha^{h_1}-1\right) C_M \\ \left(\alpha^{h_2}-1\right) C_M \\ \vdots \\ \left(\alpha^{h_{n_h}}-1\right) C_M \end{pmatrix}^T \nu. \quad (7)$$

In (7), $\nu \in \mathbb{R}^{h_{n_h}}$ is given by

$$\nu = \begin{pmatrix} y_B(h_1) & \cdots & y_B(h_{n_h}) \end{pmatrix}^T \left(\sum_{j=1}^{n_h} y_B\left(h_j\right) y_B\left(h_j\right)^T \right)^{-1} U_B(0) \quad (8)$$

where $y_B\left(h_j\right) = \begin{pmatrix} y_{B_1}\left(h_j\right) & \cdots & y_{B_{n_B}}\left(h_j\right) \end{pmatrix}^T \in \mathbb{R}^{h_{n_h} \times n_B}$ and $U_B(0) = \begin{pmatrix} 1 & 0 & \cdots & 0 \end{pmatrix}^T \in \mathbb{Z}^{n_B}$. Here, $y_{B_l}(i) \in \mathbb{R}$ is the forced response to the basis function of the form i^{l-1} $(l = 1, 2, \ldots, n_B)$.

The second term in the right-hand side of (6) denotes the term to compensate for the prediction error due to disturbances and/or uncertainties. The unknown coefficients $e_m(k+d)$ $(m = 1, 2, \ldots, d_e)$ are determined in the following fashion. First, the number of steps h_c used for the polynomial approximation is specified. Then, at each sampling instant, the following vectors $\varphi \in \mathbb{R}^{h_c}$, $\theta \in \mathbb{R}^{d_e}$ and a matrix $H \in \mathbb{Z}^{h_c \times d_e}$ are defined:

$$\varphi := \begin{pmatrix} e(k+d) - e(k+d-h_c) \\ e(k+d-1) - e(k+d-h_c) \\ \vdots \\ e(k+d-h_c+1) - e(k+d-h_c) \end{pmatrix}, \quad H := \begin{pmatrix} h_c & h_c^2 & \cdots & h_c^{d_e} \\ h_c - 1 & (h_c-1)^2 & \cdots & (h_c-1)^{d_e} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{pmatrix}, \quad \theta := \begin{pmatrix} e_1(k+d) \\ e_2(k+d) \\ \vdots \\ e_{d_e}(k+d) \end{pmatrix}. \quad (9)$$

The problem is to approximate φ by $H\theta$ in the least-squares sense. The vector of unknown coefficients θ that minimizes the square error $J(\theta) := (\varphi - H\theta)^T (\varphi - H\theta)$ is hence given by

$$\theta = \left(H^T H\right)^{-1} H^T \varphi. \quad (10)$$

Once the number of steps h_c is specified, the matrix $\left(H^T H\right)^{-1} H^T$ can be preliminarily computed off-line. Therefore, the optimal coefficients $e_m(k)$ $(m = 1, 2, \ldots, d_e)$ can be determined by just updating the vector φ and computing (10) on-line. The mechanism which extrapolates the past prediction error to the future prediction horizon by using the coefficients given in (10) is called the *auto-compensation*. The use of the auto-compensation is optional, so the PFC is often utilized without the second term in the right-hand side of (6).

2.2 Disturbance Observer

The *disturbance observer* (DOB) has the structure depicted in Figure 2 where $P(s)$ is the transfer function of the real plant, $P_n(s)$ is the nominal model of the plant, and $Q(s)$ is a proper and stable filter. The nominal model $P_n(s)$ is supposed to be minimum-phase in the following. On the assumption that disturbances acting on the system, modeling errors and nonlinearities can be regarded as an equivalent

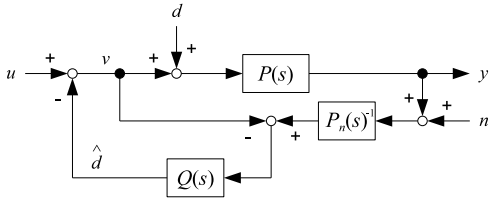


Figure 2: Structure of the disturbance observer.

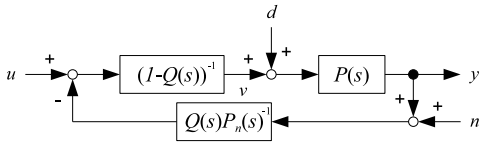


Figure 3: Equivalent form of the disturbance observer.

disturbance d at the plant input, the DOB computes the estimate $\hat{d}$ of the current disturbance d, which is subtracted from the input u to cancel the disturbance.

The designed parameter of the DOB is the filter $Q(s)$. This filter is closely related to the sensitivity properties of the DOB. For example, the transfer function from the disturbance d to the measurement output y is given by

$$G_{yd}(s) = \frac{(1 - Q(s)) P(s) P_n(s)}{P_n(s) + (P(s) - P_n(s)) Q(s)}. \tag{11}$$

Accordingly, to reject the disturbance d asymptotically, the gain of $Q(s)$ should be unity at low frequencies. On the other hand, the transfer function from the measurement noise n to the measurement output y is given by

$$G_{yn}(s) = \frac{-P(s) Q(s)}{P_n(s) + (P(s) - P_n(s)) Q(s)}. \tag{12}$$

Hence, to suppress the measurement noise, the gain of $Q(s)$ should be 0 at high frequencies. Therefore, $Q(s)$ should be a low-pass filter with the DC gain of 1. Since the closed-loop transfer function from the input u to the measurement output y in Figure 2 is given by

$$G_{yu}(s) = \frac{P(s) P_n(s)}{(1 - Q(s)) P_n(s) + P(s) Q(s)}, \tag{13}$$

the transfer property approximates the nominal plant $P_n(s)$ at low frequencies if $Q(s)$ is chosen as stated above. This means that the DOB can nominalize the real plant at low frequencies.

When the plant has no unstable zeros, the low-pass filter $Q(s)$ can be selected as [19]

$$Q(s) = \frac{1 + \sum_{m=1}^{n_q - \rho_q} f_m s^m}{1 + \sum_{m=1}^{n_q} f_m s^m} \tag{14}$$

where $n_q \in \mathbb{Z}$, $\rho_q \in \mathbb{Z}$ and $f_m \in \mathbb{R}$ are respectively the order of $Q(s)$, the relative degree of $Q(s)$ and unknown coefficients to be determined. To make $Q(s) P_n(s)^{-1}$ proper, ρ_q must be chosen as $\rho_q \geqslant \rho_P$ where $\rho_P \in \mathbb{Z}$ is the relative degree of $P_n(s)$. Figure 3 shows the equivalent form of the disturbance observer. In the figure, $(1 - Q(s))^{-1}$ can be written as

$$\frac{1}{1 - Q(s)} = \frac{1 + \sum_{m=1}^{n_q} f_m s^m}{s^{n_I} \left(f_{n_I} + \sum_{m=1}^{\rho_q} f_{n_I + m} s^m \right)} \tag{15}$$

where $n_I = n_q - \rho_q + 1$, if the Q-filter of the form given in (14) is used. Hence, we can see that the loop transfer function includes n_I integrators in Figure 3, which means that the DOB can asymptotically reject up to the $1/s^{n_I}$-type disturbances.

Various types of analogue filters can be utilized to realize $Q(s)$, and in many cases, the Butterworth or the binomial low-pass filters are used. More specifically, the Butterworth filter is a reasonable option

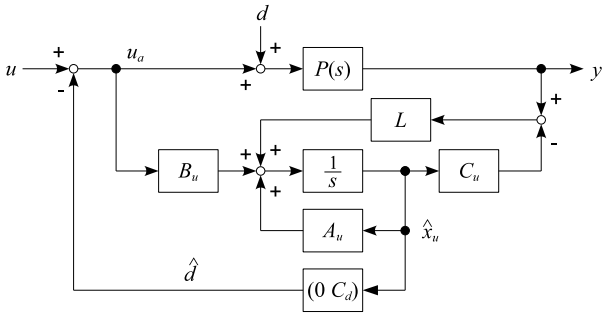


Figure 4: Structure of the unknown input disturbance observer.

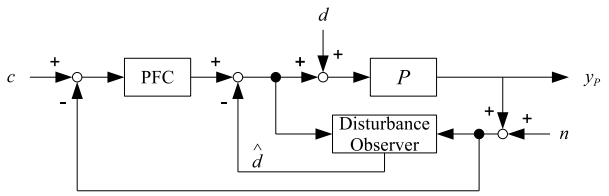


Figure 5: Structure of the proposed control system.

when $n_q = \rho_q$, which means that the degree of the numerator polynomial of $Q(s)$ is 0. Otherwise, the binomial filter of the following form is useful:

$$Q(s) = \frac{1 + \sum_{m=1}^{n_q-\rho_q} a_m \left(s\tau_{n_q}\right)^m}{1 + \sum_{m=1}^{n_q} a_m \left(s\tau_{n_q}\right)^m} \tag{16}$$

where $a_m\,(m = 1, 2, \ldots, n_q)$ are the binomial coefficients (*i.e.*, $a_m = n_q!/m!(n_q - m)!$) and τ_{n_q} is a design variable to be determined.

Aside from the DOB, another type of disturbance observer based on the unknown input observer, which is referred to as the *unknown input disturbance observer* (UIDOB), has also been proposed [6]. The UIDOB is a natural extension of the Luenberger observer, and it is based on the following state-space models of the plant and the fictitious disturbance generator:

$$\text{plant}\begin{cases} \dot{x}(t) = Ax(t) + B\left(u_a(t) + d(t)\right), \\ y(t) = Cx(t), \end{cases} \qquad \text{disturbance generator}\begin{cases} \dot{z}(t) = A_d z(t), \\ d(t) = C_d z(t) \end{cases} \tag{17}$$

where $x \in \mathbb{R}^n$ is the state variable of the plant, $u_a \in \mathbb{R}$ is the control input, $y \in \mathbb{R}$ is the plant output, $z \in \mathbb{R}^{n_d}$ is the state variable of the disturbance generator and $d \in \mathbb{R}$ is the disturbance. Notice that the eigenvalues of A_d are not restricted to those on the origin (*i.e.*, disturbances are not necessarily the $1/s^k$-type). It is assumed that the pair (C, A) and (C_d, A_d) are both observable and that no eigenvalues of A_d coincide with zeros of the plant. The UIDOB is based on the augmented system obtained from (17) and is given by

$$\begin{cases} \begin{pmatrix} \dot{\hat{x}}(t) \\ \dot{\hat{z}}(t) \end{pmatrix} = \begin{pmatrix} A & BC_d \\ 0 & A_d \end{pmatrix} \begin{pmatrix} \hat{x}(t) \\ \hat{z}(t) \end{pmatrix} + \begin{pmatrix} B \\ 0 \end{pmatrix} u_a(t) + L\left\{ y(t) - \begin{pmatrix} C & 0 \end{pmatrix} \begin{pmatrix} \hat{x}(t) \\ \hat{z}(t) \end{pmatrix} \right\}, \\ \begin{pmatrix} \hat{x}(t) \\ \hat{d}(t) \end{pmatrix} = \begin{pmatrix} I & 0 \\ 0 & C_d \end{pmatrix} \begin{pmatrix} \hat{x}(t) \\ \hat{z}(t) \end{pmatrix} \end{cases} \tag{18}$$

where $\hat{x}$, $\hat{z}$ and $\hat{d}$ are the estimates of x, z and d, respectively. $L \in \mathbb{R}^{n+n_d}$ is the observer gain which is chosen to stabilize the observer system. Figure 4 shows the structure of the UIDOB where

$$A_u = \begin{pmatrix} A & BC_d \\ 0 & A_d \end{pmatrix},\ B_u = \begin{pmatrix} B \\ 0 \end{pmatrix},\ C_u = \begin{pmatrix} C & 0 \end{pmatrix},\ \hat{x}_u(t) = \begin{pmatrix} \hat{x}(t) \\ \hat{z}(t) \end{pmatrix}. \tag{19}$$

The estimate $\hat{d}$ is fed back to the control input to compensate for the input disturbance d. Unlike the DOB, the UIDOB estimates $\hat{x}$ as well as $\hat{d}$, so the observer-based state feedback technique can be applicable to stabilize the plant.

The equivalence and difference between the DOB and UIDOB has been already investigated by Schrijver and van Dijk [9][2], and they have shown that the DOB is equivalent to the UIDOB for some specific design choices. Actually, the DOB is more general than the UIDOB for the following reasons [9]:

- Given any UIDOB designed for the $1/s^k$-type disturbance model, we can always find exactly the same DOB as the given UIDOB.
- It is possible to design a DOB by freely specifying the relative degree of the $Q(s)$. However, in the UIDOB case, the relative degree always becomes $\rho_P + 1$ where ρ_P is the relative degree of the plant. Hence, it is generally impossible to convert a given DOB into an equivalent UIDOB structure.
- There is no freedom to choose the order of $Q(s)$ in the UIDOB structure, and the order of the UIDOB is higher than that of the DOB for plants with stable zeros.

We therefore use the DOB in this paper. Refer to Schrijver and van Dijk [9] for more details about the DOB and UIDOB.

2.3 Structure of Proposed Control System

The structure of the overall control system we propose is shown in Figure 5 where P represents the actual plant, c is the set-point, y_P is the plant output, d is the disturbance, n is the noise, and $\hat{d}$ is the estimated disturbance. The control system consists of an inner loop formed by a disturbance observer and an outer loop formed by a PFC controller.

The advantages of using the disturbance observer along with the predictive functional control are twofold.

First, an improvement of prediction accuracy in the PFC algorithm is expected since the input/output behavior of the actual plant approaches its nominal characteristics by using the DOB. This can be understood in the following way. As explained in the previous subsection, the design parameter $Q(s)$ is a low-pass filter, and its DC gain is usually chosen to be 1 (= 0 dB) to reject the $1/s^k$-type disturbances asymptotically. Hence, in the low-frequency band on which disturbances generally concentrate, the transfer function from the control input to the plant output given in (13) can be approximated by

$$G_{yu}(s) = \frac{P(s)P_n(s)}{(1-Q(s))P_n(s) + P(s)Q(s)} \simeq P_n(s) \tag{20}$$

since $Q(s) \simeq 1$. This means that the behavior of the actual plant equipped with the DOB shown in Figure 2 is close to that of the nominal model $P_n(s)$ as long as the bandwidth of $Q(s)$ is properly designed. Although gain mismatches or modeling errors between the internal model and the real plant are unavoidable in real-world applications, the PFC has tolerance for such uncertainties to some extent. However, the transient performance may deteriorate even though the steady-state response is satisfactory. For instance, suppose that the real plant $P(s)$ and the internal model $P_n(s)$ are both stable first-order systems and given by as follows:

$$P(s) = \frac{k}{s+p}, \quad P_n(s) = \frac{k_n}{s+p} \tag{21}$$

where $k > 0$, $k_n > 0$ and $p > 0$. If there exists a gain mismatch between these two transfer functions such that $k < k_n$, then the closed-loop response is slower than the designed response based on the internal model $P_n(s)$. This difference is due to the inaccuracy of the prediction in the PFC algorithm. The use of

[2]The DOB is denoted as the disturbance estimation filter (DEF) in [9].

Table 1: Specifications of linear actuator (left) and DC motor (right).

Description	Value	Description	Value
ball screw lead	10 mm	nominal voltage	24 V
stroke	500 mm	rated current	2.96 A
rail length	670 mm	rated torque	0.28 Nm
maximum rotational speed	500 mm/s	rated speed	1810 r/min

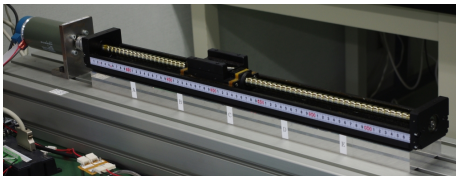


Figure 6: Single axis table drive system.

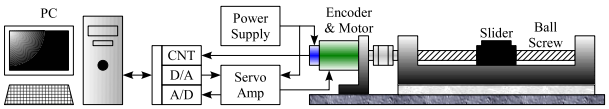


Figure 7: Schematic diagram of experimental setup.

the disturbance observer improves the prediction accuracy and, in consequence, leads to the improvement of the transient response in the PFC scheme.

Second, the disturbance observer can compensate for high-order (*e.g.*, ramp, parabola, etc.) disturbances. It is known that, in the case of non-integrating processes, the steady-state error caused by stepwise disturbances becomes zero in the PFC scheme. Unfortunately, this property does not hold true for high-order disturbances. Also, for integrating processes, the steady-state error remains even though the disturbance is stepwise. To eliminate the steady-state error in such cases, one or more integrators must be added in the loop transfer function. This can be accomplished within the local loop formed by the disturbance observer when $n_I \geqslant 1$ in (15), and the disturbance is asymptotically eliminated in the inner-loop shown in Figure 5. Consequently, the disturbance rejection property of the PFC can be enhanced, and the above-mentioned difficulty is remedied by the simultaneous use of the disturbance observer.

3 Controller Design for Single Axis Table Drive System

3.1 Plant Description

Figure 6 shows the external view of the single axis table drive system which is used in the experiment shown in Section 4. Also, Figure 7 shows the schematic diagram of the experimental setup. As is clear from Figure 7, the system is controlled by the semi-closed-loop control method. The core of this experimental apparatus is a single axis linear actuator (NSK Ltd., Monocarrier MCM08050H10K), which is driven by an 80 Watt DC servo motor (maxon motor ag, F2260.885) equipped with an optical encoder with the resolution of 1000 pulses/revolution. The DC motor is driven by a DC servo amplifier (maxon motor ag, ADS 50/5). The specifications of the linear actuator and DC motor are summarized in Table 1. The encoder pulse is received by a 24-bit encoder counter board (Interface Co., Ltd., PCI-6204) and counted by quad edge evaluation. The command signal to the DC servo amplifier is sent from a 12-bit digital-to-analogue converter, and the current monitor signal from the DC servo amplifier is received by a 12-bit analog-to-digital converter. Those D/A and A/D converters are both implemented on one analogue I/O board (Interface Co., Ltd., PCI-3521). The control PC is equipped with dual 1.7 GHz Intel Xeon® processors and is running on Microsoft Windows 2000® operating system. A real-time control environment is constructed by using MATLAB®, Simulink® and Real-Time Windows Target® (Math-

Works, Inc.), and the table drive system is controlled at the sample rate of 1 kHz. Two S-function blocks for the use of PCI-6204 (encoder counter board) and PCI-3521 (A/D and D/A board) were developed by the authors since Simulink is not shipped with blocks for these boards.

We use the DC servo amplifier in the current control mode to command and control the motor torque instead of directly manipulating the motor current. Hence, on the assumption that the table drive system can be approximated by a one-inertia system, its equation of motion subject to friction torque is given by

$$\ddot{\theta}(t) = \frac{K_S}{J} e_a(t) - \frac{1}{J} T_d(t) \tag{22}$$

where θ denotes the angle of the motor shaft, e_a is the command voltage to the DC servo amplifier, J is the equivalent moment of inertia, K_S is the conversion factor from the command voltage to the motor torque, and T_d is the nonlinear friction torque. Notice that T_d includes the viscous friction torque. In the case of our experimental apparatus, the conversion factor is given by $K_S = 0.0801$ Nm/V, and the nominal value of the moment of inertia is identified as $J = 1.6928 \times 10^{-4}$ kg m^2 by using a standard system identification technique.

We define the state vector and output variables as $(x_1\ x_2)^T := (\theta\ \dot{\theta})^T$ and $y := \theta$, respectively. Also, the time delay of the experimental apparatus can be negligible as long as the sampling period is greater than or equal to 1 ms. Then the state-space description of the single axis table drive system is given by

$$\begin{cases} \begin{pmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} + \begin{pmatrix} 0 \\ K_S/J \end{pmatrix} e_a(t) + \begin{pmatrix} 0 \\ -1/J \end{pmatrix} T_d(t), \\ y(t) = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix}. \end{cases} \tag{23}$$

Here, the nonlinear friction characteristics of this single axis table drive system can be approximated by, for example, the general kinetic (GK) friction model [17] of the form

$$T_d(t) = \begin{cases} \left\{ T_c + (T_b - T_c)\, e^{-\left| \frac{\dot{\theta}(t)}{\dot{\theta}_{str}} \right|^2} \right\} \operatorname{sgn} \dot{\theta}(t) + D\dot{\theta}(t), & \text{if } \dot{\theta} \neq 0, \\ T_e(t), & \text{if } \dot{\theta} = 0 \text{ and } |T_e| < T_s, \\ T_b \operatorname{sgn} T_e(t), & \text{if } \dot{\theta} = 0 \text{ and } |T_e| > T_s \end{cases} \tag{24}$$

where $\dot{\theta}_{str}$ is the Stribeck velocity, T_c is the Coulomb friction level, T_b is the breakaway torque, D is the viscous friction coefficient, T_e is the external torque generated by the driving motor. However, to evaluate the effectiveness of the disturbance observer, and to compare the prediction accuracy in the PFC algorithm with or without the disturbance observer, we treat the friction torque as unknown disturbance in this paper.

3.2 Design of Disturbance Observer

The nominal transfer function from the applied voltage to the angular velocity can be described as

$$P_n(s) = \frac{473.2}{s}. \tag{25}$$

Hence, the relative degree of the Q-filter must be chosen as $\rho_q \geqslant 1$, and we take ρ_q to be 1 in this paper.

The required number of the integral action introduced by the DOB depends on the nature of disturbances, and it is difficult to decide its appropriate number beforehand. So, we will design two DOBs to see the difference due to the number of integral action under the following design conditions:

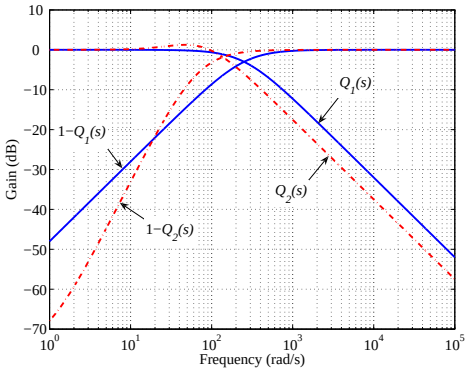


Figure 8: Frequency characteristics of $Q_i(s)$ and $1 - Q_i(s)$ $(i = 1, 2)$.

1. relative degree $\rho_q = 1$, number of integral action $n_I = 1$.
2. relative degree $\rho_q = 1$, number of integral action $n_I = 2$.

Since the transfer function from the input voltage e_a to the rotor angle θ is given by the double integrator model $P_n(s)/s = 473.2/s^2$, the closed-loop system equipped with the DOB can asymptotically reject the stepwise disturbance when $n_I = 2$. The order of the Q-filter n_q can be determined by the relation $n_q = n_I + \rho_q - 1$, so $n_q = 1$ for the former case, and $n_q = 2$ for the latter case.

First, we design a first order filter $Q_1(s)$ as an analogue Butterworth low-pass filter. The target cut-off frequency ω_c is chosen to be 40 Hz. Then, $Q_1(s)$ is given by

$$Q_1(s) = \frac{251.33}{s + 251.33}. \tag{26}$$

Next, we design a second order filter $Q_2(s)$. In this case, we use the prototype of the binomial filter given in (16). For the case of $Q_2(s)$, the binomial coefficients a_1 and a_2 are given as $a_1 = 2$ and $a_2 = 1$, respectively. To determine the parameter τ_{n_q}, we numerically solve the following optimization problem:

$$\min_{\tau_{n_q}} \sum_{i=1}^{N} \left(|Q_d(\omega_i)| - \left| Q_2\left(j\omega_i, \tau_{n_q}\right) \right| \right)^2 \tag{27}$$

where $|Q_d(\omega)|$ is the gain of the ideal low-pass filter whose value is 1 over the frequency range from 0 to ω_c and is 0 at higher frequencies than ω_c, and ω_i $(i = 1, 2, \ldots, N)$ are N logarithmically-spaced points between decades 10^{-1} rad/s and 10^4 rad/s. Solving the optimization problem with $N = 3000$, we obtain the optimal value of the parameter $\tau_{n_q} = 0.015$, which results in the following second order filter:

$$Q_2(s) = \frac{133.38(s + 33.35)}{(s + 66.69)^2}. \tag{28}$$

Figure 8 shows the gains of $Q_i(s)$ and $1 - Q_i(s)$ $(i = 1, 2)$. We can see form Figure 8 that the property of the disturbance suppression of the DOB using $Q_2(s)$ is better than that using $Q_1(s)$.

3.3 PFC Parameters

Using the control structure depicted in Figure 5, we implement the PFC algorithm in the position control loop with the double integral internal model $P_n(s)/s$. Also, the time delay steps d is fixed to 0 in the control law given in (6).

To cope with non-constant references, both the step and ramp basis functions are employed which will be expected to achieve better tracking performance than the step basis function alone. We choose the desired closed-loop time response T_{CLTR} to be 0.02 s to make a compromise between the stability of the closed-loop system and the tracking performance, and the sampling period T_s of the system is fixed to 0.001 s. The same approaching ratio and the coincidence points as Dieulot *et al.* [3] have used, which are already shown in Subsection 2.1, are employed. So, the approaching ratio becomes $\alpha = 0.8607$, and the three coincidence points given in (4) are defined as $(h_1\ h_2\ h_3) = (6\ 10\ 20)$.

In the experiment shown in the next section, we will also use the standard PFC control scheme with the auto-compensation for comparison, and the predicted error is approximated by a first degree polynomial. The number of steps used for the polynomial approximation in (9) is taken to be $h_c = 20$. It can be seen that the tracking performance is improved as h_c becomes small. However, in exchange for the improvement, the control effort becomes large and shows extremely oscillatory behavior. The number of steps 20 is therefore the result of compromise between the tracking performance and control effort.

3.4 Design of Industrial Controller

As well as the PFC plus auto-compensation scheme, we will use an industrial controller for comparison in the experiment. It consists of a proportional-integral controller in the inner speed loop and a proportional controller in the outer position loop. The current loop is neglected since the current is already controlled in the motor driver. The PI controller in the speed loop is defined as

$$C_{PI}(s) = K_{PI}\left(1 + \frac{D}{Js}\right) \tag{29}$$

where K_{PI} is the proportional gain. This PI controller is constructed to cancel the plant dynamics and force the open-loop transfer function to be $K_S K_{PI}/Js$. It follows that the viscous friction is theoretically canceled. Here, the proportional gain K_{PI} is chosen to be 1. As the value of K_{PI} is increased, the speed of response is improved, but the maximum tracking error is not. Also, the stability of the inner loop is sensitive to the change of this gain and easily collapses when $K_{PI} > 1$. Hence, K_{PI} is fixed to 1.

The P controller in the position loop is defined as $C_P(s) = K_P$. On the simulation model, we can increase the proportional gain up to 100. However, due to uncertainties in the actual plant and the nonlinear friction, the responses of the closed-loop system becomes highly oscillatory, and the large control effort is applied to the system when K_P is greater than 20. Hence, K_P is fixed to 20 in the experiment.

4 Experimental Results

To test the proposed control scheme, an experiment was conducted using a sinusoidal reference input. The amplitude and the angular frequency are respectively 5 rad (7.96 mm in displacement) and $\pi/4$ rad/s.

Figure 9 shows the reference and the motor angles obtained by different kinds of control schemes. In this and the subsequent figures, the correspondence between the legends and control schemes are as follows: 'PFC' is the standard PFC, 'PFC+AC' is the PFC with the auto-compensation, 'PFC+DOB1' is the PFC combined with the disturbance observer using $Q_1(s)$, 'PFC+DOB2' is the PFC combined with the disturbance observer using $Q_2(s)$, 'P+PI' is the industrial control. We can see from the top graph that only the standard PFC can hardly actuate the table drive system and cannot track the reference angle. This is mainly because the stiction torque (breakaway torque) is larger than the driving torque generated by the motor and, thus, the table drive system does not move. Every control scheme except the standard PFC tracks the reference, and it is hard to distinguish from each other in the top graph. So, a detailed

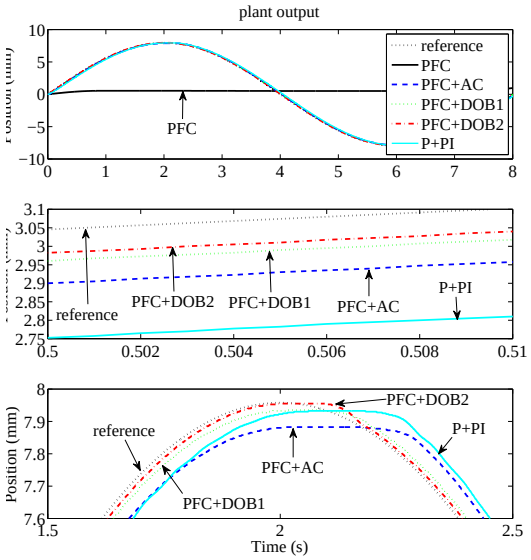


Figure 9: Comparison of control performances (top graph) and its closeups (middle and bottom graphs).

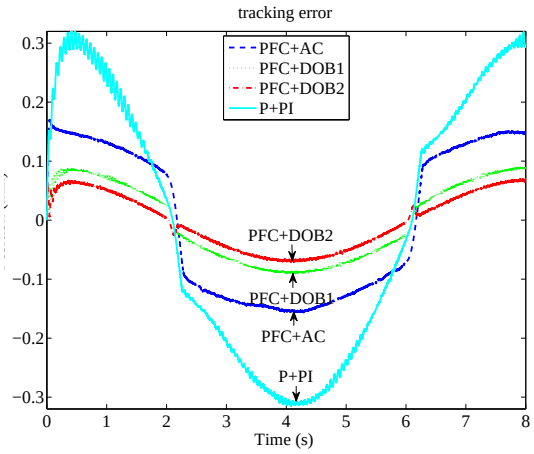


Figure 10: Comparison of tracking errors.

view near 0.5 seconds where the Coulomb friction torque and the viscous friction torque respectively act as the stepwise disturbance and the ramp disturbance is shown in the middle graph of Figure 9. In addition, a detailed view around 2 seconds where a change of direction of the movement occurs and therefore the friction torque takes the maximum is shown in the bottom graph of Figure 9. It can be seen from Figure 9 that the PFCs along with the disturbance observer track the reference better than other control schemes without the disturbance observer.

Figure 10 shows the tracking error of each control scheme where the result of the PFC is excluded from the graph since its tracking error is significant. Comparing the two PFC plus disturbance observer schemes, we see that the tracking performance of PFC+DOB2 is better than that of PFC+DOB1. The difference arises from the number of integrators in the velocity loop, and, as a result, the disturbance observer using $Q_2(s)$, which adds two integrators in the loop, has lower sensitivity and better disturbance rejection property at low frequencies. We also see that PFC+AC is a simple but effective control scheme, considering the fact that it uses no active disturbance estimation or cancellation. However, the peak-to-peak value of its tracking error is about twice as large as that of PFC+DOB1. Also, since the auto-compensation is based on the past prediction errors and extrapolation, a certain amount of degradation of the tracking error is unavoidable when the reference signal changes its direction; see around 2 seconds in Figure 10. Finally, we can see that the industrial controller (P+PI) provides the worst tracking performance.

Figure 11 shows the control effort of each control scheme. Basically, the magnitude is similar to each other. Every control effort presents oscillatory behavior during the phase of initiation of motion (see from 0 to 2 seconds), though that of the industrial controller especially draws our attention. It can be seen from the bottom graph that, with the disturbance observer, the control effort quickly responds to the change of direction of the reference to cancel the disturbance torque. This leads to the improvement of the tracking error on the time interval between 2.1 s and 2.2 s.

Figure 12 shows the actual plant output and the predicted output of the inner model. Here, to compare the prediction accuracy inside the pure PFC algorithm, the PFC scheme using the auto-compensation (PFC+AC) is excluded from the comparison. The top graph is in the case of the PFC without the distur-

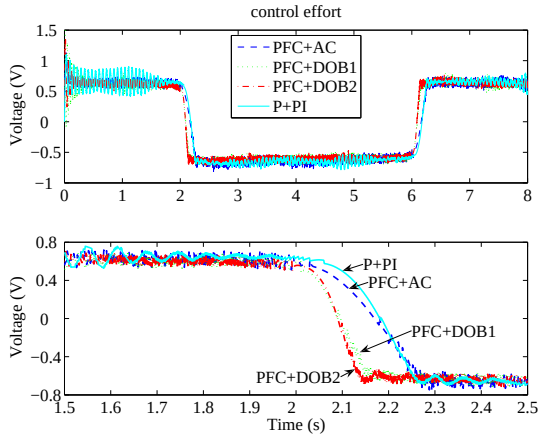


Figure 11: Comparison of control inputs (top graph) and its closeup (bottom graph).

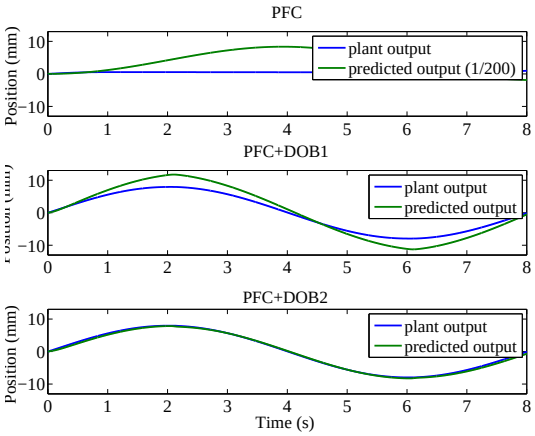


Figure 12: Comparison of prediction accuracy.

Table 2: Root Integrated Squared Values.

Control Scheme	Tracking Error	Control Effort
PFC	11.45	1.78
PFC+DOB1	0.14	2.12
PFC+DOB2	0.10	2.13
P+PI	0.48	2.23
PFC+AC	0.32	2.19

bance observer. Since the amplitude of the predicted output is too large in this case, two-hundredth of its value is drawn in the figure. As we have already seen in Figure 9, the ordinary PFC cannot overcome the friction torque, and the table drive system does not start to move. Hence, the predicted output is completely different from the plant output. On the other hand, we can see from the middle and bottom figures that the predicted output approximates the actual plant output. Also, the disturbance observer using $Q_2(s)$ provides a more accurate result than the one using $Q_1(s)$. In the middle figure, the magnitude of the predicted output is larger than the plant output, which, roughly speaking, indicates that the gain of the nominal model is larger than the actual plant. It follows that the speed of response of the plant becomes slower than the designed response. This can be confirmed by comparing the response of the PFC using $Q_1(s)$ with that of the PFC using $Q_2(s)$ in Figure 9. Even though the specified desired closed-loop time response T_{CLTR} is the same in both cases, the former slightly lags behind the latter, which is very close to the designed response owing to the accurate prediction.

To see the difference of the control performances quantitatively, we show the root integrated squared values of the tracking error e and that of the control effort u computed on the time interval between 0 s and 8 s in Table 2. Clearly, the PFC plus the disturbance observer using $Q_2(s)$ provides the best performance of all the control schemes. On the other hand, in terms of the control effort, we can hardly recognize a difference apart from the standard PFC. As a consequence, we can see that, in this experiment, the PFC combined with the disturbance observer successfully improved the tracking performance without putting extra control energy into the system.

5 Conclusions

We have presented a disturbance observer-based approach for improving the tracking performance of the predictive functional control. The role of the disturbance observer is to reject unknown disturbances including the plant-model mismatch or nonlinearities that can be regarded as an input disturbance and to nominalize the real plant. Owing to the disturbance observer, the predicted output error in the PFC algorithm is reduced, and, as a result of this, the tracking performance is improved. The proposed control scheme was implemented and validated on a single axis table drive system. It was confirmed that the predicted output error was effectively reduced and that better tracking performance was provided than the standard PFC using the auto-compensation mechanism and the conventional P plus PI control scheme even in the severe environment where the system was not driven by the standard the PFC scheme due to friction. Future works involves the extension to non-minimum phase linear systems.

Acknowledgment

This work was supported by the Japan Society for the Promotion of Science under Grant-in-Aid for Scientific Research (C) 21560247.

Bibliography

[1] S. Abu el Ata-Doss, P. Fiani and J. Richalet, "Handling input and state constraints in predictive functional control," *Proc. of the 30th Conference on Decision and Control*, pp. 985-990, Brighton, England, 1991.

[2] N. Bigdeli and M. Haeri, Predictive functional control for active queue management in congested TCP/IP networks, *ISA Transactions*, vol. 48, no. 1, pp. 107-121, 2009.

[3] J.Y. Dieulot, T. Benhammi, F. Colas, P.J. Barre, Composite predictive functional control strategies, application to positioning axes, *International Journal of Computers, Communications & Control*, vol. III, no. 1, pp. 41-50, 2008.

[4] D. Dovžan and I. Škjrank, Predictive functional control based on an adaptive fuzzy model of a hybrid semi-batch reactor, *Control Engineering Practice*, vol. 18, no. 8, pp.979-989, 2010.

[5] M. Hadjiski and V. Asenov, Predictive functional control using a blending approach, *Cybernetics and Information Technologies (Bulgarian Academy of Science)*, vol. 5, no. 2, pp. 32-41, 2005.

[6] C. Johnson, "Accommodation of external disturbances in linear regulator and servomechanism problems," *IEEE Transactions on Automatic Control*, vol. 16, no. 6, pp. 635-644, 1971.

[7] M. Lepetič, I. Škrjanc, H. G. Chiacchiarini and D. Matko, Predictive functional control based on fuzzy model: comparison with linear predictive functional control and PID control, *Journal of Intelligent and Robotic Systems*, vol. 36, pp. 467-480, 2003.

[8] M. Primucci and M. Basualdo, Thermodynamic predictive functional control applied to CSTR with jacket system, *Proc. of IFAC 15th Triennial World Congress*, Barcelona, Spain, 2002.

[9] E. Schrijver and J. van Dijk, "Disturbance observers for rigid mechanical systems: equivalence, stability, and desigh," *ASME Journal of Dynamic Systems, Measurement, and Control*, vol. 124, pp. 539-548, 2002.

[10] I. Škrjanc and D. Matko, Predictive functional control based on fuzzy model for heat-exchanger pilot plant, *IEEE Transactions on Fuzzy Systems,* vol. 8, no. 6, pp. 705-712, 2000.

[11] J. Richalet, S. Abu el Ata-Doss, C. Arber, H.B. Kuntze, A. Jacubasch and W. Schill, Predictive functional control: application to fast and accurate robot, *Proc. of IFAC 10th World Congress*, Munich, Germany, 1987.

[12] J. Richalet, Industrial applications of model based predictive control, *Automatica*, vol. 29, no. 5, pp. 1251-1274, 1993.

[13] J. Richalet and D. O'donovan, *Predictive Functional Control: Principles and Industrial Applications.* London, England: Springer-Verlag, 2009.

[14] J. M. Maciejowski, *Predictive Control with Constraints.* Harlow, England: Pearson Education, 2002.

[15] K. Ohishi, M. Nakao, K. Ohnishi and K. Miyachi, Microprocessor-controlled DC motor for load-insensitive position servosystem, *IEEE Transactions on Industrial Electronics*, vol. 34, no. 1, pp.44-49, 1987.

[16] T. Murakami and K. Ohnishi, Advanced motion control in mechatronics – A tutorial, *Proc. of the IEEE International Workshop on Intelligent Control*, Istanbul, Turkey, vol. 1, pp. SL9-SL17, 1990.

[17] E. G. Papadopoulos and G. C. Chasparis, Analysis and model-based control of servomechanisms with friction, *ASME Journal of Dynamic Systems, Measurement, and Control*, pp. 911-915, 2004.

[18] T. Satoh, N. Saito and N. Saga, Predictive functional control with disturbance observer for pneumatic artificial muscle actuator, *Proc. of the 1st International Conference on Applied Bionics and Biomechanics*, Venice, Italy, no page number, 2010.

[19] T. Umeno and Y. Hori, Robust speed control of DC servomotors using modern two-degrees-of-freedom controller design, *IEEE Trans. on Industrial Electronics*, vol. 38, no. 5, pp. 363-368, 1991.

[20] A. Vivas and P. Poignet, Model based predictive control of a fully parallel robot, *Control Engineering Practice*, vol. 13, no. 7, pp. 863-874, 2005.

INT J COMPUT COMMUN, ISSN 1841-9836
Vol.7 (2012), No. 3 (September), pp. 565-573

Packet-Layer Quality Assessment for Networked Video

H. Su, F. Yang, J. Song

Honglei Su, Fuzheng Yang, Jiarun Song
State Key Laboratory of Integrated Service Networks
Xidian University, Xi'an, Shaanxi 710071, China
E-mail:{hlsu,fzhyang}@mail.xidian.edu.cn,
sjrxidian@hotmail.com

Abstract:
To realize real-time and non-intrusive quality monitoring for networked video, a content-adaptive packet-layer model for quality assessment is proposed. Considering the fact that the coding distortion of a video is dependent not only on the bit-rate but also on the motion characteristic of the video content, temporal complexity is evaluated and incorporated in quality assessment in the proposed model. Since very limited information is available for a packet-layer model, an adaptive method for frame type detection is first applied. Then the temporal complexity which reflects the motion characteristic of the video content is estimated using the ratio of the bit-rate for coding I frames and P frames. The estimated temporal complexity is incorporated in the proposed model, making it adaptive to different video content. Experimental results show that the proposed model achieves an advanced performance in comparison with the ITU-T G.1070 model.
Keywords: Packet-Layer Model, Networked Video, Video Quality Assessment, Coding Distortion, Temporal Complexity.

1 Introduction

Recently, with the development of advantage multimedia processing technologies [1], multimedia services such as videophone, mobile conference and Internet Protocol Television (IPTV) have gained significant popularity in our daily life. However, the quality of these applications cannot be guaranteed in an IP network due to its best-effort delivery. It is therefore crucial to establish an objective model for video quality assessment targeting system design, QoS (Quality of Service) planning and quality monitoring [2], [3].

Objective video quality assessment can be categorized into media-layer models, bitstream-layer models, packet-layer models, parametric models and hybrid models from the viewpoint of the input information. To estimate the perceptual quality of service (QoS) for users, the media-layer models use media signals [4], where characteristics of the video content and decoder strategies such as error concealment are usually taken into account. The bitstream-layer models, on the other hand, perform an analysis on the bitstream without resorting to a complete decoding [5], which can be used in situations where one does not have access to decoded video sequences. The packet-layer models exploit the packet headers to obtain information about the service quality [6], making them well suited for in-service non-intrusive monitoring. The parametric models employ parameters from the network or the application [7], [8]. Parameters from the network may include the packet loss rate and the delay information, while those from the application usually cover the coding bit rate, frame rate, and so on. The hybrid models use a combination of information from the bitstream and the media data, and therefore have an advanced performance as well as combined features of the other models [9].

Since the packet-layer model only utilizes information from packet headers, it is very efficient in quality monitoring due to its low complexity, especially suitable for quality monitoring at network internodes. The other advantage is that the packet-layer model does not need decryption and decoding,

making it favorable when packet payloads are encrypted. In this paper, a packet-layer model is proposed for efficient quality assessment for networked video. Utilizing the limited information which can be provided by packet headers, the frame type and temporal complexity are estimated based on the coding bit-rate. The temporal complexity is incorporated in the proposed model to make it content-adaptive.

The remainder of this paper is organized as follows. The framework for proposed packet-layer video quality assessment is introduced in Section 2. Section 3 discusses the relationship between the coding distortion and the bit-rate. The proposed packet-layer model for video quality assessment is described in Section 4. The experimental results are presented in Section 5. This paper closes with conclusions given in Section 6.

2 Packet-Layer Model for Video Quality Assessment

Packet-layer model for video quality assessment is especially suitable for application scenarios like in-service video quality monitoring and network service planning. It predicts the networked video quality from packet-header information, without resorting to any media-related payload information. Since only the packet header is exploited, the packet-layer model is very useful at network inter-nodes due to its low complexity, where it can monitor thousands of video streams at the same time. The other advantage is that the packet-layer model does not need decryption and video decoding, making it favorable when packet payloads are encrypted.

As an example, Figure 1 shows the structure of a packet in RTP/UDP/IP protocol stacks. In this case, the IP (Internet Protocol) header, the UDP (User Datagram Protocol) header, and the RTP (Real-time Transport Protocol) header can be accessed by a packet-layer model. The length of payload is easily obtained since the UDP length field indicates the length of the UDP header [10], the RTP header and the payload [11]. The marker bit in the RTP header indicates the end of a video frame, and all packets related to one video frame are with the same RTP timestamp. Using this information, the packets can be assembled to frames.

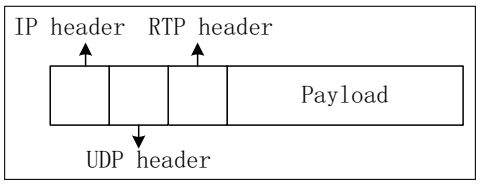


Figure 1: The structure of a packet under RTP/UDP/IP protocol stacks

By analyzing packet headers, the information needed by the parametric model can be obtained and used to estimate the video quality [12]. Apart from the parameters of bit-rate, frame rate, and packet loss rate which are usually employed in parametric models [13], [14], other information can also be employed in a packet-layer model, such as coding parameters (e.g., the frame type, and the bit-rate of each frame), information about the video content characteristics (e.g., the ratio of the bit-rate for coding I frames and P frames), and the detailed positions of lost packets in a video.

The framework of the proposed packet-layer model is shown below in Figure 2. Firstly, after packet header analysis, the bit-rate for coding each frame can be obtained. Then, it is employed to detect the frame type and calculate the ratio of the bit-rate for coding I frames and P frames. This ratio is employed in the proposed model to estimate the temporal complexity which reflects the motion characteristic of the video content. Finally, the coding distortion of networked video is evaluated using the bit-rate information and the estimated temporal complexity.

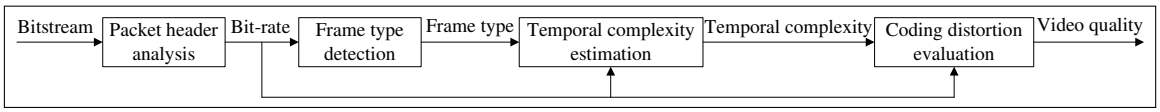


Figure 2: Framework of the proposed packet-layer model

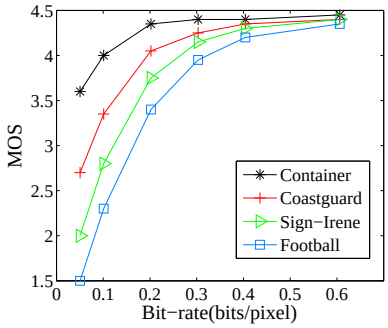


Figure 3: Relationship between the MOS and the bit-rate for each sequence

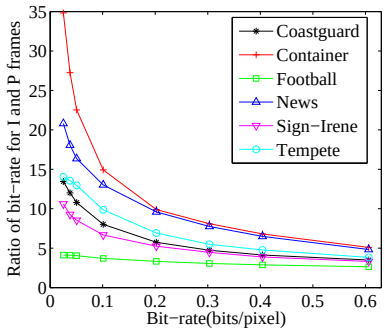


Figure 4: The ratio of the bit-rate for coding I frames and P frames

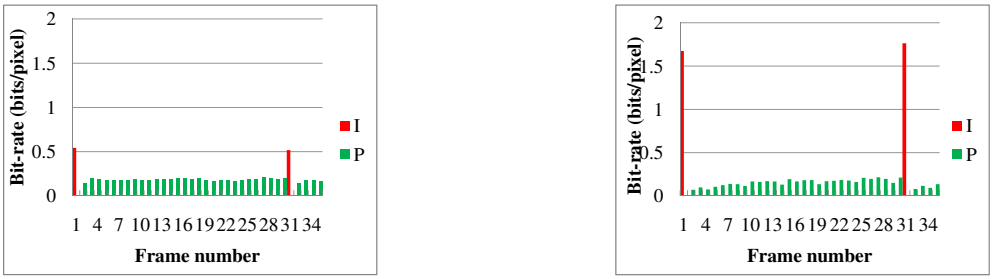
3 Coding Distortion and Bit-Rate

The bit-rate is a key parameter for estimating the coding distortion. It has been well recognized that there is a relationship between the bit-rate and the average Mean Opinion Score (MOS) for different video sequences. Therefore, several functions have been proposed to approximate the relationship to predict the average coding distortion using the bit-rate, such as the computational model proposed in the ITU-T Recommendation G.1070 [15] and its enhancement [16]. Other model forms can also be found, such as the "m-n model" [17], as well as the exponential model [12], [18]. A detailed performance comparison of those models is provided in [19], where superior performance of the enhanced G.1070 model is observed.

Although the average MOS can be predicted using models, the subjective quality of individual videos cannot be well formulated when provided only with the coding bit-rate. Considering the video quality for each sequence, the relationship between the bit-rate and the MOS is shown in Figure 3. It is observed that there are obvious differences in the video quality at a same bit-rate for different sequences. Therefore using the bit-rate only is not suitable for estimating the quality of a certain video service.

It has now been widely acknowledged that content features must be taken into account for an accurate prediction of the perceived video quality [20]. Building on this argument, video clips are classified into three classes according to the subjective movement content (High, Medium and Low movement content), and the model parameters are calculated for each class [18], [19]. However, it is not described how to obtain the information about movement content for each video clip based on objective parameters [18]. Although the average SAD (sum of absolute differences) can be employed in [19] to reflect the motion characteristic of video content, this value is not available for a packet-layer model.

According to Figure 3, it can be seen that the video clip which has a higher motion complexity such as "Football" has a comparatively lower quality at the same bit-rate. Correspondingly, "Container" having a lower motion has a higher quality over the others under the same coding bit-rate. Therefore, the temporal complexity estimated from the packet headers is expected to reflect the motion extent of video clips. How to measure this variable and then establish a packet-layer model based on video content is proposed in the next section.



(a) Football (b) News

Figure 5: Bit-rate histogram of different frame type at 0.202(bits/pixel)

4 Packet-Layer Model Based on Video Content

Figure 4 shows the values of the ratio of the bit-rate for coding I frames and P frames for different video clips. This ratio is always comparatively lower for the "Football" sequence due to its high temporal complexity which results in more bit-rates in coding the P frames. On the other hand, the values of this ratio for the "Container" sequence are consistently larger because of its lower temporal complexity. As the observation, low values of this ratio may correspond to a high temporal complexity. Therefore, the temporal complexity can be roughly estimated using the ratio of the bit-rate for coding I frames and P frames. For a packet-layer model, however, the frame type information is not readily available. Consequently, a frame type detection method based on the bit-rate distribution for each frame is introduced as follows.

4.1 Frame Type Detection

As a general principle, video coding exploits spatial redundancy using intra coding and temporal redundancy using inter coding, where the inter coding modes are usually more efficient in removing redundancy. Accordingly, the bit-rate for coding an I frame is usually much higher than that for a P frame, as shown in Figure 5.

Consequently, a threshold-based method is proposed to detect the frame type using the information about the coding bit-rate. However, Figure 5 also shows that the bit-rate related to a certain frame type varies with the video content. Because of the high motion complexity of "Football", more bit-rates are distributed to P frames. As the result, at a same bit-rate, the values of bit-rate coding I frames of "Football" are lower than the values of "News" which has a lower motion complexity under the same coding bit-rate. Therefore, to make the detection more effective, the threshold for video clips of higher temporal complexity should be lower than the threshold for clips of lower temporal complexity. So fixed thresholds may fail in detecting for different video clips. Dynamic thresholds which are adaptively adjusted [21] are applied in this paper.

4.2 Temporal Complexity Estimation

After frame type detection, the ratio of the bit-rate for coding I frames and P frames can be calculated using the information about the frame type and bit-rate of each frame. This ratio is defined as:

$$r = \frac{R_I}{R_P}, \tag{1}$$

where R_I is the average bit-rate for coding I frames in a certain duration and R_P is the average bit-rate for coding P frames in the same duration. However, it can be seen from Figure 4 that there is obvious

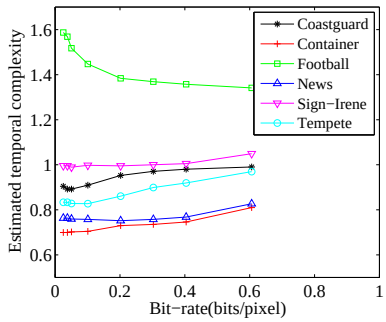


Figure 6: Relationship between estimated temporal complexity and bit-rate

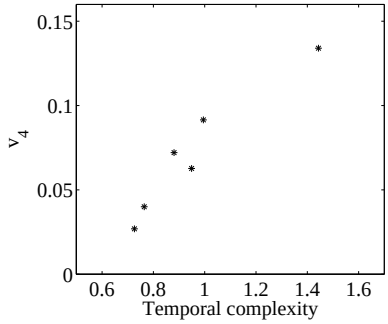


Figure 7: Relationship between v_4 and temporal complexity

differences in the values of r at different bit-rates for each sequence. Due to the fact that the temporal complexity should be evaluated for the sequence with a given related value, r can not be directly applied as the measure of temporal complexity.

Consequently, a mathematical mapping is used to unify the values of r for all the range of bit-rates for each sequence. Firstly, the natural logarithm function acts on the values of r to eliminate the enormous differences due to their distribution of different orders of magnitude. Then, adaptively adjusted factors which are related to the average bit-rate are introduced to make the values generated by the first step for each sequence as near as possible. Both of these two steps are implemented without influencing the relatively ranking of the curves of different sequences. However, to give the temporal complexity a practical significance (a high value corresponds to a higher motion extent sequence), the inverse function is employed at last. So the temporal complexity is formulated as:

$$\sigma_T = \frac{a_1 \cdot \ln(R) + b_1}{\ln(R_I) - \ln(R_P)}, \tag{2}$$

where R is the average bit-rate, and a_1 and b_1 are constants obtained by the experiments.

As shown in Figure 6, the values of estimated temporal complexity calculated by Formula 2 for each sequence are roughly consistent for all bit-rate. And different sequences have different average values which can reflect the motion characteristic of the video content. So the estimated temporal complexity can be employed for the packet-layer model based on video content.

4.3 Proposed Model for Quality Assessment

It is obvious that the average MOS for different video sequences increases as the bit-rate increases and saturates at the maximum MOS, which has been formulated in ITU-T Recommendation G.1070 as:

$$V_q = 1 + v_3 \cdot (1 - \frac{1}{1 + (\frac{R}{v_4})^{v_5}}), \tag{3}$$

where V_q is the video quality, R is the bit-rate, and v_3, v_4, v_5 are empirical parameters. This model can estimate the average video quality for different contents at each bit-rate.

However, video quality strongly depends on the video content. Best values for v_3, v_4 and v_5 are calculated for the video sequences are presented in Table 1. It can be found from Figure 6 and Table 1 that a higher value of v_4 corresponds to a video sequence with higher temporal complexity (e.g., the "Football" sequence). On the contrary, a video sequence whose temporal complexity is low usually has a low value of v_4 (e.g., the "Container" sequence). Figure 7 shows the relationship between v_4 and the temporal complexity, and a linear model approximates this relationship well as follows,

$$v_4 = a_2 \cdot \sigma_T + b_2, \tag{4}$$

where a_2, b_2 are obtained by the experiments, and v_4 is not a constant but a variable varied with σ_T.

Table 1: The values of v_3, v_4 and v_5 for each video sequence

Video sequence	v_3	v_4	v_5
Container	3.546	0.027	1.237
News	3.371	0.040	2.232
Coastguard	3.464	0.063	1.733
Tempete	3.456	0.072	1.687
Sign-Irene	3.546	0.091	1.884
Football	3.481	0.134	2.230

In addition, Table 1 shows that there is relative small difference between the values of v_3 for different sequences, which is the maximum MOS of the sequence. Though there is a relative large difference between the values of v_5 for different sequences, the difference influences the value of V_q slightly. Therefore, v_3 and v_5 are set as constants for all video clips in the proposed model and both of them are obtained by the experiments.

Consequently, submitting Formula 4 into Formula 3, the proposed model is established as:

$$V_q = 1 + v_3 \cdot (1 - \frac{1}{1 + (\frac{R}{a_2 \cdot \sigma_T + b_2})^{v_5}}). \tag{5}$$

Apart from the bit-rate, the temporal complexity, which reflects the motion characteristic of the video content, is considered in this model to make the evaluation more accurate.

5 Experimental Results

The video sequences chosen for experiments covered a wide range of scenes from high motion to low motion events. Specifically, standard video sequences of "Carphone", "Coastguard", "Container", "Football", "Foreman", "Hall_Monitor", "Mother&Daughter", "News", "Paris", "Sign_Irene", "Silent", "Soccer" and "Tempete" were used for performance evaluation. The sequences were all in the Common Intermediate Format (CIF) at 25 frames per second (fps), and encoded using x264 coder [22] with a GOP (Group of Picture) structure of "IPPP" sized of 30. For each sequence, the first 8 seconds were used for evaluation.

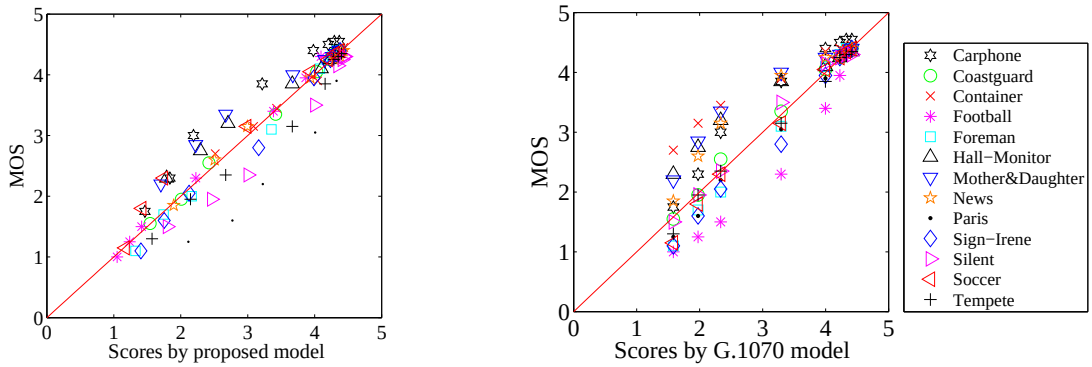
The subjective scores were collected for comparison purposes. The guidelines specified by the Video Quality Experts Group (VQEG) in [23] were followed for the subjective tests. Twenty-five non-expert viewers were involved in these tests, using the Absolute Category Rating (ACR) with a 5-point scale to obtain the MOSs of reconstructed sequences [24], [25].

The parameters were obtained according to the experiments, and their values are shown in Table 2. These parameters were set fixed for all carried experiments. However, if applied to videos generated by the other codecs, they may need to be adjusted.

Table 2: Parameter values

v_3	v_5	a_1	b_1	a_2	b_2
3.477	1.834	-0.334	1.137	0.142	-0.065

Pearson correlation coefficient (PCC) and the root-mean-squared error (RMSE) were used to evaluate the performance of the proposed model. By comparison with the G.1070 model, the proposed model gets an increment about 0.024 in PCC and a decrement about 0.082 in RMSE, as shown in Table 3. The



(a) Proposed model (b) G.1070 model

Figure 8: Scatter plot of MOSs vs objective scores

scatter plots of the objective scores versus the subjective scores are shown in Figure 8, from which the same conclusion can be drawn that using the proposed model the perceived coding distortion can be more accurately measured.

Table 3: Performance comparison of proposed model and G.1070 model

Video quality assessment model	PCC	RMSE
Proposed model	0.9582	0.3250
G.1070 model	0.9338	0.4067

6 Conclusions

A packet-layer model based on characteristics of the video content is proposed in this paper to measure the perceived coding distortion for networked video. Without resorting to the payload information, the temporal complexity is estimated using the ratio of the bit-rate for coding I frames and P frames to reflect the motion characteristic of the video content. Based on analysis of the parameters in the original G.1070 model, the measure of temporal complexity is integrated in the proposed model. Extensive experimental results have demonstrated that the proposed model shows an advanced performance in comparison with the G.1070 model. Further work may include the application of the proposed model to practice by considering both coding distortion and packet loss.

Acknowledgement

This work was supported by the National Science Foundation of China (60902081, 60902052), the Fundamental Research Funds for the Central Universities (72004885), the International Science and Technology Cooperation Program of China (2010DFB10570), and the 111 Project (B08038).

Bibliography

[1] C. Grava, A. Gacs¨˜di, I. Buciu, "A homogeneous algorithm for motion estimation and compensation by using cellular neural networks", *International Journal of Computers Communications & Control*, ISSN 1841-9836, Vol. 5, No. 5, pp.719-726, 2010.

[2] H. R. Wu, K. R. Rao, Eds., Digital video image quality and perceptual coding, *CRC Press*, 2005.

[3] A. Marchand, M. Chetto, "Quality of service scheduling in real-time systems", *International Journal of Computers Communications & Control*, ISSN 1841-9836, Vol. 3, No. 4, pp. 353-365, 2008.

[4] ITU-T Recommendation J.148, "Requirements for an objective perceptual multimedia quality model", 2003.

[5] O. Verscheurei, X. Garcia, "User-oriented QoS in packet video delivery", *IEEE Network*, pp. 12-21, Nov. 1998.

[6] A. Clark, "Modeling the effects of burst packet loss and recency on subjective voice quality", *IP Telephony Workshop*, 2001.

[7] K. Yamagishi, T. Hayashi, "Analysis of psychological factors for quality assessment of interactive multimodal service", *Electronic Imaging 2005*, pp. 130-138, Jan. 2005.

[8] K. Yamagishi, T. Hayashi, "Opinion model using psychological factors for interactive multimodal services", *IEICE Trans. Commun.*, Vol. E89-B, No. 2, pp. 281-288, Feb. 2006.

[9] A. Takahashi, A. Kurashima, H. Yoshino, "Objective assessment methodology for estimating conversational quality in VoIP", *IEEE Trans.on SALP*, Nov. 2006.

[10] RFC 768, UDP, User datagram protocol, 2003.

[11] RFC 3550, RTP, A transport protocol for real-time applications, 2003.

[12] A. Raake, M. Garcia, J. Berger, F. Kling, P. List, J. Johann, C. Heidemann, "T-V-Model: parameter-based prediction of IPTV quality", *Proc. International Conference on Acoustics, Speech, and Signal Processing*, pp. 1149-1152, Mar. 2008.

[13] M. N. Garcia, A. Raake, "Parametric packet-layer video quality model for IPTV", *Proc. Information Sciences Signal Processing and their Applications*, Kuala Lumpur, Malaysia, May 2010.

[14] K. Yamagishi, T. Hayashi, "Parametric packet-layer model for monitoring video quality of IPTV services", *Proc. International Communications Conference*, Beijing, China, May 2008.

[15] ITU-T Recommendation G.1070, "Opinion model for video-telephony applications", Apr. 2007.

[16] J. Joskowicz, J. C. Lopez-Ardao, "Enhancements to the opinion model for video-telephony applications", *Proc. the International Latin American Networking Conference*, Pelotas, Brazil, Sep. 2009.

[17] J. Joskowicz, J. C. Lopez-Ardao, M. A. G. Ortega, C. L. Garcia, "A mathematical model for evaluating the perceptual quality of video", *Proc. International. Workshop on Future Multimedia Networking*, Coimbra, Portugal, June 2009.

[18] H. Koumaras, A. Kourtis, D. Martakos, J. Lauterjung, "Quantified PQoS assessment based on fast estimation of the spatial and temporal activity level", *Multimedia Tools and Applications*, Vol. 34, No. 3, Sep 2007.

[19] J. Joskowicz, J. C. Lopez-Ardao, "A general parametric model for perceptual video quality estimation", *Proc. Communications Quality and Reliability*, Vancouver, BC, June 2010.

[20] M. N. Garcia, A. Raake, P. List, "Towards content-related features for parametric video quality prediction of IPTV services", *Proc. Acoustics, Speech and Signal Processing*, Las Vegas, USA, pp. 757-760, April 2008.

[21] N. Liao, Z. Chen, "A packet-layer video quality assessment model based on spatiotemporal complexity estimation", *Proc. Visual Communications and Image Processing*, Huangshan, China, July 2010.

[22] VideoLAN, X264 CODEC, http://www.videolan.org/x264.html.

[23] VQEG, "Hybrid perceptual/bitstream group TEST PLAN 1.1", http://www.its.bldrdoc.gov/vqeg/, Sep. 2007.

[24] ITU-T, Recommendation P. 910, "Subjective video quality assessment methods for multimedia applications", April 2008.

[25] ITU-R Recommendation BT.500-11, "Methodology for the subjective assessment of the quality of television pictures", 2002.

INT J COMPUT COMMUN, ISSN 1841-9836
Vol.7 (2012), No. 3 (September), pp. 574-585

Bionic Wavelet Based Denoising Using Source Separation

M. Talbi, A.B. Aicha, L. Salhi, A. Cherif

Mourad Talbi, Lotfi Salhi, Adnene Cherif
Faculty of Sciences of Tunis,
Laboratory of Signal Processing,
University Campus, 2092 El Manar II, Tunis, Tunisia
E-mail: mouradtalbi196@yahoo.fr, lotfi.salhi@laposte.net,
adnane.cher@fst.rnu.tn

Anis Ben Aicha
Université de Carthage, Ecole Supérieure des Communications
Laboratoire de recherche COSIM
Route de Raoued 3.5 Km, Cité El Ghazala, Ariana, 2083,
Tunisie, Tél. : +216 71 857 000 - Fax : +216 71 856 829
E-mail: ben.aicha.anis@gmail.com

Abstract:
We consider the problem of speech denoising using source separation. In this study we have proposed a hybrid technique that consists in applying in the first step, the Bionic Wavelet Transform (BWT) to two different mixtures of the same speech signal with noise. This speech signal is corrupted by a Gaussian white noise with two different values of the Signal to Noise Ratio (SNR) in order to obtain those two mixtures. The second step consists in computing the entropy of each bionic wavelet coefficient and finds the two subbands having the minimal entropy. Those two subbands are used to estimate the separation matrix of the speech signal from noise by using the source separation. Our proposed technique is evaluated by comparing it to the denoising technique based on source separation in time domain.
Keywords: Bionic wavelet transform, Blinde Source Separation, entropy, speech enhancement.

1 Introduction

In signal processing, the source separation constitutes an attractive problem. Its goal is to extract from many signals mixture, the meaningful signals. This is performed with minimum a priori information on the mixture process. In the case of instantaneous mixture, many approaches employing the ICA algorithm can solve the problem of the source separation. One of those approaches permits to estimate the unmixing matrix by minimizing the mutual information between the separated sources [1, 2]. Others exploit the non-Gaussianity of the source signals and perform separation by maximizing this non-Gaussianity [2]. For example, a technique using a subband decomposing in combination with ICA, has been developed by Tanaka et al [3]. Kisilev et al [4] have employed geometric algorithms for separating mixed signals. Rachid Moussaoui et al [5] have proposed an algorithm using the idea of applying a preprocessing in the transformed domain but the separation is performed in the time domain. In this paper we have used the source separation with Bionic Wavelet Transform (BWT) for enhancement of speech signal corrupted by white noise. The source separation is performed with ICA and instead of using the wavelet packet transform as used in the technique proposed by Rachid et al [5], we have used in this work the BWT.

2 Restrictions of ICA

The ICA standard formulation needs at least as many sensors as sources. Therefore, we suppose in this paper that the source number is equals to the sensor number. In the instantaneous mixture case, the sources are not directly observed but as a linear combination such as:

$$x_i(t) = \sum_{j=1}^{j=N} a_{ij} S_j(t), \tag{1}$$

where x are the observed signals, s are the source signals and $A = [a_{ij}]$ is unknown full rank mixing matrix.

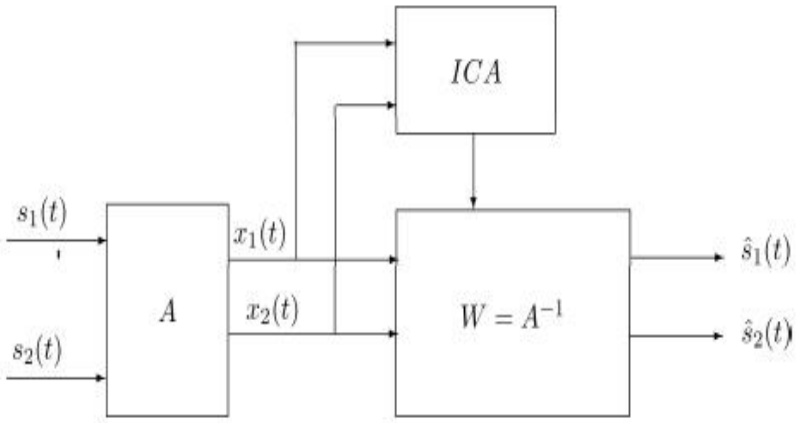


Figure 1: ICA Principle.

Figure 1 shows the ICA principle. The ICA aim is practically to find the inverse matrix of A, which is the unmixing matrix $W = A^{-1}$. To make an estimation of W, certain assumptions have to be made and some restrictions have to be imposed [2]: we assume that the individual components $s_i(t)$ are statistically independent over the observation time and the individual components must have non Gaussian distributions. In comparison to previous work, the novelty of the approach of Rachid Moussaoui et al [5] resides in the preprocessing implementation before the source separation process in to:

- Relax the previous restrictions by increasing the non-Gaussanity which is a pre-requirement for ICA.

- Initiate a preliminary separation by decreasing the mutual information between the resultant signals from the preprocessing.

The preprocessing transforms the observed signals to find an adequate representation where the signals distributions are non-Gaussian. For this reason, the wavelet transform is used in order to emphasize the non Gaussian nature of the observed signals. Once we have found the inverse matrix W with the wavelet packets based ICA then, the separation is performed in the time domain [5]. Figure 2 illustrated an overview of the system proposed by Rachid Moussaoui et al [5].

In this paper we have chosen $s_2(t)$ to be a white noise that corrupted the clean speech signal $s_1(t)$ with two different values of the Signal to Noise Ratio (SNR).

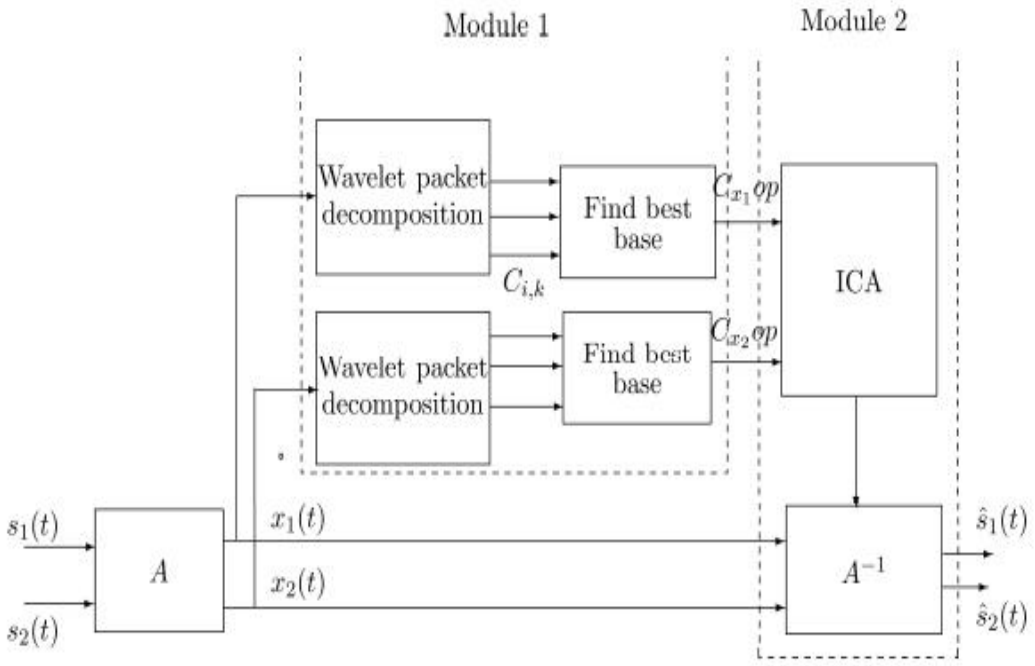


Figure 2: Overview of the source separation system proposed by Rachid Moussaoui et al [5].

3 The proposed technique

The proposed speech enhancement system, illustrated in Figure 4, is inspired from that of Rachid Moussaoui et al [5]. The latter is conceived for multi-channel source separation and is based on wavelet based independent component analysis. It comprises two modules shown in dotted boxes in Figure 4. The first module (pre-processing) extracts appropriate signals from the observed signals in order to facilitate the separation of the speech and noise signals. For this, the observed signals are projected on suitable bases, more specifically on bionic wavelet bases. The second module (speech and noise separation) performs the source separation using standard ICA [1]. The input of this module is the extracted signals from module 1 and the observed signals. Its output is the cleaned or the enhanced speech signal $\hat{s}_1(t)$. Figure 4 illustrated an overview of the proposed speech enhancement system which is summarized by the following steps:

i Decompose the observed signals (the two noisy speech signals) into bionic wavelet subbands by applying the BWT.

ii Compute the entropy value of each subband and select the two subbands having the minimum entropy.

iii Use those two subbands as the inputs of the ICA system in order to estimate the separation matrix, A^{-1}.

iv Estimate the enhanced speech signal $\hat{s}_1(t)$ by applying A^{-1} to the temporal mixtures $x_1(t)$ and $x_2(t)$.

The used entropy in this work, is the Shannon entropy which is defined for each subband $w_{.,j}$, $1 \leq j \leq 30$ as:

$$H(j) = -\sum p_i \log(p_i). \tag{2}$$

Note that in the expression $w_{\cdot,j}$, $1 \leq j \leq 30$, $\cdot$ is replaced by 1 if we apply the BWT to $x_1(t)$ and is replaced by 2 if we apply the BWT to $x_2(t)$.

The probability p_i is expressed as: $p_i = \frac{w_{\cdot,j}(i)^2}{\|W\|^2}$, $1 \leq i \leq N$ and N is the number of samples in the subband $w_{\cdot,j}$ and W is obtained by concatenating all the subbands $w_{\cdot,j}$, $1 \leq j \leq 30$.

4 The bionic wavelet transform

J. Yao and Y. T. Zhang have proposed the bionic wavelet transform (BWT) as a new time-frequency technique by referring to the perceptual model [6]. The term "bionic" means that the BWT is guided by an active biological mechanism [7]. Moreover, the BWT decomposition is both perceptually scaled and adaptive [8]. The initial perceptual aspect of the transform comes from the logarithmic spacing of the baseline scale variables which are designed to match basilar membrane spacing [8]. Then, two adaptation factors control the time-support employed at each scale, based on a non-linear perceptual model of the auditory system [8]. The basis of this transform is the Giguerre -Woodland non-linear transmission line model of the auditory system [9, 10], an active-feedback electro-acoustic model incorporating the auditory canal, middle ear, and cochlea [8]. The model yields estimates of the time-varying acoustic compliance and resistance along the displaced basilar membrane, as a physiological acoustic mass function, cochlear frequency-position mapping, and feedback factors representing the active mechanisms of outer hair cells. The net result can be seen as a technique for the estimation of the time-varying quality factor Q_{eq} of the cochlear filter banks as the input sound waveform function [8]. The references [6–9] give the complete details on the elements of this model. The BWT adaptive nature is ensured by a time-varying linear factor $T(a,\tau)$ which represents the scaling of the cochlear filter bank quality factor Q_{eq} at each scale over time [8]. For each scale and time, the adaptation factor $T(a,\tau)$ of BWT is computed by using the update equation [8]:

$$T(a,\tau+\Delta\tau) = \frac{1}{\left[1 - G_1 \frac{G_s}{G_s+|X_{BWT}(a,\tau)|}\right]\left[1 + G_2|\frac{\partial}{\partial t}X_{BWT}(a,\tau)|\right]} \tag{3}$$

where C_s is a constant (typically $C_s = 0.8$) that represents non linear saturation effects in the cochlear model [6, 8].

The quantities G_1 and G_2 are respectively the active gain factor, which represents the outer hair cell active resistance function, and the active gain factor representing the time-varying compliance of the basilar membrane [8]. Practically speaking, the partial derivative in equation (3) can be approximated by using the first difference of the previous points of the BWT at that scale [8]. $X_{BWT}(a,\tau)$ represents the bionic wavelet transform (BWT) of the signal $x(t)$ and it is given by:

$$X_{BWT}(a,\tau) = \frac{1}{T(a,\tau)\sqrt{a}} \int x(t) \cdot \tilde{\varphi}^* \left[\frac{t-\tau}{a \cdot T(a,\tau)}\right] \cdot e^{-j\omega_0(\frac{t-\tau}{a})} dt, \tag{4}$$

where a denotes the parameter of scale, τ is the shifting parameter in time and $\tilde{\varphi}$ is the mother wavelet envelop given by [7]:

$$\varphi(t) = \frac{1}{T(a,\tau)\sqrt{a}} \tilde{\varphi}\left[\frac{t}{T(a,\tau)}\right] \cdot e^{j\omega_0 t} \tag{5}$$

where ω_0 is the base fundamental frequency of the unscaled mother wavelet.

In practice ω_0 is equals to 15165.4 for the human auditory system [6]. The discretization of the scale a is achieved by employing a pre-determined logarithmic spacing across the desired frequency range, so that at each scale the center frequency is expressed by [8]:

$$\omega_m = \frac{\omega_0}{(1.1623)^m}, m = 0, 1, 2, \ldots. \tag{6}$$

Based on Yao and Zhang's original work for cochlear implant coding [9], coefficients at 22 scales, m = 7,. . . ,28, are calculated employing numerical integration of the continuous wavelet transform. These 22 scales correspond to center frequencies logarithmically spaced from 225 Hz to 5300 Hz. (Although the scales used here match those from Yao and Zhang's original work, empirical variation of the number of scales and frequency placement showed minimal effect on the overall enhancement results). For this implementation, we have used coefficients at 30 scales. In the formula (4), the role of first factor $T(a,\tau)$ multiplying $\sqrt{a}$ is to ensure that the energy remains the same for each mother wavelet. The role of second factor $T(a,\tau)$ is to adjust the envelop $\tilde{\varphi}(t)$ without adjusting the central frequency of $\varphi(t)$ [7]. Thus, the main difference between (BWT) and the continuous wavelet transform (CWT) is based on the fact that the time-frequency resolution achieved by (BWT) can be adjusted in an adaptive manner not only by frequency variation of the signal but also by instantaneous amplitudes of this signal. It is the mother wavelet which makes the continuous wavelet transform adaptive, while the adaptive characteristic of the BWT comes from the mechanism of active control in the human auditory model. which adjusts the mother wavelet associated to (BWT) according to the analyzed signal. Basically, the idea of the (BWT) is inspired from the fact that we need to make the mother wavelet envelop variable in time according to the signal characteristics.

The employed mother wavelet $\varphi(t)$ in [7] is the Morlet wavelet and its envelop $\tilde{\varphi}(t)$ is given by [8]:

$$\tilde{\varphi}(t) = e^{\left[-(\frac{t}{T_0})^2\right]} \tag{7}$$

where T_0 denotes the initial time-support.

Figure 3 illustrated the real and the imaginary parts of the complex Morlet mother wavelet.

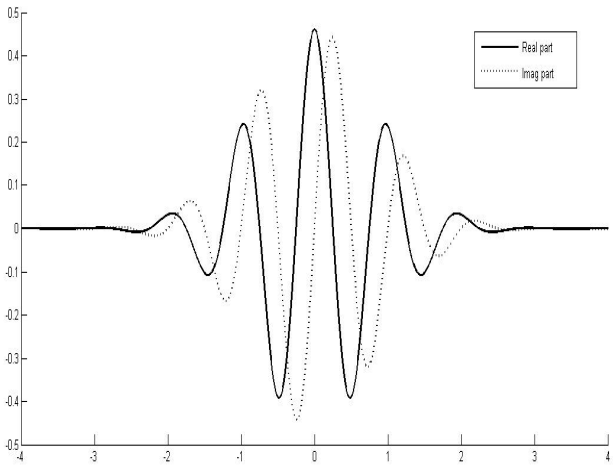


Figure 3: The Morlet wavelet.

It can be shown [7, 9] that the obtained BWT coefficients, $X_{BWT}(a,\tau)$ are derived by using the following formula [8]:

$$X_{BWT}(a,\tau) = K(a,\tau)X_{WT}(a,\tau), \tag{8}$$

where $K(a,\tau)$ is given by:

$$K(a,\tau) = \frac{\sqrt{\pi}}{C}\frac{T_0}{\sqrt{1+T^2(a,\tau)}} \tag{9}$$

where C represents a normalizing constant calculated from the squared mother wavelet integral.

This representation yields to an effective computational technique for calculating in direct manner, the BWT coefficients from those of the wavelet transform. This is performed without using the BWT definition given by equation (4). There are some key differences between the discretized CWT employing the Morlet wavelet used for the BWT and a filterbank based WPT using an orthonormal wavelet. One of them is that the WPT provides a perfect reconstruction, while the discretized CWT is an approximation whose exactness depends on the number and placement of frequency bands selected [8].

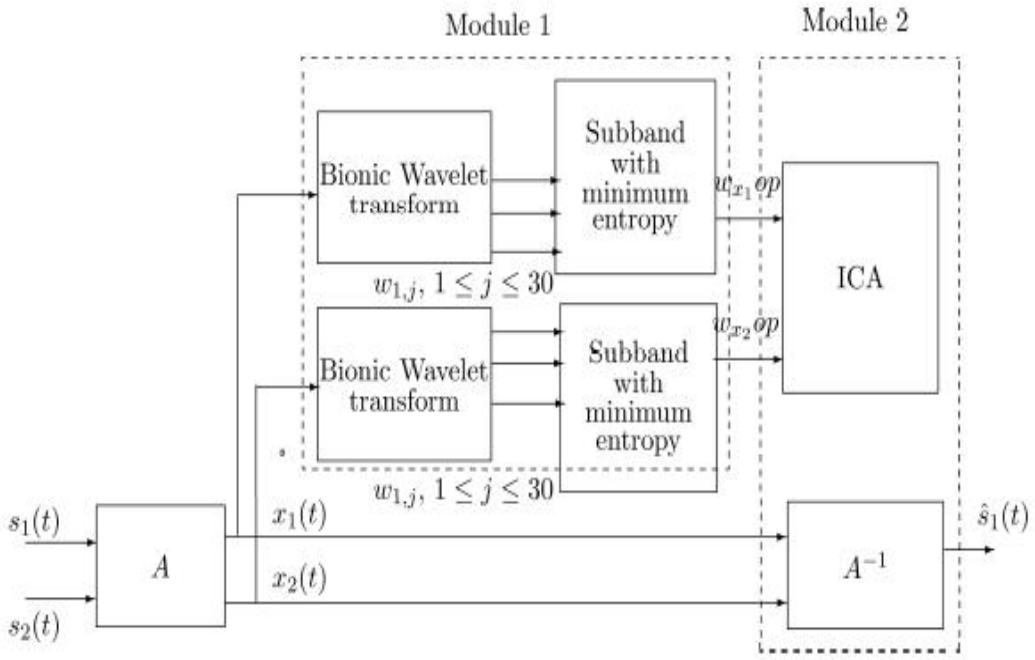


Figure 4: Overview of the proposed system.

5 Criterion of evaluation

For evaluating our proposed technique, we have compared it to the temporal technique based on runica. The evaluation is based on SNR, SSNR, ISd and PESQ computation. These parameters are defined as follow:

- **Signal-to-noise ratio**
  The signal-to-noise ratio (SNR) of the enhanced speech signal is defined by:

$$SNR_{dB} = 10\text{Log}_{10}\left[\frac{\sum_{n=0}^{N-1} x(n)^2}{\sum_{n=0}^{N-1}(x(n)-\hat{x}(n))^2}\right], \tag{10}$$

  where $x(n)$ and $\hat{x}(n)$ represent respectively the original and the enhanced speech signals, and N is the samples number per signal.

- **Segmental signal to noise ratio**
  The segmental signal-to-noise ratio (segSNR) is calculated by averaging the frame based SNRs over the signal:

$$segSNR_{dB} = \frac{1}{M}\sum_{m=0}^{M-1} 10\text{Log}_{10}\left[\frac{\sum_{n=N_m}^{N_m+N-1} x(n)^2}{\sum_{n=N_m}^{N_m+N-1}(x(n)-\hat{x}(n))^2}\right], \tag{11}$$

where M designates the number of frames, N is the size of frame, and N_m is the beginning of the $m-th$ frame. As the SNR can become negative and very small during silence periods, the segSNR values are limited to the range of [-10dB, 35dB] as per [10].

- **Itakura-Saito distance**
The distance of Itakura-Saito (ISd) measures the spectrum changes and can be computed employing the coefficients of linear prediction (LPC) and this according to the following equation:

$$ISD(a,b) = \frac{(a-b)^T R(a,b)}{a^T R a}, \tag{12}$$

where a represents the LPC vector of the original speech signal $x(n)$. R is the matrix of autocorrelation and b is the LPC coefficients vector of the enhanced speech signal $\hat{x}(n)$. In this work, a 10^{th} order LPC based measure is employed.

- **Perceptual evaluation of speech quality**
The perceptual evaluation of speech quality (PESQ) algorithm [11, 12] is an objective quality measure, that is approved as the ITU-T recommendation P.862. It is a tool of objective measurement conceived to predict the results of a subjective Mean Opinion Score (MOS) test. It was proved [13, 14] that the PESQ is more reliable and correlated better with MOS than the traditional objective speech measures.

6 Results and discussion

From Table 1 to Table 8, we report the obtained results from the application of our proposed speech enhancement technique and the temporal technique based on runica on eight noisy sentences taken from the Timit database.

Table 1: Sentence 1.

Parameters	Proposed method	Temporal method
SNRi of the first mixture	0.7672	0.7672
SNRi of the second mixture	-3.4638	-3.4638
SNRf(dB)	69.5528	58.7147
SSNRi of the first mixture	-3.5715	-3.5715
SSNRi of the second mixture	-6.3509	-6.3509
SSNRf(dB)	34.8736	34.1366
PESQi of the first mixture	1.3110	1.3110
PESQi of the second mixture	1.0983	1.0983
PESQf	4.4989	4.4936
ISdi of the first mixture	2.4029	2.4029
ISdi of the second mixture	3.9475	3.9475
ISdf	$4.181\ 10^{-10}$	$4.948\ 10^{-8}$

These results show clearly that our proposed technique outperforms the temporal technique of source separation using standard ICA [1].

Fig. 5 and Fig. 6 illustrate an example of speech enhancement using our proposed technique.

Table 2: Sentence 2.

Parameters	Proposed method	Temporal method
SNRi of the first mixture	-2.1038	-2.1038
SNRi of the second mixture	-6.6325	-6.6325
SNRf(dB)	66.5296	48.9184
SSNRi of the first mixture	-5.8749	-5.8749
SSNRi of the second mixture	-7.8815	-7.8815
SSNRf(dB)	34.2126	30.7198
PESQi of the first mixture	1.4577	1.4577
PESQi of the second mixture	1.2445	1.2445
PESQf	4.4981	4.4535
ISdi of the first mixture	2.8500	2.8500
ISdi of the second mixture	4.6523	4.6523
ISdf	$1.4043\ 10^{-9}$	$3.1847\ 10^{-6}$

Table 3: Sentence 3.

Parameters	Proposed method	Temporal method
SNRi of the first mixture	4.2400	4.2400
SNRi of the second mixture	-0.2606	-0.2606
SNRf(dB)	77.4719	49.5281
SSNRi of the first mixture	-1.4316	-1.4316
SSNRi of the second mixture	-4.6465	-4.6465
SSNRf(dB)	34.7279	31.6486
PESQi of the first mixture	1.9873	1.9873
PESQi of the second mixture	1.7460	1.7460
PESQf	4.4999	4.4621
ISdi of the first mixture	1.8622	1.8622
ISdi of the second mixture	3.1204	3.1204
ISdf	$2.5837\ 10^{-10}$	$8.2210\ 10^{-6}$

7 Conclusion

In this paper, we have proposed a new speech enhancement technique that consists in applying in the first step, the Bionic Wavelet Transform (BWT) to two different mixtures of the same speech signal with gaussian white noise with two different values of Signal to Noise Ratio (SNR). The second step consists in computing the entropy of each bionic wavelet coefficient and finds the two subbands having the minimal entropy. Those two subbands are used to estimate the separation matrix of the speech signal from noise by employing the source separation. The obtained results from the SNR, SSNR, ISd and PESQ computation, show clearly that the proposed speech enhancement technique outperforms the temporal technique of source separation using standard ICA.

Table 4: Sentence 4.

Parameters	Proposed method	Temporal method
SNRi of the first mixture	1.6366	1.6366
SNRi of the second mixture	-2.7143	-2.7143
SNRf(dB)	61.8042	55.4325
SSNRi of the first mixture	-4.3027	-4.3027
SSNRi of the second mixture	-6.3345	-6.3345
SSNRf(dB)	31.4223	29.4299
PESQi of the first mixture	1.5414	1.5414
PESQi of the second mixture	1.2476	1.2476
PESQf	4.4816	4.4506
ISdi of the first mixture	2.5180	2.5180
ISdi of the second mixture	4.0632	4.0632
ISdf	$7.4591\ 10^{-8}$	$8.9142\ 10^{-7}$

Table 5: Sentence 5.

Parameters	Proposed method	Temporal method
SNRi of the first mixture	-0.7340	-0.7340
SNRi of the second mixture	-5.7034	-5.7034
SNRf(dB)	59.5033	57.0129
SSNRi of the first mixture	-5.6047	-5.6047
SSNRi of the second mixture	-7.6901	-7.6901
SSNRf(dB)	31.4463	30.8201
PESQi of the first mixture	1.3907	1.3907
PESQi of the second mixture	1.1391	1.1391
PESQf	4.4814	4.4748
ISdi of the first mixture	2.6384	2.6384
ISdi of the second mixture	4.2292	4.2292
ISdf	$2.4528\ 10^{-8}$	$1.0487\ 10^{-7}$

Table 6: Sentence 6.

Parameters	Proposed method	Temporal method
SNRi of the first mixture	2.5544	2.5544
SNRi of the second mixture	-2.1241	-2.1241
SNRf(dB)	62.4473	62.2521
SSNRi of the first mixture	-3.3643	-3.3643
SSNRi of the second mixture	-5.8605	-5.8605
SSNRf(dB)	31.4816	31.4398
PESQi of the first mixture	1.3805	1.3805
PESQi of the second mixture	1.0564	1.0564
PESQf	4.4929	4.4926
ISdi of the first mixture	2.6047	2.6047
ISdi of the second mixture	4.1036	4.1036
ISdf	$9.3330\ 10^{-8}$	$1.0324\ 10^{-7}$

Table 7: Sentence 7.

Parameters	Proposed method	Temporal method
SNRi of the first mixture	-0.0411	-0.0411
SNRi of the second mixture	-4.9428	-4.9428
SNRf(dB)	69.0167	67.9430
SSNRi of the first mixture	-5.7362	-5.7362
SSNRi of the second mixture	-7.8201	-7.8201
SSNRf(dB)	33.9383	33.8069
PESQi of the first mixture	1.6015	1.6015
PESQi of the second mixture	1.4326	1.4326
PESQf	4.4978	4.4974
ISdi of the first mixture	2.5867	2.5867
ISdi of the second mixture	4.1275	4.1275
ISdf	$4.6205\ 10^{-10}$	$6.7472\ 10^{-10}$

Table 8: Sentence 8.

Parameters	Proposed method	Temporal method
SNRi of the first mixture	-1.7018	-1.7018
SNRi of the second mixture	-6.5995	-6.5995
SNRf(dB)	68.1109	52.2155
SSNRi of the first mixture	-5.8031	-5.8031
SSNRi of the second mixture	-7.8610	-7.8610
SSNRf(dB)	33.2544	29.8972
PESQi of the first mixture	1.3814	1.3814
PESQi of the second mixture	1.1889	1.1889
PESQf	4.4968	4.4504
ISdi of the first mixture	3.3273	3.3273
ISdi of the second mixture	5.1147	5.1147
ISdf	$3.1514\ 10^{-9}$	$5.4382\ 10^{-6}$

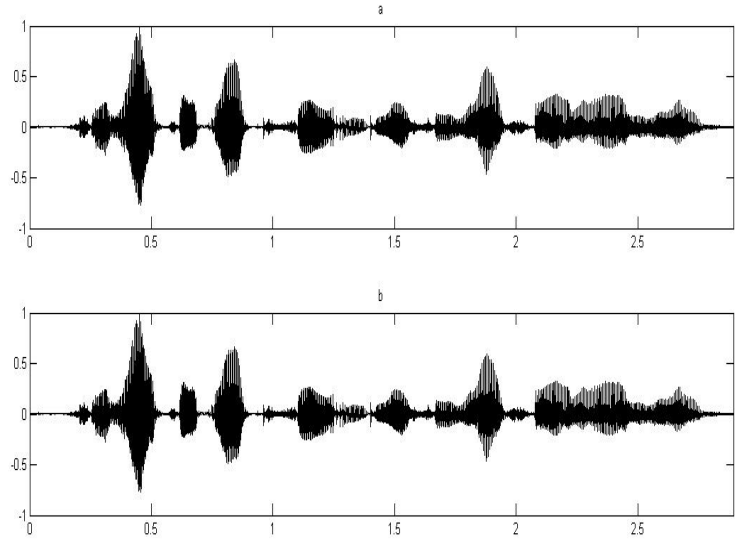


Figure 5: (a) clean speech signal, (b) enhanced speech signal.

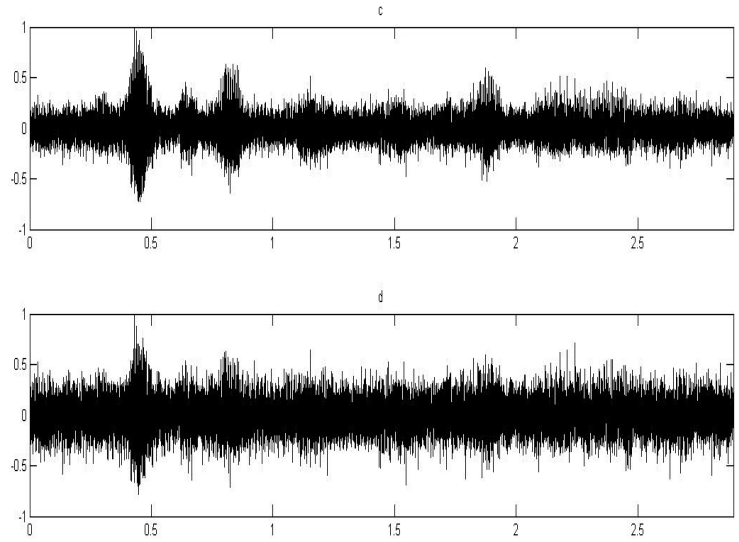


Figure 6: (c) first mixture, (d) second mixture.

Bibliography

[1] A. J. Bell, T. J. Sejnowski, An information maximization approach to blind separation and blind deconvolution, *Neural Computation*, Vol.7, pp.1004-1034, 1995.

[2] A Hyvarinen, J. Karhunen, E. Oja, *Independent component analysis*, Wiley and Sons, 2001.

[3] T. Tanaka, A. Cichocki, Subband decomposition independent component analysis and new performance criteria, *ICASSP*, pp.541-544, 2004.

[4] P. Kisilev, M. Zibulevsky, Blind source separation using multinode sparse representation, *ICIP*, 2001.

[5] R. Moussaoui, J. Rouat, R. Lefebvre, Wavelet Based Independent Component Analysis for Multi-Channel Source Separation, *ICASSP*, pp.645-648, 2006.

[6] J. Yao, Y. T. Zhang, Bionic wavelet transform: a new timefrequency method based on an auditory model, *IEEE Trans. on Biomedical Engineering* Vol.48, No.8, pp.856-863, 2001.

[7] Xiaolong Yuan, B.S.E.E. A THESIS, Ť Auditory Model-based Bionic Wavelet Transform for speech enhancement. Electrical and computer engineering.

[8] M. T. Johnsona, X. Yuanb, Y. Rena, Speech signal enhancement through adaptive wavelet thresholding. *in conference Elsevier*, pp.123-133, 2007.

[9] J. Yao, Y. T. Zhang, The application of bionic wavelet transform to speech signal processing in cochlear implants using neural network simulations, *IEEE Trans. Biomed. Eng.*, Vol.49, No.11, pp. 1299-1309, 2002.

[10] B. Chen, P. C. Loizou, A Laplacian-based MMSE estimator for speech enhancement, *Speech Communication*, Vol.49, No.2, pp.134-143, 2007.

[11] ITU-T P.862. Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrowband telephone networks and speech codecs, *ITU Recommendation P.862*, 2001.

[12] A. W. Rix, J. G. Beerends, M. P. Hollier, A. P. Hekstra, Perceptual evaluation of speech quality (pesq) - a new method for speech quality assessment of telephone networks and codecs, *ICASSP*, pp.749-752, 2001.

[13] Y. Hu, P. C. Loizou, Evaluation of objective measures for speech enhancement, *IEEE Trans. Speech, Audio Processing*, Vol.16, No.1, pp.229-238, 2008.

[14] E. Zavarehei, S. Vaseghi, Q. Yan. Inter- frame modeling of DFT trajectories of speech and noise for speech enhancement using Kalman filters, *Speech Communication*, Vol.48, No.11, pp.1545-1555, 2006.

Author index