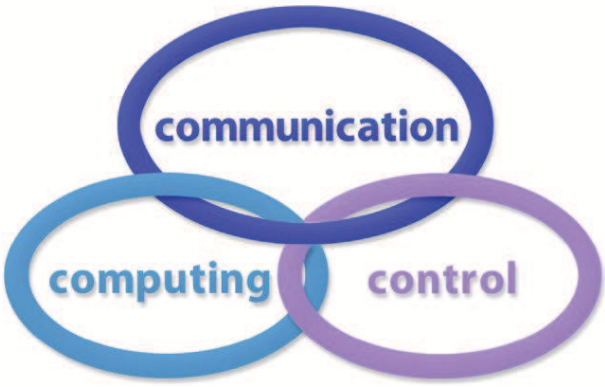
INTERNATIONAL JOURNAL

of

COMPUTERS COMMUNICATIONS & CONTROL

ISSN 1841-9836



A Bimonthly Journal
With Emphasis on the Integration of Three Technologies

Year: 2018 Volume: 13 Issue: 4 Month: August

This journal is a member of, and subscribes to the principles of, the Committee on Publication Ethics (COPE).

http://univagora.ro/jour/index.php/ijccc/

BRIEF DESCRIPTION OF JOURNAL

Publication Name: International Journal of Computers Communications & Control.
Acronym: IJCCC; **Starting year of IJCCC:** 2006.
ISO: Int. J. Comput. Commun. Control; **JCR Abbrev:** INT J COMPUT COMMUN.
International Standard Serial Number: ISSN 1841-9836.
Publisher: CCC Publications - Agora University of Oradea.
Publication frequency: Bimonthly: Issue 1 (February); Issue 2 (April); Issue 3 (June); Issue 4 (August); Issue 5 (October); Issue 6 (December).
Founders of IJCCC: Ioan DZITAC, Florin Gheorghe FILIP and Misu-Jan MANOLESCU.

Indexing/Coverage:

- Since 2006, Vol. 1 (S), IJCCC is covered by Clarivate Analytics and is indexed in ISI Web of Science/Knowledge: Science Citation Index Expanded.

  2018 Journal Citation Reports® Science Edition (Clarivate Analytics, 2017):
  Subject Category: (1) Automation & Control Systems: Q4(2009, 2011, 2012, 2013, 2014, 2015), Q3(2010, 2016, 2017); (2) Computer Science, Information Systems: Q4(2009, 2010, 2011, 2012, 2015), Q3(2013, 2014, 2016, 2017).

  Impact Factor/3 years in JCR: 0.373(2009), 0.650 (2010), 0.438(2011); 0.441(2012), 0.694(2013), 0.746(2014), 0.627(2015), 1.374(2016), **1.29 (2017)**.
  Impact Factor/5 years in JCR: 0.436(2012), 0.622(2013), 0.739(2014), 0.635(2015), 1.193(2016), **1.179(2017)**.

- Since 2008 IJCCC is indexed by Scopus: **CiteScore 2017 = 1.04**.
  Subject Category:
  (1) Computational Theory and Mathematics: Q4(2009, 2010, 2012, 2015), Q3(2011, 2013, 2014, 2016, 2017);
  (2) Computer Networks and Communications: Q4(2009), Q3(2010, 2012, 2013, 2015), Q2(2011, 2014, 2016, 2017);
  (3) Computer Science Applications: Q4(2009), Q3(2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017).

  SJR: 0.178(2009), 0.339(2010), 0.369(2011), 0.292(2012), 0.378(2013), 0.420(2014), 0.263(2015), 0.319(2016), 0.326 (2017).

- Since 2007, 2(1), IJCCC is indexed in EBSCO.

Focus & **Scope:** International Journal of Computers Communications & Control is directed to the international communities of scientific researchers in computers, communications and control, from the universities, research units and industry. To differentiate from other similar journals, the editorial policy of IJCCC encourages the submission of original scientific papers that focus on the integration of the 3 "C" (Computing, Communications, Control).

In particular, the following topics are expected to be addressed by authors:

(1) Integrated solutions in computer-based control and communications;

(2) Computational intelligence methods & Soft computing (with particular emphasis on fuzzy logic-based methods, computing with words, ANN, evolutionary computing, collective/swarm intelligence, membrane computing, quantum computing);

(3) Advanced decision support systems (with particular emphasis on the usage of combined solvers and/or web technologies).

FAMOUS FORMER MEMBER IN THE EDITORIAL BOARD OF IJCCC

Lotfi A. Zadeh (Feb. 4, 1921 - Sept. 6, 2017)
The inventor of Fuzzy Sets, Fuzzy Logic, and Soft Computing
Former member in the Editorial Board of IJCCC between 2008-2017

FAMOUS EXCELLENCE AWARD PROPOSED FOR IJCCC

Journals Excellence Awards proposed by Scopus (2015)

EDITORIAL STAFF OF IJCCC (2018)

EDITORIAL ADDRESS:
Agora University, Cercetare Dezvoltare Agora, Tineretului 8, 410526 Oradea, Bihor, Romania,
Tel./ Fax: +40 359101032, E-mail: ijccc@univagora.ro, rd.agora@univagora.ro
URL: http://univagora.ro/jour/index.php/ijccc/

EDITORIAL BOARD OF IJCCC (MEMBERS, 2018):

Constantin GAINDRIC
IMMAS, Republic of MOLDOVA
Kishinev, 277028, Academiei 5
gaindric@math.md

Xiao-Shan GAO
Academia Sinica, CHINA
Beijing 100080, China
xgao@mmrc.iss.ac.cn

Enrique HERRERA-VIEDMA
University of Granada, SPAIN
Av. del Hospicio, s/n, 18010 Granada
viedma@decsai.ugr.es

Kaoru HIROTA
Tokyo Institute of Tech., JAPAN
G3-49,4259 Nagatsuta
hirota@hrt.dis.titech.ac.jp

Arturas KAKLAUSKAS
VGTU, LITHUANIA
Sauletekio al. 11, LT-10223 Vilnius
arturas.kaklauskas@vgtu.lt

Gang KOU
SWUFE, CHINA
Chengdu, 611130
kougang@swufe.edu.cn

Heeseok LEE
KAIST, SOUTH KOREA
85 Hoegiro, Seoul 02455
hsl@business.kaist.ac.kr

George METAKIDES
University of Patras, GREECE
Patra 265 04, Greece
george@metakides.net

Shimon Y. NOF
Purdue University, USA
610 Purdue Mall, West Lafayette
nof@purdue.edu

Stephan OLARIU
Old Dominion University, USA
Norfolk, VA 23529-0162
olariu@cs.odu.edu

Gheorghe PĂUN
Romanian Academy, ROMANIA
IMAR, Bucharest, PO Box 1-764
gpaun@us.es

Mario de J. PEREZ JIMENEZ
University of Seville, SPAIN
Avda. Reina Mercedes s/n, 41012
marper@us.es

Radu-Emil PRECUP
Pol. Univ. of Timisoara, ROMANIA
Bd. V. Parvan 2, 300223
radu.precup@aut.upt.ro

Radu POPESCU-ZELETIN
Technical University Berlin, GERMANY
Fraunhofer Institute for Open CS
rpz@cs.tu-berlin.de

Imre J. RUDAS
Obuda University, HUNGARY
Budapest, Becsi ut 96b, 1034
rudas@bmf.hu

Yong SHI
Chinese Academy of Sciences, CHINA
Beijing 100190
yshi@gucas.ac.cn, yshi@unomaha.edu

Bogdana STANOJEVIC
Serbian Academy of SA, SERBIA
Kneza Mihaila 36, Beograd 11001
bgdnpop@mi.sanu.ac.rs

Athanasios D. STYLIADIS
University of Kavala, GREECE
65404 Kavala
styliadis@teikav.edu.gr

Gheorghe TECUCI
George Mason University, USA
University Drive 4440, Fairfax VA
tecuci@gmu.edu

Horia-Nicolai TEODORESCU
Romanian Academy, ROMANIA
Iasi Branch, Bd. Carol I 11, 700506
hteodor@etc.tuiasi.ro

Dan TUFIŞ
Romanian Academy, ROMANIA
13 Septembrie, 13, 050711 Bucharest
tufis@racai.ro

Edmundas K. ZAVADSKAS
VGTU, LITHUANIA
Sauletekio ave. 11, LT-10223 Vilnius
edmundas.zavadskas@vgtu.lt

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, Volume 13, Issue 4, August 2018.

Contents

Trajectory Tracking Control for Seafloor Tracked Vehicle by Adaptive Neural-Fuzzy Inference System Algorithm
Y. Dai, X. Zhu, H. Zhou, Z. Mao, W. Wu **465**

An Improved ABC Algorithm for Energy Management of Microgrid
R. Gao, J. Wu, W. Hu, Y. Zhang **477**

An Optimized DBN-based Coronary Heart Disease Risk Prediction
K. Lim, B.M. Lee, U. Kang, Y. Lee **492**

A Simulation based Analysis of an Multi Objective Diffusive Load Balancing Algorithm
I.D. Mironescu, L. Vinţan **503**

An Approach for Detecting Fault Lines in a Small Current Grounding System using Fuzzy Reasoning Spiking Neural P Systems
H. Rong, M. Ge, G. Zhang, M. Zhu **521**

Enhanced Interconnection Model in Geographically Interdependent Networks
D.F. Rueda, E. Calle, X. Wang, R.E. Kooij **537**

Improving DTNs Performance by Reduction of Bundles Redundancy using Clustering Algorithm
R.O. Schoeneich, P. Prus **550**

Nonparametric Regression-based Step-length Estimation for Arm-swing Walking using a Smartphone
P.H. Truong, N.D. Nguyen, N.H. Ho, G.-M. Jeong **566**

Arithmetic Operations with Spiking Neural P Systems with Rules and Weights on Synapses
H.F. Wang, K. Zhou, G.X. Zhang **574**

An ABC Algorithm with Recombination
X. You, Y. Ma, Z. Liu, M. Xie **590**

Author index **602**

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 13(4), 465-476, August 2018.

Trajectory Tracking Control for Seafloor Tracked Vehicle by Adaptive Neural-Fuzzy Inference System Algorithm

Y. Dai, X. Zhu, H. Zhou, Z. Mao, W. Wu

Yu Dai*
1. College of Mechanical and Electrical Engineering
Central South University
Changsha 410083, China
2. State Key Laboratory of Ocean Engineering
Shanghai Jiao Tong University
Shanghai 200240, China
*Corresponding author: daiyu_6@aliyun.com

Xiang Zhu, Haibo Zhou, Zuoli Mao, Wei Wu
1. College of Mechanical and Electrical Engineering
Central South University
Changsha 410083, China
zhuxiang_csu@163.com, 1584269607@qq.com
zhouhaibo@csu.edu.cn, 819964861@qq.com

Abstract: Trajectory tracking control strategy and algorithm for the tracked vehicle moving on the seafloor has aroused much concerns due to the commonly occurred serious slip and trajectory deviation caused by the seafloor extremely soft and cohesive sediment. An improved multi-body dynamic model of a seafloor tracked vehicle (STV) has been established in a simulation code RecurDyn/Track. A particular terramechanics model with a dynamic shear displacement expression for the vehicle-sediment interaction has been built and integrated into the multi-body dynamic model. The collaborative simulation between the mechanical multi-body dynamic model in RecurDyn/Track and the control model in MATLAB/Simulink has been achieved. Different control algorithms performances including a PID control, a fuzzy control and a neural control, have been compared and proved the traditional or individual intelligent controls are not particularly suitable for the tracked vehicle on the seafloor. Consequently, an adaptive neural-fuzzy inference system (ANFIS) control algorithm with hybrid learning method for parameter learning which is an integrated control method combined with the fuzzy and neural control, has been adopted and designed. A series of collaborative simulations have been performed and proved the ANFIS algorithm can achieve a better trajectory tracking control performance for the STV as its trajectory deviation can be maintained within a permissible range.
Keywords: seafloor tracked vehicle, multi-body dynamic model, adaptive neural-fuzzy inference system (ANFIS), collaborative simulation, trajectory tracking control.

1 Introduction

Tracked vehicles are widely used in the deep seafloor engineering fields, such as seafloor exploration, seafloor cable laying and installation, seafloor dredging, seafloor mineral resources exploitation, etc. The deep seafloor extremely soft and cohesive sediment is completely different from land-surface soils, which makes the tracked vehicle more likely to be involved in serious slip, large sinkage and motion trajectory deviation. Its locomotion performance and control characteristics directly affect the continuous operation performance and operation safety for the tracked vehicle on the seafloor.

Influenced by the seafloor complex and changeable environmental loads, it is particularly difficult to master and evaluate the mobility and locomotion of the seafloor tracked vehicle,

Sup et al. adopted a new technology based on Euler parameters for evaluating the dynamic properties of a tracked vehicle on seafloor [13] and further put forward a subsystem synthesis method to analyse a multi-body model of a tracked vehicle [15]. Kim et al. researched the complicated dynamics of an articulated tracked vehicle crawling on the seafloor inclined and undulating terrain [14]; furthermore the effects of the buoyancy position layout on the dynamics of the vehicle were analysed [16]. Li et al. built a virtual prototype of a seafloor tracked mining vehicle and conducted simulations to estimate the vehicle's locomotion and trafficability [17]. Li et al. studied the effect of the grouser height of a seafloor tracked mining vehicle on its tractive performance through an established relationship between the total driving force and the slip of the mining vehicle [18]. Dai et al. developed new multi-body dynamic models for three types of seafloor tracked vehicles and performed simulations to evaluate their locomotion and trafficability performances [5–7]; besides, the complex integrated dynamic performances for seafloor tracked vehicles connecting to pipeline systems and surface ships were investigated and evaluated [8, 9].

To control the locomotion state and trajectory of the tracked vehicle on the seafloor, Herzog et al. established an automatic hydraulic drive mode with slip control of the driving track for a seafloor tracked vehicle; meanwhile an experimental system for the slip control development along with the logic of the automatic driving model was presented [12]. Yeu et al. proposed a path tracking method for tracked vehicle on the seafloor with a vector pursuit algorithm to make the vehicle's motion following the specified path [28]. Yeu et al. further based on the kinematics of the seafloor tracked mining vehicle to propose two navigation algorithms, known as dead-reckoning and extended Kalman filter [29]. Yoon et al. used the indirect Kalman filter method with the inner measuring sensors to underwater localization of a seafloor tracked mining vehicle [30]. Zhang et al. presented a control method for a straight-line path tracking of a seabed mining vehicle based on the ANFIS control; however only a single straight-line path was tracked without comparisons to other control methods [19]. Han et al. proposed a PID control algorithm and achieved an anticipated goal for a seafloor tracked miner moving along a desired path [10, 11]. Wang et al. presented a fuzzy and a predictive controller, further the efficiency of the control method was verified by the computer simulation and experimental results [24]. Li et al. built a hydraulic system model of a seafloor self-propelled tracked mining vehicle, and its kinematic control by a fuzzy algorithm was discussed [20]. Besides, various path tracking control method researches for vehicles or robots have been conducted. Cui et al. investigated a trajectory tracking problem for a fully actuated autonomous underwater vehicle (AUV), and two neural networks (NNs) were integrated into an adaptive control design, the robustness and effectiveness of the proposed control method were tested and validated through extensive numerical simulation results [4]. Sokolov et al. presented a neuro-evolution approach for a crawler robot motion that can autonomously solve the sequences of the navigation and flipper control tasks to overcome obstacles [23]. Bozic et al. based on a combination of neural networks and genetic algorithm to intelligent modelling and optimization of energy usage for a wheel-legged (Wheg) robot running; simulation of neuro-fuzzy control system was developed for minimization of energy usage during the Wheg's running [2]. Chen et al. proposed a robust adaptive position/force control algorithm to track the desired posture and force in opening a door for a mobile robot manipulator, and co-simulation between MATLAB and RecurDyn were performed to verify the dynamic model and control method [3]. Barai et al. proposed a two-degree-of-freedom fuzzy controller for foot trajectory tracking control of a hydraulically actuated hexapod robot, and the fuzzy pre-filter was designed by a genetic algorithm (GA) based optimization [1]. Wang et al. developed an adaptive position tracking system and a force control strategy for a non-holonomic mobile manipulator robot, which combined the merits of Recurrent Fuzzy Wavelet Neural Networks (RFWNNs), and the simulation and experimental results verified the effectiveness and robustness of the proposed method [25]. Widyotriatmo et al. proposed a control method for a team of multiple mobile

robots, the individual mobile robots tracked the assigned trajectories and also should to collision among the mobile robots by the artificial potential field algorithm [26]. Ngo et al. developed a robust adaptive self-organizing control system based on a novel wavelet fuzzy cerebellar model articulation controller for a robot manipulator, and verified its effectiveness through simulation and experimental results [21, 22].

However, until now, it still remains typical problems need to be further resolved, an optimized trajectory tracking control strategy for the tracked vehicle on the seafloor has not yet been designed; furthermore, a collaborative model with real-time information interactions between the mechanical-control systems has not yet been achieved, so the trajectory tracking control performance and accuracy can not be evaluated and optimized. As it is costly and extremely difficult to perform seafloor in-situ tests, the collaborative simulation for the combined mechanical-control systems is an effective way. An optimized trajectory tracking control strategy and a mechanical-control systems collaborative simulation research were conducted in the paper.

2 Multi-body dynamic model of a STV

The dynamic simulation code RecurDyn/Track, based on a relative coordinate system and a recursive algorithm relative, was adopted to establish an improved multi-body dynamic model of a STV as shown in Fig. 1. Table 1 gives its main structural parameters.

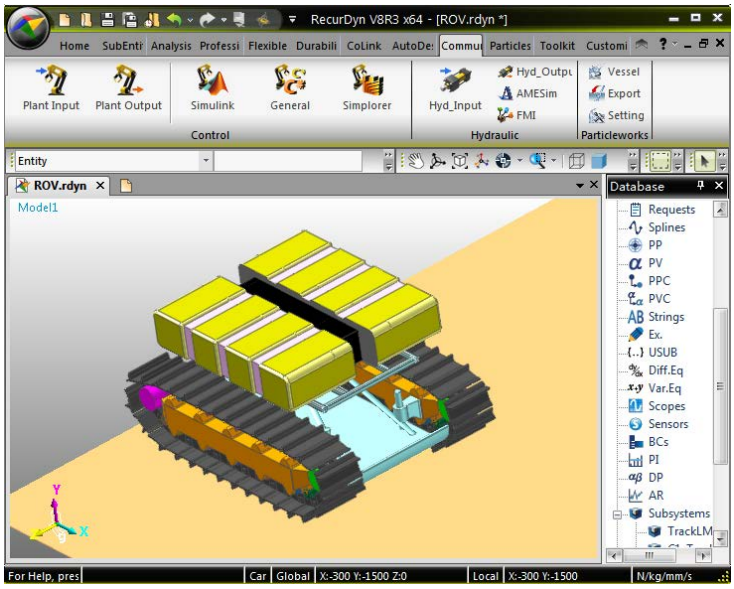


Figure 1: A 3D improved multi-body dynamic model of a STV

A user-written subroutine for characterizing the particular terramechanics model of the seafloor sediment was developed in the C language in the Visual Studio.Net environment and then integrated into the RecurDyn/Track environment. Meanwhile the dynamic processes between the track-sediment interactions were taken into account. Through laboratory simulant experiments as shown in Fig. 2, a pressure-sinkage relationship and a shear stress-shear displacement relationship between the track-sediment interactions have been obtained.

The normal force F_{ni} acting on each track link ith element can computed by multiplying the pressure with area of each track link as [27]:

$$F_{n_i} = p_{x_i} \cdot \Delta A_i = \left[\left(\frac{k_c}{b} + k_\varphi \right) \cdot (\Delta z_i)^n \right] \cdot \Delta A_i \tag{1}$$

Table 1: Structural parameters of the STV

Parameters	Values
Total weight underwater (tons)	2.35
Overall dimension (m): Length × width × height	2.3×1.6×1.2
Track contact length (m)	1.6
Track width (m)	0.36
Distance between centre lines of tracks (m)	1.2
Track pitch (m)	0.15
Grouser height (m)	0.15
Diameter of road wheel (m)	0.04
Number of road wheel per track	7
Diameter of support roller (m)	0.04
Number of support roller per track	5

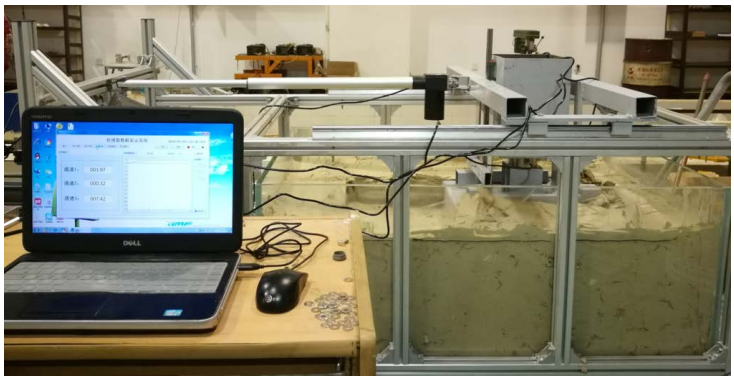


Figure 2: Laboratory experimental system for simulant track-sediment interaction mechanics

Where p_{x_i} is the normal pressure, ΔA_i is the area of each tack link, b is the width of track link, k_c is the sediment cohesion deformation modulus, k_φ is the sediment friction deformation modulus, Δz_i is the sinkage, n is the sediment deformation exponent.

The longitudinal shear force F_{long_i} is computed by multiplying shear stress with area of each track link:

$$\begin{aligned} F_{long_i} &= \operatorname{sgn}\left(j_{x_i}\right) \tau_{x_i} \cdot \Delta A \\ &= \operatorname{sgn}\left(j_{x_i}\right) \tau_{\max} \cdot K_r \cdot \left\{1 + \left[\frac{1}{K_r(1-e^{-1})} - 1\right] e^{1-j_{x_i}/K_\omega}\right\} \left(1 - e^{-j_{x_i}/K_\omega}\right) \cdot \Delta A \end{aligned} \tag{2}$$

Where "sgn" is the signum function, j_{x_i} is the dynamic longitudinal shear displacement, $\tau_{\max}$ is the maximum shear stress, K_r is the ratio of the residual shear stress τ_{res} to $\tau_{\max}$, and K_ω is the shear displacement when $\tau_{\max}$ occurs.

The dynamic longitudinal shear displacement can be expressed as a differential equation:

$$\frac{d}{dt} j_{x_i}\left(x_i, t\right) + \frac{r_s \omega_s\left(t\right)}{x_i} \cdot j_{x_i}\left(x_i, t\right) = r_s \omega_s\left(t\right) - v_x\left(t\right) \tag{3}$$

Where r_s and $\omega_s\left(t\right)$ are the radius and angular velocity of the vehicle's sprocket, x_i and $v_x(t)$ represent the distance and actual velocity of the centre of each track link.

Similarly, the lateral shear force F_{lat_i} acting on each track link is computed as:

$$\begin{aligned} F_{lat_i} &= -\mathrm{sgn}\left(j_{y_i}\right)\tau_{x_i}\cdot\Delta A \\ &= -\mathrm{sgn}\left(j_{y_i}\right)\tau_{\max}\cdot K_r\cdot\left\{1+\left[\frac{1}{K_r(1-e^{-1})}-1\right]e^{1-j_{y_i}/K_\omega}\right\}\left(1-e^{-j_{y_i}/K_\omega}\right)\cdot\Delta A \end{aligned} \tag{4}$$

Where j_{y_i} is the dynamic lateral shear displacement.
While the dynamic lateral shear displacement as be expressed as:

$$\frac{d}{dt}j_{y_i}\left(x_i,t\right)+\frac{r_s\omega_s\left(t\right)}{x_i}j_{y_i}\left(x_i,t\right)=v_{y_i}\left(t\right) \tag{5}$$

Where v_{y_i} represent actual lateral velocity of the centre of each track link.

Fig. 3 presents a group of turning simulations trajectories of the STV with different turning velocity ratios (TVRs). The input velocity for the inner track was set to 0.5 m/s, while, the input velocities for the outer track were set to 0.6 m/s, 0.65 m/s and 0.7 m/s, respectively.

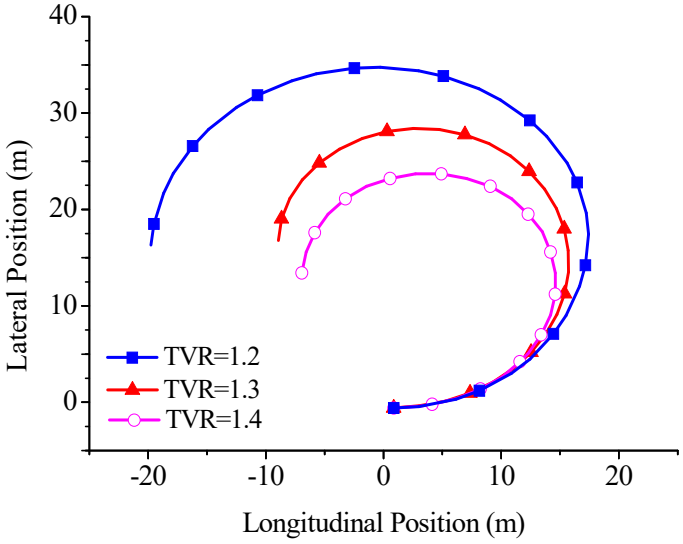


Figure 3: Simulation trajectories of the STV with different turning velocity ratios

It can be seen with the increase although a small value of the turning velocity ratio, the turning radius will increase obviously. According to the requirement of the turning velocity ratio for the STV that should not exceed 1.4, the minimum turning radius for the STV is about 12 m, which is much larger than that on land-surface soft soil and also much larger than the theoretical computational turning radius. Fig.4 presents the slips of the inner and outer tracks when the TVR is only 1.2.

It can be observed a serious slip condition occurred for the outer track of the STV in spite of a low TVR, and further exhibited a serious slip commonly occurred on the seafloor will result in a much larger turning radius for the tracked vehicle compared to move on the land-surface soft soils.

3 Trajectory tracking collaborative control simulations for a STV

A control design code MATLAB/Simulink was adopted to establish different control models for the STV. An interface toolkit RecurDyn/Control was designed to realize the information

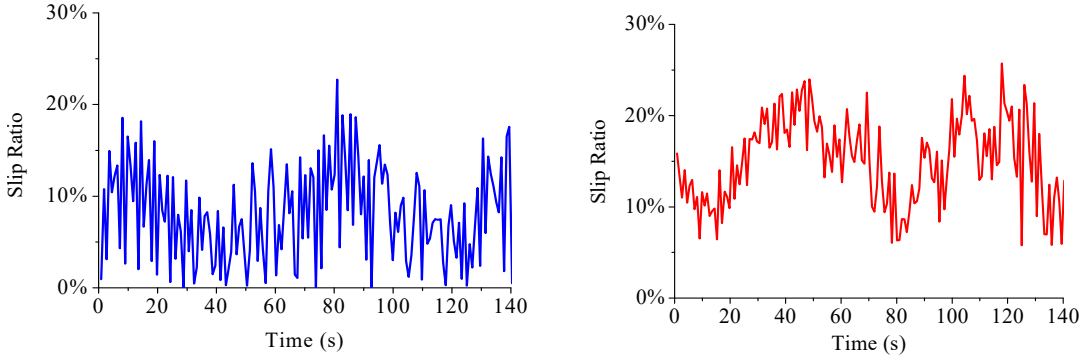


Figure 4: Slips of the left and right tracks of the STV

communication between the control system model and mechanical dynamic model. The MATLAB/Simulink was taken as the main interface. The plant input and plant output of the mechanical dynamic model were set, while, a M file for exporting the mechanical dynamic model was compiled.

3.1 Comparisons of different control algorithms

A RecurDyn/Track plant block representing a mechanical dynamic model of the STV was created in the MATLAB/Simulink; then the mechanical-control collaborative simulation can be achieved. The theoretical input velocity for the STV is 0.6 m/s in the collaborative simulation. Several different control methods including a PID control, a fuzzy logic control and a neural network control were performed and compared. Fig.5 shows the co-simulation model interface of a PID control with mechanical dynamic model, and corresponding simulation results with and without external jamming signal for the input velocity also compared. It can be seen the PID control is insensitive to the external disturbance.

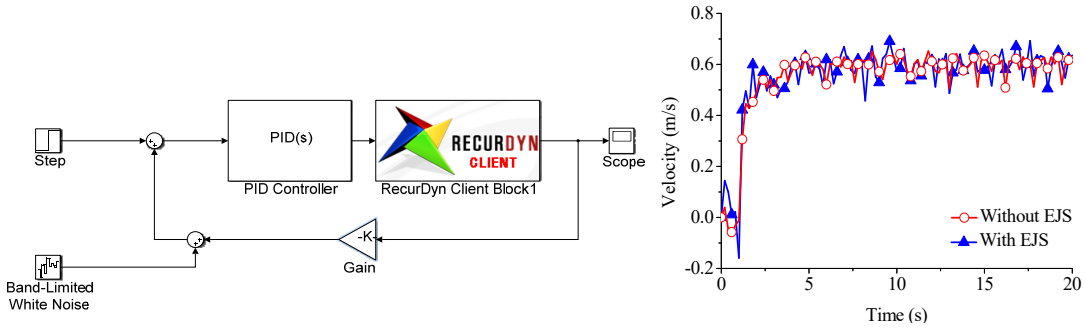


Figure 5: Co-simulation model interface and results of PID control with dynamic model

Fig. 6 shows a co-simulation model interface of an adaptive fuzzy logic control model with mechanical dynamic model, and the simulation results with and without external jamming signal compared. The inputs for the fuzzy logic controller were velocity error (E) and velocity error derivative (EC), which were the difference between the target velocity and ideal velocity. The output was the velocity compensation (U). Compared with the PID control algorithm, the fuzzy logic algorithm has a stronger anti-disturbance ability and better robustness, which was more suitable for the STV motion control. Nevertheless, the control precision of the conventional fuzzy logic is not high.

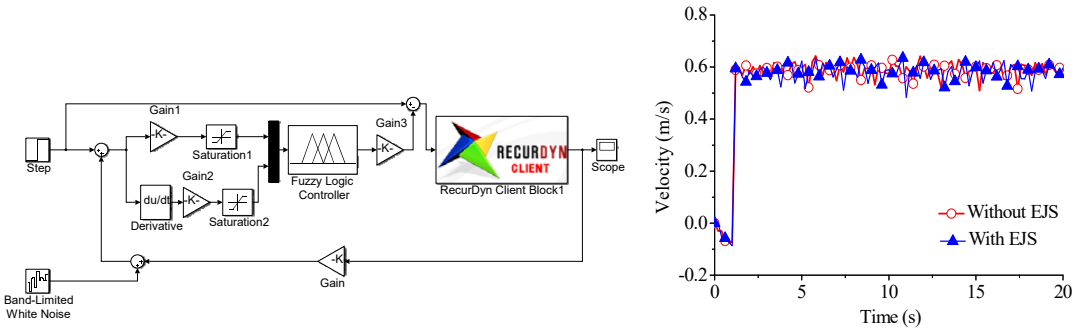


Figure 6: Co-simulation model interface and results of fuzzy logic control with dynamic model

Fig.7 shows a co-simulation model interface of a BP neural network algorithm control model with dynamic model, and simulation results with and without external jamming signal compared. The neural network has the features of self-adapting and self-learning; however, it is weak in expressing the rule knowledge. As it can be seen the neural network algorithm has a weak anti-disturbance ability compared to a fuzzy control method.

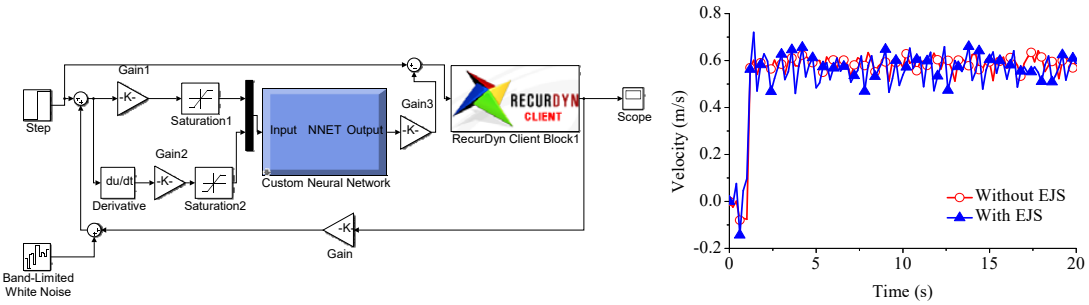


Figure 7: Co-simulation model interface and results of neural network control with dynamic model

In order to overcome the shortcomings of a single fuzzy control or a single neural control, a Fuzzy-Neural control method that incorporates the fuzzy control and the neural control was adopted and designed. The fuzzy Takagi-Sugeno (T-S) model that is more simple for calculation and better for mathematical analysis, was combined with an adaptive neural control systems, then an Adaptive Neural-Fuzzy Inference System (ANFIS) was presented for controlling the STV motion state in the paper. The critical step in this ANFIS architecture is to realize the self-learning and adaptive of the control parameter. Hybrid learning algorithm, namely, a combination of least-squares estimation and back-propagation, was developed for the parameter learning of the membership function.

The simulation model interface of the ANFIS collaborated with the dynamic model for the STV was presented in Fig. 8. It can be seen a desirable control effect can be obtained by using this control scheme. However, this hybrid algorithm method required more computation and analysis.

3.2 Trajectory tracking collaborative control simulations and comparisons

Actually, mechanics properties of the seafloor sediment are always uneven distribution, which will cause the differences of the shear forces and traction forces under two tracks. As a result, a turning moment will be generated, and make the STV move deviate from its predetermined

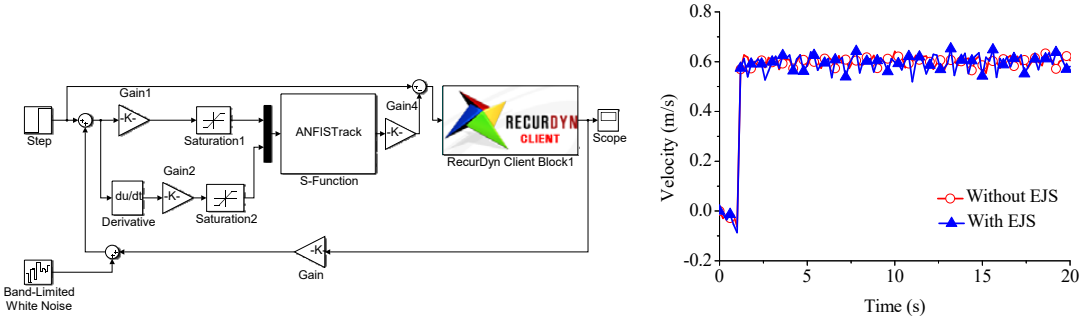


Figure 8: Co-simulation model interface and results of Neural-Fuzzy control with dynamic model

straight-line and circle trajectory. The shear strengths under the left track were set to 3.0 kPa and 4.0 kPa respectively, while the shear strength under the right track was set to a constant value 5 kPa. Fig.9 shows the simulation trajectories of the STV under these two conditions. While the green dashed lines both in Fig.9 and Fig.10 represent the predetermined straight-line paths. It can be seen the actual motion trajectories will deviate from its predetermined straight-line trajectory due to the uneven distribution of the sediment mechanics properties. With the longitudinal displacement increases, the deviations will continuously increase. According to the trajectory deviation control requirement for the STV, the deviation should within the allowable range of $-1m$ to $1m$. So, it is necessary to perform an effective control for the STV to keep its actual motion trajectory to tracking a predetermined path.

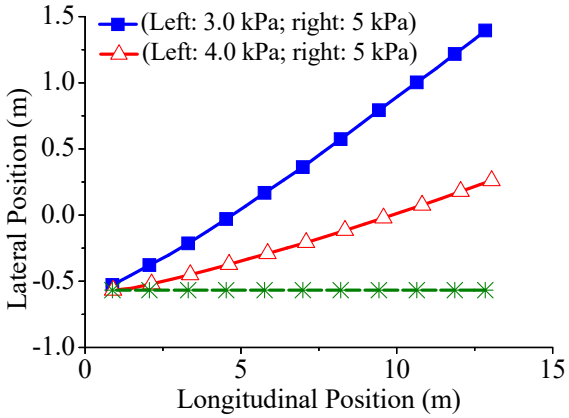


Figure 9: Straight-line motion trajectory deviation simulations of the STV

Straight-line motion trajectory control simulations comparisons between different controls

The above used fuzzy logic control, neural control and ANFIS control were carried out respectively for the trajectory control as shown in Fig. 11. It can be seen obviously that with the ANFIS control, the trajectory deviation is the minimum; when the longitudinal displacement is about 16 m, its lateral trajectory deviation is just about 0.01 m, which indicate the ANFIS control has a better effect for the STV compared to other control methods. If without an optimized or proper control for the STV, its lateral trajectory deviation will continuously enlarge along with its longitudinal motion.

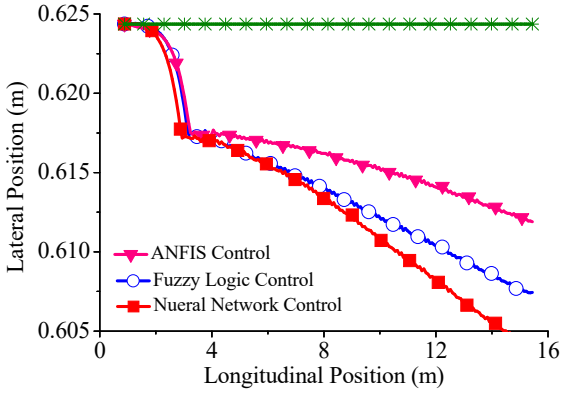


Figure 10: Straight-line motion trajectory control simulations comparisons between different controls

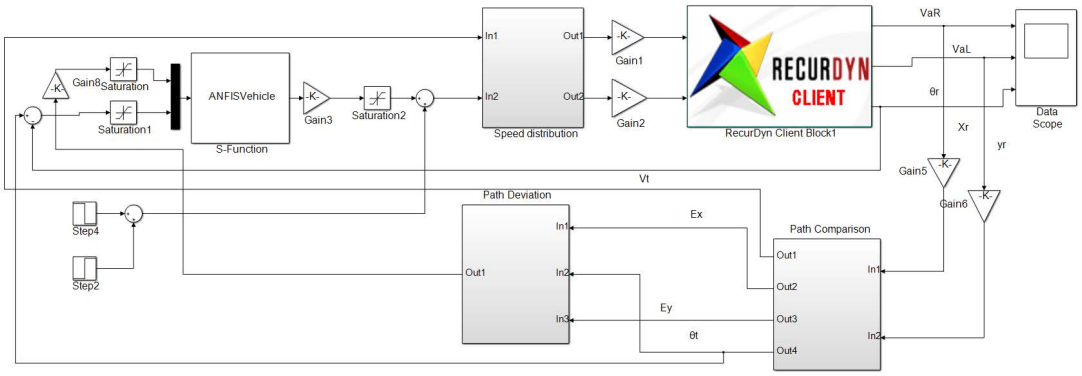


Figure 11: Combined path co-simulation model interface of the ANFIS control model with dynamic model

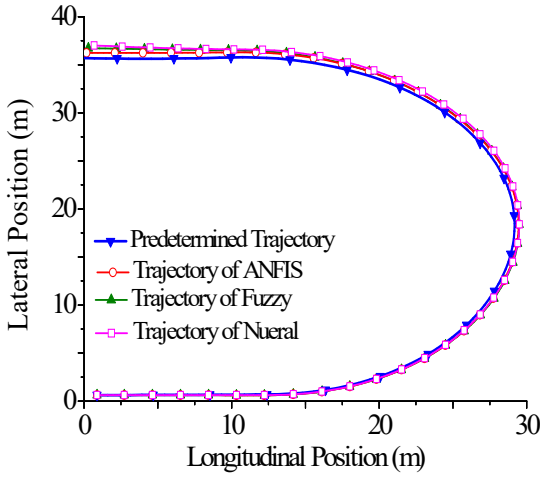


Figure 12: Simulation motion trajectories of the STV with different controls

A predetermined combined path including straight-line and turning paths was set for tracking simulation. The input velocity of the STV was set to 0.5 m/s; the turning velocity ratio was set to 1.2. The straight-line and turning motions time were set to 20 s and 90 s, respectively. The motion trajectories simulations between the fuzzy logic control, neural network control and ANFIS, were conducted and compared relative to the predetermined path in Fig. 12.

It can be seen that with the ANFIS control, the trajectory deviation for the STV relative to its predetermined path was the minimum compared to other control methods. For above one round of the predetermined path tracking, the maximum trajectory deviation under ANFIS control can be maintained around 0.5 m within the allowable range of -1 m to 1m. It can be predicted that with the continuous motions of the STV along with more rounds of above predetermined combined paths, the control effect of the ANFIS will be more obvious and efficient.

4 Conclusions

The major conclusions as follows can be drawn from this work.

(1) An improved multi-body dynamic model of a STV with integration of a dynamic terramechanics model of the particular sediment has been established and verified. A new collaborative simulation model of the STV integrating a mechanical multi-body dynamic model in RecurDyn/-Track and a control model in MATLAB/Simulink has been developed and co-simulations were achieved. Different control algorithms performances including a PID control, a fuzzy control and a neural control, have been compared and proved that the traditional or individual intelligent controls are not particularly suitable for the STV motion control.

(2) An adaptive neural-fuzzy inference system (ANFIS) control algorithm incorporating the fuzzy control and neural network control has been adopted and designed for the STV motion control. A straight-line path tracking controls for the STV by the fuzzy, neural network and ANFIS controls have performed and proved the ANFIS control algorithm can achieve a desired control performance for the STV compared to the other controls.

(3) A predetermined combined path, including the straight-line and turning paths, has been tracked for the STV by an ANFIS control algorithm compared to a fuzzy logic and neural network controls. The collaborative simulations have proved the ANFIS control method can achieve a better control effect among these control algorithms, with its actual maximum trajectory deviation can be maintained around 0.5 m within the permissible range.

Acknowledgment

This research is supported by the National Natural Science Foundation of China (Grand No. 51774324 and No. 51105386), the National Key Research and Development Program of China (Grand No. SQ2016YF010109). The authors would like to thank the comments from the anonymous reviewers that improved the quality of the paper.

Bibliography

[1] Barai, R.K.; Nonami, K. (2007); Optimal two-degree-of-freedom fuzzy control for locomotion control of a hydraulically actuated hexapod robot, Information Sciences, 177(8), 1892-1915, 2007.

[2] Bozic, M.; Ducic, N.; Djordjevic, G.; Slavkovic, R. (2017); Optimization of Wheg robot running with simulation of neuro-fuzzy control, *International Journal of Simulation Modelling*, 16(1), 19–30, 2017.

[3] Chen, C. C.; Li, J.S.; Luo, J.; Xie, S.R; Li, H.Y.; Pu, H.Y; Gu, J. (2016); Robust adaptive position and force tracking control strategy for door-opening behaviour, *International Journal of Simulation Modelling*, 15(3), 423–435, 2016.

[4] Cui, R.; Yang, C.; Li, Y. (2017); Adaptive neural network control of AUVs with control input nonlinearities using reinforcement learning, *IEEE Transactions on Systems Man & Cybernetics Systems*, 47(6), 1019–1029, 2017.

[5] Dai, Y.; Chen, L. S.; Zhu, X. (2016); Modelling and simulation of a mining machine excavating seabed massive sulfide deposits, *International Journal of Simulation Modelling*, 15(2), 377–387, 2016.

[6] Dai, Y.; Zhu, X.; Chen, L.S. (2016); A mechanical-hydraulic virtual prototype co-simulation model for a seabed remotely operated vehicle, *International Journal of Simulation Modelling*, 15(3), 532–541, 2016.

[7] Dai, Y.; Zhu, X.; Chen, L. S. (2015); A new multi-body dynamic model for seafloor miner and its trafficability evaluation, *International Journal of Simulation Modelling*, 14(4), 732–743, 2015.

[8] Dai, Y.; Chen, L. S.; Zhu, X. (2016); A New Multi-Body Dynamic Model of a Deep Ocean Mining Vehicle-Pipeline-Ship System and Its Integrated Motion Simulation, *Strojniški vestnik - Journal of Mechanical Engineering*, 62(6), 757–763, 2016.

[9] Dai, Y.; Liu, H.; Zhang, T.; Liu, S. J.; Li, Y. (2016); Three-Dimensional Coupled Dynamic Analysis of Deep Ocean Mining System, *Technical Gazette*, 23(4), 1037–1045, 2016.

[10] Han, Q. J.; Liu, S. J. (2015); Path tracking control algorithm of the deep sea tracked vehicle, *Journal of Central South University*, 46(2), 472–478, 2015.

[11] Han, Q. J.; Liu, S. J.; Dai, Y. (2011); Dynamic analysis and path tracking control of tracked underwater miner in working condition, *Journal of Central South University*, 42(2), 307–312, 2011.

[12] Herzog, K.; Schulte, E.; Atmanand, M. A. (2007); Slip control system for a deep-sea Mining machine, *IEEE Transaction on Automation Science and Engineering*, 4(2), 282–286, 2007.

[13] Kim, H.; Hong, S.; Choi, J. S. (2005); Dynamic analysis of underwater tracked vehicle on extremely soft soil by using Euler parameters, *Proceedings of the* 6^{th} *ISOPE Ocean Mining Symposium*, 141–148, 2005.

[14] Kim, H. W.; Hong, S.; Lee, C.H. (2011); Dynamic analysis of an articulated tracked vehicle on undulating and inclined ground, *Proceedings of the* 9^{th} *ISOPE Ocean Mining Symposium*, 97–103, 2011.

[15] Kim, H. W.; Lee, C. H.; Hong, S. (2013); Dynamic analysis of a tracked vehicle based on a subsystem synthesis method, *Proceedings of the* 10^{th} *ISOPE Ocean Mining Symposium*, 279–285, 2013.

[16] Lee, C. H.; Kim, H. W.; Hong, S. (2011); A study on the driving performance of a tracked vehicle on an inclined plane according to the position of buoyancy, *Proceedings of the 9th ISOPE Ocean Mining Symposium*, 104–109, 2011.

[17] Li, L.; Zhong, J. (2005); Research of China's pilot-miner in the mining system of polymetallic nodule, *Proceedings of the* 6^{th} *ISOPE Ocean Mining Symposium*, 124–131, 2005.

[18] Li, J. Z.; Liu, S. J.; Dai, Y. (2017); Effect of grouser height on tractive performance of tracked mining vehicle, *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, 39(7), 2459–2466, 2017.

[19] Li, L.; Zhang, M.; Shuang, Z. (2011); Moving control of seabed mining vehicle based on ANFIS, *Control Engineering of China*, 18(5), 660–663, 2011. (in Chinese)

[20] Li, L.; Zou, X. L. (2007); Seafloor robot's control on tracking automatically planning mining paths, *Journal of Mechanical Engineering*, 43(01), 152–7, 2007.

[21] Ngo, T. Q.; Phuong, T. V. (2015); Robust Adaptive Self-Organizing Wavelet Fuzzy CMAC Tracking Control for Deicing Robot Manipulator, *International Journal of Computers Communications & Control*, 10(4), 567–578, 2015.

[22] Ngo, T. Q.; Wang, Y. N.; Mai, T. L.; Nguyen, M. H.; Chen, J. (2012); Robust Adaptive Neural-Fuzzy Network Tracking Control for Robot Manipulator, *International Journal of Computers Communications & Control*, 7(2), 341–352, 2012.

[23] Sokolov, M.; Afanasyev, I.; Klimchik, A. (2017); HyperNEAT-based flipper control for a crawler robot motion in 3D simulation environment, *Proceedings of the IEEE International Conference on Robotics and Biomimetics*, 2017.

[24] Wang, S.; Gui, W. H.;Zang, T. (2005); Fuzzy And Predictive Control On the Deep-Sea Vehicle, *Transactions of the American Ophthalmological Society*, 9, 83–91, 2005.

[25] Wang, Y. N.; Mai, T. L.; Mao, J. X. (2014); Adaptive motion/force control strategy for non-holonomic mobile manipulator robot using recurrent fuzzy wavelet neural networks, *Engineering Applications of Artificial Intelligence*, 34(9), 137–153, 2014.

[26] Widyotriatmo, A.; Joelianto, E.; Prasdianto, A.; Bahtiar, H.; Nazaruddin, Y. Y. (2017); Implementation of Leader-Follower Formation Control of a Team of Nonholonomic Mobile Robots, *International Journal of Computers Communications & Control*, 12(6), 871–885, 2017.

[27] Wong, J. Y. (2010); Terramechanics and Off-Road Vehicle Engineering (Second Edition), Butterworth-Heinemann, 2010.

[28] Yeu, T. K.; Park, S. J.; Hong, S. (2006) ; Path Tracking using Vector Pursuit algorithm for tracked vehicles driving on the soft cohesive soil, *SICE-ICASE International Joint Conference*, 2781–2786, 2006.

[29] Yeu, T. K.; Yoon, S. K.; Park, S. J. (2011); Study on underwater navigation of crawler type mining robot, *Oceans 2011*, 7985(1): 1–6, 2011.

[30] Yoon, S. K.; Yeu, T. K.; Hong, S. (2014); Path tracking control test of underwater mining robot, *Oceans*, 1–4, 2014.

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 13(4), 477-491, August 2018.

An Improved ABC Algorithm for Energy Management of Microgrid

R. Gao, J. Wu, W. Hu, Y. Zhang

Ren Gao*
Hubei University of Economics
No. 8, Yangqiaohu Avenue, Jiangxia District, Wuhan, China
*Corresponding author: gr@hbue.edu.cn

Juebo Wu
1. Department of Geography, National University of Singapore
Arts Link, Singapore 117570
2. ZTE Corporation
No.55, Science and Technology South Road, Shenzhen, China

Wen Hu
Hubei University of Economics
No. 8, Yangqiaohu Avenue, Jiangxia District, Wuhan, China

Yun Zhang
Wuhan University
No. 299, Bayi Road, Wuchang District, Wuhan, China

Abstract: Microgrids are an ideal way of electricity generation, distribution, and regulation for customers by means of distributed energy resources on the community level. However, due to the randomness of photovoltaic and wind power generation, it is a crucial and difficult problem to achieve optimal economic dispatch in microgrids. In this paper, we present an optimal economic dispatch solution for a microgrid by the improved artificial bee colony (ABC) optimization, with the aim of satisfying load and balance demand while minimizing the cost of power generation and gas emission. Firstly, we construct a mathematical model according to different characteristics of distributed generation units and loads, and improve the performance of global convergence for ABC in order to fit such model. Secondly, we explore how to solve the optimal economic dispatch problem by the improved ABC and give the essential steps. Thirdly, we carry out several simulations and the results illustrate the benefits and effectiveness of the proposed approach for optimal economic dispatch in microgrid.
Keywords: Artificial Bee Colony (ABC), optimization, economic dispatch, microgrid, swarm intelligence.

1 Introduction

The energy crisis and environmental degradation is a global burning issue confronting humanity today, and the microgrid provides an effective way to solve this problem [12]. The microgrid is a micro system consisting of a set of distributed power supply, loads and energy storage systems and control devices, which can supply various forms of power (electricity, heat, etc.) to loads with high reliability. It is able to meet the electricity and heat demand through the output of distributed generation units and energy storage units, as well as improve the energy efficiency and reduce power generation cost and discharge system. At the same time, users in microgrid are free to choose the power supply mode by the optimization demand, and all these factors are the impetus for the development of smart grid [6]. However, there are still many problems on operation control, energy management and scheduling in microgrid. The diversity of distributed

generation units and flexibility of their composition makes its energy management become more complex. Thus, it is a challenging issue to achieve optimal economic dispatch in microgrid [11,20].

Significant research has been conducted and reported in the fields of economic dispatch for microgrids, related to the operation costs as well as minimizing emissions. Dynamic programming method was introduced to economic dispatch in microgrid, which obtained the global optimal solution by partitioning the complex problems into smaller stages [3, 18]. But the computation time and memory requirement will increase greatly when the dimension of stage, state and decision-making variable becomes bigger, that is, the dimension disorder. Compared with the traditional optimization algorithms, evolution algorithm is a kind of effective global search technology, which has been successfully applied in the areas of unit combination of electric power system and optimization scheduling, such as Genetic algorithm (GA) [2, 14], Differential Evolution (DE) [5,16], and Bacterial Foraging algorithm (BFA) [4,7]. Although evolution algorithm is a good solution to economic dispatch, it may lead to a long iterative time and poor convergence due to the presence of prematurity and random walk. By evolution algorithm, the searching process starts at a larger random state and ends in a small random state, in which this mechanism of combination search is easy to cause the algorithm converges into local optimal solution. In addition, swarm intelligence was also introduced to microgrid for economic dispatch, such as Particle Swarm Optimization (PSO) [1,19]. Swarm intelligence has the advantages of simple and easy to realize quick convergence and relying on few parameters, which has received more and more attention in the field of microgrid. In addition, some autonomous systems are developed to achieve management automatically, such as Cyber-Physical System [8].

Artificial bee colony optimization, as a specific application of swarm intelligence, has the ability to solve optimization issues by means of local optimization searching behavior of individual artificial bee with a fast convergence speed [15, 17]. The contribution of this paper is as below. 1) A mathematical model of optimal economic dispatch is established in a microgrid based on ABC, by considering time-of-use pricing strategy for both grid-connected and islanded microgrid. 2) The conventional ABC is improved in order to speed up the convergence and avoid local minimum. 3) Simulations are carried out to demonstrate the feasibility and benefits of the proposed approach.

The remainder of the paper is organized as follows. Section 2 improves the conventional ABC algorithm. The mathematical model and economic dispatch strategy in microgrid is presented in section 3. Section 4 describes the optimal economic dispatch solution for a microgrid by the improved ABC. Simulation and results analysis are performed in section 6. A brief conclusion is given as well as the future work.

2 Improvement of ABC

The goal of economic dispatch is to solve the optimal composition problem of distributed generation units in microgrid, in order to reduce the cost and emissions. But the traditional ABC is easy to fall into local optimum and with slow convergence. In this section, we firstly modify ABC by introducing the choice mechanism of the bee's neighborhood individual. Then, we improve the global convergence performance and dynamic regulation of sharing information among individual bees by integrating cross operation, mutation ability and greedy strategy.

2.1 Neighborhood factor

In ABC algorithm, the location update of food source for employed bees and onlookers is defined as:

$$l_{ij} = p_{ij} + r \times (p_{ij} - p_{kj}) \tag{1}$$

where i is the location of food source, $k \in \{1, 2, \ldots, M\}$, $j \in \{1, 2, \ldots, D\}$ and $k \neq j$. r is a random number within $[-1, 1]$. p_{ij} is the current position for the bee and p_{kj} is the food source position by random selection in the neighborhood individuals.

The selection probability for food source is defined as:

$$P_i = Fit_i / \sum_{i=1}^{M} Fit_i \tag{2}$$

where Fit_i means the higher earnings ratio is, the larger probability will be.

When the continuous search for food source by an individual is more than a certain number, a new food source is produced, defined as:

$$x_i^j = r \times (x_{max}^j - x_{min}^j) + x_{min}^j \tag{3}$$

where $j \in \{1, 2, \ldots, D\}$, r is a random number within $[-1, 1]$. x_{max} and x_{min} are the maximum and the minimum value respectively.

In ABC, the optimal solution is calculated by the food source update according to generating new food source from bees' neighborhood. The neighborhood individuals are chosen randomly without considering the relationship between the quality of food source in neighborhood and in current position. Meanwhile, the bee's food source doesn't combine with the search process in food source update. Therefore, the process control is insufficient in ABC algorithm, which causes the relatively slow speed of convergence.

Neighborhood factor can adjust search process dynamically according to the quality of individual food source in the neighborhood. The onlooker compares the quality of the current food source with the food sources in the neighborhood when it choosing the neighborhood individual. If the quality of the food source in the neighborhood is better, the food sharing information should be increased. On the contrary, the food sharing information will be increased.

The location update of food source for onlookers is re-defined as:

$$v_{ij} = \tau \times x_{ij} + \eta \times r \times (x_{ij} - x_{kj}) \tag{4}$$

where τ is forgetting factor, and it means the memory strength of the current food source to search the next food source. η is neighborhood factor that determines the strength of sharing information according to the quality of food sources in neighborhood. The forgetting factor is going down dynamically in the next search in order to make the bees make full use of the search information in neighborhood individuals and get the better global optimization results, defined as:

$$\tau = \lambda \times w_\tau \tag{5}$$

In order to get a better global optimization results in later search period, the neighborhood factor is changing from small to big, defined as:

$$\eta = \lambda \times w_\eta \tag{6}$$

λ is a constant in Eq. 5 and Eq. 6. $\lambda > 1$ when the quality of food source in the neighborhood is better than the current food source. Otherwise, $\lambda < 1$.

w_τ and w_η is changing dynamically along with the search process.

$$w_\tau = w_2 - ((iter_{max} - iter)/iter_{max})^\alpha \times (w_2 - w_1) \tag{7}$$

$$w_\eta = w_3 - ((iter_{max} - iter)/iter_{max})^\beta \times (w_4 - w_3) \tag{8}$$

where w_1, w_2, w_3, w_4 are constants limited within $[0.1, 1.5]$ and $w_2 > w_1$, $w_4 > w_3$. In forgetting factor τ, w_τ decreases from w_2 to w_1 along with the search process, which makes the bees move to the optimal food source area rapidly. To achieve quick convergence, the value of α is less than 1. Since the value is too small to global convergence, the value of α is usually set to $[0.8, 1]$.

In η, w_η increases from w_3 to w_4 along with the search process, which will reach the information sharing degree between the bee and its neighborhood individual. The value of β is bigger than 1 in order to avoid bees missing some optimal food source and limited within $[1, 1.2]$.

2.2 ABC combined with genetic algorithm

(1) Cross operation

Cross operation refers to the mutual exchange of genes between two pairs of chromosomes in certain way, so as to form two new individuals. Cross operator can enhance the global search ability in wide range with global parallelism. In ABC algorithm, role conversion mechanism is realized after bees starting searching the paths and the number of iterations is the same as the one of conversions. In general applications, less iterations can make the algorithm have strong ability in local search but with weak global search ability. By adding cross operator, it can make the ABC algorithm have better ability for global optimization. The detail of the process is as follows.

Step 1: Perform cross operation for all paths by combining any two paths together as P_i and P_j.

Step 2: Select a node position p randomly, and add the area (from P_j to the end) to the left. Delete the node that P_i has done cross operation with P_j.

Step 3: Do the same operation for P_j.

(2) Mutation ability

For ABC algorithm, there may be better solution or even the global optimal solution in the neighborhood of current solution. We add mutation factor to further realize global optimization operation for optimal economic dispatch in microgrid. Mutation operation is conducted for all paths and the procedure is as below.

$j \neq 1$: Select the jth food source and change the position j and $j - 1$.

$j = 1$, change the first food source with the last food source.

j is the food source position.

(3) Greedy strategy

During computing process, the worst path will be replaced by the optimal path in each population. There are two optimal paths being preserved for the employed bee in each iteration. For onlookers and scouts, greedy algorithm is adopted for cross operation. If the candidate solution is better than the current solution, then choose the candidate solution as optimal one. The bee doesn't record the path information of the current solution any longer. Otherwise, the path information doesn't need to be changed by the candidate solution. By greedy algorithm and cross operation, the ABC algorithm can be enhanced by sub-operator.

3 Mathematical model and economic dispatch strategy in microgrid

Fig. 1 shows the microgrid architecture studied in this paper, which encompasses wind turbine (WT), micro turbine (MT), photovoltaic (PV), fuel cell (FC), battery (BT), other storage components, general loads and essential loads with diverse features.

In this architecture, the connection point, called point of common coupling (PCC), is the bride between the utility system and the microgrid system. Thus, there are two kinds of modes

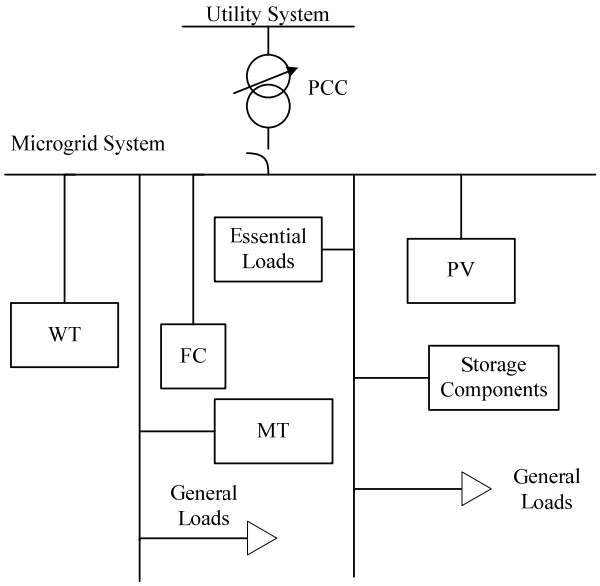


Figure 1: Typical architecture of microgrid system

in microgrid according to the state of PCC, that is, connected state (grid-connected) and disconnected state (islanded mode).

3.1 Mathematical model of economic dispatch

The aim of optimal economic dispatch in MicroGrid is to get the best composition of distributed generation units and reduce the consumption and emission at the same time. We present the mathematical model of economic dispatch by considering both grid-connected mode and islanded mode.

Proposed objective function

a) Grid-connected mode

In grid-connected mode, the microgrid can exchange electricity with the utility system through the PCC, so the selling and the purchasing behaviour are considered in order to get better economic dispatch. The objective function of operating cost is defined as:

$$F = \min \left\{ \sum_{k=1}^{T} \Delta t_k \left[\sum_{i=1}^{N} (CF_i + COM_i + PO_i) + CB_{i,k} - CS_{i,k} \right] \right\} \tag{9}$$

where T is the total operating time of MicroGrid, N is the total number of generating units and Δt_k is the duration of time interval k. CF_i means the fuel cost of the micro generation unit i while COM_i denotes the operation and maintenance cost of the micro generation unit i. PO_i is environmental cost generated by exhaust emissions. $CB_{i,k}$ stands for the purchased electricity of the micro generation unit i in Δt_k if the load demand goes beyond the generated power, and $CS_{i,k}$ implies the income from sold electricity of the micro generation unit i in Δt_k if the generated power is too much for the load demand.

The fuel cost of CF_i is computed by:

$$CF_i = KF_{i,k} \times P_{i,k} \tag{10}$$

where $KF_{i,k}$ is the fuel coefficient (\$/Kwh) of the micro generation unit i in Δt_k and $P_{i,k}$ is the generated power (Kwh) for the micro generation unit i in Δt_k.

The operation and maintenance cost of COM_i is calculated by:

$$COM_i = KOM_{i,k} \times P_{i,k} \tag{11}$$

where $KOM_{i,k}$ is the operation and maintenance coefficient (\$/Kwh) of the micro generation unit i in Δt_k and $P_{i,k}$ is the same meaning as in Eq. 10.

The environmental cost of PO_i is computed by:

$$PO_i = KR_{i,k} \times EP_{i,k} \times P_{i,k} \tag{12}$$

where $KR_{i,k}$ is emission coefficient, $EP_{i,k}$ is emission price and $P_{i,k}$ is the same meaning as in Eq. 10.

The purchased electricity of $CB_{i,k}$ is computed by:

$$CB_{i,k} = KB_{i,k} \times PB_{i,k} \tag{13}$$

where $KB_{i,k}$ is the buying coefficient (\$/Kwh) of the micro generation unit i in Δt_k and $PB_{i,k}$ is the bought power (Kwh) for the micro generation unit i in Δt_k.

The income from sold electricity of $CS_{i,k}$ is computed by:

$$CS_{i,k} = KS_{i,k} \times PS_{i,k} \tag{14}$$

where $KS_{i,k}$ is the buying coefficient (\$/Kwh) of the micro generation unit i in Δt_k and $PS_{i,k}$ is the income (Kwh) for sold by the micro generation unit i in Δt_k.

b) Islanded mode

Under this circumstance, the MicroGrid is disconnected to the utility system and controlled by MicroGrid itself as an islanded entity. No power is exchanged between MicroGrid and utility system through the PCC. Thus, the objective function of operating cost is defined as:

$$F = \min \left\{ \sum_{k=1}^{T} \Delta t_k \left[\sum_{i=1}^{N} (CF_i + COM_i + PO_i) \right] \right\} \tag{15}$$

where all the parameters are the same meanings as in Eq. 9 except removing the cost for purchasing power and the income for selling surplus power.

Equality constraints

a) Grid-connected mode

According to the charge-discharge of battery, there are two kinds of situations for power balance constraints, namely charging balance constraints and discharging balance constraints.

For charging battery, the power balance constraints are defined as:

$$P_{load} = \sum_{i=1}^{N} P_{i,k} - \alpha_{ch} P_{ch,k} + PB_{i,k} - PS_{i,k} \tag{16}$$

where P_{load} is the load power in MicroGrid, $P_{i,k}$ is the generated power (Kwh) for the micro generation unit i in k, $P_{ch,k}$ is the power for charging the battery, α_{ch} is charging efficiency coefficient, $PB_{i,k}$ and $PS_{i,k}$ are the same meanings as in Eq. 13 and Eq. 14.

For discharging battery, the power balance constraints are defined as:

$$P_{load} = \sum_{i=1}^{N} P_{i,k} - \alpha_{dis} P_{dis,k} + PB_{i,k} - PS_{i,k} \tag{17}$$

where all parameters are the same as in Eq. 16 except $P_{dis,k}$ is the power for discharging the battery and α_{dis} is discharging efficiency coefficient.

b) Islanded mode

In islanded mode, MicroGrid cannot exchange any power with utility system, so it doesn't need to consider purchasing power and selling power.

For charging battery, the power balance constraints are redefined as:

$$P_{load} = \sum_{i=1}^{N} P_{i,k} - \alpha_{ch} P_{ch,k} \tag{18}$$

where the parameters are the same meanings as in Eq. 16.

For discharging battery, the power balance constraints are redefined as:

$$P_{load} = \sum_{i=1}^{N} P_{i,k} - \alpha_{dis} P_{dis,k} \tag{19}$$

where the parameters are the same meanings as in Eq. 17.

Inequality constraints

The constraints of micro generation units, buying and selling electricity, and charging and discharging battery constitute the inequality constraints in MicroGrid.

a) Inequality constraints for micro generation units

$$P_{i,k}^{\min} \leq P_{i,k} \leq P_{i,k}^{\max} \tag{20}$$

where $P_{i,k}^{\min}$ and $P_{i,k}^{\max}$ are the minimum and the maximum operating power of the micro generation unit i.

b) Inequality constraints for buying and selling electricity

$$PB_{i,k}^{\min} \leq PB_{i,k} \leq PB_{i,k}^{\max} \tag{21}$$

$$PS_{i,k}^{\min} \leq PS_{i,k} \leq PS_{i,k}^{\max} \tag{22}$$

where $PB_{i,k}^{\min}$ and $PS_{i,k}^{\min}$ are the minimum buying and selling electricity from/to utility system while $PB_{i,k}^{\max}$ and $PS_{i,k}^{\max}$ are the maximum buying and selling electricity from/to utility system.

c) Inequality constraints for charging and discharging battery

$$P_{bt,k}^{\min} \leq P_{bt,k} \leq P_{bt,k}^{\max} \tag{23}$$

$$E_{bt,k}^{\min} \leq \left| E_{bt,0} - \sum_{k=1}^{j} P_{bt,k} T \right| \leq E_{bt,k}^{\max} \tag{24}$$

where $P_{bt,k}^{\min}$ and $P_{bt,k}^{\max}$ are the minimum and the maximum charging/discharging efficiency. $E_{bt,k}^{\min}$ and $E_{bt,k}^{\max}$ are the minimum and the maximum battery capacity.

3.2 Economic dispatch strategy

(1) Grid-connected strategy

In grid-connected microgrid, the economic dispatch problem should consider not only the power scheduling strategy of distributed generation units, but also the influence of the system performance of power trading between microgrid and external network.

a) Utilize WT and PV units as output at first, track and control the maximum power.

b) Determine whether to use MT for generating power according to actual load.

c) When the generated power by WT, PV and MT units is beyond the load demand, make battery storage discharge to supply the loads and monitor the state of battery charging and discharging.

d) If the energy from storage battery is enough to meet the load demands, increase the output by storage battery to sell the electricity to the utility system.

e) If the energy from storage battery is insufficient to satisfy the demands in microgrid, purchase the power from the utility system.

f) If the generating cost is higher than the price to purchase electricity from utility system, the microgrid buys the power from outside to meet internal load demand. And if the internal cost is smaller than external price, the microgrid generates the power and sells to utility system.

(2) Islanded mode strategy

In islanded microgrid, it doesn't need to take power trading into account and load demands are completely supplied by the internal micro sources and energy storage devices. The energy control system is responsible for real-time management and maintenance of system reliable operation. The economic dispatch is realized by the optimum combination of power supply.

a) Utilize WT and PV units as output at first.

b) When the generated power by WT, PV and MT units is beyond the load demand, turn off the units with high generating cost to achieve electricity balance.

c) When the generated power by WT, PV and MT units is not enough for the load demands, make battery storage discharge to supply the loads.

d) If the energy from storage battery is enough to meet the load demands, utilize all the distributed generation units and storage batteries together to meet residual load demands according to comprehensive generating cost.

e) Perform load shedding if the generated power is insufficient when all distributed generation units are running and the storage batteries are discharging.

4 Optimal economic dispatch solution for a microgrid by the improved ABC

Optimal economic dispatch in microgrids can be regarded as optimal decisions for multistage decision problems to get different combinations of distributed generation units in different periods [9, 21]. The ABC algorithm is a useful way to deal with multistage decision problems that can compare the performance of combination in each stage. By means of the idea in dynamic programming, we firstly explore the optimal economic dispatch solution based on the improved ABC and then present the key steps of algorithm.

4.1 Implementation of optimal economic dispatch

The ABC algorithm is mainly to solve the non-constraint problem, while the economic dispatch in microgrids is a kind of constraint optimization problem. The key point is to convert the parts of constraint optimization problem to non-constraint problem. The ABC algorithm

can solve the equality constraints of distributed generation units directly, and the necessary conversion has to do for the inequality constraints of distributed generation units.

In the proposed approach, the definition of path for optimal composition of distributed generation units is each state of unit combination in a time period constitutes for a decision set from time 1 to time T. So the problem of optimal economic dispatch is a multistage search problem, with the goal of finding a decision path with minimum total cost. The dynamic model is defined as:

$$F_t\left(U_t^l\right) = \min\left\{\varphi_t^l\left(U_{t-1}^k, U_t^l\right)\right\} \qquad t \in T \tag{25}$$

where U_t^l is the state l in the t period, $\varphi_t^l\left(U_{t-1}^k, U_t^l\right)$ is the accumulation of operating cost from the state k in the $t-1$ period to the state l in the t period. $F_t\left(U_t^l\right)$ is the minimum cost in the t period. The constraint conditions are followed the presented mathematical model of economic dispatch.

(1) Conversion of objective function

The objective function of operating cost is redefined as:

$$\min\left(\sum_{i=1}^{n-1} tc\left(s_{\pi(i)}, s_{\pi(i+1)}\right) + tc\left(s_{\pi(n)}, s_{\pi(1)}\right)\right) \tag{26}$$

where $tc(s_i, s_j)$ is the transfer cost from the state i to the state j, and $\pi(i)$ is the optional state set.

(2) Treatment of constraint conditions

We build *tabu* list to restrict the state that cannot meet some constraints. *tabu* is used to record all the states that allow to be transferred in the period t. There is no direct connection between any two periods.

If the state meets the demand, its element in *tabu* is set to 1. Otherwise, its element in *tabu* is set to 0. On one hand, the number of transfer can be restricted by *tabu*. On the other hand, the search behaviour can be run in a feasible solution, which plays certain guidance for bees search behaviours.

In the process of economic dispatch, when the output of a micro source is beyond its upper limit, other micro sources have to adjust their output to keep the system balance.

4.2 Key steps of algorithm

(1) Divide the loads into diverse classes, general loads and essential loads, ordered by the importance.
(2) Create composition states by different distributed generation. Compute the maximum and the minimum output for each state to generate the upper limit and lower limit.
(3) Compare states with loads. If lower limit of one state is bigger than the maximum loads or upper limit is smaller than the minimum loads, remove this state. Otherwise, record this state and its upper limit and lower limit.
(4) Determine the initial state.
(5) Initialize bee number, swarm evolution parameters, random food sources etc.
(6) Search objective food source by scouts and generate employed bees. If the current food source is better than the previous one, store it. Otherwise, ignore this food source.
(7) If the process reaching final time in the iteration, go to step 9. Otherwise, go to step 8.
(8) Compute list of the current period and the earnings rate by the proposed constraint processing methods. Calculate the output and generating cost at this state and update the path information. Go to step 6.

Table 1: Time-of-use pricing (RMB/Kwh)

Time	07:00-10:00	10:00-15:00	15:00-19:00	19:00-21:00	21:00-23:00	23:00-07:00
Usage	Mid peak	On-peak	Mid peak	On-peak	Mid peak	Off-peak
B-price	0.47	0.8	0.47	0.8	0.47	0.16
S-price	0.35	0.64	0.35	0.64	0.35	0.12

Table 2: Emission factor and external cost of NO_X, SO_2, CO_2

Type	EC (RMB/kg)	EF-FC (kg/MWh)	EF-MT (kg/MWh)
CO_2	0.023	489	724
NO_X	8	0.014	0.2
SO_2	6	0.0027	0.0036

(9) Record the shortest path from step 8 and update the path information for global optimization according to the optimization algorithm.
(10) If not reaching the end of iterations and not appearing stagnation phenomenon, go to step 6 for the next iteration. Otherwise, stop the process.

5 Simulations and results analysis

To demonstrate the feasibility and effectiveness of the proposed approach in optimal economic dispatch for microgrids, we carry out several case studies based on the real data, for both grid-connected and islanded microgrid. The basic data are chosen for the typical summer and winter days from the central region of china.

5.1 Testing environment

The collection of the basic data in this experiment includes typical day load curve in winter/summer, local energy data, and technical performance parameters for micro sources. In grid-connected mode, since the battery charging/discharging is only restricted by the performance of itself, we assume the battery charging and discharging have the same situations in winter and summer. The gas price is 2.05 RMB/m^3.

We adopt the time-of-use pricing here, which is classified into three groups: on-peak, mid-peak and off-peak. In different time period, the prices for selling and buying power to/from the utility system are also different, as shown in Table 1.

Because MT and FC power are relying on burning fuel, it is inevitable to produce atmospheric pollutants such as CO_2, NO_X, SO_2, and other solid dust particles. The handling cost of emissions is computed by the estimated external discount cost multiplied by the total generated power and the emission factor (EF). Table 2 gives the emission of FC and MT as well as external cost (EC) [10, 13].

5.2 Case study 1: Typical winter days

According to the proposed algorithm, the output power generated by micro sources is calculated for typical days in winter, also including the best economic dispatch for operation and optimal operation cost. In winter, the micro sources are controlled dynamically followed by the heating requirement.
(1) Grid-connected strategy

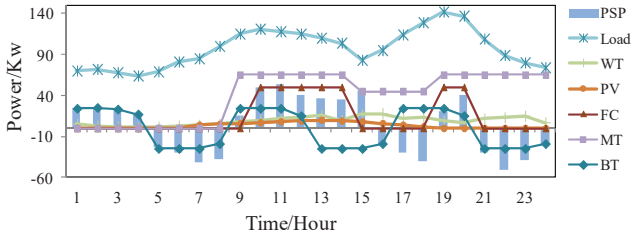


Figure 2: Power balance of grid-connected microgrid in typical winter days

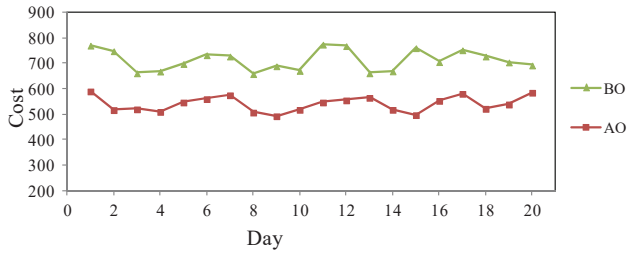


Figure 3: Comparison of total cost of daily power generation in the grid-connected MicroGrid

The power balance of the grid-connected microgrid in typical winter days is shown in Fig. 2, where PSP means the buying power (negative value) and the selling power (positive value) from/to the utility system.

It can be seen from the results that the microgrid bought the power from outside during the off-peak hours and sold the power to utility system during the peak hours, in order to get the best economic dispatch.

We carried out a comparison of total cost of daily power generation for no optimization and after optimization in winter days, in which the number of observed days is 20. The average cost is RMB 712.7 before optimization (BO) while it is RMB 540.95 after optimization (AO), as shown in Fig. 3.

(2) Islanded mode

The power balance of the islanded microgrid in typical winter days is shown in Fig. 4 and the related system cost is given in Fig. 5, including comprehensive cost (C_Cost), fuel cost (F_Cost), operation and maintenance cost (OM_Cost), and environmental cost (E_Cost).

5.3 Case study 2: Typical summer days

In summer, the micro sources are controlled dynamically followed by the cooling requirement.

(1) Grid-connected strategy

The power balance of the grid-connected microgrid in typical summer days is shown in Fig. 6.

It can be seen from the results that the MT and FC run together in peak hours because the power price is higher than it generated by micro sources. The microgrid system purchased the power from outside for supplying loads and battery charging in off-peak time.

The average cost of daily power generation of the grid-connected MicroGrid is RMB 1007.05 before optimization (BO) while it is RMB 810.95 after optimization (AO) in summer, as shown in Fig. 7.

(2) Islanded mode

The power balance of the islanded microgrid in typical summer days is shown in Fig. 8 and the related system cost is given in Fig. 9.

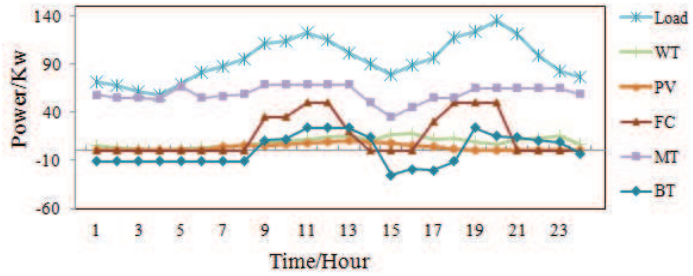


Figure 4: Power balance of islanded microgrid in typical winter days

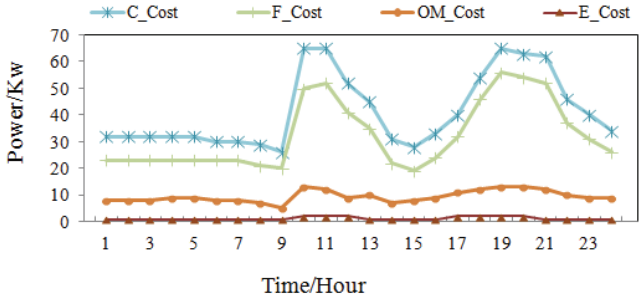


Figure 5: System cost of islanded microgrid in typical winter days

5.4 Results analysis

The simulation results showed that the proposed approach can achieve the optimal economic dispatch for grid-connected and islanded microgrid. In grid-connected mode, the microgrid system used the generated power firstly in peak time in order to reduce purchasing power from the utility system, and it also sold the surplus electricity to the utility system. In mid-peak or off-peak hours, it bought the electricity from the utility system as much as possible so as to reduce the cost of generating power. In winter, the characteristics of charging and discharging storage battery had great impact on electric power trade. In summer, since the full running time of MT is longer than it in winter and the efficiency of PV is better than in winter, the purchasing power from outside system is obviously decreased. The decision factors of buying or selling power in grid-connected microgrid is the generating cost of micro sources, outputs and power price in the utility system.

In islanded mode, it can be seen from the results the load shedding basically happened from 16:00 to 21:00 when the battery needed charging and the power generated by WT was decreasing. We can also learn that the fuel cost occupied most of the total cost. The second expense was operation and maintenance cost, and the environmental cost was minimum. By the influence of battery charging and discharging process, the high cost of the system has occurred in battery charging process. Therefore, to reduce the comprehensive cost in islanded microgrid, the key is to find a better way for flexible and effective scheduling of the storage battery.

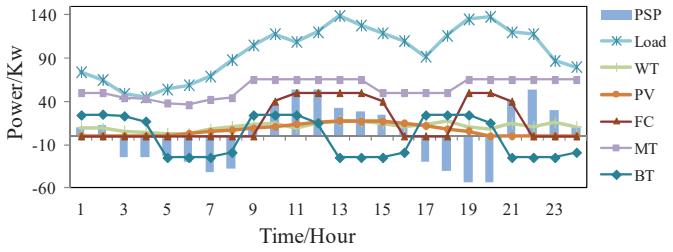


Figure 6: Power balance of grid-connected microgrid in typical summer days

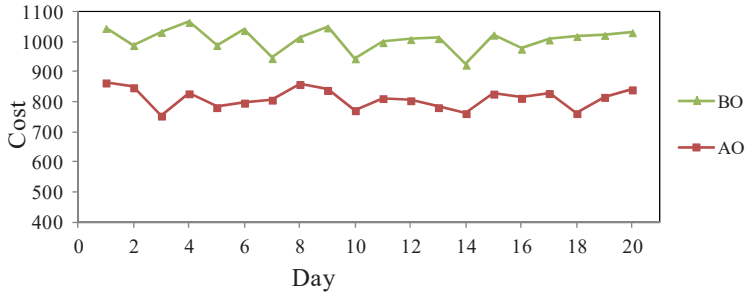


Figure 7: Comparison of total cost of daily power generation in the grid-connected MicroGrid

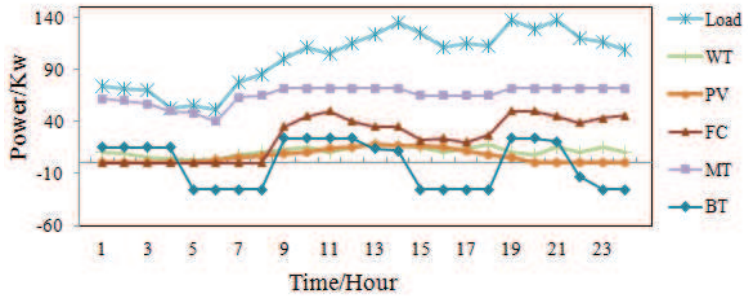


Figure 8: Power balance of islanded microgrid in typical summer days

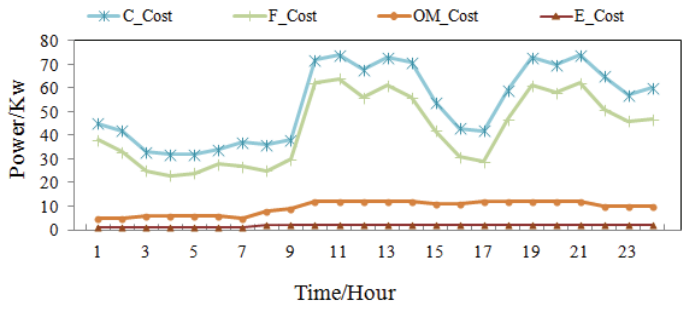


Figure 9: System cost of islanded microgrid in typical summer days

6 Conclusions and future work

Microgrids are the component of the Smart Grid revolution offering a number of important advantages in reducing energy consumption and environmental pollution, and improving power system reliability and flexibility, which changes the way to deal with the increasingly growth of loads. In this paper, we presented a novel approach of optimal economic dispatch in a microgrid with the improved ABC, including grid-connected and islanded mode. In order to overcome the problems of slow convergence and easy to fall into local optimum in traditional ABC, we improved it by introducing neighborhood factor, cross operation, mutation ability and greedy strategy. We constructed the mathematical model of optimal economic dispatch and also gave the economic dispatch rules for a microgrid both in grid-connected and islanded mode. The ABC-based optimal economic dispatch solution was discussed and the detail steps were given. Simulations of typical cases were carried out and the results showed that the improved ABC-based approach is feasible and effective in optimal economic dispatch for microgrids.

Future work will focus on exploring hybrid swarm intelligence algorithms to economic dispatch in microgrid, such as Altruism algorithm and Cuckoo search algorithm. Moreover, flexible switching strategies for real-time scheduling by considering more constraint conditions in microgrid would be a promising direction of future research.

Acknowledgments

This paper is supported by the Natural Science Foundation of Hubei Province (2016CFB208, 2017CFB773).

Bibliography

[1] Abid, S.; Zafar, A. (2017); Managing Energy in Smart Homes Using Binary Particle Swarm Optimization, *Conference on Complex, Intelligent, and Software Intensive Systems*, pp.189-196, 2017.

[2] Ahmad, M.; Khan, A. et al. (2017); A Hybrid Genetic Based on Harmony Search Method to Schedule Electric Tasks in Smart Home, *International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*, pp.154-166, 2017.

[3] Al-Saadi, M.; Luk, P.; Economou, J. (2017); Integration of the Demand Side Management with Active and Reactive Power Economic Dispatch of Microgrids, *International Conference on Intelligent Computing for Sustainable Energy and Environment*, pp.653-664, 2017.

[4] Ali, W.; Rehman, A. et al. (2017); Home Energy Management Using Social Spider and Bacterial Foraging Algorithm, *International Conference on Network-Based Information Systems*, pp.245-256, 2017.

[5] Azeem, M.; Amin, S. (2017); Scheduling of Appliances in Home Energy Management System Using Elephant Herding Optimization and Enhanced Differential Evolution, *International Conference on Intelligent Networking and Collaborative Systems*, 132-142, 2017.

[6] Banerjee, B.; Jayaweera, D.; Islam, S. (2016); Micro Grid Planning and Operation, *Smart Power Systems and Renewable Energy System Integration*, 29-47, 2016.

[7] Batool, S.; Khalid, A. et al. (2017); Pigeon Inspired Optimization and Bacterial Foraging Optimization for Home Energy Management, *Advances on Broad-Band Wireless Computing, Communication and Applications*, 14-24, 2017.

[8] Chen, B.; Yang, Z.; Huang S. et al. (2017); Cyber-Physical System Enabled Nearby Traffic Flow Modelling for Autonomous Vehicles, *36th IEEE International Performance Computing and Communications Conference, Special Session on Cyber Physical Systems: Security, Computing, and Performance (IPCCC-CPS 2017)*, 2017.

[9] Das, V.; Karuppanan, P. et al. (2017); Energy Grid Management, Optimization and Economic Analysis of Microgrid, *Smart Energy Grid Design for Island Countries*, 289-325, 2017.

[10] Hongze, L.; Sen, G.; Bao, W. (2011); Analysis of sensitivity of the environmental value of wind power, *Energy Procedia 5*, 2576-2580, 2011.

[11] Hu, B.; Wang, H.; Yao, S. (2017); Optimal economic operation of isolated community microgrid incorporating temperature controlling devices, *Protection and Control of Modern Power Systems*, 2-6, 2017.

[12] Jadav, K. A.; Karkar, H. M.; Trivedi, I. N. (2017); A Review of Microgrid Architectures and Control Strategy, *Journal of The Institution of Engineers (India): Series B*, 98(6), 591-598, 2017.

[13] Jansen, R.; Karki, R. (2017); Sustainable Energy Optimization in a Smart Microgrid, *Sustainable Power Systems*, 111-132, 2017.

[14] Khan, S.; Khan, A. et al. (2017); Genetic Algorithm and Earthworm Optimization Algorithm for Energy Management in Smart Grid, *International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*, 447-459, 2017.

[15] Kremljak, Z.; Palcic, I.; Kafol, C. (2014); Project Evaluation Using Cost-Time Investment Simulation, *International Journal of Simulation Modelling*, 13(4), 447-457, 2014

[16] Reddy, S. S.; Park, J. Y.; Jung, C. M. (2016); Optimal operation of microgrid using hybrid differential evolution and harmony search algorithm, *Frontiers in Energy*, 10(3), 355-362, 2016.

[17] Sharma, T. K.; Pant, M.; Singh, V. P. (2011); Artificial Bee Colony Algorithm with Self Adaptive Colony Size, *International Conference on Swarm, Evolutionary, and Memetic Computing*, 593-600, 2011.

[18] Trivedi, I. N.; Jangir, P.; Bhoye, M.; Jangir, N. (2016); An economic load dispatch and multiple environmental dispatch problem solution with microgrids using interior search algorithm, *Neural Computing and Applications*, 1-17, 2016.

[19] Xu, L. Z.; Yang, G. Y. et al. (2017); A Coordinated Heat and Electricity Dispatching Model for Microgrid Operation via PSO, *Life System Modeling and Intelligent Computing*, 213-219, 2017.

[20] Zhang, D.; Liu, S.; Papageorgiou, L. G. (2016); Energy Management of Smart Homes with Microgrid, *Advances in Energy Systems Engineering*, 507-533, 2016.

[21] Zupancic, D.; Buchmeister, B.; Aljaz, T. (2017); Reducing the Time of Task Execution with Existing Resources ¨C Comparison of Approaches, *International Journal of Simulation Modelling*, 16(3), 484-496, 2017

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 13(4), 492-502, August 2018.

An Optimized DBN-based Coronary Heart Disease Risk Prediction

K. Lim, B.M. Lee, U. Kang, Y. Lee

Kahyun Lim[1]*, Byung Mun Lee[2], Ungu Kang[2], Youngho Lee[2]
1. Department of IT Convergence Engineering; 2. Department of Computer Engineering
Gachon University, Republic of Korea
1342 Seongnamdaero, Sujeong-gu, Seongnam-si, Gyeonggi-do, Korea
bmlee@gachon.ac.kr, ugkang@gachon.ac.kr, lyh@gachon.ac.kr
*Corresponding author: lyh@gachon.ac.kr

Abstract: Coronary Heart Disease (CHD) is the world's leading cause of death according to a World Health Organization (WHO) report. Despite the evolution of modern medical technology, the mortality rate of CHD has increased. Nevertheless, patients often do not realize they have CHD until their condition is serious due to the complexity, high cost, and the side effects of the diagnosis process. Thus, research on predicting CHD risk has been conducted. The Framingham study is a widely-accepted study in this field. However, one of its limitations is its overestimation of risk, which threatens its accuracy. Therefore, this study suggests a more advanced CHD risk prediction algorithm based on Optimized-DBN (Deep Belief Network). Optimized-DBN is an algorithm to improve performance by overcoming the limitations of the existing DBN. DBN does not have the global optimum values for number of layers and nodes, which affects research results. We overcame this limitation by combining with a genetic algorithm. The result of genetic algorithm for deriving the number of layers and nodes of Optimized-DBN for CHD prediction was 2 layers, 5 and 7 nodes to each layers. The accuracy of the CHD prediction algorithm based on Optimized-DBN which is developed by applying results of genetic algorithm was 0.8924, which is better than Framingham's 0.5015 and DBN's 0.7507. In the case of specificity, Optimized-DBN based CHD prediction was 0.7440, which was slightly lower than 0.8208 of existing DBN, but better than Framingham's 0.65. In the case of sensitivity, Optimized-DBN is 0.8549, which is better than Framingham 0.4429 and DBN 0.7468. AUC of suggesting algorithm was 0.762, which was much better than Framingham 0.547 and DBN 0.570.
Keywords: Artificial Neural Networks (ANN), Deep Belief Network (DBN), Coronary Heart Disease (CHD), computational intelligence, genetic algorithm, CHD prediction.

1 Introduction

Coronary Heart Disease (CHD) is a disease in which a waxy substance called plaque builds up inside the coronary arteries [11]. It has been a leading cause of death globally for 15 years according to a World Health Organization report [1]. Once CHD is developed, it is nearly impossible to cure completely and patients are at risk of sudden death due to sudden myocardial infarction or etc. Therefore, the accurate risk prediction of CHD is very important in that it gives patients a chance to live better life by caring themselves more cautiously [6].
With this reason, there have been many researches to develop the method for predicting the risk of CHD. Among these, the Framingham Heart Study which was developed through prospective cohort study and the US Adult Treatment Panel (ATP) are widely used as standards when predicting the risk of CHD [15]. And as Artificial Neural Networks (ANN) are becoming increasingly popular because of its outstanding classifying performance, it is begin to be adopted into establishing the risk prediction model for CHD [20]. Research on developing a CHD risk

prediction model based on various ANN algorithms such as Random Forest (RF), Support Vector Machine (SVM), and Deep Belief Network (DBN) have been actively conducted. Although the risk prediction model based on ANN algorithms boasts a much better performance than the previous model, nevertheless, there is still room for improvement [9]. Therefore, these days, to develop a more improved version of prediction model, an ensemble technique which means combination of two different algorithms has been attempted [21]. The performance depends on how to combine two algorithms and if those are able to complement each other's disadvantages, it can significantly improve the prediction performance. However, developing CHD risk prediction model by combining two ANN algorithms have not tried much yet.

Among many ANN algorithms, Deep Belief Network is one of the most widely-used ANN algorithm because it has advantage in overfitting issue compared to other ANN algorithms [5]. Overfitting is one of the bothersome issue to most of the ANN algorithm researchers because it degrades prediction performance when applied to actual field [14]. Through regulating the parameter values of hidden or visible layers, DBN can be able to solve the overfitting problem to a certain degree. Though, DBN also has a limitation that there is no optimal value adequate to all datasets. The optimal values of each parameters are differed depending on datasets. Thus, researchers set values randomly and try several times, and select the values which show the best performance. However, there is no conviction that it is the optimal value set.

Therefore, in this study, our goal was to develop an CHD risk prediction model based on Optimized-DBN by combining DBN with Genetic Algorithm (GA). GA is an optimization algorithm which is useful for finding the optimal value. We utilized the GA to find the optimal value of the DBN parameter values of hidden and visible layers which is adequate to the dataset to predict the risk of CHD.

In section 2, we explained background knowledge about DBN and GA which are needed to understand the Optimized-DBN algorithm we suggest through this paper. In section 3, we described process and methodology of research to develop and verifying Optimized-DBN. The results of research such as performance comparison with the traditional DBN and other ANN algorithms which are widely being used now are recorded in section 4. Lastly, we concluded the paper with suggestion of future works in section 5.

2 Backgrounds

2.1 Deep belief network (DBN)

Deep learning refers to a collection of algorithms that pursues the improvement of performance by setting multiple hidden layers and passing information to those hidden layers several times. In 2006 Jeffrey Hinton, a professor at the University of Toronto who is called the master of artificial intelligence, first proposed the DBN in the paper "A fast learning algorithm for deep belief nets" [7]. With this paper, he proved that a DBN can greatly improve existing artificial neural networks, which take a lot of time to optimize and perform worse through the unsupervised pre-learning method. A DBN consists of a restricted Boltzmann machine (RBM) and artificial neural network and the RBM is composed of visible neurons and hidden neurons. Once the RBM obtains the training result of the input data, it uses the result as the input data for the next layer. It accumulates hidden neurons and learns [16]. The energy function of RBM is as in the following equation (1).

$$E(V,H) = -\sum_{i,j} v_i h_j - \sum_{i \in v} a_i v_i - \sum_{j \in H} b_j h_j \quad (1)$$

In equation (1), v_i and v_j are the binary values of the visible layer of i and the hidden layer

of j. In addition, h_i and h_j mean the bias of the visible layer of i and the hidden layer of j. W means a weighting matrix. Therefore, the energy function of the RBM and the Boltzmann machine without bias is the same as the following equation (2).

$$E(V,H) = -\frac{1}{2}\sum_{i,j} w_{ij} v_i h_j = -\frac{1}{2} V^T W H \quad (2)$$

In this case, the network V and H vectors follow the probability of the following equation (3).

$$p(V,H) = \frac{1}{Z^e} - E(V,H) \quad (3)$$

Z is the partition function, and p satisfies a probability value between 0 and 1. The updating of the weights uses k-step contrastive divergence (CD-k). $\langle v_i h_j \rangle$ data is the mean value of the input patterns of the visible layer of i and the hidden layer of j, and $\langle v_i h_j \rangle$ model represents the average value of the network [3].

The DBN sets the initial weights by performing unsupervised pre-training before performing the error-propagation algorithm [9]. After the initial weight value is determined, it performs an error back-propagation algorithm with the supervised learning method and performs fine-tuning to optimized performance [22]. However, research on the optimum value of the number of layers and nodes is still lacking, especially when performing pre-training through RBM.

The number of layers and nodes is a variable that influences research results, but there is a limit to the global optimal value because the optimal value depends on the type and characteristics of the dataset to be learned. Most studies using DBN have been conducted by researchers to set several values randomly and apply values which showed the best performance. However, this method has limitations because it is difficult to confirm whether its result is the optimum result. Our study overcomes this problem of DBN by applying a genetic algorithm.

A genetic algorithm (GA) is a method to find the optimum value. GA simulates a mathematical evolutionary method using crossover, mutation and selection. In other words, GA can be used to generate an optimal value of the number of layers and nodes of the DBN to maximize its performance. Through a genetic algorithm, we improve the performance of the CHD risk prediction by finding the optimal number of layers and nodes for the data through a genetic algorithm and applying it to the DBN.

2.2 Genetic algorithm (GA)

Genetic algorithms were created by John Holland [8], a professor of computer science and psychology at the University of Michigan. This is an algorithm in which the initial population of artificial chromosomes is reproduced through crossover and mutation, and in the process, chromosomes with a low fitness value are extinguished and chromosomes with a high fitness value survive. Survived chromosomes thus dominate the population. A genetic algorithm is an optimization algorithm performed in this form [13]. The genetic algorithm assigns the rank of each object through the fitness function and sets as many genes as the researcher set. It transmits the characteristics of genes having high fitness by conveying those to the next generation. After that, the next generation of chromosomes is generated through a crossover process. The crossover process has the effect of accelerating the convergence speed to its optimal value. Finally, through the mutation process which changes chromosomes at a certain probability, the global search effect is maximized in this stage, converging toward its optimum value. The genetic algorithm is based on the schema theorem. The schema is a set of strings consisting of 0, 1, and asterisks (*), and the genetic algorithm is executed by adjusting these schema. When the reproduction rate in the

genetic algorithm is proportional to the fitness, the probability that a particular schema (H) will survive the next generation can be predicted by the following equation (4) [19].

$$P_H^{(C)} = 1 - P_c \left(\frac{l_d}{l-1} \right) \quad (4)$$

The probability that schema H will survive after the mutation is as following equation (5). The genetic algorithm maximizes the global search effect by the mutation process.

$$P_H^{(m)} = (1 - P_m)^n \quad (5)$$

It is possible to get the probability of the schema growing according to the above two expressions, The formula is as following equation (6)

$$m_H (i+1) = \frac{\hat{f}_H (i)}{\hat{f} (i)} m_H (i) \left[1 - p_c \left(\frac{l_d}{l-1} \right) \right] (1 - p_m)^n \quad (6)$$

In this study, a genetic algorithm was used to derive the optimal DBN layer and node number values. The initial population size, N, was set to 256, Pc was set to 0.7, Pm was set to 0.001, and the termination condition was set to 200 generations following regular settings. The sigmoid function was also used as a fitness function.

3 Optimized DBN based coronary heart disease risk prediction

3.1 Experimental design

The purpose of this study was to develop and validate a predictive algorithm for coronary heart disease risk based on an optimized-DBN. The optimized-DBN improves performance by applying optimal values for the layer and node number derived by genetic algorithm to DBN. We developed a predictive algorithm for CHD risk through CHD risk factor data and verified it using a confusion matrix and ROC curve. The research method for this paper is shown in Figure 1. The training data was applied to the RBM, and when the RBM configuration was completed, the DBN was constructed through the error propagation algorithm. Then, the optimum value of the number of nodes and layers of the DBN was derived through the genetic algorithm, and the optimized result was applied to generate the optimized-DBN. The performance of the optimized-DBN based CHD risk prediction was verified by the confusion matrix and the ROC curve. And we utilized Matlab as a toolbox for developing ANN model and R 3.0 to visualize results.

3.2 Data

The optimized-DBN based CHD risk prediction proposed by this study predicts the risk CHD of patients through relevant variables. Data from the 6th National Health and Nutrition Examination Survey (KNHANES) was used for training and validation [12]. The KNHANES is a national health and nutrition survey conducted by the Korea Center for Disease Control and Prevention (KCDC) to identify the status of and trends in people's health and nutritional status [10]. This data includes the CHD risk factor data and CHD incidence data for each subject, which is useful for training and testing for the CHD risk prediction [17]. This study also utilized the latest data from the National Health and Nutrition Examination Survey of Korea. Variables for predicting the risk of CHD were as in the Framingham study. Eight factors including gender, age, systolic blood pressure (SBP), diastolic blood pressure (DBP), total cholesterol (TCL), high density cholesterol (HDL), obesity, and smoking were set as predictors of CHD. We defined

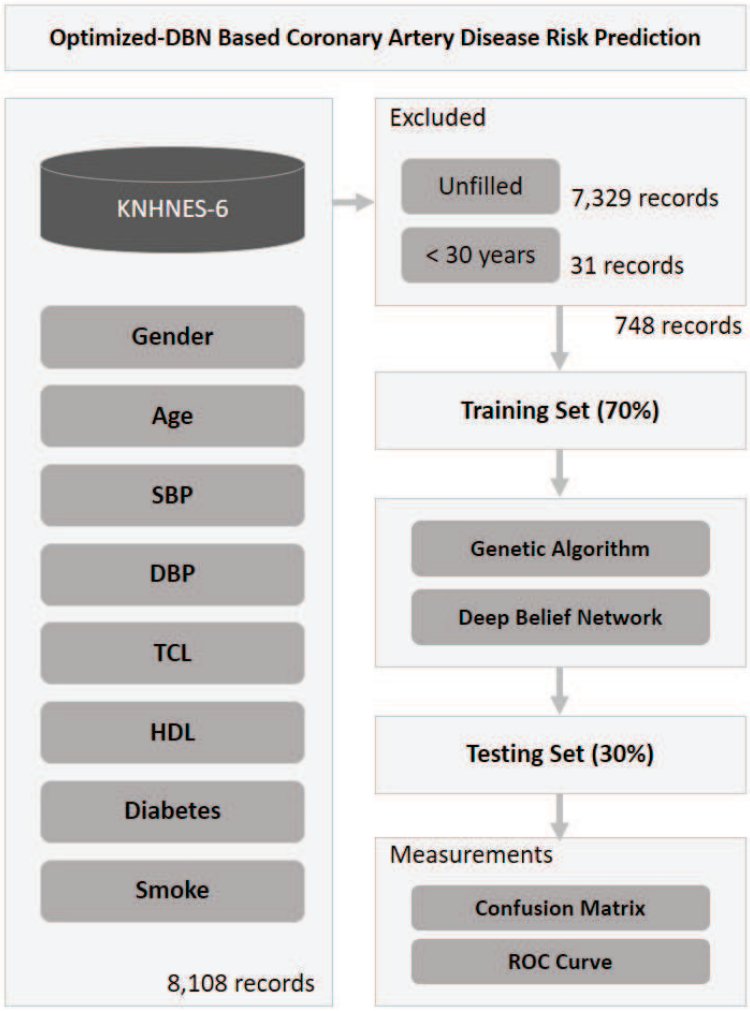


Figure 1: Experimental design for developing optimized-DBN based coronary heart disease risk prediction

age, total cholesterol (TCL), high density cholesterol (HDL), systolic blood pressure (SBP), and diastolic blood pressure (DBP) as continuous variables and gender, smoking, and obesity as nominal variables. The above 8 variables were extracted from KNHANES-6 data.

A total of 8,108 patients were enrolled in the KNHANES-6 study. 7,329 patients were excluded from the study because of the lack of data on risk factors for CHD suggested by the Framingham study or the presence of CHD. Furthermore, as in the Framingham study, data for subjects under the age of 30 was excluded. The remaining 748 data were used for analysis and prediction. In addition, 70% of the data was randomly extracted and used as training data for the creation of the optimized-DBN. The remaining 30% of the data was used to evaluate the performance of the optimized-DBN.

Of the KNHANES-6 data, 748 data points met the conditions of the study. Therefore, this data was used for research. A randomly chosen 70% of this training data was applied to the RBM first. The RBM uses the training result as training data for the next layer. In this way, the RBM is constructed by stacking hidden layers. Once the RBM configuration is complete, the DBN is configured via the error propagation algorithm. In this process, the sigmoid function is used as the activation function. A genetic algorithm is then performed to derive the optimum value of the number of nodes and layers of the DBN. The initial population size (N) of the

	Low Risk (288 patients)		High Risk (260 patients)	
	Mean	SD	Mean	SD
AGE	50.11	14.102	53.06	13.775
TCL	206.69	39.587	201.40	39.569
HDL	43.99	9.569	43.54	9.947
SBP	123.78	16.040	123.11	15.696
DBP	79.56	11.200	78.50	11.066
GENDER	M 302, W 186		M 135, W 107	
SMOKING	Y 318, N 170		Y 169, N 91	
DIABETES	Y 463, N 25		Y 196, N 64	

Figure 2: Summary of data from Korea National Health and Nutrition Examination Survey according to Framingham Risk Score Guidelines

genetic algorithm was set to 256, the possibility of crossover (Pc) was set to 0.7, the possibility of mutation (Pm) was set to 0.001, and the termination condition was set to 200 generations. After the genetic algorithm is implemented, the optimal value of the layer and node values with the lowest error rate is selected, and this value is applied to generate Optimized-DBN. When the error rate gradually decreases over the generations and converges toward its lowest value, the RBM performs learning for CHD risk prediction. As a result of the genetic algorithm, the number of layers was 2 and the number of nodes was 5 and 7 nodes to each layers, and this value was applied to the optimized-DBN configuration. After the optimized-DBN is constructed, an error backpropagation algorithm is performed, performance is improved through fine-tuning, and the optimized-DBN configuration is completed.

3.3 Performance measure

In this study, confusion matrix and receiver operating characteristic (ROC) curves were used as performance evaluation indexes. The confusion matrix can be used to assess the performance of the proposed algorithm by comparing the predicted risks with actual values. TP (true positive) means that an actual CHD risk patient is correctly predicted as a CHD risk patient. TN (true negative) means that patients with low CHD risk are correctly predicted as those with low CHD risk. These two indicators are values that are precisely classified [18]. FP (false positive) means that a person with a low CHD risk is misdiagnosed as a CHD risk patient. False negative (FN) means that patients with high CHD risk are incorrectly classified as those with low CHD risk. These two indicators are not well classified [2]. The ROC curve is a graph showing how sensitivity and specificity are related to each other. In this study, the ROC curve was used to show the accuracy of the optimized-DBN based CHD risk prediction. Sensitivity means the probability that the prediction is correct when the prediction algorithm assumes that the CHD risk is high. It is a measure of how accurately the proposed algorithm predicts patients with high CHD risk. Specificity means the probability that the prediction is correct when the prediction algorithm determines that the CHD risk for a patient is low. It is a measure of how accurately the proposed algorithm predicts patients with low CHD risk. The ROC curve can be interpreted

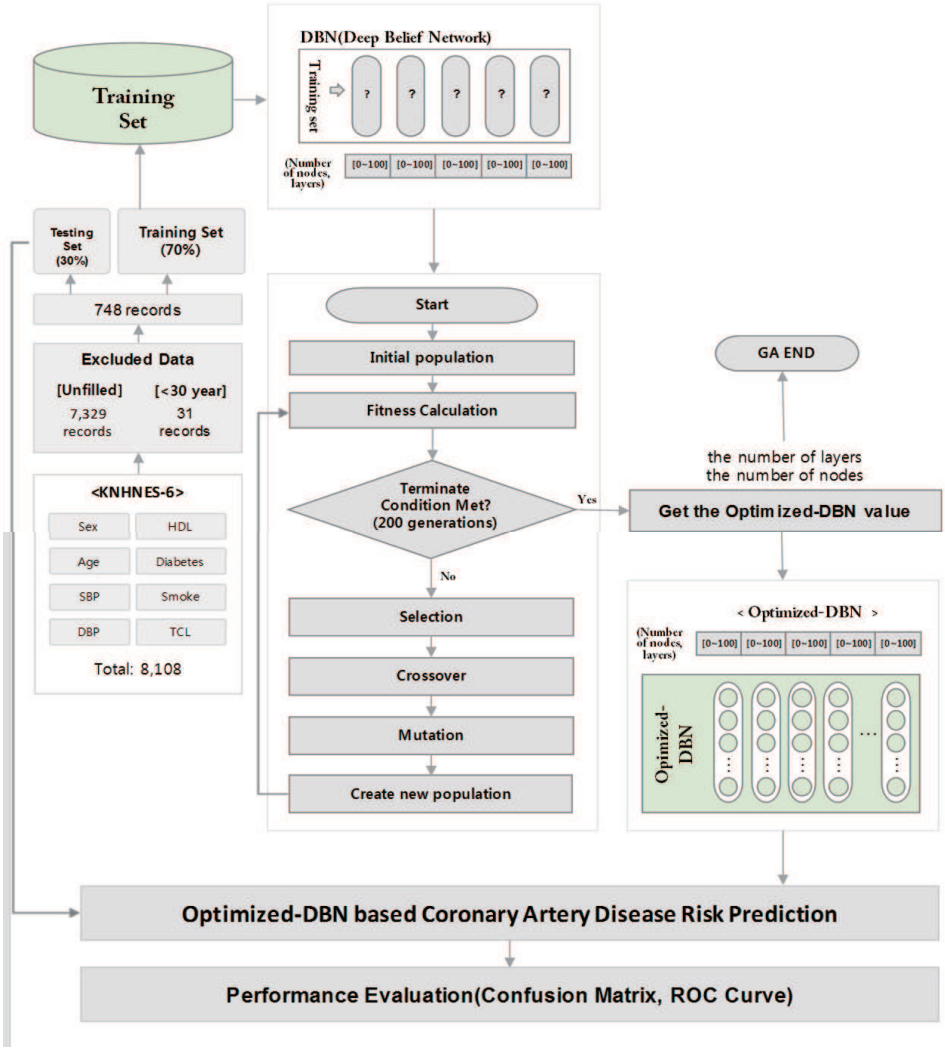


Figure 3: Process for developing optimized DBN based coronary heart disease risk predictions

as the proposed algorithm's accuracy increasing as the graph converges to 1. In other words, as the area under the curve (AUC), which is the area under the line of the ROC curve graph, approaches 1, it can be estimated that the accuracy of the prediction algorithm is high.

4 Results

4.1 Genetic algorithm results

Our results show that the proposed CHD risk prediction algorithm performs better than the existing CHD risk prediction algorithm. The proposed CHD risk prediction algorithm is based on DBN, which is an artificial neural network algorithm. We overcome the limitations of DBN by using a genetic algorithm. After constructing the RBM through the training data and finishing the DBN configuration, the genetic algorithm was applied to derive the optimal value of the number of nodes and layers. The initial population (N) of the genetic algorithm was set to 256, the possibility of crossover (Pc) was set to 0.7, the possibility of mutation (Pm) was set to 0.001, and the termination condition was set to 200 generations. As shown in Figure 4, the error rate gradually decreased over the generations. The error rate, which was close to the initial 3.8,

decreased as the number of generations increased. It can be seen that the error rate converges to 0.3033 near the termination condition of 200 generations. After the genetic algorithm was terminated, the layer and node values with the lowest error rate were selected as the optimal solution. The value was 2 layers and 5 and 7 nodes to each layers. After applying these values to the DBN, the optimized-DBN generation was completed by supervised learning.

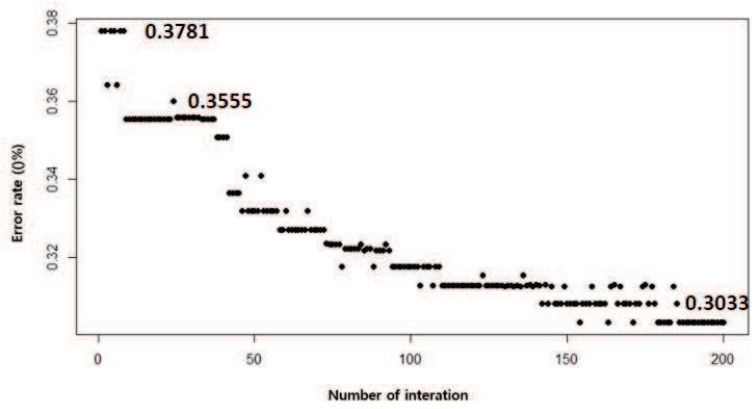


Figure 4: Error rate of genetic algorithm to develop optimized-DBN

• Sensitivity : 0.855 • Specificity : 0.744 • Accuracy : 0.829		Actual result / Classification	
		High (1)	Low (0)
Predictive Result / Classification	High (1)	TP 837	FP 75
	Low (0)	FN 142	TN 218

Figure 5: Confusion Matrix of optimized-DBN based CHD risk prediction

The confusion matrix of the optimized-DBN based CHD risk prediction is shown in Figure 5. TP, which means that the actual CHD risk patient is correctly predicted as a CHD risk patient, was 837; TN, which means that patients with low CHD risk are correctly predicted as those with low CHD risk, was 218; FP, which means that a person with a low CHD risk is misdiagnosed as a CHD risk patient, was 75; and FN, which means that patients with high CHD risk are incorrectly classified as those with low CHD risk, was 218.

The sensitivity of the optimized-DBN was 0.855, Specificity was 0.744 and accuracy was 0.829. This was confirmed to be superior to the results of the Framingham study, which showed sensitivity as 0.4430, specificity as 0.65, and accuracy as 0.5015. In addition, it performed well when compared with other classifiers. In the case of sensitivity, all of the classifiers are generally high. In particular, the Bolzmann Perceptron Network is slightly higher than suggesting algorithm. But the performance of the proposed algorithm is relatively superior to BPN in other factors such as Specificity, Accuracy and AUC. Even in case of specificity, the performance of the proposed algorithm is superior to other classifiers, but slightly lower than DBN's 0.8209. However, suggesting algorithm is also superior to DBN in other factors. In the case of Accuracy and AUC, it is confirmed that the proposed algorithm shows much better performance than all other classifiers. The performance comparison between the proposed optimized-DBN and other

Table 1: Performance comparison by classifiers

Algorithm	Sensitivity	Accuracy	AUC	
NB	0.8482	0.6385	0.7917	0.736
LR	0.8283	0.6962	0.8003	0.716
BPN	0.8654	0.6211	0.7909	0.701
RF	0.8219	0.6144	0.7720	0.696
DBN	0.7469	0.8209	0.7508	0.570
Framingham	0.4430	0.65	0.5016	0.548
Optimized DBN	0.8550	0.7440	0.8294	0.762

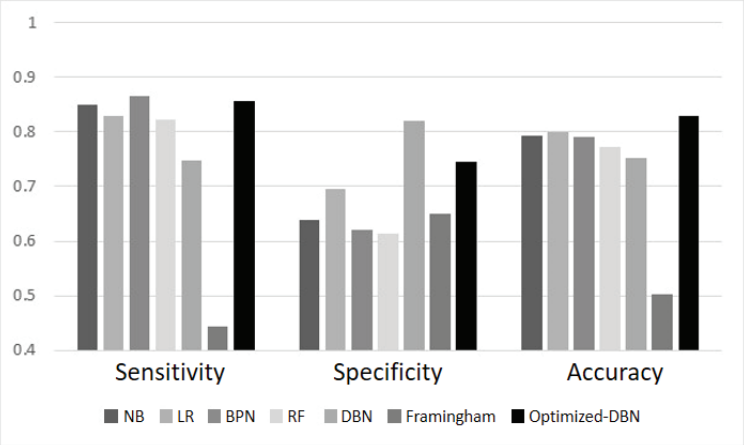


Figure 6: Comparison graph to evaluate the optimized-DBN based CHD risk prediction

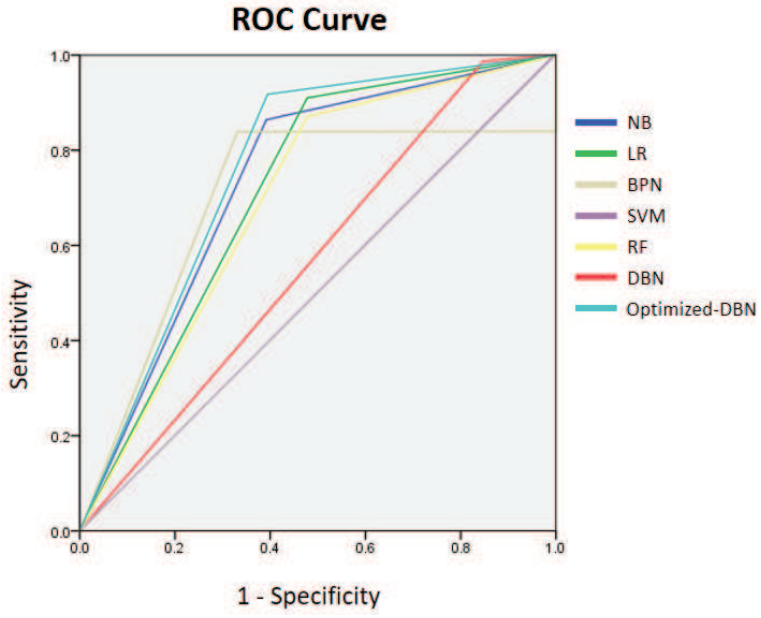


Figure 7: Comparison graph to evaluate the optimized-DBN based CHD risk prediction

prediction algorithms is shown in the figure 6 and figure 7.

The AUC of the optimized-DBN based CHD prediction was 0.762. The Framingham AUC

was 0.546, which confirms that the optimized-DBN based CHD prediction was more accurate. In addition, it performed well when compared with other classifiers such as the support vector machine (SVM) and random forest(RF) which shows 0.501 and 0.696. More precise results are listed the figure 6.

5 Conclusions and future work

The purpose of this study is to develop the Optimized-DBN, a more advanced ANN based prediction model, by combining two ANN algorithms, DBN and GA. And we validate its performance through predicting CHD risk. The results show the improved performance of Optimized-DBN when comparing to not only the conventional DBN but also other major ANN algorithms and Famingham risk score based prediction model which is widely used in healthcare system. This study demonstrates the use of a GA as a tool to draw the optimal parameter values for setting the DBN classifier to improve the performance. It is very useful since it can be applied regardless of datasets because the Optimized-DBN is able to bring the optimal value sets out which are tailored to the data set.

This study will contribute to helping not only medical staffs to make decisions but also patients to prevent CHD by realizing their risk in advance. And it is also able to be applied to any other diseases, if there is enough patient medical data to train and validate the prediction model. However, since this study used only the national health data of Republic of Korea, to more stable and reliable verification, more precise, objective evaluation through further samples is needed. Furthermore, despite its superiority, still, the prediction accuracy is not perfect so we should keep working to develop a more improved way of classification and prediction.

Acknowledgements

This research was supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2018-2017-0-01630) supervised by the IITP(Institute for Information & Communications Technology Promotion).

Bibliography

[1] Blackwell, D. L.; Lucas, J. W. (2014); Summary health statistics for U.S. adults - national health interview survey, *Vital Health statistics*, 10(260), 1–161, 2014.

[2] Fielding, A. H. (1997); A review of methods for the assessment of prediction errors in conservation presence/absence models *Environmental conservation*, 24(1) 38–49, 1997.

[3] Freeman J. A.; Skapura, D. M. (1991); *Algorithms, Applications, and Programming Techniques*, Addison-Wesley Publishing Company, 1991.

[4] Hecht-Nielsen, R. (1992) Theory of the Backpropagation Neural Network *Neural Networks for Perception*, 65–93, 1992.

[5] Hinton, G. E.; Osindero, S.; Teh, Y.-W. (2006); A fast learning algorithm for deep belief nets *Neural computation*, 18(7), 1527–1554, 2006.

[6] Eom, J.-H.; Rhee, J.-K. (2006); AptaCDSS-A Cardiovascular Disease Level Prediction and Clinical Decision Support System using Aptamer Biochip, *Korean Institute of Information Scientists and Engineers*, 33, 28–32, 2006.

[7] Hinton, G.E.; Osindero S.; Teh, Y.W. (2006); A fast learning algorithm for deep belief nets, *Neural computing*, 18(7), 1527-1554, 2006.

[8] Holland J.H. (1984); Genetic Algorithms and Adaptation, In: Selfridge O.G., Rissland E.L., Arbib M.A. (eds), *Adaptive Control of Ill-Defined Systems. NATO Conference Series (II Systems Science)*, Springer, Boston, MA, 16, 317-333, 1984.

[9] Ki, S.K.; Lee, S.M. (2014); Voice Activity Detection based on DBN using the Likelihood Ratio *Journal of Rehabilitation Welfare Engineering& Assistive Technology*, 8(3), 145–150, 2014.

[10] Korea Center for Disease Control and Prevention (2013); *Guidelines for using raw data of Korean National Health and Nutrition Examination Survey - the first survey of the sixth phase (KNHANES VI-1)*, Ministry of Health and Welfare, 2013.

[11] Korea Encyclopedia Research Center (1996); *Korea Encyclopedia Research Center: Nursary Encyclopedia*, Korea Encyclopedia Research Center, 1996.

[12] Korea National Health and Nutrition Examination Survey (2013); [Online]. Available: https://knhanes.cdc.go.kr/knhanes/

[13] Lewis, P.O. (1998); A genetic algorithm for maximum-likelihood phylogeny inference using nucleotide sequence data *Molecular Biology and Evolution*, 15(3), 277–283, 1998.

[14] Mohamed, A.R.; Dahl, G.E.; Hinton, G. (2011); Acoustic modeling using deep belief networks *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1), 14–22, 2011.

[15] National Institutes of Health (2001); *NIH: National Cholesterol Education Program ATP III Guidelinesr*, United States National Institutes of Health, 2001.

[16] Liu, N.; Jiang-ming Kan, J.-M. (2016); Improved Deep Belief Networks and Multi-Feature Fusion for Leaf Identification *Neurocomputing*, 216, 460–467, 2016.

[17] Park, R.W. (2017); Sharing Clinical Big Data While Protecting Confidentiality and Security: Observational Health Data Sciences and Informatics *Healthcare Informatics Research*, 23(1), 1–3, 2017.

[18] Townsend, J. T. (1974); Theoretical analysis of an alphabetic confusion matrix *Perception & Psychophysic*, 9(1), 40–50.

[19] Whitley, D. (1994); A genetic algorithm tutorial *Statistics and computing*, 4(2), 65–85, 1994.

[20] Wulsin, D.F.; Gupta, J. R.; Mani, R.; Blanco, J. A. (2011); Modeling electroencephalography waveforms with semi-supervised deep belief nets fast classification and anomaly measurement *Journal of neural engineering*, 8(3), 036015, 2011.

[21] Yan, X.; Chao, T.; Tu, K.; Zhang, Y. (2007); Improving the prediction of human microRNA target genes by using ensemble algorithm *Federation of European Biochemical Societies*, 581(8), 1586–1593, 2007.

[22] You, H.; Koo, M.-M.; Yi, K.; Nam, K. (2016); The Frequency based Study of the Applicability of DBN Algorithm on Language Acquisition Modeling *The Korean Journal of Cognitive and Biological Psychology*, 28 (4), 617–651, 2016.

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 13(4), 503-520, August 2018.

A Simulation based Analysis of an Multi Objective Diffusive Load Balancing Algorithm

I.D. Mironescu, L. Vinţan

Ion Dan Mironescu*
Lucian Blaga University of Sibiu, Romania
*Corresponding author: ion.mironescu@ulbsibiu.ro

Lucian Vinţan
Lucian Blaga University of Sibiu, Romania
lucian.vintan@ulbsibiu.ro

Abstract: In this paper, we presented a further development of our research on developing an optimal software-hardware mapping framework. We used the Petri Net model of the complete hardware and software High Performance Computing (HPC) system running a Computational Fluid Dynamics (CFD) application, to simulate the behaviour of the proposed diffusive two level multi-objective load-balancing algorithm. We developed an meta-heuristic algorithm for generating an approximation of the Pareto-optimal set to be used as reference. The simulations showed the advantages of this algorithm over other diffusive algorithms: reduced computational and communication overhead and robustness due to low dependence on uncertain data. The algorithm also had the capacity to handle unpredictable events as a load increase due to domain refinement or loss of a computation resource due to malfunction.
Keywords: Petri Net simulation, High Performance Computing, load balancing, diffusive algorithm, multi-objective optimisation

1 Introduction

1.1 The problem

The main problem approached through our research is how we can optimally distribute the computational effort on an application among the multiple processing units (PUs) of a High Performance Computing (HPC) machine. Computational Fluid Dynamics (CFD) applications [8], which are the focus of our research, perform alternatively intensive computation and communication (data exchange) steps for a high number of points of a discretized computational domain. The hardware on which these applications run is a collection of computing nodes connected through a hierarchical communication network. Each node has a heterogeneous collection of processing units with different processing speed communicating through a hierarchy of global and local memories. Shorter computation time, lower energy consumption and an equilibrated load are all important for the application user and for the hardware operator. Therefore, this represents a multi-objective optimization problem.

1.2 State of the art

Taking into account that the computation platform and the application have many parameters, each one with many distinct discrete values, the load-balancing problem is a NP-hard one [7]. This type of application needs a large-scale distributed computing system. As the underlying numerical method is an approximation method, the computational load can be known completely only at computation time. To handle this problem's particularities, modern solutions use metaheuristic [3], distributed [16] and dynamic load balancer [17]. The scale of the system

and the heuristic behaviour of the used algorithm make modelling and simulation indispensable in the development and assessment of a load balancer algorithm. The modern tools for modelling and simulation of computing systems are using the discrete event formalism [5] [9] [22]. From the current existing methods, Petri Nets presents the advantage of having graphical representation and sound formal definition [2]. The graphical representation facilitates rapid development of models and expressive simulation. The formal definition allows formal verification of the resulted models.

2 The proposed high performance computing system

In this research, we continue and complement our work presented in [20]. The hardware we consider in our approach is a cluster of heterogeneous nodes, implementing a High Performance Computer System (HPC). For the simulations, we modelled the nodes of this HPC after those of an existing system, the COKA cluster [25]. A node from COKA has two CPU sockets and eight PCIe slots. The CPU sockets are occupied by Intel Xeon E5 2630v3 CPUs with 8 cores. Nvidia K80 GPU boards, each having two GPU processors, occupy the PCIe slots. The CPUs and GPUs are communicating through the PCIe bus. Four GPU cards are sharing the same bus to a CPU socket. Each node has two network interfaces – one on each socket; they connect each node to an Infiniband (IB) interconnection network with a fat tree topology. We selected this model because performance and energy consumption models and measurements were presented in [4] for the same class of application we use.
The software runtime allowing the transparent and distributed use of the hardware resources has a two level architecture. The internode level is implemented in the CHARM++ framework [14]. The CHARM++ runtime supports the transparent, asynchronous, message based communication among chares distributed on the system nodes. Chares are the basic entities at this level and follow the Actor paradigm. The heterogeneous node level is managed with the support of the StarPU framework [1]. In this framework, the application is represented as a Directed Acyclic Graph of tasks. The tasks are dispatched to the processing units by a scheduler component that uses a scheduling/load balancing algorithm. Required memory transfers and dependencies between tasks are managed transparently.
The application we designed implements the Lattice Boltzmann Method (LB) CFD numerical method [8]. The computation in LB is an iterative process. In each iteration, for each point of the discrete computational space, two consecutive computational steps are performed: collision (only local data to the point is used) and *propagation* (data from neighbours is needed). The distributed character of the computation is implemented through domain decomposition. At the upper level (CHARM++) the workload is distributed / redistributed among the nodes. The global domain is divided in subdomains. Each subdomain is distributed to a chare, which will manage it until the end of computations. The chare divides the subdomain in blocks, creates for each the corresponding tasks and feeds them to the lower level (StarPU); this dispatches the task on PU and executes them.

3 Proposed load balancer

The proposed load balancing algorithm described in [20] and schematically presented in figure 1 has two levels, corresponding to the software architecture levels. At the CHARM++ level, the load distribution is performed by the chares. In the first iteration, one of the chares receives the domain and starts, as initiator, a negotiation process with other nodes; the negotiations are following the Contract Net Protocol [24]. In each negotiation round the initiator selects its

neighbours from the nodes bidding to its announcement; the initiator then sends to its neighbours bigger subdomains to be further redistributed. The process continues recursively with each responder becoming initiator until no node responds to the call. The resulting neighbourhood relation will be kept until the computation ends and each node will communicate only with its neighbours implementing thus the diffusive character of the algorithm [12]. After receiving its subdomain, each node's chare decomposes it in blocks and creates tasks for the next level. At the StarPU level, the local scheduler dispatches the tasks received from the node's chare to the queues of the node's PUs.

We developed a scheduler based on the dequeue model data aware sorted decision (dmdasd) scheduler [15] with some adaptations. The scheduler is using a cost function representing a weighted sum of the three optimization objectives. The function has the following form:

$$f_{obj} = \alpha \cdot t_{computation} + \beta \cdot E_{computation} + \gamma \cdot Load \tag{1}$$

where f_{obj} is the scheduling function, $t_{computation}$ is the estimated computation time , $E_{computation}$ is the estimated energy consumption, $Load$ is the estimated queue load. The weighting coefficients must satisfy the normalisation (equilibrium) relation $\alpha+\beta+\gamma = 1$. The scheduler estimates the values of the objective function for each PU; it sends the task for computing the block to the PU with the lowest value of the objective function. Our variant includes a supplementary border queue, priority based dispatching and work stealing. Exterior blocks are placed in the border queue based on assigned priority, but they can be transferred to another node if this becomes idle and requests ghost border zones [18]. When all PUs become idle, StarPU signals to the node's chare the end of computation for the current iteration. Then, the chare demands the ghost border zones from its neighbours and its frequency is lowered. If the neighbour responds with a ghost border zone, the computation continues normally; if it responds with a bigger memory zone, the load is redistributed and the neighbour frequency is raised. The receiving chare integrates the new memory zone in its subdomain. If the neighbour is not responding in a pre-set time, it is considered malfunctioning; its subdomain is recovered from backup by its neighbours and recomputed.

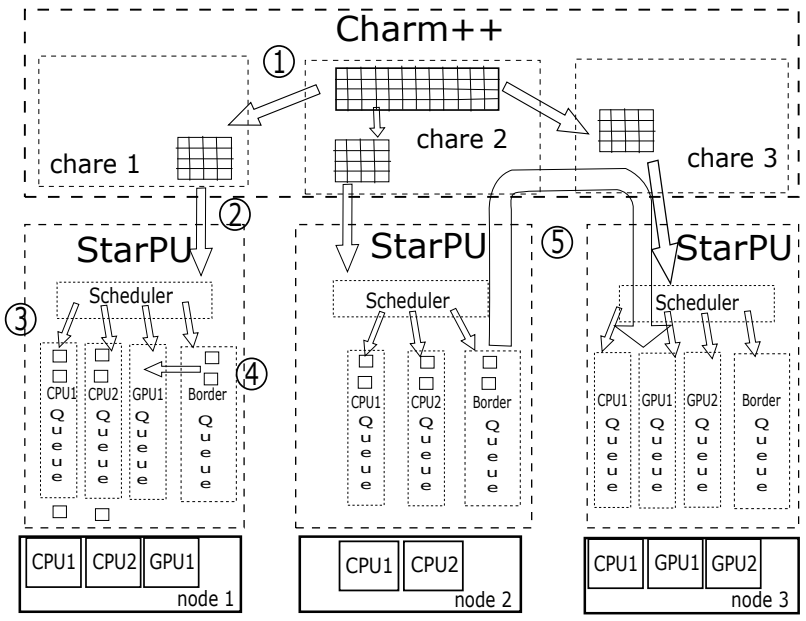


Figure 1: Scheme of the load balancing two level architecture: 1) distribution; 2) task creation; 3) scheduling; 4) work stealing; 5) redistribution

4 Modelling of the system

4.1 Underlying mathematical model

The execution of the HPC application on the given hardware architecture was modelled as a combination of atomic computation and communication steps. These steps can be performed concurrently on the available units. We adopted from literature the models for the time and the energy needed to perform these steps on a given PU [4] [21]. The computation is divided in two phases: *collision* and *propagation*. On the CPU, the unit optimally processed by each core is the 1024x1024 sites block. The time of performing the computation for this number of sites in the *collision* phase (t_{col_CPU}, in s) depends on the clock's frequency (f, in MHz) through the formula:

$$t_{col\_CPU}(f) = \frac{1380}{f}, \tag{2}$$

The energy used in this phase (E_{col_CPU}, in J) is related to the clock's frequency (f, in MHz) through the following relations:

$$E_{col\_CPU}(f) = P_{col\_CPU}(f) \cdot t_{col\_CPU}(f) \tag{3}$$

$$P_{col\_CPU}(f) = 30 + 0.0062 \cdot f^{1+0.2}, [W] \tag{4}$$

The time needed to perform the *propagation* phase for the same block (t_{prop_CPU}) is independent of the frequency:

$$t_{prop\_CPU}(f) = 0.15s \tag{5}$$

The energy used by the CPU in the *propagation* phase E_{prop_CPU} is related to the clock's frequency (f, in MHz) through the following relations:

$$E_{prop\_CPU}(f) = P_{prop\_CPU}(f) \cdot t_{prop\_CPU} \tag{6}$$

$$P_{prop\_CPU}(f) = 30 + 0.055 \cdot f^{1+0.2}, [W] \tag{7}$$

The GPU processes in one step a block of 8192x1024 sites. The time needed by the GPU in the *collision* phase (t_{col_GPU}, in s) depends on the clock's frequency (f, 'in MHz) by the expression:

$$t_{col\_GPU}(f) = \begin{cases} \frac{71.4}{f} & f < 650MHz \\ 0.033 & f \geq 650MHz \end{cases} \tag{8}$$

The energy used in this phase (E_{col_GPU}, in J) is given by the following formulas:

$$E_{col\_GPU}(f) = P_{col\_GPU}(f) \cdot t_{col\_GPU}(f) \tag{9}$$

$$P_{col\_GPU}(f) = \begin{cases} 42.5 + 0.109f & f < 650MHz \\ 42.5 + 0.109f + +0.005e^{0.0099f} & f \geq 650MHz \end{cases} \tag{10}$$

where (P_{col_GPU}) is the power (in W) required for the *collision* step on the GPU. For the *propagation* phase on the GPU, the time needed (t_{prop_GPU}, in s) is:

$$t_{prop\_GPU}(f) = \begin{cases} \frac{27.2}{f} & f < 800MHz \\ 0.085 & f \geq 800MHz \end{cases} \tag{11}$$

The energy used for the *propagation* phase (E_{prop_GPU}, in J) is calculated on the basis of the power required (P_{prop_GPU}, in W) with the relation:

$$E_{prop\_GPU}(f) = P_{prop\_GPU}(f) \cdot t_{prop\_GPU}(f) \tag{12}$$

$$P_{prop\_GPU}(f) = 42.94 + 0.096f \tag{13}$$

The blocks to be processed are in the main memory. Time and energy for accesses to the local memory of the PUs are included in the models presented above. In order to be processed by the GPUs, blocks must be transferred from the main memory (host memory) to the GPU's local memory (device memory). The results should be transferred back from device to host memory. In order to model these transfers over PCIe, we used the model described in [23]; this model adapts the latency (L), overhead (o), short message gap (g), long message gap (G) model to the host-device communication over the PCIe bus. The expression of the transfer time from host to device (t_{hd}) as a function of the number of transferred bytes (B) is given by the following formula:

$$t_{hd}(B) = L_{hd} + o_{hd} + G_{hd} \cdot B + g_{hd} \tag{14}$$

With the values from [23], the expression (14) is:

$$t_{hd}(B) = 0.009420 + 8.318392E - 008B + 0.002503, [ms] \tag{15}$$

The energy used for the host-device transfer ($E_{hd}(B)$, in J) is:

$$E_{hd}(B) = P_{PCI} \cdot t_{hd}(B) \tag{16}$$

P_{PCI} is the power consumed on the PCIe Bus; $P_{PCI} = 25W$ [26].
The transfer time from device to host (t_{dh}) is given by the formula:

$$t_{dh}(B) = L_{dh} + o_{dh} + G_{dh} \cdot B + g_{dh} \tag{17}$$

With the values from [23] the expression is:

$$t_{dh}(B) = 0.009023 + 7.924734E - 008B + 0.002674, [ms] \tag{18}$$

The energy used for the host-device transfer ($E_{dh}(B)$) is:

$$E_{dh}(B) = P_{PCI} \cdot t_{dh}(B) \tag{19}$$

For the network, the LoOgGP model proposed in [19] was used. This model introduces a second overhead (O) supplementary to the latency (L), overhead (o), short message gap (g), long message gap (G) parameters; O is linearly dependent on data size. The transfer time between two nodes over IB (t_{IB}) for B bytes is then:

$$t_{IB}(B) = L_{IB} + o_{IB} + O_{IB} \cdot B + G_{IB} \cdot B + g_{IB} \tag{20}$$

With the values from [19], $t_{IB}(B)$ is (in ms):

$$t_{IB}(B) = 181.5 + 34.7 + 1.88B + 37.9B + 1.88 \tag{21}$$

4.2 Reference model

As reference we developed a model estimating the time and the energy needed to perform one iteration for a given system configuration. A system configuration specifies: the number N_N of nodes and for each node i with $0 \leq i < NN$: the number of PUs of each type ($n_{CPU}(i)$ and $n_{GPU}(i)$); the clock frequencies of each PU($f_{CPU}(j,i)$,$0 \leq j < n_{CPU}(i)$ and $f_{GPU}(k,i)$,$0 \leq k < n_{GPU}(i)$); the number of blocks allocated to each node NB(i). The time of computing a block on the CPU j of the node i ($t_{CPU}(j,i)$) is:

$$t_{CPU}(j,i) = t_{col\_CPU}(f_{CPU}(j,i)) + t_{prop\_CPU}(f_{CPU}(j,i)) \tag{22}$$

where t_{col_CPU} and t_{prop_CPU} are calculated with the formulas (2), respectively (5). The energy is given by the following formula:

$$E_{CPU}(j,i) = E_{col\_CPU}(f_{CPU}(j,i)) + E_{prop\_CPU}(f_{CPU}(j,i)) \tag{23}$$

where E_{col_CPU} and E_{prop_CPU} are calculated with the formulas (3) , respectively (6). If the GPU is alone on the link to the host memory, the total time of computing eight blocks ($Bl_{GPU} = 8 \cdot Bl_{CPU}$ bytes) on the GPU k of the node i ($t_{GPU}(k,i)$) is:

$$t_{GPU}(k,i) = t_{hd}(Bl_{GPU}) + t_{col\_GPU}(f_{GPU}(k,i)) + t_{prop\_GPU}(f_{GPU}(k,i)) + t_{dh}(Bl_{GPU}) \tag{24}$$

where t_{hd} and t_{dh} are the transfer times from host to device and from device to host, respectively, given by equations (14) and (17), t_{col_GPU} and t_{prop_GPU} are the times for computing in *collision* and in *propagation* phase respectively, calculated with the formulas (8) and (11). The energy used to compute eight blocks on the GPU k of the node i ($E_{GPU}(k,i)$) is given by the following equation:

$$E_{GPU}(k,i) = E_{hd} + E_{col\_GPU}(f_{GPU}(k,i)) + E_{prop\_GPU}(f_{GPU}(k,i)) + E_{dh} \tag{25}$$

where the computation energies E_{col_GPU} and E_{prop_GPU} are calculated with the formulas (9) and (12) and the transport energies E_{hd} and E_{hd} are calculated with the formulas (16) and (19). Therefore, for the maximal time needed for the computation on the GPU (t_{max_GPU}), we derived the formula for the case that m GPUs are sharing the same link to memory:

$$\begin{aligned} t_{max\_GPU} = (m \mod 2) \cdot t_{hd}(Bl_{GPU}) + floor(\frac{m}{2}) \cdot t_{hd}(2 \cdot Bl_{GPU}) + t_{col\_GPU}(f_{GPU}(k,i)) \\ + t_{prop\_GPU}(f_{GPU}(k,i)) + ((m-1) \mod 2) \cdot t_{dh}(2 \cdot Bl_{GPU}) + (m \mod 2) \cdot t_{dh}(Bl_{GPU}) \end{aligned} \tag{26}$$

With these considerations, if we denote with $N_{BO}(i)$ the number of blocks which completely occupy the units of the node i, with q the quotient of $N_B/N_{BO}(i)$ and with r the reminder of $N_B/N_{BO}(i)$, the time to compute N_B blocks on node i is given by the following formula:

$$t(N_B,i) = (q+1) * max(t_{CPU}(j,i), t_{max\_GPU}), j = 1, n_{CPU}(i) \tag{27}$$

The energy is given by the equation:

$$\begin{aligned} E(N_B,i) = q * (\sum_{j=1}^{n_{CPU}(i)} E_{CPU}(j,i) + \sum_{k=1}^{n_{GPU}(i)} E_{GPU}(k,i)) + \sum_{k=1}^{\frac{r}{8}} E_{GPU}(k,i) \\ + \sum_{j=1}^{r \mod 8} E_{CPU}(j,i) + \sum_{j=(r \mod 8)+1}^{n_{CPU}(i)} E_{CPUidle}(j,i) + \sum_{k=\frac{r}{8}+1}^{n_{GPU}(i)} E_{GPUidle}(k,i) \end{aligned} \tag{28}$$

where $E_{CPUidle}(j,i)$ is the energy consumed by the CPU j of the node i in idle mode and $E_{GPU}(k,i)$ is the energy consumed by the GPU k of the node i in idle mode.
The CPU and GPU idle energies are calculated with the equations:

$$E_{CPUidle}(j,i) = P_{CPUidle} \cdot t_{CPUidle}(j,i) \tag{29}$$

$$E_{GPUidle}(k,i) = P_{GPUidle} \cdot t_{GPUidle}(k,i) \tag{30}$$

The CPU and GPU idle times are given by the formulas:

$$t_{CPUidle}(j,i) = \begin{cases} t(N_B,i) - (q+1)\cdot t_{CPU}(j,i) & if j \leq r \mod 8 \\ t(N_B,i) - q\cdot t_{CPU}(j,i) & if j > r \mod 8 \end{cases} \quad (31)$$

$$t_{GPUidle}(k,i) = \begin{cases} t(N_B,i) - (q+1)\cdot t_{GPU}(k,i) & if k \leq \frac{r}{8} \\ t(N_B,i) - q\cdot t_{GPU}(k,i) & if k > \frac{r}{8} \end{cases} \quad (32)$$

We considered $P_{CPUidle} = 30$ W and $P_{GPUidle} = 42$ W, the free coefficients from equations (4) and (10).
After all PUs have terminated their local computation, the exchange phase takes place. In this phase, each node transfers the border zones to its neighbours. The averaged dimension (d_{bord}) of the border zone for a subdomain with N_B blocks is:

$$d_{bord}(N_B) = 4\cdot\sqrt{N_B}\cdot 1024\cdot w_{bord}\cdot d_{site}, \text{ bytes} \quad (33)$$

where w_{bord} is the border width (stencil dimension for the computation of one site) and d_{site} is the dimension in bytes of the data for one site. The average is computed considering the square with the equivalent number of blocks. Considering a duplex channel of communication, the time for the node i to be ready for the next iteration ($t_{iter}(i)$) is the time needed to compute the N_B blocks and the time to transfer the border zones:

$$t_{iter}(N_B,i) = t(N_B,i) + t_{IB}(d_{bord}(N_B)) \quad (34)$$

The energy consumed for this is:

$$E_{iter}(N_B,i) = E(N_B,i) + E_{IB}(d_{bord}(N_B)) \quad (35)$$

An iteration is finished when all nodes have completed their transfers. The performance objectives of the whole system for a given distribution of blocks $N_B(i)$, $1 < i \leq N_N$ are: a) the time of computation for one iteration (t_{iter}) (Eq. 36); b) the energy of computation for one iteration(E_{iter}) (Eq. 37), with the idle energy E_{idle} given in (Eq. 38); c) the load imbalance $Li = 1 - Lb$ where the load balancing Lb is given by (Eq. 39) as defined in [27] . We used Li in order to perform the multi-criterial minimisation for all the objectives, as Lb is an objective to be maximized.

$$t_{iter} = \max_{i\in[0...N_N-1]} t_{iter}(N_B(i),i) \quad (36)$$

$$E_{iter} = \sum_{i=0}^{N_N-1} E_{iter}(N_B(i),i) + E_{idle} \quad (37)$$

$$E_{idle} = \sum_{i=0}^{N_N-1} (t_{iter} - t_{iter}(N_B,i))\cdot(\sum_{j=1}^{n_{CPU}(i)} P_{CPUidle}) + \sum_{k=1}^{n_{GPU}(i)} P_{GPUidle}) \quad (38)$$

$$Lb = \frac{\underset{i\in[0..N_N-1]}{AVG} t_{iter}(N_B(i),i)}{t_{iter}} \quad (39)$$

4.3 Petri Net model

The system was modelled using the Petri net formalism. Specifically, Extended Petri Nets were used [2]. The modelling and simulation were performed using the Snoopy tool [9]. Hardware and software components were modelled as described in [20]. Each PU is represented by a token; this token is initially located in the idle place and travels to the other places representing points of the computation reached by the PU. The process of computation is represented by two transitions corresponding to the two phases: propagation and collision. Each of the computation phases introduces a time delay and an energy consumption; to model them, we implemented in the net the relations described in Section 4.1. The frequency of each PU is modelled by tokens residing in the Frequency place; this place is connected to the transitions representing each of the two computing phases, collision and propagation, through a read arc. In this way, transition's delay and number of generated tokens are expressed as function of the number of frequency tokens. The process of modifying the frequency of PU for Dynamic Voltage and Frequency Scaling (DVFS) is simulated by increasing or decreasing the number of tokens in the frequency place. In the CPU model, each transition corresponds to the computation of an elementary block. The delay of the transitions corresponds to the time given by the formulas (2) for *collision* and (5) for *propagation*. For the GPU model, each transition corresponds to the computation of eight blocks representing a GPU block. The delay corresponds to the time obtained with the formulas (8) for *collision* and (11) for *propagation*. The transition for each computing phase produces tokens that accumulate in an *Energy* place. The number of produced energy tokens depends on the number of frequency token through the formulas (3) for collision on CPU, ((6) for propagation on CPU, (9) for collision on GPU and respectively (12) for propagation on GPU. By counting the tokens in the *Energy* place, the energy consumed by each computational process can be evaluated.
The PUs, global memories and network interfaces are connected through a simplified model of the PCIe bus. The tokens flowing through this part of the model represent data packets. Each component connected to the bus can gain the exclusive access to the bus by putting a token in a place which inhibits the transition for other units connected to the bus. The values of the coefficients from the formulas (15) and (18) were used as timing values for the transitions of the CPN model. Overhead (o) is used for the transition representing the preparing of the message. This transition needs a CPU core token in the idle place. Latency (L) is used as a delay in the next transition modelling the transport on the PCI. The bandwidth for a large message (G) is used to calculate the passing rate through the transition. The bandwidth for short messages (g) is used for the transition enabling sending of a new data stream. The transition delay simulates the bus contention. The specific mainboard architecture is also considered; it has a separate link to each processor. In this way, only half of the GPU boards from a node shares the same host-device link.
For the network simulation, the values of the parameter from the formula (21) were used for the transitions delays. The two overheads (o, O) were used in the transmission preparing transition, which also consumes a CPU core token from the idle state. The two gaps (g, G) were used in the transmission transition simulating the network bandwidth and contention.
The application is composed of the scheduler net, which connects to the StarPU run time net, and the load balancer, which connects to the Charm++ runtime. We re-used the Charm++ structure presented in [20]. Details of the scheduler for one PU are presented in Figure 2. The scheduling is done by computing an objective function for each PU. The next arriving task is dispatched to the PU with the minimal value for the objective function. The computation of the objective function is implemented through a net simulating the addition of values through the accumulation of corresponding number of tokens. For each PU, one such net was build. This net has an accumulator place for each optimisation criteria. The place contains the current value

of the criteria for the PU. For each task in the waiting queue and for each PU, a number of tokens are generated for energy consumption, completion time and load. The number of tokens corresponds to the estimated values of the criteria for the task execution on the PU. These tokens are transferred to an estimation place. An amount equal to the tokens for the current values is also transferred in the same place. This gives the estimation of the cumulative values for energy, completion time and load after the task execution on the given PU. All estimated values are connected to a cumulative transition. This transition functions like a minimum operator. It transfers tokens until the place with the smallest number of tokens becomes empty. The empty place activates the scheduling transition. The transition transfers the task token from the waiting queue to the corresponding PU queue. The accumulator places of the selected PU are then updated with the number of tokens corresponding to the scheduled task. If a frequency change takes place, the computational speed and the energy used by the PU are changing, meaning that the estimation for the completion time and energy for the tasks still in queue are not accurate any more. The number of tokens in energy and completion time accumulators places are increased correspondingly. In addition, if a task is transferred from queue to execution, a token is removed from the load accumulator place. By explicitly modelling the scheduling process, it is possible to model the incurring overhead. Each operation corresponding to a computational process is represented through a transition. The transition is enabled only if a token for a CPU core is present in the idle core place; it consumes this token and returns it to the idle place only after the processing time passed.

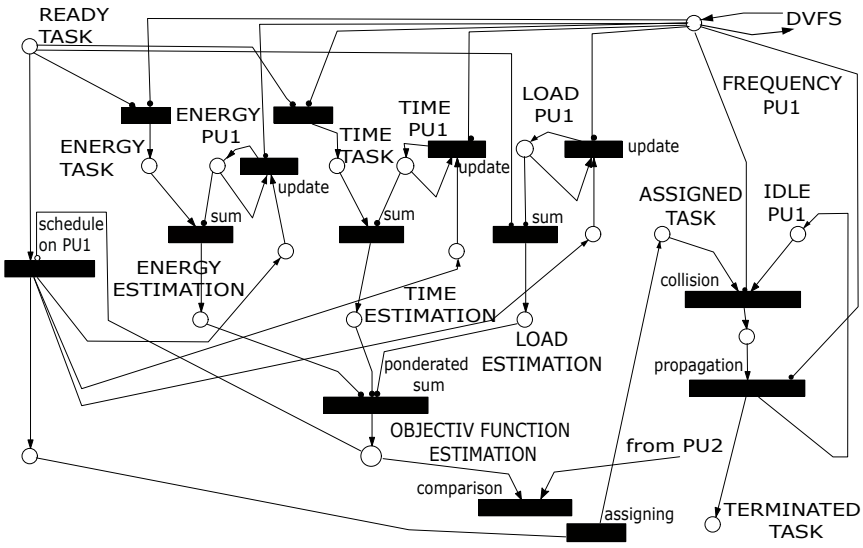


Figure 2: Schematics of the scheduling function for a PU

5 Simulation

5.1 Computing system modelled

We modelled a system with 16 nodes. To give relevance to the computational case, each node was configured differently from the other in respect to the available CPUs and GPUs. This configuration becomes relevant in the context of the partition of computing resources among different applications [10] or in the context of temporary or permanent resource unavailability. In both cases, even if the nodes are identical at start, the resources available to an application differ from a node to the other. The configuration for each node is given in Table 1, where

Table 1: Nodes configuration

Node i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$n_{CPU}(i)$	8	8	8	8	8	8	8	8	16	16	16	16	16	16	16	16
$n_{GPU}(i)$	2	4	6	8	10	12	14	16	2	4	6	8	10	12	14	16
$NBO(i)$	23	38	53	68	83	98	113	128	31	46	61	76	91	106	121	136

nCPU(i) and nGPU(i) are the number of PUs of each type and NBO(i) the number of blocks completely occupying the units of the node i.

Considering the optimal blocks load of each PU given in [4] and that a CPU core is needed to manage the operation of a GPU board (=two processors), $N_{BO}(i)$ is given by the formula:

$$N_{BO}(i) = n_{CPU}(i) - \frac{n_{GPU(i)}}{2} + 8 \cdot n_{GPU}(i) \tag{40}$$

5.2 Design space

The input variables for the three objective functions of the multi-criterial optimisation problem are the partition of the LB grid in 16 subdomains and the running frequency for each PU of the system. The dimension of the design space (i.e. the number of possible hardware-software combinations) is equal to the number of possible partitions x number of possible frequency configurations. The number of possible frequency configurations is equal to ((number of CPU frequency levels) x (number of GPU frequency levels))^{16}. Even by considering only the frequencies effectives for DVFS (as shown in [4]), we have four levels for CPU and 12 levels for GPU giving a set of 48^{16} possible configurations. The number of possible partitions depends on the dimension of the partition unit(the smallest part to be transferred between nodes). If a node is idle, the minimal optimal load to be transferred to it will be of 16 blocks(the load of a GPU card with two GPU processor). The node 15 can host maximal 256 blocks over its optimal load of 136; this gives 256/16 = 16 units of partition. For the 16 nodes, we determined experimentally 300540195 possibilities of distributing these 16 units. This gives a total of $48^16 \cdot 300540195 \approx 10^{35}$ possible configurations. This design space obviously cannot be explored exhaustively in order to obtain the Pareto set.

5.3 Reference surface

To have a measure of the algorithm efficiency, we built an approximation of the Pareto surface in the space of the three objectives: energy consumption, completion time and load balancing. We designated this surface as reference. Given the design space dimension, we used a metaheuristic algorithm to generate a reference surface approximating the Pareto set. We preferred this approach to other metaheuristics (e.g. NSGA2 multi-objective genetic algorithm) because it can find points in the Pareto set like the configurations using minimal energy and the configurations with minimal computing time. The pseudocode is presented in Algorithm 1

To reduce calculation, we used a dictionary type structure *perf_ data* to store the values for the time needed and energy consumed by each individual node for the computing of each of the possible subdomain's dimension (in blocks), with each of the possible frequency configurations. The values were calculated with formulas (27) and (28) from Section 4.2. The dictionary allows the retrieving for each node i and for each subdomain dimension B of the minimal computing time (t_{min}) and corresponding energy ($E_{t_{min}}$), of the minimal energy (E_{min}) and corresponding time ($t_{E_{min}}$) and of the corresponding frequencies configurations, $freq_{t_{min}}$ and $freq_{E_{min}}$. We obtained a set of starting points by generating a manageable set of partitions. We considered

Algorithm 1 Metaheuristic search

generate perf_data dictionary
generate schedules
for all schedules, S **do** ▷ global candidate selection
  $(freq_{t_{min}}, t_{min}, E_{t_{min}}, Li_{t_{min}}) \leftarrow$ generate tmin configuration$(S, perf_data)$
  $(freq_{E_{min}}, t_{E_{min}}, E_{min}, Li_{E_{min}}) \leftarrow$ generate Emin configuration$(S, perf_data)$
  if $insert(ref\_set, (S, freq_{t_{min}}, t_{min}, E_{t_{min}}, Li_{t_{min}}))$ **then**
    $new\_freq \leftarrow freq_{t_{min}}$
    repeat ▷ local search for candidates that minimise Li
      $new\_freq \leftarrow decrease_frequency(new\_freq)$
      $(t_{new}, E_{new}, Li_{new}) \leftarrow generatenewLiconfig(S, new\_freq, perf\_data)$
    until $insert(ref\_set, (s, new\_freq, t_{new}, E_{new}, Li_{new}))$
  end if
  if $insert(ref\_set, (S, freq_{E_{min}}, t_{E_{min}}, E_{min}, Li_{E_{min}})$ **then**
    $new\_freq \leftarrow freq_{E_{min}}$
    repeat ▷ local search for candidates that minimise Li
      $new\_freq \leftarrow increase_frequency(new\_freq)$
      $(t_{new}, E_{new}, Li_{new}) \leftarrow generatenewLiconfig(S, new\_freq, perf\_data$
    until $insert(ref\_set, (s, new_c\_Emin, t_new, E_new, li_new)$
  end if
end for

packets of 16 blocks as the partition units and we generated all possible distributions of these 16 units on the 16 nodes (300540195 schedules). The reference set *ref_ set* is the structure that will contain the approximation of the Pareto set at the algorithm completion. For the *ref_ set* we used a dictionary of dictionaries structure. Each of the three optimization objectives is key at one level. The keys are sorted, allowing efficient comparison of a given element with the set members. This is used by the insert operation testing the argument element before insertion. If the element is not dominated, it is inserted in the set and the operation returns true. If the element is dominated, it is discarded and the operation returns false. If the element dominates a set member, this member is deleted from *ref_ set*. Using the same algorithm, we built reference sets with less blocks by decreasing the dimension of the partition unit. These configurations were used as starting point for the investigation of load increase case (see Section 5.5). With a similar process, reference sets were calculated for partitions on 10 to 15 nodes to be used for the node failure case (see Section 5.6).

5.4 Initialization case

The simulation aims to test if the developed algorithm can achieve a mapping (domain decomposition plus scheduling) near to the reference set and how fast it can reach this mapping. An initialization message is sent to one of the nodes with the location and dimension of the complete domain. The node starts the Contract Net Protocol initiation. After receiving biddings, the starting node divides the domain, considering its processing power (type and number of PUs) and the processing power of the bidding nodes. The distribution phase continues recursively until no nodes are bidding. Each node starts computation in parallel with redistribution. The computations start with the blocks situated at the domain core, allowing the outer blocks to be redistributed. The normal redistribution takes place automatically at the end of each iteration. A simulation was run for each node as first node.

5.5 Load increase case

The simulation goal was to show how the algorithm reacts in case of a dynamic increase in load; this situation arises when the calculation imposes the mesh refining in order to increase computation's accuracy. Refining is done by halving the distance between the grid sites (one step of refining). The number of sites in each direction is doubled. The total number of block sites is multiplied by four. We started with a configuration that is on the reduced reference set and increased four times the load of some selected nodes. In the CPN model, the load increase was simulated by marking the places corresponding to the selected nodes with four times more tokens than in the reference mapping. The effect of a single affected node was investigated at first. Each node was loaded with four times its complete load indicated in Table 1. The cases of multiple nodes were then investigated. As reference we used the reference set computed for the maximal number of sites. As starting configuration we used a configuration in the reference set for a lower number of sites.

5.6 Node failure case

In this case, we investigated how the load balancing will react to a node failure. We used the reference set computed with the same number of sites for a system with fewer nodes. In the simulation case, a scheduled transition injects at a random time a token, blocking the functioning of one or more nodes. The first goal is to find if the system could recover after losing one or more nodes; the second one is to see in how much time the mapping reaches the reference surface for the corresponding number of nodes.

6 Results and discussion

6.1 Initialisation case

In the first step, the chare on the init node distributed the data to eight other nodes with the shortest path to it. Each of these nodes already have some fixed neighbours (the node from which it received the task and some of the nodes that accepted the task.) The recursive process ends when no other nodes respond or a node has a subdomain that it can certain compute.
Figure 3 presents one example of the distribution process for a 4x4 grid. In each step, the announcing nodes are in blue, the nodes that have finished distribution and started computing are in green and the nodes that cannot distribute further are in red. The arrows are presenting established contracts.

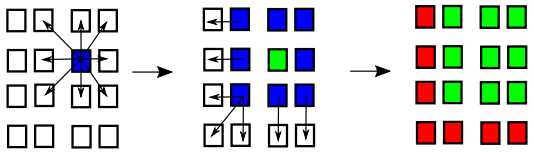


Figure 3: Initial distribution of load on available nodes

After all nodes start calculation, blocks are also exchanged in the communication step to balance the load. This happens until the computation time is approximately equal on all nodes. After the computations were started, the simulations were run until a stable mapping (meaning no full block exchange) took place in the communication step. The simulations were run for each node as starting node. A mean of 40 iterations are needed for reaching a grid distribution that is near to the reference set.

Figure 4 presents one of the reached mapping as a red point, compared with the reference set in blue. We considered the time and the used energy as more important than the load; so, we selected the values $\alpha = 0.40$; $\beta = 0.40$; $\gamma = 0.2$ for the weighting coefficients of the objective function given by formula (1). Therefore we expected that the resulted mapping will be nearer to the points representing also the weighted combination of the three objectives on the reference set. The cause of not reaching the configuration on the reference set is that the redistribution stopped before reaching the optimal schedule. The faster node has finished earlier and requested the border block. The request arrived only after the slower node started to process the last blocks so they were not transferred anymore.

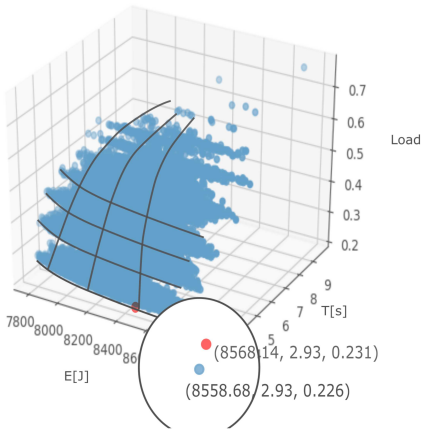
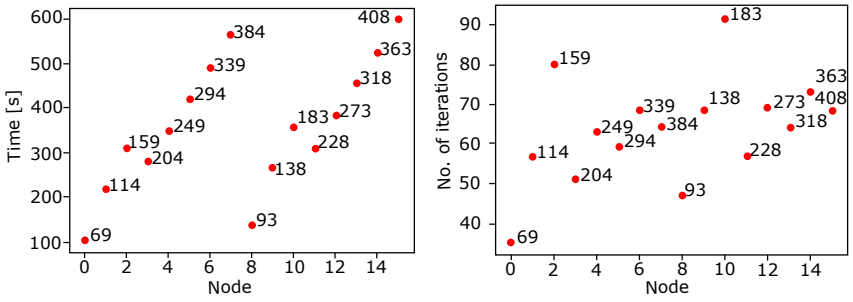


Figure 4: Reference set (blue) with stable configuration (red) reached after redistribution

6.2 Load increase case

Figure 5 presents time (a) and iteration number (b) needed to reach the proximity of the Pareto front for a single affected node.



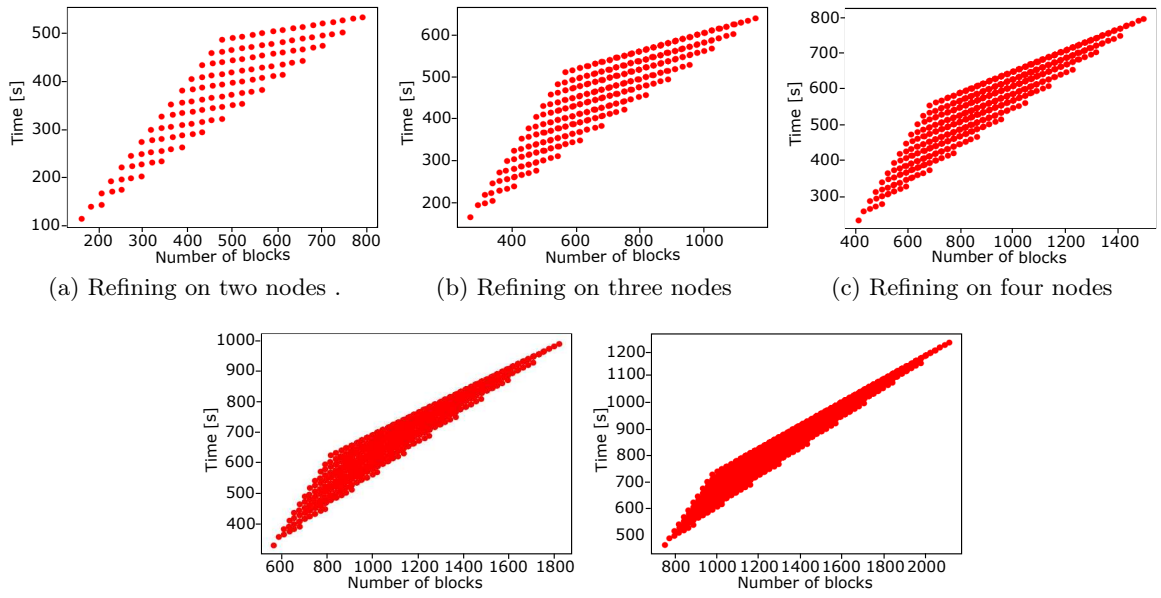
(a) Time needed to reach the near optimal solution.

(b) Number of iterations needed to reach optimal solution.

Figure 5: Handling of the load increase on one node. Point label indicates number of blocks to be transferred

The numbering of the nodes are in the order from the slowest node (node 0 only two GPUs available) to the fastest node (node 15 - all possible GPUs=16 available). In each case, the affected node was considered to have the minimal optimal load ($N_{BO}(i)$ from Table 1) before the refinement was started. In Figure 5, each point is labelled with the number of blocks to be

transferred which is approximated to 3*optimal load. Point annotation is the number of supplementary blocks to be redistributed (3*optimal load, see Table 1). The break in the ascending trend at the nodes 3 and 11 can be explained by some architectural details. Starting with the two nodes, there is a better distribution of GPUs on the CPU-GPU interconnect network. In this way, the transfer time between the host and devices is smaller for the same amount of data. The time needed to reach a stable near optimal mapping, in the proximity of the reference set, for the refinement started in two(a) to six(f) nodes is presented in Figure 6. Number of blocks on the x-axis is the total number of blocks to be redistributed from the affected node. All possible distinct combinations of two, three, four, five and respectively six nodes that can be formed from the 16 nodes, were simulated. As the configurations are limited, there are more combination



(a) Refining on two nodes . (b) Refining on three nodes (c) Refining on four nodes

(d) Refining on five nodes (e) Refining on six nodes

Figure 6: Handling of the load increase on 2 to 6 nodes

of nodes with the same total number of blocks. Depending of the node combination, we have different timings for the same total point number (grid sites). This is the result of their hardware configuration and of the number of neighbours from each of them. The time is increasing with the number of nodes as they share the same communication bandwidth.

6.3 Node failure case

Figure 7 shows the time until the stable, near optimal mapping is reached depending on the number of points to be redistributed. The points label indicates the number of defective nodes. Three classes of nodes failure behaviour are observed:

- slow recovering nodes: 0, 1, 3, 13 and 15;
- intermediate recovering nodes: 8, 9, 2, 10, 4, 12, 6, 14 and 7;
- fast recovering nodes: 11 and 5.

The three classes are given by the variation in the number and the hardware configuration of the node neighbours. Failure in nodes with many neighbours with better architectural characteristics

(such as nodes 11 and 5) is recovered faster than failure in nodes that have less and/or suboptimal neighbours (such as nodes 0 and 5).

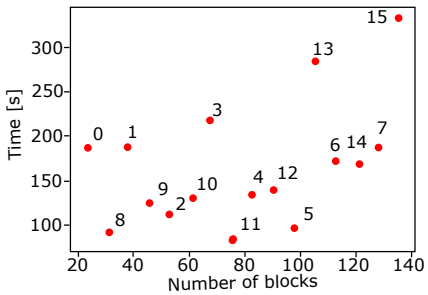


Figure 7: Time for the redistribution of the load of a failed node. The label of each point indicates the node that has failed.

7 Conclusion

We have investigated through simulation the behaviour of the proposed algorithm for the multi-objective optimization of the workload distribution on a heterogeneous computing system. The simulations have shown that the algorithm can perform as expected. The algorithm succeeds:

- To distribute the workload by passing it from neighbour to neighbour. The maximal number of iterations needed is equal to the maximal distance between the starting node and the other nodes in the tree of neighbourhood that is forming;
- To reach a stable state in the proximity of the reference set of a tri-objective optimisation problem. The factors influencing the iteration number and the time needed for reaching a stable configuration were also revealed by this investigation. They are the hardware configuration and the position in the node neighbourhood hierarchy/topology. From the hardware perspective, the number and distribution of GPUs on the available slots is determined. The optimum is for a moderate number of GPUs uniformly distributed on the slots, to fully exploitation of the communication hierarchy. From the topological perspective, nodes with fewer neighbours recover slower from the increase of load as they cannot efficiently evacuate the load;
- To handle well sudden load increases caused by mesh refining or node failure.

The behaviour observed through the simulation reveals that our algorithm has some advantages over other algorithms:

- It does not need an explicit, distinct imbalance check. This is implicitly detected by the slower node that receives a border zone request from is faster neighbours. The load balancing is also resolved with the transfer that is already taking place only with an increased payload. Therefore, our developed algorithm has a speed and resource advantage over all algorithms that make an explicit balance check and have an explicit load balancing phase;
- It can detect and solve imbalances resulted from special situations, like mesh refining or node failure without explicit checks. The detection is implicitly codified in the exchanges protocol between neighbours. After the supplementary load is dispatched to some queues, the load balancing is taking place like in a normal situation. The consideration of special cases- refining and failure - is an original aspect. This is important because these cases are unexpected, but not rare for the application investigated;

- It uses less uncertain data. It starts the transfer only when the imbalance is certain and directs the data only to the nodes that can handle the supplementary load. So, our proposed algorithm is more robust than algorithms basing their decision exclusive on estimations. The second level scheduler uses estimates in its decision, but at this level predictability is higher. At first level, the decision of transferring the load is not based on estimations but triggered by a real imbalance;

- It follows three objectives for optimisation, instead of classical solution that considers only time and energy [6] [13]. Analysis of computing load is also important in the context of system cooling and of a datacenter reliability.

The heuristic algorithm used to approximate the Pareto surface is also original and has showed potential at least for producing reference sets and for off-line scheduling. It must be also further analysed. More investigation has to be done on the model of a system with a much higher number of nodes in the magnitude order of a real HPC configuration. This will be done using a simulator that can handle this number of nodes and a metaheuristic automatic design space tool (FADSE) [11] to build the reference set.

Conflict of interest

The authors declare no conflict of interest.

Bibliography

[1] Augonnet, C.; Samuel, T.; Namyst, R.; Wacrenier, P.-A. (2011); StarPU: A Unified Platform for Task Scheduling on Heterogeneous Multicore Architectures, *Concurrency and Computation: Practice and Experience, Special Issue: Euro-Par 2009*, 23, 187-198, 2011.

[2] Blätke, M.A.; Heiner, M.; Marwan, W. (2015); Engineering with Petri Nets, In *R. Robeva (Ed.), Algebraic and Discrete Mathematical Methods for Modern Biology*, Elsevier Inc., 141–193, 2015.

[3] Brahambhatt, M.; Panchal, D. (2015); Comparative Analysis on Heuristic Based Load Balancing Algorithms in Grid Environment, *International Journal of Engineering Research & Technology (IJERT)*, 4(4), 802–806, 2015.

[4] Calore, E.; Gabbana, A.; Schifano, F.S.; Tripiccione, R. (2017); Evaluation of DVFS techniques on modern HPC processors and accelerators for energy-aware applications, *Concurrency and Computation: Practice and Experience*, DOI: https://doi.org/10.1002/cpe.4143, 29(12), 1–19, 2017.

[5] Casanova, H.; Giersch, A.; Legrand, A.; Quinson, M.; Suter, F. (2014); Versatile, Scalable and Accurate Simulation of Distributed Applications and Platforms, *Journal of Parallel and Distributed Computing*, DOI: https://doi.org/10.1016/j.jpdc.2014.06.008, 74(10), 2899–2917, 2014.

[6] Chatterjee, N.; Paul, S.; Mukherjee, P.; Chattopadhyay, S.(2017); Deadline and energy aware dynamic task mapping and scheduling for Network-on-Chip based multi-core platform, *Journal of Systems Architecture*, DOI: https://doi.org/10.1016/j.sysarc.2017.01.008, 74, 61–77, 2017.

[7] Chen, B.; Potts, C.N.; Woeginger, G.J. (1998); A Review of Machine Scheduling: Complexity, Algorithms and Approximability, In *D.Z. Du, P.M. Pardalos (Eds.), Handbook of Combinatorial Optimization*, Springer, 21–129, 1998.

[8] Guo, Z.; Shu, C.(2013); *Lattice Boltzmann Method and Its Applications in Engineering Advances in computational fluid dynamics Volume:3*, World Scientific, 2013.

[9] Heiner, M.; Herajy, M.; Liu, F.; Rohr, C.; Schwarick, M.(2012); Snoopy - a unifying Petri net tool, In *S. Haddad, L. Pomello, (Eds.) Application and Theory of Petri Nets*, Springer, 7347, 398–407, 2012.

[10] Hugo, A. E.; Guermouche, A.; Wacrenier, P.A.; Namyst, R. (2013); Composing Multiple StarPU Applications over Heterogeneous Machines: A Supervised Approach, In *2013 IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum*, 1050–1059, 2013.

[11] Jahr, R.; Calborean, H.; Vintan, L.; Ungerer, T. (2012); Finding Near-Perfect Parameters for Hardware and Code Optimizations with Automatic Multi-Objective Design Space Explorations, In *Concurrency and Computation: Practice and Experience*, DOI: 10.1002/cpe.2975, 27(9), 2196–2214, 2012.

[12] Jeannot, E.; Vernier, F., (2006); *A Practical Approach of Diffusion Load Balancing Algorithms*, INRIA, RR5875, 2006.

[13] Juarez, F.; Ejarque, J.; Badia, R.M. (2018); Dynamic energy-aware scheduling for parallel task-based application in cloud computing, *Future Generation Computer Systems*, 78, 257–271, 2018.

[14] Kale, L.V.; Bhatele, A.; (2013); *Parallel Science and Engineering Applications: The Charm++ Approach (1st ed.)*, CRC Press, 2013

[15] Kasmi, N.; Zbakh, M.; Samadi, Y.; Cherkaoui, R.; Haouari, A. (2017) Performance evaluation of StarPU schedulers with preconditioned conjugate gradient solver on heterogeneous (multi-CPUs/multi-GPUs) architecture, In *3rd International Conference of Cloud Computing Technologies and Applications (CloudTech)*, 1–6, 2017.

[16] Kaur, N.; Chhabra, A. (2017); Comparative Analysis of Job Scheduling Algorithms in Parallel and Distributed Computing Environments, *International Journal of Advanced Research in Computer Science*, 8(3), 948–956, 2017.

[17] Khan, S.; Nazir, B.; Khan, I. A.; Shamshirband, S.; Chronopoulos, A. T. (2017); Load balancing in grid computing: Taxonomy, trends and opportunities, *Journal of Network and Computer Applications*, 88, 99–111, 2017.

[18] Kjolstad, F.B.; Snir, M.(2010); Ghost Cell Pattern, In *Proceedings of the 2010 Workshop on Parallel Programming Patterns (ParaPLoP'10)*, DOI=http://dx.doi.org/10.1145/1953611.1953615, 4, 2010.

[19] Martinez, D. R.; Cabaleiro, J.C.;Pena, T.F.; Rivera, F.F.; Blanco,V. (2009); Accurate analytical performance model of communications in MPI applications, In *2009 IEEE International Symposium on Parallel & Distributed Processing*, DOI: https://doi.org/10.1109/IPDPS.2009.5161175, 1–8. 2009.

[20] Mironescu, I.D.; Vintan, L. (2017); A task scheduling algorithm for HPC applications using colored stochastic Petri Net models, In *Proceedings of 13th International Conference on Intelligent Computer Communication and Processing*, 479–486, 2017.

[21] Rauber, T.; Rünger, G.; Schwind, M.; Xu, H.; Melzner, S. (2014); Energy measurement, modeling, and prediction for processors with frequency scaling, *The Journal of Supercomputing*, 70, 1451–1476, 2014.

[22] Ubal, R.; Byunghyun, J., Mistry, P.; Schaa, D.; Kaeli, D., (2012); Multi2Sim: a simulation framework for CPU-GPU computing, In *Proceedings of the 21st international conference on Parallel architectures and compilation techniques (PACT '12)*, ACM, 335–344, 2012.

[23] van Werkhoven, B.V.; Maassen, J.; Seinstra, F.J.; Bal, H.E. (2014); Performance Models for CPU-GPU Data Transfers, In *14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, 11–20, 2014.

[24] Wu,J.; Contract Net Protocol for Coordination in Multi-Agent System, In *2008 Second International Symposium on Intelligent Information Technology Application*, doi: 10.1109/I-ITA.2008.273, 1052-1058, 2008.

[25] [Online]. Available: http://www.fe.infn.it/coka/doku.php?id=start, Accesed on 26 february 2018

[26] [Online]. Available: https://pcisig.com/specifications/pciexpress/base2/, Accesed on 26 february 2018

[27] [Online]. Available: https://pop-coe.eu/node/69, Accesed on 26 february 2018

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 13(4), 521-536, August 2018.

An Approach for Detecting Fault Lines in a Small Current Grounding System using Fuzzy Reasoning Spiking Neural P Systems

H. Rong, M. Ge, G. Zhang, M. Zhu

Haina Rong, Mianjun Ge, Gexiang Zhang*
School of Electrical Engineering
Southwest Jiaotong University
Chengdu, 610031, China
ronghaina@126.com, 641294684@qq.com, zhgxdylan@126.com
*Corresponding author: zhgxdylan@126.com

Ming Zhu
Chengdu University of Information Technology
Chengdu 610225, China
zhuming@126.com

Abstract: This paper presents a novel approach for detecting fault lines in a small current grounding system using fuzzy reasoning spiking neural P systems. In this approach, six features of current/voltage signals in a small current grounding system are analyzed by considering transient and steady components, respectively; a fault measure is used to quantify the possibility that a line is faulty; information gain degree is discussed to weight the importance of each of the six features; rough set theory is applied to reduce the features; and finally a fuzzy reasoning spiking neural P system is used to construct fault line detection models. Six cases in a small current grounding system prove the effectiveness of the introduced approach.
Keywords: Membrane computing; P system; spiking neural P systems; fault line detection; feature analysis; information gain degree; rough set theory

1 Introduction

As a rapidly developing branch of natural computing, membrane computing, initiated by Păun [3, 12], focuses on the investigations of a computational model, called a membrane system or a P system, abstracting from the structure and functioning of living cells, as well as from the way the cells are organized in tissues or high order structures. Numerous variants of P systems have Turing computing power [5, 15] or can solve computationally hard problems in a polynomial time [16, 26]. Inspired by the information processing principles in a living cell, a P system has certain characteristics, such as distribution, parallelism and expansibility, which make it suitable to solve a variety of practical problems [14, 46], such as engineering optimization [7, 28], languages generation [12, 40] and modeling biological and ecological systems [6].

In recent years, much attention is paid to spiking neural P systems (SN P systems), which were introduced in 2006 [5, 9] by considering the neurophysiological behavior of neurons sending electrical impulses (spikes) along axons from presynaptic neurons to postsynaptic neurons. Except for theoretical results [10, 13], SN P systems have been widely used to solve various application problems, such as combinatorial and engineering optimization problems [30, 38], signal recognition [2], arithmetic operations [17, 21, 28] and fuzzy knowledge representation [23]. Of a particular interest is the combination of SN P systems with fuzzy set theory, called fuzzy membrane computing [36], to solve fault diagnosis problems with respect to transformers [16, 42], transmission lines [8, 24, 25], traction power supply systems of high-speed railways [39] and metro

traction systems [10], in electric power systems. In general, there are two kinds of fuzzy reasoning spiking neural P systems (FRSN P systems) [36]: fuzzy reasoning spiking neural P system with real numbers (rFRSN P systems) [16] and fuzzy reasoning spiking neural P systems with trapezoidal fuzzy numbers (tFRSN P systems) [24].

The present work is motivated by two reasons. On the one hand, the application extension of FRSN P systems requires further discussions. On the other hand, the fault line detection in a small current grounding system is a very important problem and its detection accuracy needs to be enhanced [4, 9, 32, 35]. In China, most of medium-voltage distribution networks are small current grounding systems. Furthermore, about 70%–80% of the faults in distribution networks result from single-phase grounding. Therefore, the fault line detection problem with respect to single-phase grounding in a small current grounding system is a very common type of fault diagnosis in an electrical power system [1, 15, 30, 31]. Thus, a novel approach for detecting fault lines in a small current grounding system using FRSN P systems is proposed to enhance the accuracy. This approach first analyzes six features of current/voltage signals in a small current grounding system by considering transient and steady components, respectively. Then, the possibility that a line is faulty is quantified by using a fault measure. Next, information gain degree is used to weight the importance of each of the six features and rough set theory is discussed to reduce the features. Finally, FRSN P systems are used to construct fault line detection models. The effectiveness of the introduced approach is verified by several typical cases of fault line detection in a small current grounding system.

The organization of this article is as follows. Section 2 describes the fault line detection problem in a small current grounding system. Section 3 presents the proposed approach in detail. In Section 4, six typical cases are used to conduct the experiments. Conclusions are finally drawn in Section 5.

2 Problem description

In this paper, a 110kV/35kV distribution network with 6 feeders is considered. The simplified model of a distribution network is shown in Fig. 1. When a single-phase-to-ground fault occurs in the distribution network, any one of the 6 lines, 1–6, or the bus may be the faulty line. If the fault lasts for a long time, there will be much effect on the safe and stable operation of the distribution network, and even significant security incidents may happen, therefore, it is required to take a proper action to deal with the fault so as to restore power supply as soon as possible. When a line is faulty, the distribution network will generate zero sequence voltage and current. As usual, there means much difference in phase and amplitude of zero sequence current between the fault line and normal fault line.

Currently the widely used techniques are that a certain number of features are used to detect fault lines. There are three main techniques: steady state method, transient method and information fusion method. The steady state method is to select fault line by extracting the steady state fault characteristics of the distribution network, such as zero sequence current amplitude and phase, admittance, harmonic, power [1, 15, 30, 31]. But these fault characteristics are easily affected by the way of neutral point grounding and different fault grounding methods. With the development of signal processing technology, the transient component that contains a lot of fault information is considered to detect the fault line. In [41], the wavelet transform was used to extract the features from zero sequence current of each feeder and the fault line detection is realized by comparing the polarity. Wavelet analysis has a good characteristic of time-frequency localization and can decompose the signal into different frequency bands, so it is especially suitable for the analysis of non stationary signals and transient signals. In [27], a neural network was used to detect the fault line detection. In [47], a comprehensive fault line

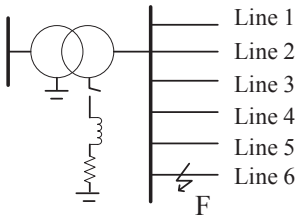


Figure 1: Simplified model of a distribution network

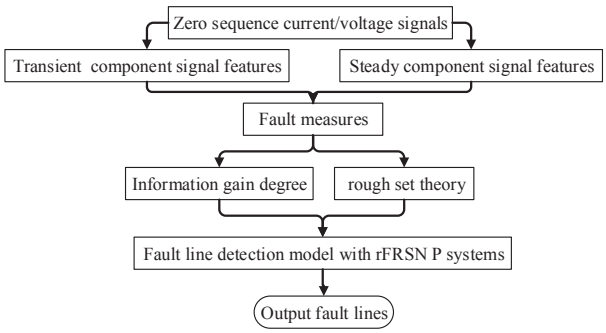


Figure 2: Framework of fault line detection approach

selection method based on fuzzy theory was discussed. In [14], an information fusion method based on D-S Evidence Theory was presented to detect fault lines in a small current grounding system.

3 The proposed approach

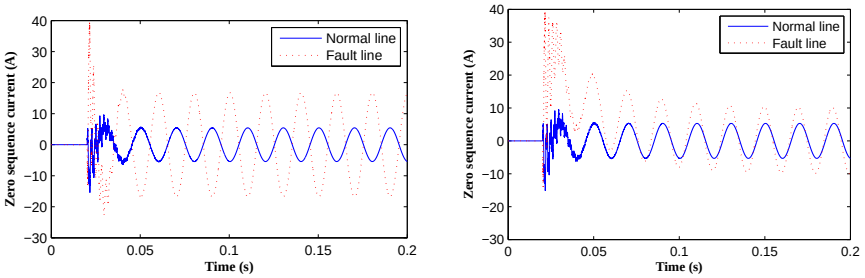
The fault line detection approach for a small current grounding system we propose here consists of four processes: feature analysis, fault measure calculation, feature information fusion and fault line detection model construction. Fig. 2 shows the framework of this approach. This paper considers both transient and steady components features of current or voltage signals. In the process of feature information fusion, the information gain degree is used to weight the importance of features and rough set theory is applied to reduce the features. A fault line detection model is constructed by using a fuzzy reasoning spiking neural P systems with real numbers (rFRSN P systems). In what follows, the processes will be described step by step.

3.1 Feature analysis

When a single-phase-to-ground fault occurs in the distribution network, the steady-state characteristics are mainly analyzed in the time domain. The fault current in the fault line is the ground capacitance current with the direction from the bus to the line, or it is the sum of the capacitance current in other lines with the direction from the line to the bus. The transient characteristics are analyzed in the frequency domain, including the amplitude, phase and energy of transient zero sequence current. To intuitively show the features, this study considers an example of a system, which has a single phase to ground fault at 0.02s, the transition resistance (R_0) is 0.2Ω, 20Ω, 2000Ω, representing the metal grounding, low impedance grounding and high impedance grounding, respectively. In the sequel, the first four features of steady components and the last two features of transient components are analyzed by using different methods, due to space limitations, this paper only gives a simulation diagram that the transition resistance

(R_0) is 0.2Ω.

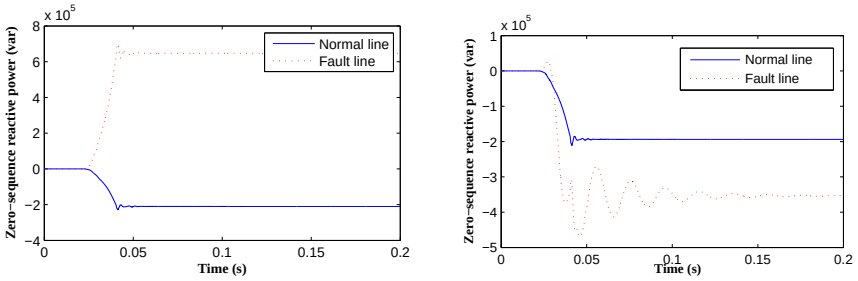
(1) *Zero sequence current analysis*: when the neutral non-grounding system has a single phase grounding fault, the amplitude of the zero sequence current in the fault line equals the sum of the zero sequence currents in all the normal lines. The direction of the zero sequence current in the fault line is from the line to the bus and the direction of the zero sequence current in the normal line is from the bus to the line. Thus, if the amplitude of the zero sequence current in a line is equal to the sum of the amplitude of the zero sequence current in other lines, the direction is opposite to the other lines, which indicates that the line is a fault one. Fig. 3 shows the comparisons of zero sequence current between normal and fault lines, which refer to the lines in a neutral non-grounding system and a neutral grounding system, respectively.



(a) Non-grounding system with $R_0 = 0.2\Omega$ (b) Grounding system with $R_0 = 0.2\Omega$

Figure 3: Zero sequence current comparisons between normal and fault lines

(2) *Zero-sequence reactive power analysis*: the zero sequence reactive power is the product of the corresponding zero sequence current and voltage. Fig. 4 shows the comparisons of zero sequence reactive power between normal and fault lines, which refer to the lines in a neutral non-grounding system and a neutral grounding system by using arc extinction coil, respectively. In the neutral non-grounding system, the zero sequence reactive component of the fault line is opposite to the zero sequence reactive component of the normal line, and its amplitude is bigger than that of the zero sequence reactive component of the normal line.

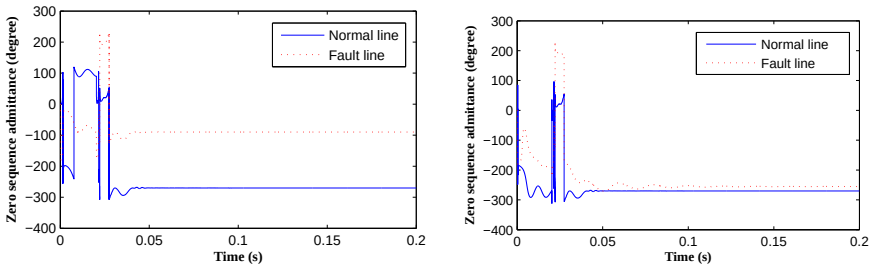


(a) Non-grounding system with $R_0 = 0.2\Omega$ (b) Grounding system with $R_0 = 0.2\Omega$

Figure 4: Zero sequence reactive power comparisons between normal and fault lines

(3) *Zero sequence admittance analysis*: In the neutral non-grounding system, the admittance angle of the normal line is in the first quadrant of the admittance plane angle, while

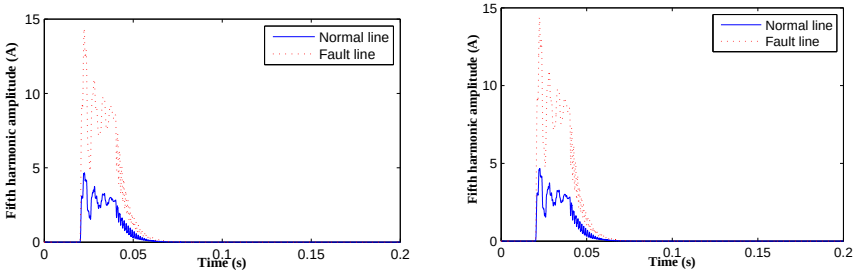
the admittance angle of the fault line is in the third quadrant. In the neutral grounding system by arc extinction coil, the admittance angles of the normal line and the fault line are in the first and second quadrant, respectively. Fig. 5 shows the comparisons of zero sequence admittance angles between normal and fault lines, which refer to the lines in a neutral non-grounding system and a neutral grounding system by using arc extinction coil, respectively.



(a) Non-grounding system with $R_0 = 0.2\Omega$ (b) Grounding system with $R_0 = 0.2\Omega$

Figure 5: Zero sequence admittance angle comparisons between normal and fault lines

(4) *The fifth harmonic analysis*: the zero sequence current in a fault line contains a large number of odd harmonics, while the arc suppression coil will not greatly affect the magnitude and direction of the harmonic, so the harmonic can be used to detect the line. With the increase of harmonic frequency, the harmonic content becomes lower, so the fifth harmonic is usually used to realize fault line detection. Fig. 6 shows the comparisons of the fifth harmonic between normal and fault lines, which refer to the lines in a neutral non-grounding system and a neutral grounding system by using arc extinction coil, respectively.

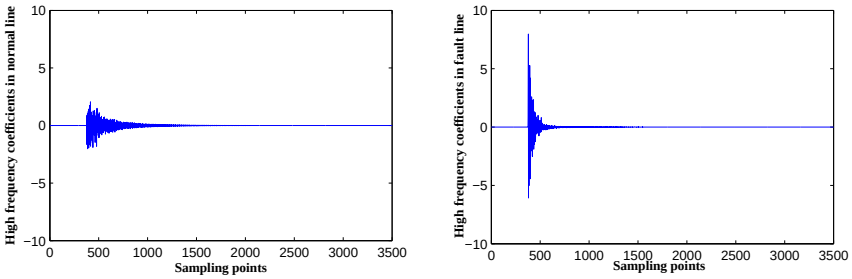


(a) Non-grounding system with $R_0 = 0.2\Omega$ (b) Grounding system with $R_0 = 0.2\Omega$

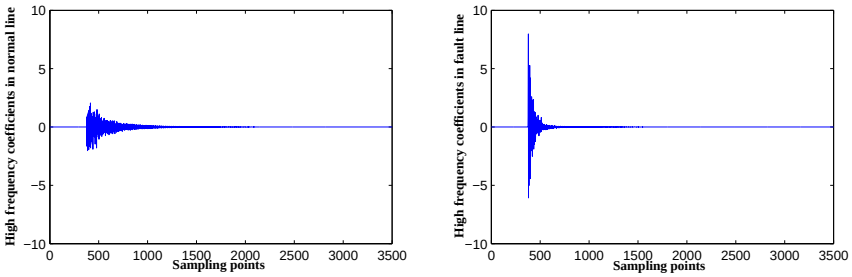
Figure 6: The fifth harmonic amplitude comparisons between normal and fault lines

(5) *Wavelet waveform analysis of zero sequence current*: the zero sequence current is decomposed into four layers by using wavelet transform, and the corresponding wavelet decomposition waveforms at scale 4 is shown in Fig. 7, where the waveforms in (a)–(f) are obtained from the neutral unearthed system and the figures (g)–(l) are gained from the neutral grounding system by using arc extinction coil. The wavelet energy polarity of the zero sequence current in the fault line is the opposite of the normal line, and the amplitude is the maximum. Thus, the fault line can be identified by comparing the energy

amplitude and polarity of the zero sequence current.



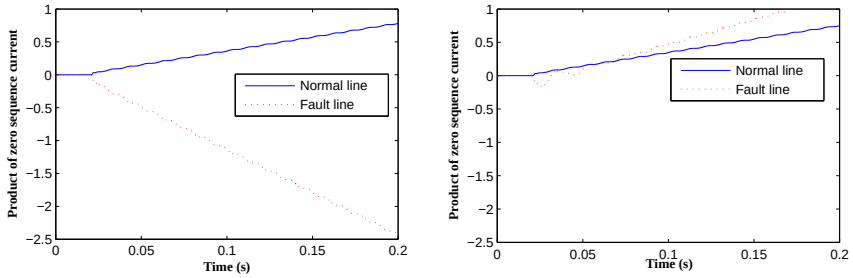
(a) Non-grounding system with $R_0 = 0.2\Omega$ (b) Grounding system with $R_0 = 0.2\Omega$



(a) Non-grounding system with $R_0 = 0.2\Omega$ (b) Grounding system with $R_0 = 0.2\Omega$

Figure 7: Wavelet waveform comparisons of zero sequence current between normal and fault lines

(6) *Transient zero sequence current analysis*: the direction and distribution of transient zero sequence currents shows similar phenomena to those of steady-state zero sequence currents. Fig. 8 show the comparisons of the products of zero sequence current between normal and fault lines, which refer to the lines in a neutral non-grounding system and a neutral grounding system by using arc extinction coil, respectively.



(a) Non-grounding system with $R_0 = 0.2\Omega$ (b) Grounding system with $R_0 = 0.2\Omega$

Figure 8: Comparisons of the products of zero sequence current between normal and fault lines

The analysis of the features discussed in this section indicates that there exists a difference between normal and fault lines so that the features can be used to detect the fault line in a small grounding system.

3.2 Fault measure

Following the feature analysis, a fault measure is discussed to quantify the possibility of a fault line [5, 33]. This article defines a fault measure as a real variable between 0 and 1. The smallest and biggest values 0 and 1 indicate that the line is normal and is faulty, respectively. Suppose that the number of lines in a distribution network is N and the number of fault line detection methods is H. The fault measure function is constructed by adopting line detection method h ($h = 1, 2, \ldots, N$) for line k ($k = 1, 2, \ldots, N$) is $X_h(k) = X_{rh}(k)X_{ah}(k)$, where $X_{rh}(k)$ is the relative fault measure function reflecting the difference of the fault measures between one line and the other lines in the distribution network; and $X_{ah}(k)$ is the reliability fault measure function reflecting the reliable degree of the fault measure.

As usual, different specific calculation methods are used for different fault features. To elaborate on the fault measure, we take the zero sequence current as an example. In a small current grounding system, the absolute values of the magnitudes of the zero sequence current in the fault line are equal to the sum of the absolute values of the amplitudes of all the normal lines. For line k, the magnitude of the zero sequence current is $|I_k|$. If $x = |I_k|/\sum_{n=1}^{N} |I_n|$ is big, the line tends to a fault line, conversely, the line is inclined to a normal one. The relative fault measure function of the line is described as follows:

$$X_{r1}(k) = \begin{cases} 1, & x \geq 0.5; \\ \frac{2N}{N-1}x - \frac{1}{N-1}, & \frac{1}{2N} < x < 0.5; \\ 0, & x \leq \frac{1}{2N}. \end{cases} \tag{1}$$

About the reliability fault measure function, when a single-phase-to-ground fault occurs in the system, the amplitudes of the zero sequence current in all lines reach to the maximum. If $x = |I_{max}|/\sum_{n=1}^{N} |I_n|$ is big, the fault characteristics of this method are obvious, on the contrary, they are not obvious. The reliability fault measure function of the line is described as follows:

$$X_{a1}(k) = \begin{cases} 1, & x \geq 0.5; \\ \frac{10N}{5N-6}x - \frac{6}{5N-6}, & \frac{3}{5N} < x < 0.5; \\ 0, & x \leq \frac{3}{5N}. \end{cases} \tag{2}$$

3.3 Information gain degree

Information gain degree is introduced to weight the importance of the six types of fault features in a small current grounding system. Suppose that X is the data for a particular sample group like the lines in this paper and A is the attribute set for the fault line detection methods, $A = \{H_1, H_2, \ldots, H_m\}$. The lines are divided into two types: fault and normal ones. So $X = \{X_1, X_2\}$, where $|X_i|$ is the number of sample instances of class i and $|X|$ is the total number of sample instances of X.

If $H(X_i)$ is the probability that a sample instance belongs to class i, then $H(X_i) = |X_i|/|X|$. Assume that $M(X)$ is the degree of uncertain information about X and it is $\sum_{i=1}^{m}[-H(X_i)\log_2 H(X_i)]$, where method H has the characters $h_1, h_2, \ldots, h_j$. In the case of $H = h_j$, C_{ij} is the number of sample instances of class i and Y_j is the total number of sample instances, thus, the probability $H(X_i|H = h_j)$ is $C_{ij}/|Y_i|$, the conditional entropy of H and the classification information entropy are described as $M(X, H = h_j) = \sum_j[-H(X_i|H = h_j)\log_2 H(X_i|H = h_j))]$,

and $B(X,H) = \sum_j [H(X_i|H = h_j)M(X, H = h_j)]$. The information gain of H is $\Delta Q_h = M(X) - B(X,H) = M(X) - \sum_j [H(X_i|H = h_j)M(X, H = h_j)]$. The information gain degree is normalized and the weight of the line detection method is $X_{mh}(k) = \Delta Q_h / \Sigma \Delta Q$.

3.4 Rough set theory

In this study, rough set theory is used to reduce the fault features in a small current grounding system. Rough set theory is a tool to analyze the incomplete and imprecise data [24, 43]. It is used to simplify the rule reduction to information system. The main ideas of rough set theory include knowledge reduction and domain reduction, which are described in a very brief way as follows. Knowledge reduction: suppose that R is an equivalence relation, $ind(R)$ represents the intersection of all the equivalence relations in R, $r \in R$. If $ind(R) = ind(R - \{r\})$, then r can be omitted, otherwise, r cannot be omitted. Domain reduction: suppose that $F = \{X_1, X_2, \ldots, X_n\}$ is a set family, $X_j \in U$ (U represents the domain). If $\bigcap(F - \{X_i\}) = \bigcap F$, then X_i can be omitted, domain reduction can delete the same decision rules and redundant attribute values in an information system so as to obtain the minimal solution.

In this paper, the condition attributes are the fault line detection methods and the decision attribute decides whether a fault detection method corresponding to a kind of features is correct or not. Conditional attributes require to be discretized. Thus, this method is used to simplify the condition attributes and eliminate some unnecessary condition attributes and redundant decision rules. Finally, the decisive solution is obtained.

3.5 Fault line detection models with rFRSNPS

In this section, a fuzzy reasoning spiking neural P system with real numbers (rFRSNPS) [16] is used to build the fault line detection model. The definition of rFRSN P system is first briefly described. Subsequently, the reasoning algorithm and fuzzy production rules of fault line detection are discussed. Finally, the fault line detection model is presented.

Fuzzy reasoning Spiking neural P systems

The definition and reasoning algorithm of rFRSNPS is described in [16], due to space limitations, this paper no longer gives a specific process.

Fuzzy production rules of fault line detection

The rFRSN P systems contain two types of neurons: proposition neurons and rule neurons, the rule neurons also express the fuzzy production rules. In this paper, we will describe three different kinds of fault line selection rules for rFRSN P system.

(1) (*General Rules*) R_i ($CF = c_i$): IF $p_j(\theta_j)$ THEN $p_k(\theta_k)$, where p_j and p_k represent propositions, c_i represents the certainty factor of rule R_i, as a real number belonging to [0,1], θ_j and θ_k are real numbers in [0,1] representing the truth values of p_j and p_k, the truth value of p_k is calculated as $\theta_k = \theta_j * c_i$

(2) (*And Rules*) R_i ($CF = c_i$): IF $p_1(\theta_1)$ and ... and $p_{k-1}(\theta_{k-1})$ THEN $p_k(\theta_k)$, where p_1, ... p_k are proposition, c_i represents the certainty factor of rule R_i, as a real number belonging to [0,1], θ_1, ... θ_k are real numbers in [0,1] representing the truth values of p_1, ... p_k, the truth value of p_k is calculated as $\theta_k = min(\theta_1, \ldots, \theta_{k-1}) * c_i$.

(3) (*Or Rules*) R_i ($CF = c_i$): IF $p_1(\theta_1)$ or ... or $p_{k-1}(\theta_{k-1})$ THEN $p_k(\theta_k)$, where p_1, ... p_k are proposition, c_i represents the certainty factor of rule R_i, as a real number belonging to [0,1], θ_1, ... θ_k are real numbers in [0,1] representing the truth values of p_1, ... p_k, the truth value of p_k is calculated as $\theta_k = max(\theta_1, \ldots, \theta_{k-1}) * c_i$.

Fault line detection model

Based on the fuzzy production rules of fault line detection discussed above, we can establish the fault line detection model with rFRSNPS for a line. The model is shown in Fig. 9, where b, c, d, f and P_1 represent the fault line measure values of the zero sequence current phase, the zero sequence reactive power, the zero sequence admittance amplitude, the transient zero sequence current and fused fault measure, respectively. The rFRSNPS for fault line detection is described as follows:

$$\Pi_1 = (O, \sigma_1, \sigma_2, \ldots, \sigma_{18}, syn, in, out)$$

where

(1) O=$\{a\}$ is the singleton alphabet (a is called spike);

(2) $\sigma_1, \ldots, \sigma_{11}$ are proposition neurons corresponding to the propositions with fuzzy truth values $\theta_1, \ldots, \theta_{11}$;

(3) $\sigma_{12}, \ldots, \sigma_{18}$ are rule neurons, where σ_{12}, σ_{15} are *general* rule neurons; σ_{16}, σ_{17} are *and* rule neurons; σ_{18} are *or* rule neurons;

(4) $syn = \{(1,12), (2,13), (3,14), (4,15), (5,16), (5,17), (6,16), (7,17), (8,16), (8,17), (9,18), (10,18), (12,5), (13,6), (14,7), (15,8), (16,9), (17,10), (18,11),\}$;

(5) in=$\{\sigma_1, \ldots, \sigma_4\}$;

(6) out=$\{\sigma_{11}\}$.

4 Case studies

The distribution network system shown in Fig. 1 is simulated on MATLAB/Simulink to obtain training and testing samples for verifying the fault line detection. The transformer ratio is 110kV/35kV. The simulation time is 0.2s. The lines length are 10km, 15km, 20km, 28km, 35km, 50km, respectively, the positive sequence parameters of line are $0.17ohms/km$, $1.21mH/km$ and 36.6+j$172F/km$, the zero sequence parameters of line are $0.23ohms$, $5.48mH/km$ and $6pF/km$. In the neutral ungrounded system, the testing samples are considered for several values of fault initial phase(0, 45 and 90 degrees), fault location(10%, 50% and 90% of lines 1 to 6) and transition resistance(0.2, 20 and 2000 ohms).

In the process of the simulation experiment, we collect 162 features from each of lines 1 to 6 and totally 972 zero sequence signal features as data sets. Next, the fault line measure values of the zero sequence current amplitude, the zero sequence current phase, the zero sequence reactive power, the zero sequence admittance amplitude, the wavelet energy of zero sequence current and the transient zero sequence current can be calculated by using the zero sequence signal. In what follows, six cases in the small current grounding system are used to test the introduced approach.

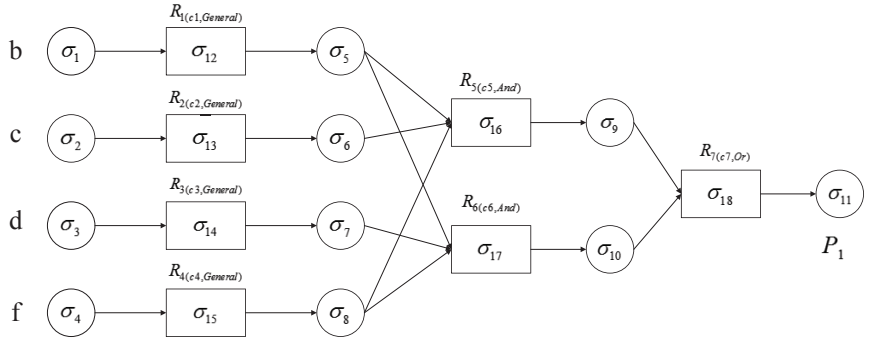


Figure 9: A certain line of fault line selection model with rFRSNPS

Case 1: single-phase-to-ground fault occurring in line 1: initial phase is 90 degrees; the fault is located at 90% of the line; transition resistance is 20 ohms.

Table 1 gives fault measure values of zero sequence current signals in lines 1–6. Then, the values between 0∼0.3, 0.3∼0.7 and 0.7∼1 are discretized as 0, 1, and 2, respectively. The results are shown in Table 2. Next, Eqs. subsection 3.3 are used to compute the weights of the six feature analysis methods and the results are 0.4833, 0.6500, 0.6500, 0.4833, 0.6500 and 0.6500, respectively. Subsequently, rough set theory is applied to reduce the condition attributes consisting of the features shown in Table 1. The reduced result is (b∧f∧(c∨d)), where b and f are the core attributes. Finally, the reasoning algorithm described above and fault line detection model with rFRSNPS are used to obtain the detection result. The detailed steps are described as follows:

Step 1: $g = 0$, $\boldsymbol{\theta}_0 = (0.9583, 0.9667, 0.6467, 0.9439, 0, 0, 0, 0, 0, 0, 0,)^T$,
$\boldsymbol{C} = diag(0.6500, 0.6500, 0.4833, 0.6500, 1, 1, 1)$,$\boldsymbol{\delta}_0 = 0$.

$$\boldsymbol{D}_1 = \begin{bmatrix} \boldsymbol{A}_1 & 0 \\ 0 & 1 \end{bmatrix}_{11\times7} \quad \boldsymbol{D}_2 = \begin{bmatrix} 0 & 0 \\ 0 & \boldsymbol{A}_2 \\ 0 & 0 \end{bmatrix}_{11\times7} \quad \boldsymbol{D}_3 = \begin{bmatrix} 0 & 0 \\ 0 & \boldsymbol{A}_3 \end{bmatrix}_{11\times7} \quad \boldsymbol{E}^T = \begin{bmatrix} 0 \\ \boldsymbol{B}_1 \end{bmatrix}_{11\times7}$$

$$\boldsymbol{A}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}_{4\times4} \quad \boldsymbol{A}_2 = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix}_{4\times3} \quad \boldsymbol{A}_3 = \begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}_{3\times2} \quad \boldsymbol{B}_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}_{7\times7}$$

Step 2: The firing condition of proposition neuron is satisfied and there is a postsynaptic rule neuron, so the proposition neuron fires and transmits a spike to the next rule neuron by the directed arc.

Step 3: $\boldsymbol{\delta}_{g+1} = (\boldsymbol{D}_1^T \otimes \boldsymbol{\theta}_g) + (\boldsymbol{D}_2^T \oplus \boldsymbol{\theta}_g) + (\boldsymbol{D}_3^T \odot \boldsymbol{\theta}_g)$, $\boldsymbol{\delta}_1 = (0.9583, 0.9667, 0.6467, 0.9439, 0, 0, 0)^T$.

Step 4: $\boldsymbol{\delta}_1 \neq 0$. The reasoning algorithm continues.

Step 5: $g = 1$.

Step 6: $\boldsymbol{\theta}_g = \boldsymbol{E}^T \odot (D \otimes \boldsymbol{\delta}_g)$, $\boldsymbol{\theta}_1 = (0, 0, 0, 0, 0.6229, 0.6284, 0.3126, 0.6135, 0, 0, 0)^T$.

Step 7: $\boldsymbol{\delta}_{g+1} = (\boldsymbol{D}_1^T \otimes \boldsymbol{\theta}_g) + (\boldsymbol{D}_2^T \oplus \boldsymbol{\theta}_g) + (\boldsymbol{D}_3^T \odot \boldsymbol{\theta}_g)$, $\boldsymbol{\delta}_2 = (0, 0, 0, 0, 0.6135, 0.3126, 0)^T$.

Step 8: $\boldsymbol{\delta}_2 \neq 0$. The reasoning algorithm continues.

Step 9: $g = 2$.

Step 10: $\boldsymbol{\theta}_g = \boldsymbol{E}^T \odot (D \otimes \boldsymbol{\delta}_g)$, $\boldsymbol{\theta}_2 = (0, 0, 0, 0, 0, 0, 0, 0, 0.6135, 0.3126, 0)^T$.

Step 11: $\boldsymbol{\delta}_{g+1} = (\boldsymbol{D}_1^T \otimes \boldsymbol{\theta}_g) + (\boldsymbol{D}_2^T \oplus \boldsymbol{\theta}_g) + (\boldsymbol{D}_3^T \odot \boldsymbol{\theta}_g)$, $\boldsymbol{\delta}_3 = (0, 0, 0, 0, 0, 0, 0.6135)^T$.

Table 1: Fault measure values of zero sequence current signals in lines 1–6,
where CA, CP, RP, AA, WE and TC represent current amplitude, current phase, reactive power, admittance amplitude, wavelet energy and transient current, respectively.

line	CA	CP	RP	AA	WE	TC
1	0.4371	0.9583	0.9667	0.6467	0.9182	0.9439
2	0.0675	0.0167	0.4234	0.3175	0.0483	0.3203
3	0.0945	0.1083	0.1092	0.0945	0.2531	0.0000
4	0.0506	0.3917	0.0823	0.0506	0.0658	0.0000
5	0.1182	0.1518	0.3167	0.3182	0.2146	0.0186
6	0.3691	0.2417	0.2631	0.1694	0.0731	0.3034

Table 2: Discretized fault measure values,
where CA, CP, RP, AA, WE and TC represent current amplitude, current phase, reactive power, admittance amplitude, wavelet energy and transient current, respectively.

line	CA	CP	RP	AA	WE	TC
1	1	2	2	1	2	2
2	0	0	1	1	0	1
3	0	0	0	0	0	0
4	0	1	0	0	0	0
5	0	0	1	1	0	0
6	1	0	0	0	0	1

Step 12: $\boldsymbol{\delta}_3 \neq 0$. The reasoning algorithm continues.

Step 13: $g = 3$.

Step 14: $\boldsymbol{\theta}_g = \boldsymbol{E}^T \odot (D \otimes \boldsymbol{\delta}_g)$, $\boldsymbol{\theta}_3 = (0,0,0,0,0,0,0,0,0,0,0.6135)^T$.

Step 15: $\boldsymbol{\delta}_{g+1} = (\boldsymbol{D}_1^T \otimes \boldsymbol{\theta}_g) + (\boldsymbol{D}_2^T \oplus \boldsymbol{\theta}_g) + (\boldsymbol{D}_3^T \odot \boldsymbol{\theta}_g)$, $\boldsymbol{\delta}_4 = (0,0,0,0,0,0,0)^T$.

Step 16: $\boldsymbol{\delta}_4 = 0$. The reasoning algorithm ends. The reasoning result can be obtained from output neuron P_1. The pulse value of the spike in P_1 is $0.6135(> 0.5)$. The rest may be inferred in the same way. The reasoning results of the other five lines can also be obtained and the pulse values of the spike in P_2 to P_6 are 0.0109, 0, 0, 0.0121, $0.1571(< 0.2)$, respectively. So line 1 is the fault line.

Case 2: single-phase-to-ground fault occurring in line 2: initial phase is 90 degrees; the fault is located at 90% of the line; transition resistance is 20 ohms.

Similar to *Case 1*, the weights of the six feature analysis methods are 0.4833, 0.6500, 0.6500, 0.4971, 0.6500 and 0.6500, respectively. The reduced result is (c∧f∧(b∨d)), where c and f are the core attributes. The initial parameter matrices of rFRSNPS for fault line detection are as follows:

$\boldsymbol{\theta}_0 = (0.9252, 0.9321, 0.7172, 0.9986, 0, 0, 0, 0, 0, 0, 0)^T$,

$\boldsymbol{C} = diag(0.6500, 0.6500, 0.4971, 0.6500, 1, 1, 1)$, $\boldsymbol{\delta}_0 = 0$.

According to the reasoning algorithm, we can get:

$\boldsymbol{\theta}_1 = (0,0,0,0,0.6014, 0.6059, 0.3565, 0.6491, 0, 0, 0)^T$,

$\boldsymbol{\theta}_2 = (0,0,0,0,0,0,0,0,0.6014, 0.3565, 0)^T$,

$\boldsymbol{\theta}_3 = (0,0,0,0,0,0,0,0,0,0,0.6014)^T$, and $\boldsymbol{\delta}_4 = 0$.

So, the pulse value of the spike in P_2 is 0.6014 (> 0.5). The pulse values of the spike in P_1, P_3, P_4, P_5 and P_6 are 0.1109, 0.0251, 0.0459, 0 and 0.0241 (< 0.2), respectively. Thus, line 2 is the fault line.

Case 3: single-phase-to-ground fault occurring in line 3: initial phase is 90 degrees; the fault is located at 90% of the line; transition resistance is 2000 ohms.

Similarly, the weights of the six feature analysis methods are 0.1909, 0.6500, 0.6500, 0.4833, 0.6500 and 0.6500, respectively. The reduced result is (b∧c∧(d∨e∨f)), where b and c are the core attributes; e is the fault measure value of the wavelet energy of zero sequence current. The initial parameter matrices of the rFRSNPS for fault line selection are as follows:

$\boldsymbol{\theta}_0 = (0.9167, 0.8723, 0.6842, 0.5974, 0.8783, 0, 0, 0, 0, 0, 0, 0, 0)^T$,

$\boldsymbol{C} = diag(0.6500, 0.6500, 0.4833, 0.6500, 0.6500, 1, 1, 1, 1)$, $\boldsymbol{\delta}_0 = 0$.

According to the reasoning algorithm, we can get:

$\boldsymbol{\theta}_1 = (0,0,0,0,0,0.5959,0.5670,0.3307,0.3883,0.5709,0,0,0,0)^T$,

$\boldsymbol{\theta}_2 = (0,0,0,0,0,0,0,0,0,0,0.3307,0.3883,0.5670,0)^T$,

$\boldsymbol{\theta}_3 = (0,0,0,0,0,0,0,0,0,0,0,0,0,0.5670)^T$, and $\boldsymbol{\delta}_4 = 0$.

Thus, the pulse value of the spike in P_3 is 0.5670 (> 0.5). The pulse values of the spike in P_1, P_2, P_4, P_5 and P_6 are 0.0705, 0.1308, 0, 0.0090 and 0.1588 (< 0.2), respectively. So line 3 is the fault line.

Case 4: single-phase-to-ground fault occurring in line 4: initial phase is 0 degrees; the fault is located at 90% of the line; transition resistance is 2000 ohms.

The weights of the six feature analysis methods are 0.1909, 0.6500, 0.6500, 0.4971, 0.6500, 0.6500. The reduced result is (b∧f∧(c∨d)), where b and f are the core attributes. The initial parameter matrices of the rFRSNPS for fault line selection are as follows:

$\boldsymbol{\theta}_0 = (0.8918, 0.8816, 0.5999, 0.8871, 0, 0, 0, 0, 0, 0, 0)^T$,

$\boldsymbol{C} = diag(0.6500, 0.6500, 0.4971, 0.6500, 1, 1, 1)$, $\boldsymbol{\delta}_0 = 0$.

According to the reasoning algorithm, we can get:

$\boldsymbol{\theta}_1 = (0,0,0,0,0.5797,0.5730,0.2982,0.5766,0,0,0)^T$,

$\boldsymbol{\theta}_2 = (0,0,0,0,0,0,0,0,0.5730,0.2982,0)^T$,

$\boldsymbol{\theta}_3 = (0,0,0,0,0,0,0,0,0,0,0.5730)^T$, and $\boldsymbol{\delta}_4 = 0$.

So the pulse value of the spike in P_4 is 0.5730(> 0.5). The pulse values of the spike in P_1, P_2, P_3, P_5 and P_6 are 0.0644, 0.0203, 0.1860, 0.0164 and 0.0005 (< 0.2), respectively. Thus, line 4 is the fault line.

Case 5: single-phase-to-ground fault occurring in line 5: initial phase is 0 degrees; the fault is located at 50% of the line; transition resistance is 20 ohms.

The weights of the six feature analysis methods are 0.4833, 0.6500, 0.6500, 0.4833, 0.6500, 0.6500. The reduced result is (f∧e∧(b∨c)∧(c∨d)), where f and e are the core attributes. The initial parameter matrices of the rFRSNPS for fault line selection are as follows:

$\boldsymbol{\theta}_0 = (0.8866, 0.9167, 0.6879, 0.9336, 1, 0, 0, 0, 0, 0, 0, 0, 0)^T$,

$\boldsymbol{C} = diag(0.6500, 0.6500, 0.4833, 0.6500, 0.6500, 1, 1, 1, 1))$, $\boldsymbol{\delta}_0 = 0$.

According to the reasoning algorithm, we can get:

$\boldsymbol{\theta}_1 = (0,0,0,0,0,0.5763,0.5959,0.3325,0.6069,0.6500,0,0,0,0)^T$,

$\boldsymbol{\theta}_2 = (0,0,0,0,0,0,0,0,0,0,0.5764,0.3325,0.3325,0)^T$,

$\boldsymbol{\theta}_3 = (0,0,0,0,0,0,0,0,0,0,0,0,0,0.5764)^T$, and $\boldsymbol{\delta}_4 = 0$.

So the pulse value of the spike in P_5 is 0.5730 (> 0.5). The pulse values of the spike in P_1, P_2, P_3, P_4 and P_6 are 0.0283, 0, 0.1528, 0 and 0.0664 (< 0.2), respectively. Thus, line 5 is the fault line.

Case 6: single-phase-to-ground fault occurring in line 6: initial phase is 0 degrees; the fault is located at 10% of the line; transition resistance is 20 ohms.

The weights of the six feature analysis methods are 0.4833, 0.6500, 0.6500, 0.4833, 0.6500, 0.6500. The reduced result is (b∧c∧f∧(e∨d)), where b, c and f are the core attributes. The initial parameter matrices of the rFRSNPS for fault line selection are as follows:

$\boldsymbol{\theta}_0 = (0.9284, 0.8987, 0.5847, 0.9312, 0.9021, 0, 0, 0, 0, 0, 0, 0, 0)^T$,

$\boldsymbol{C} = diag(0.6500, 0.6500, 0.4833, 0.6500, 0.6500, 1, 1, 1, 1))$, $\boldsymbol{\delta}_0 = 0$.

According to the reasoning algorithm, we can get:

$\boldsymbol{\theta}_1 = (0, 0, 0, 0, 0, 0.6035, 0.5842, 0.2826, 0.6053, 0.5864, 0, 0, 0)^T$,

$\boldsymbol{\theta}_2 = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.2826, 0.5842, 0)^T$,

$\boldsymbol{\theta}_3 = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.5842)^T$, and $\boldsymbol{\delta}_4 = 0$.

So the pulse value of the spike in P_6 is 0.5842 (> 0.5). The pulse values of the spike in P_1, P_2, P_3, P_4 and P_5 are 0.1083, 0.0613, 0, 0.0328 and 0.0871 (< 0.2), respectively. Thus, line 6 is the fault line.

The experimental results of the six cases in a small grounding system indicate that the proposed fault line detection approach is not affected by fault locations, fault resistance and fault closing angles.

5 Conclusions

In this paper, a novel approach is introduced by fuzzy reasoning spiking neural P systems to detect fault lines in a small current grounding system. The feature analysis is performed on steady and transient components of zero sequence current of a small current grounding system. Steady state features consist of zero sequence current signal amplitudes, zero-sequence reactive power, zero sequence admittance and the fifth harmonic. Transient features are composed of wavelet energy of zero sequence current and transient zero sequence current signal amplitudes. Experiments conducted on several cases of a distribution network system verify the feasibility and correctness of the presented approach. Future work will focus on the improvement of the fault line detection accuracy and the reliability of results. Following this work, micro grids and smart grids will be also considered in the future study.

Acknowledgment

This work was supported by National Natural Science Foundation of China (61672437, 61702428) and by Sichuan Science and Technology Program (2018GZ0185, 2018GZ0085, 2017GZ0159).

Bibliography

[1] Chen, Z.L.; Fan, C.J. (2006); Fault line selection for small current neutral grounding system based on the fifth harmonic current mutation in distribution system, *Proc. CSEE*, 18(5), 37–40, 2006.

[2] Chen, Z.; Zhang, P.; Wang, X.; Shi, X.; Wu, T.; Zheng, P. (2016); A computational approach for nuclear export signals identification using spiking neural P systems, *Neural Comput Appl*, 29(3), 695–705, 2016.

[3] Dzitac, I. (2015); Impact of membrane computing and P systems in ISI WoS. Celebrating the 65th birthday of Gheorghe Păun, *International Journal of Computers Communications & Control*, 10(5): 617-626, 2015.

[4] Dong, X.; Shi, S. (2008); Identifying single-phase-to-ground fault feeder in neutral non effectively grounded distribution system using wavelet transform, *IEEE Trans on Power Deliver*, 23(4), 1829–1837, 2008.

[5] Fan, L. P.; Yuan, Z.Q.; Zhang, K. (2009); System with insulated neutral point earthing of fault line detection fusion technology study based on fuzzy and rough set theory, *Central China Electric Power*, 1, 7–11, 2009.

[6] Frisco, P.; Gheorghe, M.; Pérez-Jiménez, M.J. (Eds.) (2014); *Applications of membrane computing in systems and synthetic biology*, Springer, Heidelberg, 2014.

[7] He, J., Xiao, J., Liu, X., Wu, T., Song, T. (2015); A novel membrane-inspired algorithm for optimizing solid waste transportation, *Optik*, 126(23), 3883–3888, 2015.

[8] Huang, K.; Wang, T.; He, Y.; Zhang, G.; Pérez-Jiménez, M. J. (2016); Temporal fuzzy reasoning spiking neural P systems with real numbers for power system fault diagnosis, *J Comput Theor Nanosci*, 13(6), 3804–3814, 2016.

[9] Huang, T.; Voronca, S. L.; Purcarea, A.; Estebsari, A.; Bompard, E. (2014); Analysis of chain of events in major historic power outages, *Adv Electr Comput Eng*, 14(3), 63-70, 2014.

[10] He, Y.; Wang, T.; Huang, K.; Zhang, G.; Pérez-Jiménez, M.J. (2015); Fault diagnosis of metro traction power systems using a modified fuzzy reasoning spiking neural P system, *Rom J Inf Sci Technol*, 18(3), 256–272, 2015.

[11] Ionescu, M.; Păun, Gh.; Yokomori, T. (2006); Spiking neural P systems, *Fund Inform*, 71(2-3), 279–308, 2006.

[12] Jiang, K.; Chen, W.; Zhang, Y.; Pan, L. (2016); On string languages generated by sequential spiking neural P systems based on the number of spikes, *Nat Comput*, 15(1), 87–96, 2016.

[13] Jiang, K.; Pan, L. (2016); Spiking neural P systems with anti-spikes working in sequential mode induced by maximum spike number, *Neurocomputing*, 171, 1674–1683, 2016.

[14] Jia, Q.; Shi, L.; Wang, N.; Dong, H. (2012); A fusion method for ground fault line detection in compensated power networks based on evidence theory and information entropy, *Trans China Electrotech Soc*, 27(6): 191–197, 2012.

[15] Liang, R.; Xin, J.; Wang, C.L.; Li, G.X.; Tang, J.J. (2010); Fault line selection in small current grounding system by improved active component method, *High Voltage Eng*, 36(2), 375–379, 2010.

[16] Liu, X.; Li, Z.; Suo, J.; Liu, J.; Min, X. (2015); A uniform solution to integer factorization using time-free spiking neural P system, *Neural Comput Appl*, 26(5): 1241–1247, 2015.

[17] Liu, X.; Li, Z.; Liu, J.; Liu, L.; Zeng, X. (2015); Implementation of arithmetic operations with time-free spiking neural P systems, *IEEE Trans on Nanobiosci*, 14(6), 617–624, 2015.

[18] Păun, Gh. (2000); Computing with membranes, *J Comput System Sci*, 61(1), 108–143, 2000.

[19] Păun, Gh. (2016); Membrane computing and economics: A general view, *International Journal of Computers Communications & Control*, 11(1), 105-112, 2016.

[20] Păun, Gh.; Rozenberg, G.; Salomaa, A. (Eds.) (2010); *The Oxford handbook of membrane computing*, Oxford University Press, New York, 2010.

[21] Peng, H.; Wang, J.; Pérez-Jiménez, M.J.; Wang, H.; Shao, J.; Wang, T. (2013); Fuzzy reasoning spiking neural P system for fault diagnosis, *Inform Sciences*, 235(20), 106–116, 2013.

[22] Pan, L.; Păun, Gh. (2009); Spiking neural P systems with anti-spikes, *International Journal of Computers Communications & Control*, 4(3), 273–282, 2009.

[23] Pan, L.; Păun, Gh.; Zhang, G.; Neri, F. (2017); Spiking neural P systems with communication on request, *Int J Neural Syst*, 27(8), 1750042, 2017.

[24] Pawlak, Z. (1998); Rough set theory and its applications to data analysis, *Cybernet Syst*, 29(7), 661–688, 1998.

[25] Rong, H.; Zhu, M.; Feng, Z.; Zhang, G.; Huang, K. (2017); A novel spproach to fault classification of power transmission lines using singular value decomposition and fuzzy reasoning spiking neural P systems, *Rom J Inf Sci Technol*, 20(1), 18-31, 2017.

[26] Song, B.; Pérez-Jiménez, M.J.; Pan, L. (2015); Computational efficiency and universality of timed P systems with membrane creation, *Soft Comput*, 19(11), 3043–3053, 2015.

[27] Shu, H.; Qiu, G.; Li, C.; Peng, S. (2010); A fault line selection algorithm using neural network based on S-transform energy, *Proc. 6th Internat Conf Nat Comput*, 3, 1478–1482, 2010.

[28] Song, T.; Zheng, H.; Juanjuan (2014); Solving vertex cover problem by tissue P systems with cell division, *Appl Math Inf Sci*, ISSN 2325-0399, 8(8), 333-337, 2014.

[29] Song, T.; Zheng, P.; Wong, M. L. D.; Wang, X. (2016); Design of logic gates using spiking neural P systems with homogeneous neurons and astrocytes-like control, *Inform Sciences*, 372, 380-391, 2016.

[30] Sang, Z.; Pan, Z.; Li, L.; Zhang, H. (1997); A new approach of fault line identification, fault distance measurement and fault location for single phase-to-ground fault in small current neutral grounding system, *Power Syst Technol*, 21(10), 50–52, 1997.

[31] Tang, Y.; Chen, K.; Chen, Q.; Dong, H.B. (2005); Study on earthed fault location method in indirectly grounding power system using maximum value of absolute value summation of measurement admittance mutual difference, *Proc. CSEE* , 25(6), 49–54, 2005.

[32] Voronca, S. L.; Voronca, M. M.; Huang, T.; Purcărea, A. A. (2015); Applying the analytic hierarchy process to rank natural threats to power system security, *U P B Sci Bull Ser C*, 77(3), 269-280, 2015.

[33] Wang, B.; Yu, C.K.; Ye, J.; Bai, Y. (2011); Fault line selection method for single phase-to-ground faults of multi-criteria information integrated with lower current grounding power system based on fuzzy theory, *Guangdong Electric Power*, 9, 24–28, 2011.

[34] Wang, J.; Shi, P.; Peng, H.; Pérez-Jiménez, M.J.; Wang, T. (2013); Weighted fuzzy spiking neural P systems, *IEEE Trans on Fuzzy Syst*, 21(2), 209–220, 2013.

[35] Welfonder, T.; Leitloff, V.; Fenillet, R.; Vitet, S. (2000); Location strategies and evaluation of detection algorithms for earth faults in compensated MV distribution systems, *IEEE Trans on Power Deliver*, 15(4), 1121–1128, 2000.

[36] Wang, T.; Zhang, G.; Pérez-Jiménez, M.J.(2015); Fuzzy membrane computing: theory and applications, Solving vertex cover problem by tissue P systems with cell division, *International Journal of Computers Communications & Control*, 10(6), 144–175, 2015.

[37] Wang, T.; Zhang, G.; Zhao, J.; He, Z.; Wang, J.; Pérez-Jiménez, M.J. (2015); Fault diagnosis of electric power systems based on fuzzy reasoning spiking neural P systems, *IEEE Trans on Power Syst*, 30(3), 1182–1194, 2015.

[38] Wang, T.; Zeng, S.; Zhang, G.; Pérez-Jiménez, M.J.; Wang, J. (2015); Fault section estimation of power systems with optimization spiking neural P systems, *Rom J Inf Sci Technol*, 18(3), 240–255, 2015.

[39] Wang, T.; Zhang, G.; Pérez-Jiménez, M.J.; Cheng, J. (2015); Weighted fuzzy reasoning spiking neural P systems: application to fault diagnosis in traction power supply systems of high-speed railways, *J Comput Theor Nanosci*, 12(7), 1103–1114, 2015.

[40] Wu, T.; Zhang, Z.; Pan, L. (2016); On languages generated by cell-like spiking neural P systems, *IEEE Trans Nanobiosci*, 15(5), 455–467, 2016.

[41] Wei, X.; Yang, D. (2015); An adaptive fault line selection method based on wavelet packet comprehensive singular value for small current grounding system, *Proc. ICDRPT*, 1110–1114, 2015.

[42] Xiong, G.; Shi, D.; Zhu, L.; Duan, X. (2013); A new approach to fault diagnosis of power systems using fuzzy reasoning spiking neural P systems, *Math Probl Eng*, 2013(1), 211–244, 2013.

[43] Yao, Y.Y. (2001); Information granulation and rough set approximation, *Int J Intell Syst*, 16(1), 87–104, 2001.

[44] Zhang, G.; Cheng, J.; Gheorghe, M.; Meng, Q. (2013); A hybrid approach based on differential evolution and tissue membrane systems for solving constrained manufacturing parameter optimization problems, *Appl Soft Comput*, 13(3), 1528–1542, 2013.

[45] Zhang, G.; Rong, H.; Neri, F., Pérez-Jiménez, M.J. (2014); An optimization spiking neural P system for approximately solving combinatorial optimization problems, *Int J Neural Syst*, 24(05), 1440006, 2014.

[46] Zhang, G.; Gheorghe, M.; Pérez-Jiminez, M.J. (2017); *Real-life applications with membrane computing*, Springer International Publishing, Berlin, 2017.

[47] Zhou, Z.; He, J.; Li, X.; Zou, Y. (2006); Research on the novel comprehensive fault line selection method for the NUGS based on the fuzzy theory, *Proc. ICPST*, 1–6, 2006.

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 13(4), 537-549, August 2018.

Enhanced Interconnection Model in Geographically Interdependent Networks

D.F. Rueda, E. Calle, X. Wang, R.E. Kooij

Diego F. Rueda*, Eusebi Calle
Institute of Informatics and Applications
Universitat de Girona
P-IV Building, Campus Montilivi, Girona, Spain
u1930599@campus.udg.edu, eusebi.calle@udg.edu
*Corresponding author: u1930599@campus.udg.edu

Xiangrong Wang, Robert E. Kooij
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
Mekelweg 4, Delft, The Netherlands
X.Wang-2@tudelft.nl, robert.kooij@tno.nl

Abstract: Interconnection between telecommunication networks and other critical infrastructures is usually established through nodes that are spatially close, generating a geographical interdependency. Previous work has shown that in general, geographically interdependent networks are more robust with respect to cascading failures when the interconnection radius (r) is large. However, to obtain a more realistic model, the allocation of interlinks in geographically interdependent networks should consider other factors. In this paper, an enhanced interconnection model for geographically interdependent networks is presented. The model proposed introduces a new strategy for interconnecting nodes between two geographical networks by limiting the number of interlinks. Results have shown that the model yields promising results to maintain an acceptable level in network robustness under cascading failures with a decrease in the number of interlinks.
Keywords: Cascading failures, interdependent critical infrastructures, robustness, region-based interconnection.

1 Introduction

Interdependent networks depend on a set of Critical Infrastructures (CIs) that function collaboratively to produce and distribute the essential goods and services required for the defence and economic security of nations and the proper functioning of governments and society [11]. Natural disasters (hurricanes, earthquakes, tsunami, tornadoes, floods or forest fires), man-made disasters (Electromagnetic Pulse (EMP) or Weapons of Mass Destruction (WMD) or terrorist attacks), technology-related disasters (power grid blackouts, hardware failures, dam failures or nuclear accidents), or cyber-attacks (viruses, worms or denial of services attacks) are responsible for large-scale disasters in networks [14] [10]. Consequently, failures in critical infrastructures imply service disruptions that affect thousands of people, multiple communities, entire countries, or just one company [9].

Telecommunication networks play a vital role in supporting the control, monitoring, connectivity and data transportation services of a number of critical infrastructures, including banking and finance, emergency and government services, water supply, transportation networks, power grids and oil and gas distribution networks. The interconnection between the nodes of these CIs and telecommunication networks is usually carried out by their spatial proximity. This region-based interconnection model generates a geographically interdependent network in which two

nodes i and j, located in two separate networks are interconnected if the distance (d_{ij}) between them is less than or equal to a given radius (r). Because of such interconnections, failures that occur in one infrastructure can directly or indirectly affect the other and impact large regions with catastrophic consequences [9]. An example of a large-scale failure in interdependent networks is the Italian blackout of 2003, where a single failure in the power grid resulted in failures that propagated over a telecommunications network, ultimately affecting more than 55 million people [2]. Therefore, network topologies, the geographic locations of nodes and their interdependency relationships have a huge impact on how robust interdependent networks are designed and maintained [9].

In contrast to the one-to-one interconnection studied in previous work [2], geographically interdependent networks exhibit a one-to-multiple interdependency model i.e., one node in one network can depend on an arbitrary number of nodes in the other network [16]. In terms of the functional giant component, a geographically interdependent network is more robust with respect to cascading failures when r is large [16]. This is due to the fact that with the increase of r, a node tends to have more interconnection nodes which, in turn, will decrease the probability of that node failing as result of the failures of its interconnection nodes. However, the region-interconnection models proposed in our previous work [16] only consider the geographical distance between nodes to establish the interlink, whereas in most real scenarios, interlink allocation in geographically interdependent networks should be controlled with additional factors in order to mitigate other issues introduced by the large number of interlinks in each r e.g., high deployment cost or exceeding node capabilities.

In the literature, most of the studies have been focused on modifying the interconnection patterns, according to a certain strategy, in order to improve the robustness of interdependent networks against cascading failures. Yagan et al. [17] showed that the regular allocation of bidirectional interlinks always yields stronger robustness than random strategy and unidirectional interlinks do. Li et al. [14] allocated weighted interdependency links under limited budget to obtain a more robust interdependent cyber-physical network. Ji et al. [6] showed that the low Inter Degree-Degree difference addition strategy (IDD) and Random Inter Degree-degree difference addition strategy (RID) are superior to the existing four link addition strategies (random addition, low degree, low betweenness and algebraic connectivity based) in improving the robustness of interdependent networks with high average inter degree-degree difference. However, these studies are focused on interdependent networks where the geographical location of the nodes is not considered to establish an interlink.

J. Martín-Hernández et al. [8] showed the critical number of interlinks beyond which any further inclusion does not enhance the algebraic connectivity (λ_2) of an interdependent network. Therefore, controlling the number of interlinks in geographically interdependent networks is likely a valuable design feature in order to reduce the deployment cost of interdependent networks and not to exceed the capabilities of the nodes to be interconnected. Unlike prior efforts, the major contributions of this paper are: 1) proposing a new strategy for interconnecting nodes between two geographical networks by limiting the number of interlinks and 2) analyzing the impact of limiting the number of interlinks has on the robustness of geographically interdependent networks against cascading failures. As a study case, we focused on interdependent telecommunication networks because they can represent the interconnection of two internet service providers or can refer to multilayer networks. Moreover, in this paper we consider the vulnerability analysis of each network to a certain type of targeted attack to determine the influence the new region-based interconnection model has on the robustness of the resulting interdependent network.

The remainder of this paper is organized as follows: Section 2 describes the proposed interconnection model for region-based interdependent networks and cascading failure process in interdependent networks. Section 3 presents the topologies of the networks to be interconnected

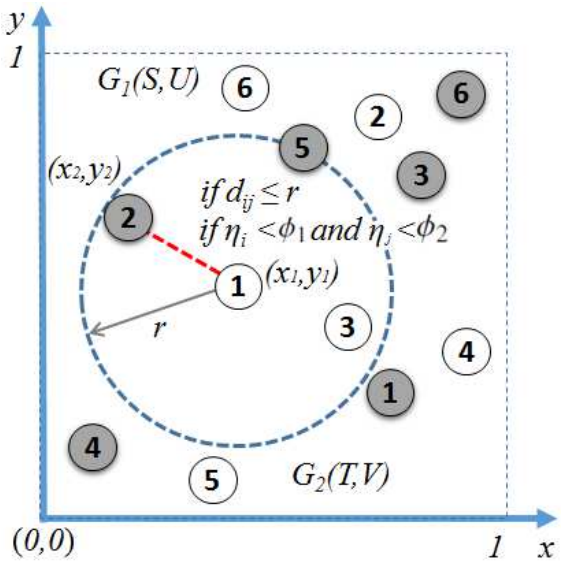


Figure 1: Enhanced interconnection model in geographically interdependent networks

and discusses the impact limiting the number of interlinks has on the robustness of region-based interdependent networks to cascading failures. Finally, Section 4 provides the conclusions and future work.

2 Concepts and models

In addition to the distance between the nodes, interlink allocation in geographic interdependent networks should be controlled by considering factors additional to the geographical constraint. This paper proposes a new region-based interconnection model in which a node i in network G_1 and a node j in network G_2 can be interconnected if 1) the distance d_{ij} between them is less than or equal to a given radius r and 2) the number of interlinks for nodes i and j do not exceed a given percentage for limiting the number of interlinks (ϕ_1 and ϕ_2, respectively). Our new strategy for interlink allocation is based on dividing the nodes in both networks into subsets in accordance with a certain nodal property. Thus, the model prevents ϕ_1 and ϕ_2 being exceeded for any node in G_1 and G_2, respectively.

The model proposed is illustrated in Fig. 1. The nodes in G_1 are represented by filled circles and the nodes in G_2 are represented by unfilled circles. For each node i in G_1, there is a set of nodes in G_2 that can be interconnected if the conditions 1) and 2) are satisfied. Consequently, in contrast to our previous work [16], an enhanced interconnection model for limiting the number of interlinks in geographically interdependent networks is generated. The remainder of this section presents the proposed region-based interconnection model in detail and describes the failure model involving cascading failures.

2.1 Interconnection model for limiting number of interlinks in geographically interdependent networks

Consider two undirected networks $G_1\,(S,U)$ and $G_2\,(T,V)$, each with a set of nodes (S,T) and a set of links (U,V) respectively. Denote N_1 and N_2 as the number of nodes in G_1 and G_2, respectively, and L_1 and L_1 as the number of links in G_1 and G_2, respectively. When G_1 and G_2 interact, a set of bidirectional interlinks I joining the two networks is introduced. Consequently,

an interdependent network is defined as $G(N, L) = (S \cup T, U \cup V \cup I)$ [8]. Let us define the adjacency matrix (A) of G as the $N \times N$ matrix:

$$A_{N\times N} = \begin{pmatrix} A_1 & \alpha B_{12} \\ \alpha B_{12}^T & A_2 \end{pmatrix}, \tag{1}$$

where α represents the coupling strength of the interaction, A_1 is the $N_1 \times N_1$ adjacency matrix of the network G_1, A_2 is the $N_2 \times N_2$ adjacency matrix of the network G_2, and B_{12} is the $N_1 \times N_2$ interconnection matrix representing the interlinks $S_i \leftrightarrow T_j$, between G_1 and G_2. Because we consider bidirectional interlinks, it follows that $B_{21} = B_{12}^T$ [8]. Let b_{ij} denote as the (i, j) entry in the B_{12} matrix, where $b_{ij} = 1$ if the node i and node j are interconnected, and $b_{ij} = 0$ if they are not. The interdependency matrix (B) of the whole system is given by:

$$B_{N\times N} = \begin{pmatrix} 0 & B_{12} \\ B_{12}^T & 0 \end{pmatrix} \tag{2}$$

In the region-based interconnection model previously proposed by us [16], the entry b_{ij} is determined by the geographical location of nodes. Let (x_i, y_i) and (x_j, y_j) denote the spatial coordinates for nodes i and j, then, b_{ij}= 1 if the Euclidean distance d_{ij} between node i in G_1 and node j in G_2 is smaller than a given threshold r. This link pattern generates a random geometric graph with a one-to-multiple interdependency model [16]. The Euclidean distances d_{ij} is given by:

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{3}$$

In the random geometric graph, a node i in G_1 can depend on an arbitrary number of nodes in G_2 that is no greater than N_2, and vice versa. When the distance between two nodes is considered as the unique interconnection constraint, some issues are evidenced. Specifically, the nodes in one network may have many interlinks from the other network, thus incurring high deployment cost. Note that the cost can be related to the economic investment required to construct an interlink. For instance, in the case of interdependent networks constructed by power grids and telecommunication networks, a new interlink has an associated deployment cost as a function of the cable length. Additionally, nodes in each network have limited capabilities to interconnect to a fixed number of nodes, and so the network's extension requires additional investments. Therefore, limiting the number of interlinks between the nodes in two networks contributes to keeping the deployment cost under control and adjusting to the operator's budget.

Let us define the new factor to be considered in the interconnection of geographically interdependent networks for limiting the number of interlinks in each network. For G_1, this factor is denoted as ϕ_1 and is given by:

$$\phi_1 = \frac{\eta_1}{N_2} \times 100\%, \tag{4}$$

where $\eta_1 \leq N_2$ is the maximum number of nodes from G_2 that each node in G_1 can interconnect to and N_2 is the number of nodes in G_2. Similary, the limit of interlinks (ϕ_2) for nodes in G_2 can be calculated analogous to (4). Therefore, the maximum number of interlinks that each node in G_1 and G_2 can interconnect to is controlled by ϕ_1 and ϕ_2.

As part of our proposal, the nodes in G_1 (G_2) are divided into μ_1 (μ_2) subsets of nodes, each with a maximum of η_1 (η_2) nodes. Subsets of nodes are a key aspect to controlling the allocation of a specific number of interlinks to each node. The number of nodes in a subset is directly related to the capacity of the nodes and the functionality performed by nodes in each network. For instance, in a fixed broadband access architecture, a subset of nodes in the access network can be interconnected to a subset of nodes in the core network. Moreover, a core network can

support the interconnection of a limited number of access nodes. Without loss of generality, a subset of nodes in a network can group nodes with similar properties or randomly. Then, the nodes of a subset in G_1 will be interconnected to the nodes of a subset in G_2 if the distance is less than or equal to a radius (r). As the number of nodes in each subset is limited, the number of interlinks in each node can be kept under control.

Let us consider that the nodes in G_1 are divided into μ_1 subsets of nodes, where μ_1 is given by:

$$\mu_1 = \begin{cases} round(\frac{N_2}{\eta_1}), & \text{if } \phi_1 < 50\% \\ 2, & \text{if } \phi_1 \geq 50\% \end{cases} \tag{5}$$

Similarly, the nodes in G_2 are divided into μ_2 subset of nodes, where μ_2 is given by:

$$\mu_2 = \begin{cases} round(\frac{N_1}{\eta_2}), & \text{if } \phi_2 < 50\% \\ 2, & \text{if } \phi_2 \geq 50\% \end{cases} \tag{6}$$

Let a_i denote the property value of node $i \in G_1$. Then, nodes in G_1 are ordered according to a_i, i.e.,, $a_1 \geq a_2 \geq ... \geq a_{i-1} \geq a_i \geq a_{i+1} \geq ... \geq a_{N_1-1} \geq a_{N_1}$. Moreover, let Γ_{S_g} denote the ordered set of nodes previously defined in G_1. If $\Gamma_{S_1}, \Gamma_{S_2}, ..., \Gamma_{S_{\mu_1}}$ represent the subsets of Γ_S, then, $\Gamma_S = \bigcup_{g=1}^{\mu_1} \Gamma_{S_g}$, and Γ_{S_g} is given by:

$$\Gamma_{S_g} = \begin{cases} \{i : (g-1) \times \eta_2 < i \leq g \times \eta_2\}, & \text{if } g < \mu_1 \\ \{i : (g-1) \times \eta_2 < i \leq N_1\}, & \text{if } g = \mu_1 \end{cases}, \tag{7}$$

where i represents the $i-th$ element in Γ_{S_g} and $g \in \{1, 2, ..., \mu_1\}$. Similarly, let c_j denote the property value of node $j \in G_2$. Then, nodes $j \in G_2$ are ordered according to c_j, i.e.,, $c_1 \geq c_2 \geq ... \geq c_{j-1} \geq c_j \geq c_{j+1} \geq ... \geq c_{N_2-1} \geq c_{N_2}$. Additionally, let Γ_{T_h} denote the ordered set of nodes previously defined in G_2. If $\Gamma_{T_1}, \Gamma_{T_2}, ..., \Gamma_{T_{\mu_2}}$ are subsets of Γ_T, then, $\Gamma_T = \bigcup_{H=1}^{\mu_2} \Gamma_{T_h}$, and Γ_{T_h} is given by:

$$\Gamma_{T_h} = \begin{cases} \{j : (h-1) \times \eta_1 < j \leq h \times \eta_1\}, & \text{if } h < \mu_2 \\ \{j : (h-1) \times \eta_1 < j \leq N_2\}, & \text{if } h = \mu_2 \end{cases}, \tag{8}$$

where j represents the $j-th$ element in Γ_{T_h} and $h \in \{1, 2, ..., \mu_2\}$

Let us define B_ϕ as an $N_1 \times N_2$ interconnection matrix, whose entries or elements are $b_{\phi_{ij}} = 1$ if nodes in the subset Γ_{S_g} are connected to nodes in the subset Γ_{T_h} for $g = h$, otherwise $b_{\phi_{ij}} = 0$. Accordingly, the B_ϕ matrix defines which nodes in the networks can be interconnected and establishes the limit for the number of interlinks that each node in the networks can handle. Thus, each node in G_1 or G_2 will have a maximum of η_1 or η_2 interconnected nodes, respectively.

Finally, let us redefine the dependency matrix B_{12}, whose entries are $b_{ji} = 1$ if $d_{ij} \leq r$ and $b_{\phi_{ij}} = 1$, otherwise $b_{ij} = 0$. Note that the new B_{12} matrix captures the interconnection conditions 1) and 2) proposed in this paper and thus the new interdependency matrix B, which is given by the equation (2), can be generated. Therefore, the nodes in each geographical network will interconnect with a limited number of interlinks, consequently improving the model defined in [16].

For simplicity, in this paper we consider that G_1 and G_2 have the same number of nodes ($N_1 = N_2$) and that all the nodes in the interdependent network have the same limit of interlinks ($\phi_1 = \phi_2$). Therefore, each network has $\mu_1 = \mu_2$ subsets of nodes with a maximum number of nodes $\eta_1 = \eta_2$. Figure 2 presents two geographical networks being interconnected by employing the interconnection proposal described in this section. As can be seen in Fig. 2, both networks

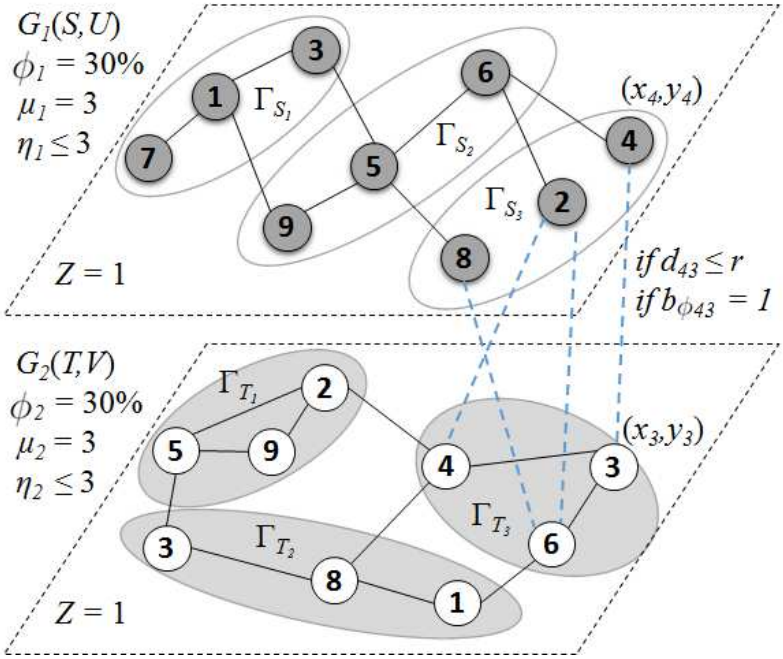


Figure 2: Subsets for limiting the number of interlinks in geographically interdependent networks

have $N_1 = N_2 = 9$ nodes and each node in G_1 and G_2 can support until $\phi_1 = \phi_2 = 30\%$ of nodes from the other. According to what has been described above, nodes in both networks are divided into $\mu_1 = \mu_2 = 3$ subsets, each one with a maximum of $\eta_1 = \eta_2 = 3$ nodes. Then, the B_ϕ matrix is generated with the subsets Γ_{S_g} and Γ_{T_h}. Finally, the interlinks between the nodes from G_1 and G_2 (dashed lines) are established if $d_{ij} \leq r$ and $b_{\phi_{ij}} = 1$.

2.2 Algorithm description

*Algorithm*1 summarizes the interconnection model proposed to limit the number of interlinks in geographically interdependent networks. *Algorithm*1 requires two networks (G_1 and G_2) to be interconnected, the percentage for limiting the number of interlinks (ϕ_1 and ϕ_2) and the radius (r). The output of *Algorithm*1 is a dependency matrix B_{12} with the conditions 1) and 2) previously described. As can be seen, *Algorithm*1 calculates the maximum number of nodes that a node can interconnect to (Lines 1 and 2) and the number of subsets (Lines 3 and 4). Then, the nodes are grouped in subsets according to one property (Lines 5 and 6). The interconnection matrix ($B_{\phi_{12}}$), in which each node in G_1 (G_2) has a maximum of η_1 (η_2) interconnected nodes (Line 7) is generated. Finally, the interdependency matrix B_{12} is generated by considering the distance constraint for a given r and the B_ϕ matrix (Lines 8 to 19). Thus, an enhanced region-based interconnection model is defined for interconnecting the G_1 and G_2 networks and the interdependency matrix B, which is given by the equation (2), can be generated from the resulting B_{12} matrix.

2.3 Cascading failure process in interdependent networks

Consider a geographically interdependent network G generated from the model proposed in this paper. When a random fraction of the nodes in G_1 fails, a cascading failure process is induced. We assume the node i in network G_1 is functional if a) at least one of its interconnected nodes in network G_2 is operative, and b) the node i belongs to the giant component of the

Algorithm 1 Interconnection model for limiting the number of interlinks in geographically interdependent networks.

Data: two geographical networks (G_1 and G_2), limit for number of interlinks (ϕ_1 and ϕ_2) and radius (r).
Result: dependency matrix B_{12}.
$\eta_1 = round(\phi_1 N_2/100)$
$\eta_2 = round(\phi_2 N_1/100)$
$\mu_1 = round(N_1/\eta_2)$
$\mu_2 = round(N_2/\eta_1)$
$\Gamma_{S_g} \leftarrow getSubsetNodes(S, \mu_1, \eta_2, nodal\_property)$
$\Gamma_{T_h} \leftarrow getSubsetNodes(T, \mu_2, \eta_1, nodal\_property)$
$B_\phi \leftarrow getB_\phi Matrix(\Gamma_{S_g}, \Gamma_{T_h}, \eta_1, \eta_2, \mu_1, \mu_2)$
for all $i \in S$ **do**
  for all $j \in T$ **do**
    $d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$
    if $d_{ij} \leq r$ and $b_{\phi_{ij}} == 1$ **then**
      $b_{ij} = 1$
    else
      $b_{ij} = 0$
    end if
  end for
end for
  return B_{12}

functional nodes in network G_1 [4]. Due to interdependency, the failed nodes in G_1 spread failures in G_2. As the assumptions *a*) and *b*) are also applied to the nodes j in network G_2, the failed nodes in G_2 spread failures back into G_1, and so on. The cascading failures continue until no more nodes fail. The remaining set of functional nodes is referred to as the *Largest Mutually Connected Component* (*LMCC*):

$$LMCC = \frac{n_1 + n_2}{N_1 + N_2}, \tag{9}$$

where n_1 and n_2 are the number of nodes that belong to the giant component of the functional nodes in G_1 and G_2, respectively, when the assumptions *a*) and *b*) are satisfied. The cascading failures described in this section can occur in real scenarios such as power grid blackouts [1] and disruptions in economic networks [15]. Note that [16] also considered the case in which a node in G_1 is functional if all of its interconnected nodes in G_2 are operational. Under that condition, in some cases, having more interconnected links makes the geographically interdependent network less robust. However, this case is outside the scope of this paper.

3 Simulation results and discussion

In this section, the topologies for geographically interdependent networks are described. Moreover, the impact limiting the number of interlinks has on the robustness of geographically interdependent network is analyzed.

Table 1: Nodes distribution in G_1 and G_2 according to interlink limits

$\phi_1 = \phi_2$	$\mu_1 = \mu_2$	$\eta_1 = \eta_2$
10%	10	5
25%	4	12
50%	2	25
75%	2	37
100%	1	50

3.1 Topologies for geographically independent networks

The geographically interdependent networks considered as the study case represent two backbone telecommunication networks being interconnected with bidirectional interlinks. The random connection property of a backbone telecommunications network is modeled using an Erdös-Rényi (ER) random graph with a Poisson nodal degree distribution [12]. This indicates that most nodes have approximately the same number of links close to the average nodal degree [3]. Although, scale-free or other graph models can be also used to model telecommunication networks, these are more associated to large networks (such as multi-autonomous systems networks). Moreover, some current backbone topologies are also scaling to other models which are out of the scope of this paper.

In order to analyze the impact the model proposed has on the robustness of interdependent networks against cascading failures, the Largest Mutually Connected Component ($LMCC$) is measured in 100 interdependent telecommunication networks. Each backbone telecommunication network to be interconnected is modeled as an ER random graph with $N_1 = N_2 = 50$ nodes and the average nodal degree ($\langle k \rangle$) equal to 6. The nodes in each network are placed uniformly in a two-dimensional square of the size $Z = 1$ i.e., each node in the G_1 and G_2 networks has as spatial coordinates (x, y), where $0 \leq x \leq 1$ and $0 \leq y \leq 1$.

The interconnection link pattern between the two ER graphs is conditioned by a given radius r. The number of interlinks in each node is limited by a given percentage ϕ. The number of subsets (μ_1, μ_2) and the maximum number of nodes that a node in G_1 and G_2 can interconnect with (η_1, η_2) are presented in Table 1. For instance, when $\phi = 25\%$, this is considered as the design constraint and, as such, the nodes in each network are divided into $\mu_1 = \mu_2 = 4$ subsets. Thus, for a given radius r, it is expected that each node in G_1 and G_2 will have a maximum of 12 interlinks.

As was described in subsection 2.1, a nodal property is also required to define how nodes in each network can be grouped. In the study case considered in this paper, node vulnerability to failures is selected as the property with which to group the nodes into subsets. In most real scenarios, the vulnerability of nodes to failures can be estimated from the historical failure database of their Operation Support Systems (OSS). However, given the difficulty of obtaining access to real data, centrality metrics could be used to measure the importance of nodes for the network connectivity under some failure scenarios [5]. Previous studies have revealed that backbone telecommunication networks modeled as ER are highly vulnerable to a sequential targeted attack based on nodal betweenness centrality (b_c) [13].

Figure 3 depicts a robustness analysis of the backbone telecommunication networks under targeted attacks when networks are not connected to other. The networks' robustness is quantified as a function of the Average Two Terminal Reliability ($ATTR$) metric [9]. As can be seen in Fig. 3, the telecommunication networks considered in this work exhibit high vulnerability to a sequential targeted attack by b_c. Whereas, the networks are more robust to a simultaneous

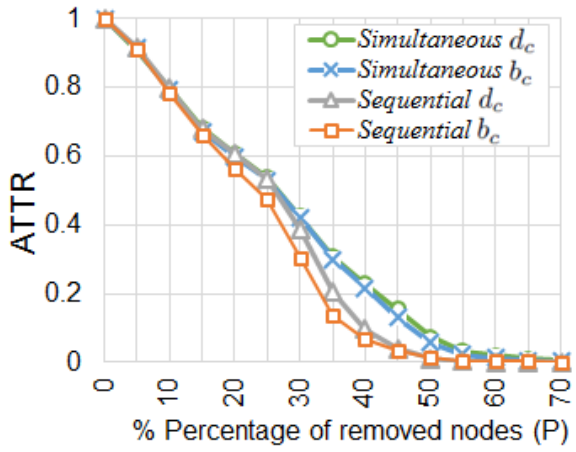


Figure 3: Robustness analysis of backbone telecommunication networks ($N_1 = 50$ and $\langle k \rangle = 6$) in a single scenario

targeted attack by b_c and sequential or simultaneous targeted attacks based on degree centrality (d_c). Consequently, node vulnerability in each ER network could be quantified by their b_c values i.e., the higher the betweenness centrality of node is, the higher the node's vulnerability is.

3.2 Analyzing the impact limiting the number of interlinks has on the robustness of geographically interdependent networks

To investigate the impact the region-based interconnection model has on the robustness of interdependent networks against cascading failures, the Largest Mutually Connected Component ($LMCC$) metric is measured when a fraction of nodes is removed. In the failure scenario considered in this paper, nodes in the network G_1 are removed (according to their vulnerability to a sequential targeted attack by b_c) until the percentage of removed nodes (P) is reached. Removing the nodes in G_1 leads to a cascading failure process as described in Section 2.3.

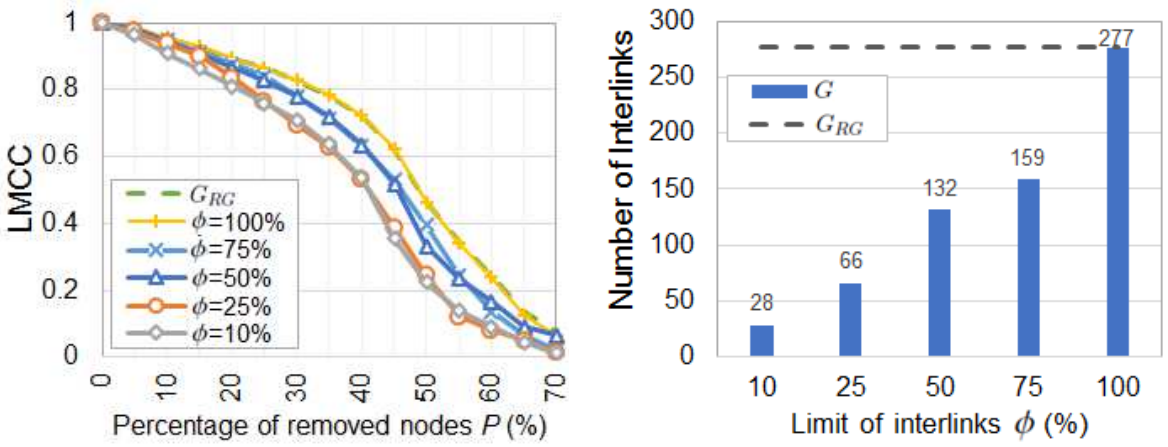
Although several geographically interdependent networks can be generated by varying the radius and the limit of the number of interlinks, the two scenarios considered as case studies are:

- Scenario 1: The radius (r) is fixed to 0.2 and the limit for the number of interlinks (ϕ) ranges from 25% to 100%. This scenario can represent a real situation in which a telecommunication network operator has a geographical area limited by a radius r and is interested in controlling the number of interlinks to other infrastructures.
- Scenario 2: The number of interlinks is limited to 25% and r is varied from 0.1 to $\sqrt{2}$. This scenario can be used by a telecommunication network operator who has a certain capacity in their network, but wants to restrict its coverage area to a certain radius r to interconnect to fewer number of nodes from other infrastructures.

Both scenarios are replicated in 100 interdependent networks. The robustness analysis presented in this section is the average of the $LMCC$ results measured in these interdependent networks.

Scenario 1: Robustness analysis in geographically interdependent networks against variations in the limit of interlinks (ϕ)

In this scenario, the radius (r) to interconnect the G_1 and G_2 networks is fixed to 0.2. Then, for a given limit in the number of interlinks (ϕ), the $LMCC$ of an interdependent network is



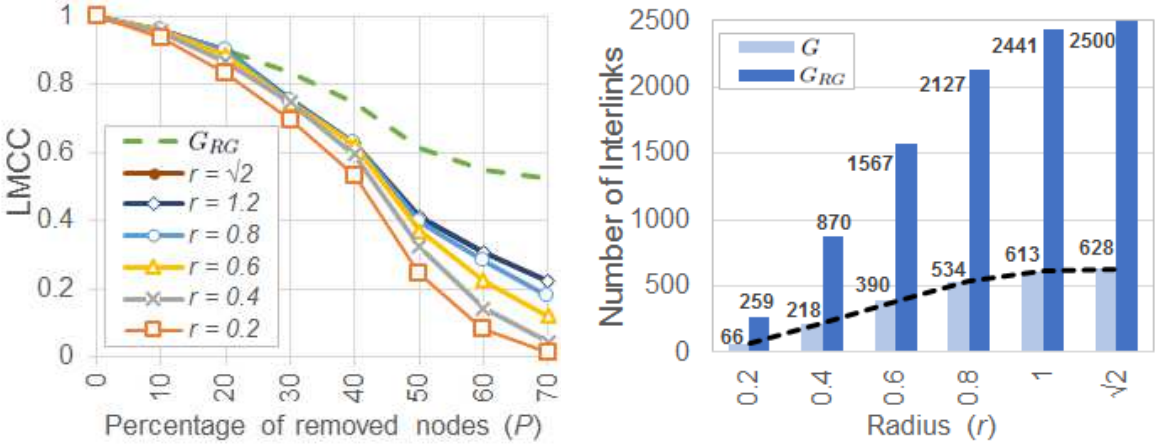
(a) (b)

Figure 4: Robustness analysis in geographically interdependent networks ($r = 0.2$) versus variations in the limit of interlinks (ϕ) a) Largest Mutually Connected Component ($LMCC$) as a function of the removed nodes (P) b) Number of interlinks as a function of ϕ

measured when a fraction of nodes (P) is removed in the G_1 network. Figure 4a depicts that for a given ϕ the $LMCC$ first decreases almost linearly with the increase in the fraction of removed nodes ($P \leq 35\%$). Later, the $LMCC$ dramatically decreases until the networks are completely disconnected. Networks with the highest slope in their $LMCC$ curves are those that have less ϕ. This is because with the decrease of ϕ, nodes in the G_1 and G_2 networks are divided into more subsets (μ_1 and μ_2, respectively) which decreases the probability for interconnecting a large number of nodes. Consequently, a node has fewer interconnected nodes and its failure probability is increased thanks to the failures of its interconnected nodes.

Also note that in Fig. 4a there is a zone ($P \leq 20\%$) in which the robustness of interdependent networks for a given ϕ is similar to the robustness reached by a network modeled according to [16] with $r = 0.2$ and without limiting the number of interlinks (G_{RG}). Moreover, in this zone all networks exhibit a high level of robustness against cascading failures ($LMCC > 0.8$). For example, when 20% of the nodes are removed from G_1 and after the cascading failure process, $LMCC = 0.89$ for $\phi = 100\%$ and 0.81 for $\phi = 10\%$. However, for $P > 20\%$, there are more differences between the $LMCC$ values reached by the networks with $\phi \leq 25\%$ and the network G_{RG}. However, in the case of networks with $\phi \geq 50\%$, their robustness remains near to that achieved by G_{RG} until $P \leq 40\%$. Therefore, for some P values, our model is able to maintain the $LMCC$ in values near those achieved by our previous work [16] when the number of interlinks is limited to a certain value of ϕ.

On other hand, as can be seen in Fig. 4b, the number of interlinks is under the maximum number of interlinks reached by the G_{RG} network for $\phi < 100\%$ (compare the dashed line G_{RG} and the blue bars G). This result is due to the strategy proposed in this paper whereby the nodes in the G_1 and G_2 networks are divided into subsets, with a maximum number of nodes η_1 and η_2, respectively. Thus, our new region-based interconnection model guarantees that the number of interlinks in geographically interdependent networks is maintained below the limit ϕ. For instance, when $\phi = 75\%$, the maximum number of interlinks in the interdependent networks is 159. Although this value is not exactly 75% of the maximum number of interlinks, it is below the limit of interlinks considered to be a design constraint.



(a) (b)

Figure 5: Robustness analysis in geographically interdependent networks ($\phi = 25\%$) versus variations in radius (r) a) Largest Mutually Connected Component ($LMCC$) as a function of removed nodes (P) b) Number of interlinks as a function of r

Scenario 2: Robustness analysis in geographically interdependent networks against variations in radius (r)

In this scenario, the interdependent telecommunication networks are the result of interconnecting the G_1 and G_2 networks by limiting the interlinks (ϕ) to 25% and varying the radius (r). The Largest Mutually Connected Component ($LMCC$) as a function of the fraction of removed nodes from the G_1 network is shown in Fig. 5a. Although the number of interlinks is limited to 25%, Fig.5a shows that geographically interdependent networks better resist cascading failures because of a major number of interlinks when the r is large. This result is to be expected as the nodes in the G_1 and G_2 networks tend to be more probable to interconnect to a greater number of nodes as a wide geographical area is defined by a larger radius r. For example, when 20% of the nodes are removed from G_1 and after the cascading failure process, $LMCC = 0.90$ for $r = 1.2$ and 0.83 for $r = 0.2$.

Additionally, Fig. 5a depicts a zone ($P \leq 20\%$) in which the robustness of geographically interdependent networks for a given radius r remains near to the robustness of a network modeled according to [16] where $r = \sqrt{2}$ and the number of interlinks is not limited (G_{RG}). In this zone, all geographically interdependent networks have the $LMCC > 0.8$. As the percentage of removed nodes in G_1 increases, the networks modeled with our new proposal maintain similar robustness levels until $P \leq 40\%$. Consequently, limiting the number of interlinks to a certain percentage ϕ trends to control interlink allocation against increases in radius r. Thus, our proposal based on subsets is effective in limiting the number of interlinks in geographically interdependent networks.

Regarding the number of interlinks, Fig. 5b shows that for a given radius r our model generates interdependent networks where the interlinks are around 25% of the maximum reached by each network G_{RG} (compare light blue and dark blue bars).The reason is because, independent of the selected radius (r), the model proposed in this paper restricts the number of nodes that a node in the G_1 and G_2 networks can interconnect with to η_1 and η_2, respectively. For example, when $r = 0.6$, the maximum number of interlinks in the interdependent networks is 390, and the number of interlinks per node is 4 on average. Consequently, for some P values our model yields promising results for maintaining network robustness under cascading failures by reducing the number of interlinks.

4 Conclusions

In this paper, an enhanced interconnection model in geographically interdependent networks has been proposed. In contrast to previous work, a new strategy based on subsets of nodes has been proposed to limit the number of interlinks in interdependent networks. The proposed region-based interconnection model has considered the percentage with which to limit the number of interlinks (ϕ) as a new factor in the design of geographically interdependent networks. Moreover, the impact limiting the number of interlinks has on the robustness of geographically interdependent networks against cascading failures has been analyzed.

The interconnection strategy proposed in this paper has proven to be effective in guaranteeing the number of interlinks in geographically interdependent networks is maintained under a certain limit ϕ. This is because for a given ϕ the nodes to be interconnected have been divided into subsets (μ_1, μ_2), each with a maximum number of nodes (η_1, η_2). Results indicate that in some scenarios ($P \leq 20\%$) the robustness for a given ϕ has been maintained at levels close to those reached by [16] ($LMCC \geq 0.80$). This is a relevant outcome because compared to the critical threshold at which $LMCC$ equals zero, quantifying the impact of a small percentage of node failures (P) is essential for network providers to prevent networks from collapsing.

Furthermore, the two scenarios that have been analyzed in this paper represent some situations in which the model proposed can be applied by network providers. Results have shown the robustness behaviour for geographically interdependent networks under cascading failures. In the first case, by limiting the coverage area to a certain radius r and varying the number of interlinks (ϕ), an interdependent networks is more robust against cascading failures when ϕ is increased. Meanwhile, in the second case, by limiting the number of interlinks to a certain ϕ and varying the radius r, the robustness increases for large values of r. In both cases, the results are because with the increase in the number of interlinks, a node tends to be less likely to fail from the failures of its interconnection nodes.

In the future work, the proposed region-based interconnection model can be studied in other interdependent networks and validated with real-world data. Moreover, an in-depth cost-benefit analysis of limiting the number of interlinks in geographically interdependent networks can be carried out.

Acknowledgements

This research was supported in part by the Spanish Ministry of Economy and Competitiveness and the DURSI Consolidated Research Group (CSI Reference SGR-1469) through the GIROS Project (TEC2015-66412-R).

Bibliography

[1] Andersson, G. et. al. (2005); Causes of the 2003 Major Grid Blackouts in North America and Europe, and Recommended Means to Improve System Dynamic Performance, *IEEE Trans. on Power Systems*, 20 (4), 1922-1928, 2005.

[2] Buldyrev, S. V.; Parshani, R.; Paul G.; Stanley, H. E.; Havlin, S. (2010); Catastrophic cascade of failures in interdependent networks, *Nature*, 464, 1025-1028, 2010.

[3] Erdos, P.; Renyi, A. (1960); On the evolution of random graphs, *Publication of the Mathematical Institute of the Hungarian Academy of Sciences*, 5, 17-61, 1960.

[4] Gao, J.; Buldyrev, S. V.; Stanley, H. E.; Havlin, S. (2012); Networks formed from interdependent networks, *Nat. Phys.*, 8, 40-48, 2012.

[5] Iyer, S.; Killingback, T.; Sundaram, B.; Wang, Z. (2013); Attack robustness and centrality of complex networks, *PLoS ONE*, 8 (4), e59613, 2013.

[6] Ji, X.; Wang, B.; Liu, D.; Chen, G.; Tang, F.; Wei, D.; Tu, L. (2016); Improving interdependent networks robustness by adding connectivity links, *Physica A*, 444, 9-19, 2016.

[7] Li, X.; Wu, H.; Scoglio, C.; Gruenbacher, D. (2015); Robust allocation of weighted dependency links in cyber-physical networks, *Physica A*, 433, 316-327, 2015.

[8] Martín-Hernández, J.; Wanga, H.; Van Mieghem, P.; D'Agostino, G. (2014); Algebraic connectivity of interdependent networks, *Physica A*, 404, 92-105, 2014.

[9] Neumayer, S.; Modiano, E. (2016); Network Reliability under Geographically Correlated Line and Disk Failure Models, *Computer Networks*, 94, 14-28, 2016.

[10] Ouyang, M. (2014); Review on modeling and simulation of interdependent critical infrastructure systems, *Reliability Engineering & System Safety*, 121, 43-60, 2014.

[11] Rinaldi, S. M.; Peerenboom, J.P.; Kelly, T. K. (2001); Identifying, Understanding, and Analyzing Critical Infrastructure Dependencies, *IEEE Control Systems Magazine*, 21 (6), 11-23, 2001.

[12] Rueda, D. F.; Calle E. (2017), Using interdependency matrices to mitigate targeted attacks on interdependent networks: A case study involving a power grid and backbone telecommunications networks, *International Journal of Critical Infrastructure Protection*, 17, 3-12, 2017.

[13] Rueda, D. F.; Calle, E.; Marzo, J. L. (2017); Robustness Comparison of 15 Real Telecommunication Networks: Structural and Centrality Measurements, *J. Netw. Syst. Manage*, 25 (2), 269-289, 2017.

[14] Sterbenz, J.P.G.; Hutchison, D.; Çetinkaya, E.K.; Jabbar, A.; Rohrer J.P.; Scholler, M.; Smith, P. (2010); Resilience and survivability in communication networks: Strategies, principles, and survey of disciplines, *Computer Networks*, 54(8), 1245-1265, 2010.

[15] Schweitzer, F.; Fagiolo, G.; Sornette, D.; Vega-Redondo, F.; Vespignani, A., White, D. R. (2009); Economic Networks: The New Challenges, *Science*, 325 (5939), 422-425, 2009.

[16] Wang, X.; Kooij, R.E.; Van Mieghem, P. (2016); Modeling region-based interconnection for interdependent networks, *Phys. Rev. E.*, 94, 042315(14), 2016.

[17] Yagan, O.; Qian, D.; Zhang, J.; Cochran, D. (2012); Optimal allocation of interconnecting links in cyber-physical systems: Interdependence, cascading failures, and robustness, *EEE Trans. Parallel Distrib. Syst.*, 23 (9), 1708-1720, 2012.

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 13(4), 550-565, August 2018.

Improving DTNs Performance by Reduction of Bundles Redundancy using Clustering Algorithm

R.O. Schoeneich, P. Prus

Radosław Olgierd Schoeneich*, Piotr Prus
Institute of Telecommunications
Warsaw University of Technology, Poland
*Corresponding author: rschoeneich@tele.pw.edu.pl
p.prus@stud.elka.pw.edu.pl

Abstract: This article presents complex clustering algorithm for Delay Tolerant Networks (DTNs) including neighborhood discovery, cluster creation, and data distribution within cluster. The general idea is to reduce the amount of messages being sent and of buffer utilization, by taking advantage of the nodes tendency to create groups and to share similar mobility patterns among each other. The main purpose of the algorithm is to improve the network performance without major changes in communication schemes between nodes. Almost no extra message type is added. Most extra features are available thanks to adding small extra fields into transmitted packages. Neighborhood discovery is being realized passively by listening to other nodes messages. The proposed algorithms allow to reduce both the bandwidth occupation, as well as the problems related to the media access. Furthermore, it can increase message delivery probability thanks to intelligent package distribution inside created cluster. Simulations were carried out to evaluate the effectiveness of the proposed solution in terms of package delivery probability, mean buffer occupancy and mean hop number to delivery the message. Results of simulation show that this solution is not necessary or recommended for small-scale networks with few nodes using clustering algorithms. However, with increasing number of nodes and messages, the performance of non-clustered DTNs drops significantly while clustered network works efficiently.
Keywords: Delay Tolerant Networks (DTNs), mobile ad-hoc networks, clustering algorithms, cluster based routing, neighborhood discovery in ad-hoc networks.

1 Introduction

Delay Tolerant Networks (DTN) [2] [9] are a kind of networks that can be used in extreme terrestrial environments with no telecommunication infrastructure provided, e.g. deserts, or rain forests. This kind of network can also be used in emergency situations, as an alternative way of communication, when conventional networks break down, for example as additional communication channel for emergency services in the case of terrorist attack. It can be also used for military purposes, as communication channel between groups of soldiers. The main advantage of this kind of network is the fact that no infrastructure is needed for its operation. Therefore, it can be easily created in any situation. Family of DTN networks that is taken into account is mobile ad-hoc networks that may lack continuous connectivity. It relates to situations when, in a particular moment, direct path between two nodes may be not available. However, communication is done through message ferring by mobile nodes. DTN are rather low transfer rate networks. The main assumption of DTN is to deliver the particular message as fast as possible, with maximum probability of delivery. However, in DTN networks it is possible that the path between two nodes will never appear, so the message might not be delivered.

The main mechanism of message forwarding is presented in Figure 1. A node that has a message to send broadcasts INV - the Invite message. This message contains a list of content identifiers, which node stores in its buffer. When a node that has received the INV message

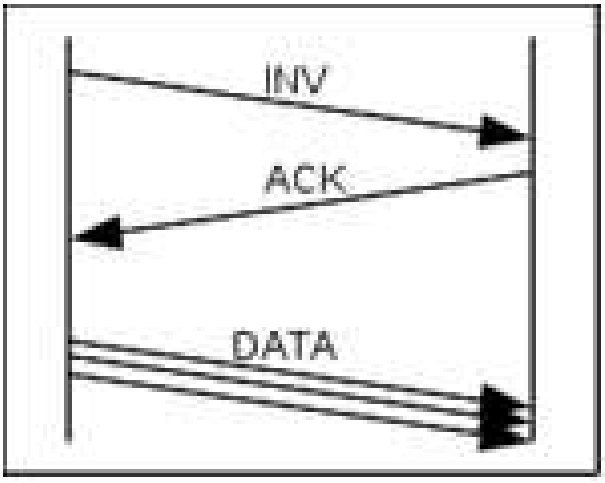


Figure 1: Basic message exchange scheme for DTN networks

wants to get particular packages, it sends ACK - the acknowledge message. The ACK message contains the list of desired content ids. After receiving ACK message, the first node forwards desired packages via DATA packages. This mechanism is being repeated until message reaches its final destination. INV messages are being sent periodically. Routing protocol that uses this mechanism is called Epidemic Routing [32]. It is a flooding protocol. It means that the message is being sent to each node met. Epidemic routing allows to reach highest possible delivery probability, along with the shortest delivery time in particular mobility pattern.

The main problem of this protocol is high redundancy. With increasing number of packages, the probability of message delivery drops significantly, because of limited buffers capacity. The second important problem is connected to the media access control. When nodes travel in a group, each node buffer will contain exactly the same content after some time. This means that each node will send identical INV messages. With increasing number of nodes, the access to the radio channel becomes more difficult. There is no need for each node of the group to send the new additional INV message. More advanced DTN routing protocols, such as Spray and Wait [29] or PRoPHET [20], limit the redundancy by reducing the number of message copies forwarded. On the one hand, this solution reduces the utilization of buffers. However, on the other hand, it removes some possible paths, which may lead the message to the destination. When mobility pattern is not completely random, two nodes will meet each other sooner or later, and the delivery probability drops. In some mobility patterns, in which nodes are constantly mobile, there are many possible paths from the source to the destination. In such patterns, the number of copies forwarded can be reduced without a significant performance loss. On the other hand, for the mobility patterns, in which only some nodes are mobile, or nodes are moving in specified directions, the number of possible paths is much lower. Reducing number of forwarded copies in such networks is highly ineffective. Moreover, advanced routing protocols do not solve the problem with radio channel access.

This considerations show two serious problems considering scalability of DTN networks: buffer and radio channel capacity. Wise use of clustering algorithms can solve both problems. Data distribution within improves significantly the message capacity and reduces the number of transmissions of the INV message, which increases the medium usage efficiency.

The idea of clustering has been extensively investigated in ad-hoc networks. First solutions like Clusterhead Gateway Switch Routing [5] and Cluster Based Routing Protocol with extensions [28] [1] [31] was reference for next development. All of these solutions introduce the idea of two

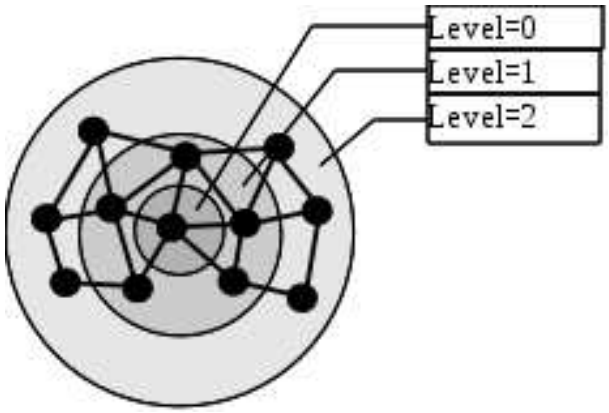


Figure 2: Two level cluster topology

level hierarchy composed of higher level managing nodes - Cluster Heads and lower-level nodes - Cluster Members with one-hop single cluster dimension. The multi-level hierarchy was proposed in [14]. Multi hop cluster range was proposed in [34].

For WSN networks there were proposed similar solutions: based on two-level single hop clusters like Low-energy Adaptive Clustering Hierarchy [13], multi hop clusters e.g. Hybrid Energy-Efficient Distributed clustering [36], or multi-level clusters like in Distributed Weight-based Energy-efficient Hierarchical Clustering protocol [8] and Two-Level Hierarchy LEACH [24]. The other very popular routing protocols which are based on clusters in sensor networks are: PANEL [3], UCS [30], EECS [35], EEUC [19], ACE [4], BCDCP [27], PEGASIS [21], TEEN [25], APTEEN [26], TTDD [23], CCS [16], HGMR [18], solutions like [10] [11] [12]. The detailed survey of clustering algorithms can be found in [33] and [22].

The clustering in DTN routing application was discussed in DTMN protocol [7] where authors proposed the solution for grouping mobile nodes with similar mobility pattern into a clusters. Similar solution for nodes clustering in vehicle-based networks was proposed in [6]. Other solution called hierarchical clustering-based forwarding scheme (HCS) was proposed in work [17]. The solution is based on implementing hierarchical clustering on social information.

All these works are focused on routing schemes in wireless networks, the message storage efficiency is less important. Our work focuses on achieving a high performance of the message storage and on the elimination of redundancy in a complex DTN network environment.

2 Proposed clustering algorithm

2.1 Cluster topology

Proposed algorithm involves grouping nodes with similar mobility patterns into two-level clusters. The general scheme of such cluster is presented in Figure 2.

A node in the area called Level=0 is called Cluster Head (CH). This node is responsible for cluster management. Address of this node is also considered as cluster id. Level=1 nodes are directly connected to CH. These nodes are responsible for data storage. Level 2 nodes are regular cluster members. The decision to organize the nodes into two-level clusters is due to the fact that such structures are still quite easy to manage and maintain in terms of their consistency. Taking into account the range of radio communication, such approach is enough to cover area of node group with a similar mobility pattern. The structure of INV message sent by DTN network with cluster algorithms is presented in Figure 3.

Figure 3: INV message structure

The package contains information about address of message sender, cluster id, and its position in cluster. DATA part of message stores a list of packages available to be sent. When a node does not belong to any cluster, the value 0 as CH address is set.

2.2 Related nodes metric

The main problem in DTN Networks is the fact that nodes are constantly mobile, which makes the cluster creation and maintenance tasks difficult. Network topology changes constantly and, furthermore, the connection between two nodes can disappear permanently. We assume that nodes do not know their position via GPS etc. The only information about the neighborhood we acquire is by listening to other nodes messages. To determine whether two nodes belong to the same cluster, some additional mechanisms have to be implemented. The solution to this problem has been presented on the basis of the Clustering and Cluster-Based Routing Protocol for Delay-Tolerant Mobile Networks [7]. Because of the fact that each node sends INV messages, there is no need to send extra HELLO messages, which are common for typical cluster-based ad-hoc networks. All the necessary/required information can be acquired from periodically sent INV messages. The period between INV messages is known. For each time slot, the related nodes metric is being evaluated. In general form, the metric formula is as follows:

$RN_y[n] = x_y[n] + \alpha^1 x_y[n-1] + ... + \alpha^N x_y[n-N]$

Where:

α - the extinction coefficient, y - the identifier of neighbouring node, $x_y[n]$ - the binary factor which indicates if node received message from node y in time slot number n, N - number of back slots, $RN_y[n]$ - value of the metric.

By changing the metric parameters, the different behavior of nodes is being set. Parameter α determines how long the information about relation is important. The higher the value of α the smaller impact on the metric value the previous meeting have. Parameter N determines the number of time slots for which the metric is being evaluated. On the one hand the higher N, the more reliable the metric value is. On the other hand, it increases the memory requirements. Additionally, the information about meetings from distant past is useless, because of a dynamic network topology change. Value of parameters have to be set individually for each situation. Depending on evaluated metric, two thresholds are defined, i.e. Y_{tresh} and Y_{min}. These thresholds will be used to make decision. If the condition:

$RN_y[n] > Y_{tresh}$

is fulfilled, then we assume the node is strongly related with node y. On the other hand, if the condition:

$RN_y[n] < Y_{min}$

is fulfilled, the relation between two nodes broke, and they no longer can be considered as strongly related nodes. These simple calculations allow to distinguish nodes with similar mobility pattern, which is very important in cluster formation and management. Metric also stores information about cluster assignment of observed node.

2.3 New cluster formation

If the number of the strongly related neighbors, which also do not belong to any cluster, exceeds threshold, the procedure of cluster initialization is carried out. Node broadcasts OR-

GANIZE message, which contains address of a sender and number of strongly related neighbors. Each node, which receives such message has to answer and sends its own ORGANIZE message. Node with the highest number of neighbors becomes a new Cluster Head, and its neighbors become Level 1 Cluster members. Such approach can lead to creation of two clusters, if two nodes are not in radio range, however, other mechanisms, which will be described later, will adjust this situation.

2.4 Joining new node to cluster

In order to join cluster, several conditions have to be fulfilled. Each node listens to broadcasted messages and evaluates related nodes metric. Nodes have to be strongly related to a sufficient number of cluster members to be concerned as strongly related to cluster. Cluster topology provides two levels, so metric is only valid for Level 0 and Level 1 nodes. If node determines that it is strongly related to the cluster, it can send a message requesting to join the cluster. Node becomes a part of the cluster of an adequate level. All packages from its buffer are being sent to the cluster and then distributed among Level 1 cluster members.

2.5 Leaving the cluster

Because of node mobility, cluster consistency has to be managed constantly. Node located at Level 1 has to be strongly related to Cluster Head, and a certain number of Level 1 nodes. If the first condition is not fulfilled, node changes its Level to 2. If second condition is not satisfied, node must leave the cluster. For Level 2 nodes only the second condition has to be fulfilled. Leaving node sends appropriate message to the Cluster Head and detaches from the cluster. It takes copies of as many packages from the cluster as its buffer can handle.

2.6 Two clusters meeting

Situation when two clusters are within radio communication range is hard to manage. CH reelection and cluster reorganization common for cluster-based ad-hoc networks will be highly infective due to packages stored inside cluster. This could lead to the package loss caused by limited buffers capacity, and high network load associated with the package transmission to new Level 1 nodes. Proposed architecture prefers absorption of smaller clusters by the bigger ones. If node determines that it is better related to an adjacent cluster, it will become a member of a new cluster. However, it is required to counteract the situation, when node oscillates between two clusters. This can be achieved by defining another threshold. Node can change cluster only if relation with new cluster is greater than current increased by threshold. This approach leads to the absorption of less numerous clusters by bigger ones.

2.7 Cluster destruction or cluster head reelection

Last node leaving the cluster is the Cluster Head node. When nodes are leaving cluster, number of strongly related nodes seen by the Cluster Head is decreasing. When the number drops below a particular value the Cluster Head makes decision to destroy the cluster and broadcasts DESTROY message. Reaction to such event can be different depending on the situation. Decreasing number of strongly related nodes seen by CH may be result of Cluster Head mobility pattern change leads to leaving the group. Node which receives DESTROY message checks a number of its strongly related neighbors. If it exceeds threshold of a new cluster creation, new Cluster Head is being reelected as in new cluster formation algorithm. If

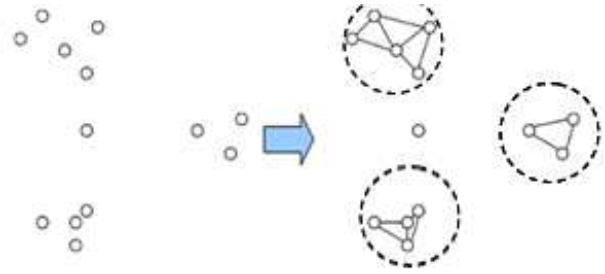


Figure 4: The network topology after cluster formation

the number of strongly related nodes is too small, and no ORGANIZE message was heard by different node, the cluster is destroyed.

2.8 Cluster head loss

Architecture must be resistant to the Cluster Head node failure. Without additional mechanisms, such situation would lead to continuous migrating nodes to Level 2 area and eventually to destruction of the cluster. To prevent the situation when Level 1 node loses a strong relation to the Cluster Head, the first node ensures that the neighboring nodes see the Cluster Head node. A broadcast message is sent and response is expected. If message is not received, the Cluster Head re-election occurs. New Cluster Head is reelected among Level 1 nodes. This makes the reorganization of cluster much easier because Level 1 nodes are better related and closer to the disappeared Cluster Head.

3 Cluster based routing

Organization of clusters changes topology of DTN networks. Nodes with similar mobility pattern are grouped into clusters creating bigger virtual nodes of network. General idea is to treat clusters like regular DTN nodes. These nodes share functionality of storing and forwarding messages, however, thanks to organization, their buffer capacity is much bigger. In clustered DTN, the number of INV messages sent by the cluster is smaller than in flat networks. This makes clustered networks more scalable than regular DTN networks. Network topology after cluster formation is presented in Figure 4.

3.1 Inter-cluster routing

Creation of virtual nodes allows coexistence of clusters and regular DTN nodes. No additional mechanisms have to be implemented. Message exchange between two nodes is based on Epidemic Routing protocol described in introduction. The choice of this protocol is associated with its high performance under condition of package delivery probability and delivery time. Furthermore, this protocol is easy to implement and do not need any additional information storage or evaluation. Problem with radio channel access control is solved by preventing some nodes to send INV message. The solution is to introduce the probability of INV message sent. Nodes with more neighbors send INV messages less often than these with lower number of neighbors. When randomly generated number is higher than threshold, the INV message is sent. Value of threshold is evaluated based on neighbors' quantity. After leaving too many time slots, the empty INV message is sent to indicate that node is still inside cluster. This approach limits the number of INV message sent without significant performance loss of message delivery. It is important to

mention that the Cluster Head has to broadcast INV messages in each time slot due to specific function of this node in cluster management. Lack of connection with Cluster Head has to be counteracted as fast as possible.

3.2 Intra-cluster routing

Nodes grouped into cluster creates consistent ad-hoc network. It is organized into two level hierarchy with one central node. Moreover nodes have information about their neighbors based on related nodes metric evaluation. This makes creation of routing tables inside cluster easy. Most appropriate routing protocol in such structure is Optimized Link State Routing Protocol (OLSR) [15] typical for ad-hoc consistent mobile networks. Path between two nodes within a cluster is almost always known. Each node know which neighbor is connected directly to the Cluster Head. Transmissions inside clusters are made on demand. Data transfer inside cluster is much faster than between two DTN nodes. There is no offset made by time slots typical for INV messages.

4 Data handling

An objective of the proposed method was to reduce the redundancy of packages in DTN networks. It is also required that network load reduction does not cause performance degradation. Single node failure, or unexpected cluster leaving, cannot lead to data loss. The purpose of this section is to describe the specifics of data handling for DTN cluster-based network. It presents the general assumptions and methods of data distribution within a cluster.

Due to their specific nature, no message delivery confirmation methods are implemented in DTN networks. Using this methods would result in message flooding in the opposite direction. That means an additional load on the network without noticeable benefit. Common concept is using Time To Live (TTL) parameter. After the time expiration, message is removed from the buffer. This solution has its explanation. If any node demands message for a long time, it may be assumed that it has already been duplicated and forwarded. Optimal value of TTL is specific for each network, and should be set individually.

4.1 New package distribution in cluster

New message in the cluster can only occur as a result of receiving the INV message from encountered node. The process of adding new nodes to a cluster takes place over several INV messages time slots and there is no need to consider the situation in which a new attached node carries packages that are not already in the cluster. Each new package received by the cluster is sent to the Cluster Head node. Then the message is forwarded to Level 1 nodes. To avoid unexpected losses associated with unexpected node failure, the message is replicated in a number of nodes simultaneously. The Cluster Head remembers package sequence number and information about nodes that possess it. The Cluster Head manages the packages and in case of node failure sends the requests to replicate the package in another Level 1 node. The CH manages the packages to be evenly distributed in nodes. The nodes through which a package has passed stores it through the entire duration of the TTL. This limits the number of transmissions and creates cache like buffers. After placing message in buffers the Cluster Head broadcasts the message about new available content.

4.2 Data search

INV messages sent by cluster members contain a list of all packages available. If ACK message is received the data search mechanism starts. First own buffer is being searched. If content is found, it is passed to requested node. If no data was found in buffer, request to the Cluster Head is sent. Level 2 nodes send request through Level 1 node. At first this node is also searching for its own buffer. If any of them finds desired content, The Cluster Head is asked. The CH responds with address of the node, which holds the content. Then package is downloaded directly from the node. Afterwards the package can be forwarded to desired location.

4.3 Buffer overflow

There may occur a situation, in which buffer of particular node overflows. Threshold of 0.95 buffer capacity is set. If the buffer occupation exceeds this number, some content has to be deleted. There are two types of package in buffers: created by the Cluster Head, and created when package was forwarded by the node. The first to delete are cached messages. The node removes as much packages as necessary to reduce buffer occupation below the threshold. Messages are deleted in order of time spent in buffer. If removal of cached packages is not enough to fulfill the buffer occupation condition, request to the Cluster Head is sent. At least two copies of one content have to be stored in the cluster to avoid data loss in case of node failure. If there are more copies, the Cluster Head permits to delete the message. Otherwise it checks if there is any other Level 1 node with free space in its buffer. If such node exists, the package is forwarded to this node. In other cases the oldest package is removed.

4.4 TTL expires

In contrast to the standard DTN network cluster member nodes do not remove the messages with expired TTL automatically. When the Cluster Head notices that package expired its TTL broadcasts the message to delete desired content. Each node, which received such message, deletes this package from its buffer. The list of available content sent in INV messages is updated.

5 Simulations

5.1 Mobility model

The research was carried out using specialized mobility model for emergency and rescue situations designed by authors. It is a mobility model proposed by us to describe the mobility pattern of nodes in an emergency situation. Scenario describes the situation when bomb explodes in the office. It takes into account different groups of people. The model describes mobility of injured civilians or people fleeing in panic. Behavior of different rescue groups was implemented: police, firefighters and emergency medical services. This model is a complex attempt to maintain the mapping of different groups behavior in life-threatening situations. We find reasonable application for clustering algorithms in such situations. Creating groups is common behavior for all living beings.

In general the scenario consists of 3 phases: stable phase, chaos and rescue operation. During the stable phase civilians are moving around the office. They do their job, go smoke a cigarette, or sit at their desk. This phase is not interesting from the research point of view, however, it is used to generate a random distribution of nodes during the explosion. Example of distribution of nodes in the stable phase is shown in Figure 5.

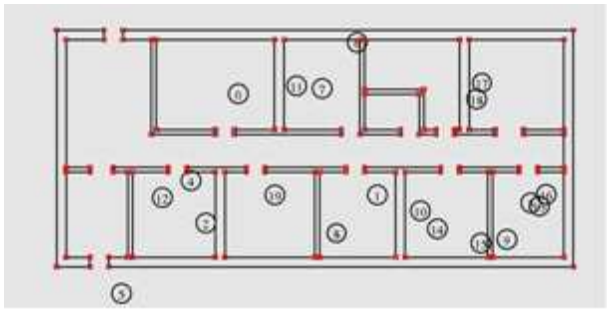


Figure 5: Node distribution during stable phase

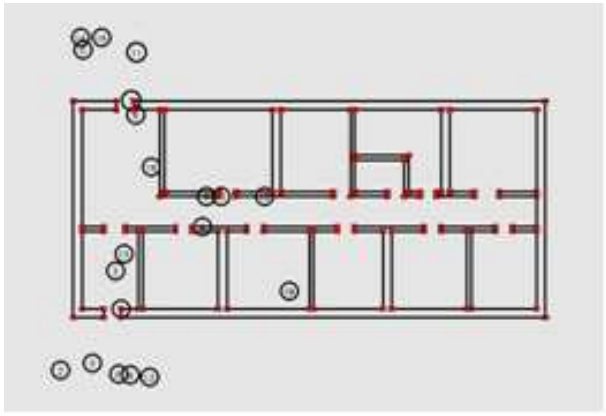


Figure 6: Node distribution during chaos phase

Stable phase is interrupted by explosion inside the office. Some civilians get injured, some are dead. Rest flees in panic. The phase of chaos starts. It lasts until arrival of rescue teams. Distribution of nodes in chaos phase is presented in Figure 6.

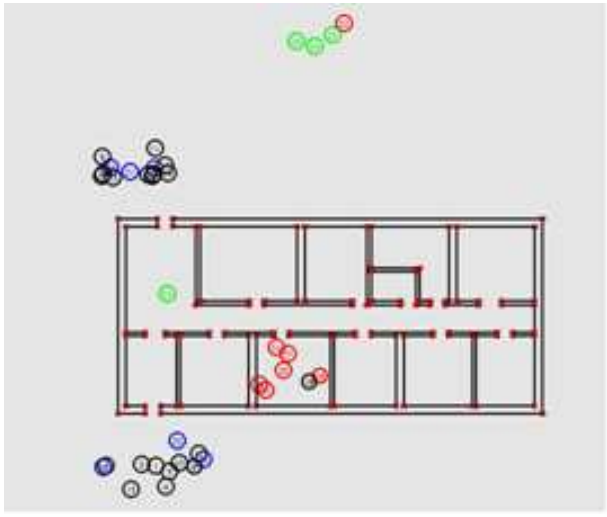


Figure 7: Node distribution during rescue phase

Evacuated people are grouping near office entrance. Here we can observe clusters. Those nodes are not highly mobile, but create a big buffer able to store message. Next, and last phase is the rescue operation. Firefighters enter the building, evacuate injured, police secure the area,

and separates onlookers. Medical services treat wounded people. Distribution of nodes in this phase is presented in Figure 7. Simulation ends when last wounded person is sent to the hospital.

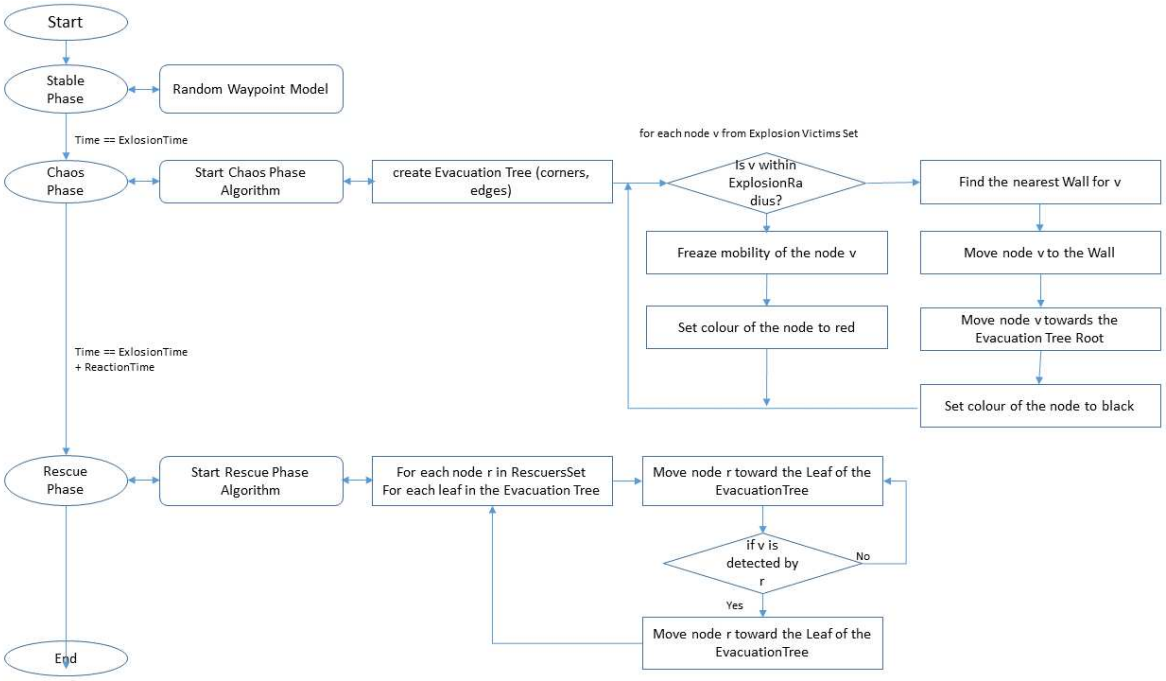


Figure 8: Mobility algorithm

The chaos phase has been implemented as follows: nodes inside a specific radius of explosion are seriously injured and are retained in place. These nodes are marked in red. Other nodes will escape using corridors out of the office. Due to smoke and reduced visibility nodes are moving along the walls. All mobile nodes in the phase are marked in black. To implement the mobility of nodes we proposed to use a set of edges of the door and edges of walls as points of the graph. All of this points are denoted as red dots. For each mobile node we executed algorithm that determines: 1) the closest distance between the black node and the nearest red point, then the algorithm 2) builds a graph between all red points and the building exit. 3) The algorithm performs the shortest path search in the graph based on the Dijsktra algorithm. After finding the shortest path, nodes are moved with speed of 2-4m/s to the exit. The rescue phase looks as follows: in simulations we propose that after a predefined period of time rescuers (green nodes) are moved. The movement is based on previous graph, but nodes are moved in opposite direction - to search all rooms in the office. Rescue nodes which hit a injured (red) nodes, take them to the exit. The mobility algorithm was presented in Figure 8.

5.2 Results

The research was performed in two stages. First for small scale network with small number of nodes and unlimited buffer capacity. This simulations show that clustered DTN works as efficient as non clustered network based on epidemic routing. The second stage is performed for a large scale network, where clustering algorithms show their advantage over regular DTN. During simulations following metrics were evaluated:

1. Package delivery probability: We counted the metric as number of messages received divided by number of messages send. The metric is important because in emergency and rescue scenarios many messages sent should reach destination nodes.

Table 1: Parameters of simulations

Parameter	Min Values	Max Values	Parameter	Value
α	0	1	Radio Radius	5m
N	1	5	Bandwith	11 Mb/s
Y_{tresh}	1.45	1.45	Packet Size	1500 b
Y_{min}	0.95	0.95	Simulations Time	750 sec.
$Y_{cluster}$	3	30	MessagePeriod	180 sec.

2. Mean buffer occupancy: The metric describes the mean number of packets stored per number of nodes. It describes how efficient the solution is in terms of minimizing buffer occupancy and total redundancy.

3. Mean hop number: The Metric specifies how many nodes have to pass the message before it was delivered to the recipient. The purpose of this metric is to determine the efficiency of packets transmission and to estimate energy requirements. It also partly determines the efficiency of use of the capacity of the radio channel.

Proposed mobility model is random and after each run different result is generated. To obtain reliable result simulation have to be run several times. After 50 simulations the mean value, and confidence interval is evaluated. For small scale networks the conditions are as follows: 1) Unlimited buffer capacity, 2) Number of messages created: 1000, 3) Number of nodes: changes from 10 to 60

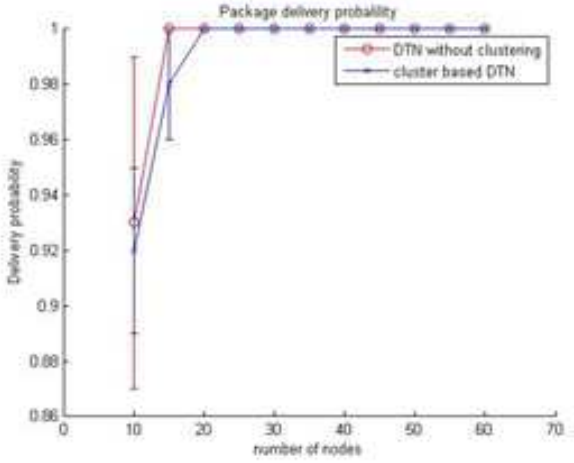
For all simulations we established nodes with different rules. Number of firefighters 6 and 1 officer, num of policemans - 6, num of rescuers - 4 persons. Total simulations time - 750sec., Messages sending period 180sec., The explossion moment was in 100 sec. after starting the simulation. Optimal values of parameters: Y_{tresh}, Y_{min}, $Y_{cluster}$ and α have been set based on preliminary simulations. The parameters of the algorithm are presented in the Table 1.

Results of simulations are presented in Figure 9 a, b, c.

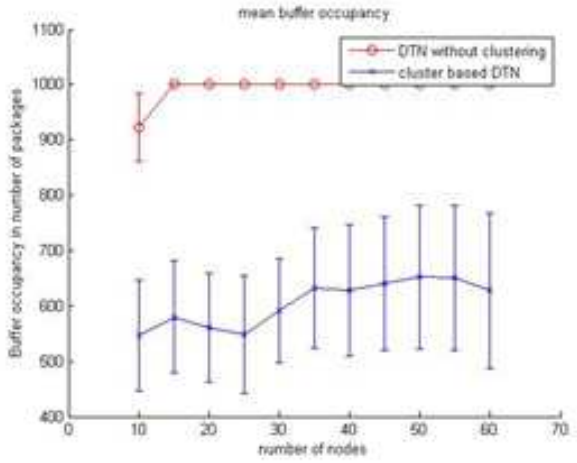
Results of simulation shows, that even in small scale networks cluster based DTN have advantage. Using clustering mechanisms do not reduce network performance within meaning of package delivery probability. Moreover it reduces the buffer occupancy and hop number. This leads to network load reduction. This advantages will be more noticeable in large scale network simulations. For those simulations conditions are: 1)Buffer capacity: 100 packages, 2) Number of messages created: 1000, 3) Number of nodes changes from 150 to 200.

Buffer capacity is smaller than number of messages. This should lead to overload of DTN network without cluster algorithms. Furthermore media access problems start being noticeable. Results of simulations are presented in Figures 10 a, b, c.

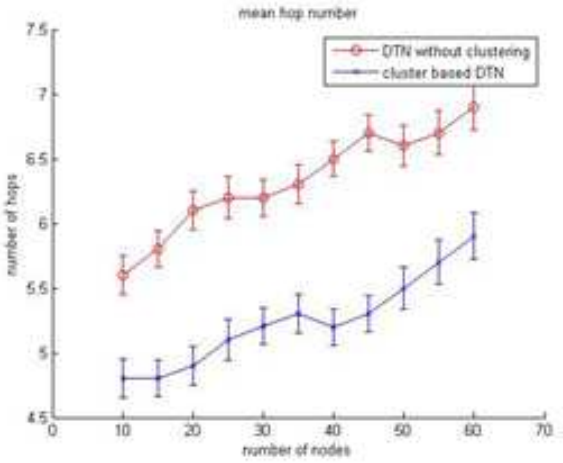
As expected regular DTN network performance drops significantly. When number of nodes exceeds 190 problems with media access are noticeable. Packages cannot be forwarded because network density is so high, that no INV message can be send without any collision. Message delivery probability is at unacceptable level. This is the result of buffers overflow. Hop numbers metric is evaluated only for successfully delivered messages, so this results do not carry any important information for regular DTN networks. More important information we get from results performed for clustered network. Hop number do not differ noticeably in comparison to small scale research. A slight increase in the number of hops is correlated with the increasing number of nodes in the network. This means that cluster based network forwards messages efficiently without additional, unnecessary transmissions.



(a) Package delivery probability

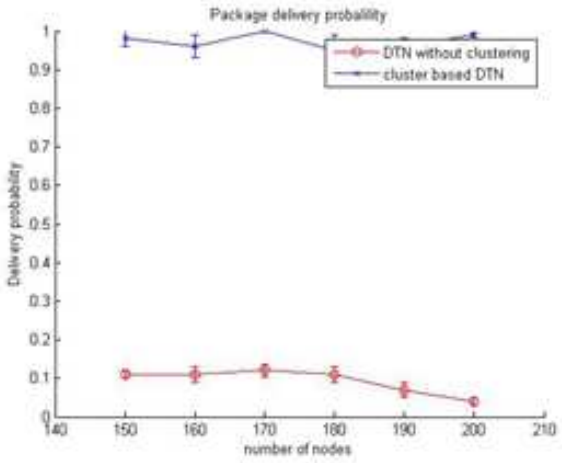


(b) Mean buffer occupancy

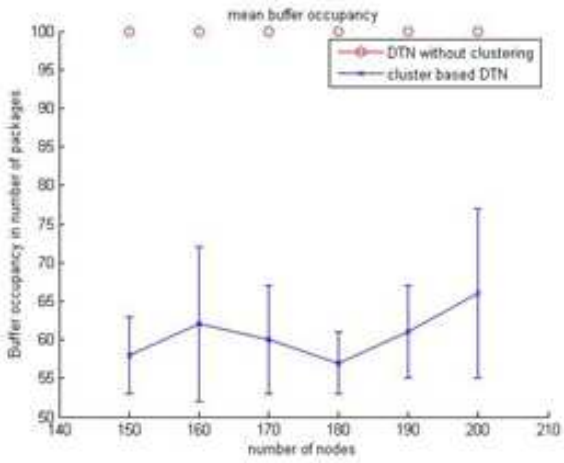


(c) Mean hop number

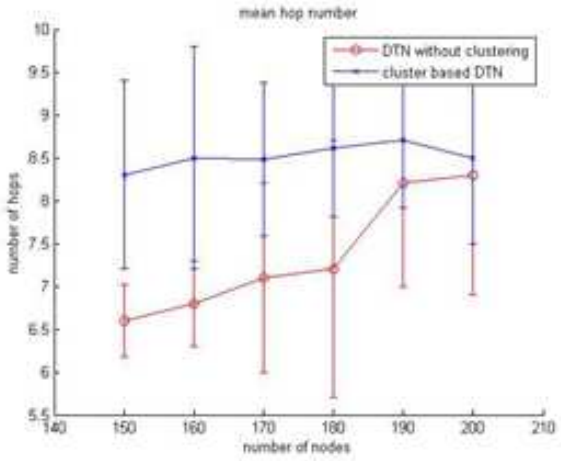
Figure 9: Results of simulations for small scale research



(a) Package delivery probability



(b) Mean buffer occupancy



(c) Mean hop number

Figure 10: Results of simulations for large scale research

6 Conclusions

We have proposed and implemented complex clustering algorithms for DTN networks. It significantly improves performance of such networks and makes them more scalable. Without many calculations, or sending extra control messages we achieved noticeable network load and buffer occupation reduction. This solution is designed to work in extreme terrestrial conditions where no telecommunication infrastructure is provided, and nodes moves in groups. This simple algorithm improves performance of regular DTN networks by reduction of unnecessary network load. Proposed solution can be easily implemented, and what is most important it can coexist with existing DTN networks

Bibliography

[1] Ahn, CW.; Ramakrishna, RS.; Kang CG. (2002); Efficient Clustering-based Routing Protocol in Mobile Ad-Hoc Networks, *in Proc. IEEE VTC 02*, DOI:10.1109/VETECF.2002.1040495, 1647–1651, 2002.

[2] Burleigh, S.; Hooke, A.; Torgerson, L.; Fall, K.; Cerf, V.; Durst, B.; Scott, K.; Weiss H. (2003); Delay-tolerant networking: an approach to interplanetary internet, *IEEE Commun. Mag.*,DOI:10.1109/MCOM.2003.1204759, 128–136, 2003.

[3] Buttyan, L.; Schaoer, P. (2010); PANEL: Position-Based Aggregator Node Election in Wireless Sensor Networks, *Int.J. Distrib. Sens. Netw*, DOI:10.1.1.158.9102, 1–16, 2010.

[4] Chan, H.; Perrig, A. (2004); An Emergent Algorithm for Highly Uniform Cluster Formation, *Wireless Sensor Network*, DOI:10.1007/978-3-540-24606-0_11, 154–171, 2004.

[5] Chiang, CC.; Wu, HK.; Liu, A.; Gerla M. (1998); Routing in clustered multi-hop mobile wireless networks with fading channel, *in Proc. of IEEE SICON 98*, 197–211, 1998.

[6] Crepaldi, R.; Bakht, M.; Kravets R. (2012); QuickSilver: application-driven inter- and intra-cluster communication in Vanets, *in Proc. ACM MobiOpp 02*, DOI:10.1145/2159576.2159591, 69–76, 2012.

[7] Dang, H.; Wu H. (2010); Clustering and Cluster-based Routing Protocol for Delay-tolerant Mobile Networks, *IEEE Trans. Wireless. Comm*, DOI:10.1109/TWC.2010.06.081216, 9(6), 1874–1881, 2010.

[8] Ding, P.; Holliday, J.; Celik A. (2005); Distributed Energy Effcient Hierarchical Clustering for Wireless Sensor Networks. *in Proc. IEEE DCOSS*, DOI:10.1007/11502593_25, 322–339, 2005.

[9] Fall, K. (2003); A delay-tolerant network architecture for chalenged internets, *in Proc. of The 2003 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM '03)*, DOI:10.1145/863955.863960, 27–34, 2003.

[10] Fan, CS. (2016); HIGH: A Hexagon-based Intelligent Grouping Approach in Wireless Sensor Networks, *Advances in Electrical and Computer Engineering*, DOI:10.4316/AECE.2016.01006, 16(1), 41–46, 2016.

[11] Fang, W.; Li, S.; Liang, X.; Li, Z. (2012); Cluster-based Data Gathering in Long-Strip Wireless Sensor Networks, *Advances in Electrical and Computer Engineering*, DOI:10.4316/AECE.2012.01001, 12(1), 3–8, 2012.

[12] Golanski, M.; Schoeneich, RO.; Siwko, M. (2010); The algorithm for distribution of large-size data in the Wireless Ad-hoc Sensor Network, *in proc. Concepts and Implementation for Innovative Military Communications and Information Technologies, Military University of Technology*, 577–584, 2010.

[13] Heinzelman, WR.; Chandrakasan, A.; Balakrishnan, H. (2000); Energy-Efficient Communication Protocol for Wireless Microsensor Networks. *in Proc. HICSS 07*, DOI:10.1109/HICSS.2000.926982, 10–19, 2000.

[14] Iwata, A.; Chiang, C-C.; Gerla, M;, Chen. TW. (1999); Scalable Routing Strategies for Ad hoc Wireless Networks, *IEEE Journal on Selected Areas in Communications*, DOI:10.1109/49.779920, 17, 1369–1379, 1999.

[15] Jacquet, P.; Muhlethaler, P.; Clausen, T.; Laouiti, A.; Qayyum, A.; Viennot, L. (2001); Optimized Link State Routing Protocol for Ad Hoc Networks, *IEEE Trans. Wireless. Comm.*, DOI:10.1109/INMIC.2001.995315, 62–68, 2001.

[16] Jung, S.; Han, Y.; Chung, T. (2007); The Concentric Clustering Scheme for Effcient Energy Consumption in the PEGASIS. *in Proc. IEEE ICACT 07*, DOI:10.1109/ICACT.2007.358351, 260–265, 2007.

[17] Kim, SK.; Yoon, JH.; Lee, J.; Yang, SB. (2015); HCS: hierarchical cluster-based forwarding scheme for mobile social networks, *Wirel. Netw.*, DOI:10.1007/s11276-014-0876-x, 21(5), 1699–1711, 2015.

[18] Koutsonikola, D.; Das, S.; Charlie, HY.; Stojmenovic, I. (2010); Hierarchical geographic multicast routing for wireless sensor networks, *Wirel. Netw.*, DOI:10.1007/s11276-008-0146-x, 16(2), 449–466, 2010.

[19] Li, C.; Ye, M.; Chen, G.; Wu, J. (2005); An Energy Effcient Unequal Clustering Mechanism for Wireless Sensor Networks, *in Proc. IEEE MASS 05*, DOI:10.1109/MAHSS.2005.1542849, 596–604, 2005.

[20] Lindgren, A.; Doria, A.; Schelen, O. (2003); Probabilistic routing in intermittently connected networks, *ACM SIGMOBILE Mobile Computing and Communications Review*, DOI:10.1145/961268.961272, 19–20, 2003.

[21] Lindsey, S.; Raghavendra, C.; Sivalingam, LM. (2002); Data gathering algorithms in sensor networks using energy metrics, *IEEE Trans. Parallel Distrib. Syst.*, DOI:10.1109/TPDS.2002.1036066, 13(9), 924–935, 2002.

[22] Liu, X. (2012); A Survey on Clustering Routing Protocols in Wireless Sensor Networks. *Sensors*, DOI:10.3390/s120811113, 12(8), 11113–11153, 2012.

[23] Luo, H.; Ye, F.; Cheng, J.; Lu, S.; Zhang, L. (2005); TTDD: Two-tier data dissemination in large-scale wireless sensor networks, *Wirel. Netw.*, DOI:10.1007/s11276-004-4753-x, 11(1), 160–175, 2005.

[24] Loscri, V.; Morabito, G.; Marano, S. (2005); A Two-Level Hierarchy for Low-Energy Adaptive Clustering Hierarchy, *in Proc. IEEE VTC -5*, DOI:10.1109/VETECF.2005.1558418, 1809–1813, 2005.

[25] Manjeshwar, E.; Agrawal, DP. (2001); TEEN: A Routing Protocol for Enhanced Effciency in Wireless Sensor Networks, *in Proc. IEEE IPDPS 01*, DOI:10.1109/IPDPS.2001.925197, 2009–2015, 2001.

[26] Manjeshwar, E.; Agrawal, DP. (2002); APTEEN: A Hybrid Protocol for Effcient Routing and Comprehensive Information Retrieval in Wireless Sensor Networks, *in Proc. IEEE IPDPS 02*, DOI:10.1109/IPDPS.2002.1016600, 195–202, 2002.

[27] Murugunathan, SD.; Ma, DCF.; Bhasin, RI.; Fapajuwo, AO. (2005); A Centralized Energy-Effcient Routing Protocol for Wireless Sensor Networks, *IEEE Radio Commun*, DOI:10.1109/MCOM.2005.1404592, 43(3), 8–13, 2005.

[28] Schoeneich, RO.; Golanski, M. (2007); Mesh Cluster Based Routing Protocol: Enhancing Multi-hop Internet Access using Cluster paradigm, *in Proc. EUROCON, 2007. The International Conference on Computer as a Tool*, DOI:10.1109/EURCON.2007.4400318, 962–965, 2007.

[29] Spyropoulos, T.; Psounis, K.; Raghavendra, CS. (2005); Spray and wait: an efficient routing scheme for intermittently connected mobile networks, *in Proc. of The 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, DOI:10.1145/1080139.1080143, 252-259, 2005.

[30] Soro, S.; Heinzelman, W. (2005); Prolonging the Lifetime of Wireless Sensor Networks via Unequal Clustering, *in Proc. IEEE WMAN 05*, DOI:10.1109/IPDPS.2005.365 ,236–243, 2005.

[31] Su, YY.; Hwang, SF.; Dow, CR. (2008); An Efficient Cluster-Based Routing Algorithm in Ad Hoc Networks with Unidirectional Links, *Journal of Information Science And Engineering*, 24(5), 1409–1428, 2008.

[32] Vahdat, A.; Becker, D. (2000); Epidemic routing for partially connected ad hoc networks. *Tech Report CS-2000-06*, DOI:10.1.1.34.6151, 2000.

[33] Wei, C.; Yang, J.; Gao, Y.; Zhang, Z. (2011); Cluster-Based Routing Protocols in Wireless Sensor Networks: A Survey, *inProc. IEEE ICCSNT 11*, DOI:10.1109/ICCSNT.2011.6182285, 1659–1663, 2011.

[34] Yau, S.; Gao, W. (2007); Multi-hop clustering based on neighborhood benchmark in mobile ad-hoc networks, *Mob. Netw. Appl*, DOI:10.1007/s11036-008-0039-3, 12(5), 381–391, 2007.

[35] Ye, M.; Li, C.; Chen, G.; Wu, J. (2005); EECS: An Energy Efficient Clustering Scheme in Wireless Sensor Networks, *in Proc. IEEE IPCCC.05*, DOI:10.1109/PCCC.2005.1460630, 535–540, 2005.

[36] Younis, O.; Fahmy, S. (2004); HEED: A hybrid, energy-efficient, distributed clustering approach for ad-hoc sensor networks, *IEEE Trans. Mobile Comput.*, DOI:10.1109/TMC.2004.41, 3(4), 366–379, 2004.

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 13(4), 566-573, August 2018.

Nonparametric Regression-based Step-length Estimation for Arm-swing Walking using a Smartphone

P.H. Truong, N.D. Nguyen, N.H. Ho, G.-M. Jeong

Phuc Huu Truong, Nhan Duc Nguyen,
Ngoc-Huynh Ho, Gu-Min Jeong*
School of Electrical Engineering,
Kookmin University,
Jeongneung-dong, Seongbuk-gu, 02707 Korea.
phtruong@kookmin.ac.kr, nhannd@kookmin.ac.kr,
ngochuynh.0303@kookmin.ac.kr
*Corresponding author: gm1004@kookmin.ac.kr

Abstract: In this paper, we propose an adaptive step-estimation method to estimate the distance traveled for arm-swinging activities at three level-walking speeds, i.e., low, normal, and high speed. The proposed method is constructed based on a polynomial function of the pedestrian speed and variance of walking acceleration. We firstly apply a low-pass filter with 10 Hz cut-off frequency for acceleration data. Then, we analyze the acceleration data to find the number of steps in each sample. Finally, the traveled distance is calculated by summing all step lengths which are estimated by the proposed method during walking. Applying the proposed method, we can estimate the walking distance with an accuracy rate of 95.35% in a normal walking speed. The accuracy rates of low and high walking speeds are 94.63% and 94.97%, respectively. Furthermore, the proposed method outperforms conventional methods in terms of accuracy and standard deviation at low, normal, and high speeds.
Keywords: Walking distance estimation, Pedestrian Dead Reckoning (PDR), nonparametric regression, arm-swinging.

1 Introduction

Recently, Pedestrian Dead Reckoning (PDR), which uses sensor information to estimate the location and orientation of a pedestrian in a global positioning system-denied scenarios [2, 6, 11, 15], has received considerable attention from researchers. Moreover, PDR technology brings many advantages on energy consumption, weight, cost compared with the inertial navigation system for indoor navigation. This leads to numerous researches on the PDR field over the last few years.

PDR algorithms commonly compute the walking distance by detecting users' steps and estimating the step lengths. Due to the location of mounted sensors, step detection and analysis can be done in many different ways. Fixing sensors on the user's foot, stance phases of the foot and the step detection can be determined [15, 17]. Zero Velocity Update (ZUPT) method is commonly used in this foot mounting configuration to bound the error accumulation [7, 13, 15]. In the second configuration, researchers fixed the inertial sensor on users' belts [1], or kept sensors in a constant pose [8]. The inertial force experienced by the sensor is directly linked to the motion of the human body's Center-of-Mass (COM) and, subsequently, to user's movement [16]. Therefore, in arm-swinging activities, where an inertial force of the sensor is not linked to the human's motion, an extra acceleration affects significantly the accuracy of walking distance estimation. Suh et al. [14] proposed a peak detection algorithm using five data points to find the number of walking steps. This algorithm utilizes the constraints between the times and consecutive peak values to detect the steps. Tian et al. [16] presented a step frequency-based

and individual-height-based distance estimation for pedestrian swinging arms. However, their method only focused on walking at the normal speed.

In this paper, we propose a distance estimation method for arm-swinging cases where a pedestrian holds a smartphone in hand and naturally walks at three speed levels. In general, data from smartphone are collected in the sensor coordinate system; thus, we firstly need to transform experimental data into the world coordinate system for synchronizing recorded data with the pedestrian movements. A low-pass filter (LPF) with 10 Hz cut-off frequency is applied to remove noise from the raw data. Then, we analyze arm-swing-related gait cycles and define step phases in human walking. Finally, we estimate the length of each step using an adaptive conversion as a polynomial function of the walking speed. The polynomial function is obtained by using a nonparametric regression called locally weighted polynomial regression [5,10]. In addition, this step-length estimation is an improvement of method [18]. The proposed method estimates the walking distance with an accuracy rate of 96.05%, 94.79%, and 94.20% in the normal, low, and high walking speeds, respectively. To demonstrate the performance of the proposed method, we compare experimental results of our method with two reference methods [16, 18].

2 Problem formulation

During walking, pedestrians hold smartphones in their hands and swing their arms forward or backward. This action varies the acceleration collected in smartphones, and thus, affects the estimation results of walking distance. We propose a distance estimation method for arm-swinging pedestrians in three speed levels. Figure 1 illustrates the walking condition of an arm-swinging pedestrian with three levels of speed. The rightmost sub-figure presents not only the phone-holding posture but also the smartphone orientation.

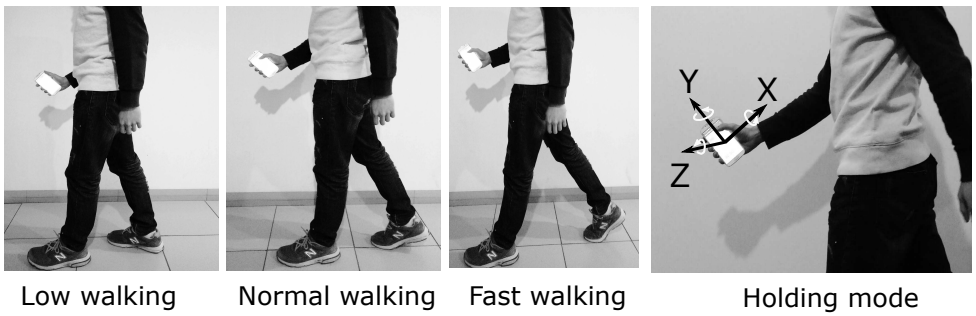


Figure 1: Walking speed levels and smartphone holding posture.

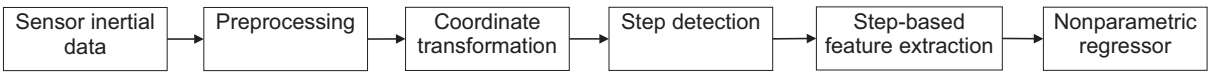


Figure 2: Block diagram of walking distance estimation

Figure 2 illustrates our proposed method for estimating the traveling distance. After collecting inertial data from the smartphone, we apply a low-pass filter to acceleration signals to isolate the force of gravity and remove the noise created during movement. Then, we transform the accelerations from smartphone's coordinates to the world coordinates using Euler's rotation theorem. By analyzing the arm-swinging phase, we find the number of steps in each trial. A feature is extracted as the magnitude of the average velocities on three dimensional axes in each step. A nonparametric regressor uses the extracted feature and ground truth to estimate the walking distance.

3 Data preprocessing

Using Euler's rotation theorem [12], we transform the raw data of the accelerometer from the smartphone's frame into the world frame. Generally, a step frequency is below 10 Hz for human walking [3]. Therefore, we design a Kaiser window-based LPF to remove noise from raw signals. The Kaiser window is defined as follows:

$$w(n) = \begin{cases} \dfrac{I_0\left(\beta\sqrt{1-\left(\frac{n-N/2}{N/2}\right)^2}\right)}{I_0(\beta)}, & 0 \le n \le N \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where, I_0 is the zeroth-order modified Bessel function; N is the length of the sequence without the first element; and β is the Kaiser window parameter that affects the sidelobe attenuation of Fourier transform of the window. To obtain Kaiser window with a sidelobe attenuation of α dB, β is determined using the following formula:

$$\beta = \begin{cases} 0.1102\,(\alpha - 8.7)\,, & \alpha > 50 \\ 5.5842(\alpha - 21)^{0.4} + 0.07886\,(\alpha - 21)\,, & 50 \ge \alpha \ge 21 \\ 0, & \alpha < 21. \end{cases} \tag{2}$$

Figure 3 shows a demonstration of magnitude response of an x-axis acceleration of the FIR filter-based Kaiser window.

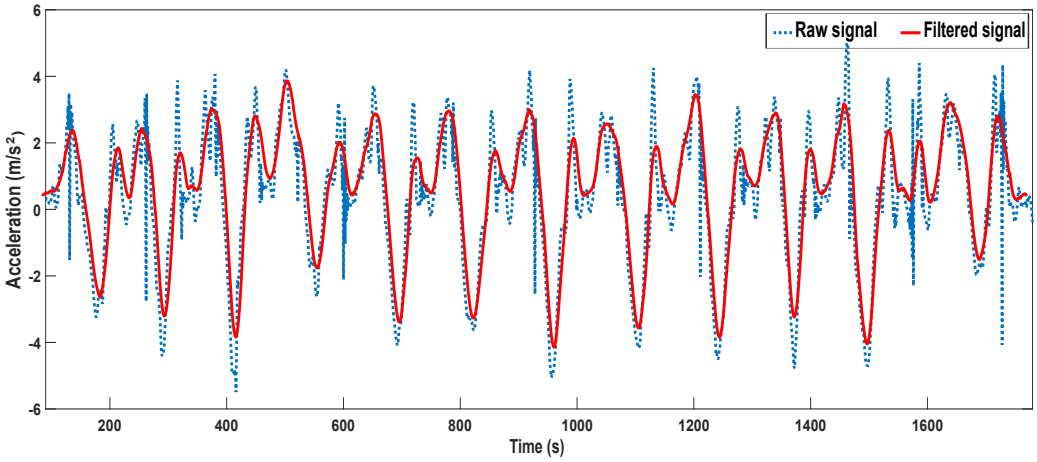


Figure 3: Magnitude response of low-pass FIR filter

4 Arm-swing analysis

In human walking, an arm-swinging cycle is divided into two phases: forward-swing and backward-swing, as shown in Figure 4. The forward-swing phase starts when the hand holding the smartphone is at the highest position behind the waist with maximum acceleration (black "X" symbol) and the same-side foot touches the ground. Then, the hand moves forward to the front of the body and the acceleration drops to a minimum (pink cycle) when the body is in a stance phase. Eventually, the hand stops at the highest forward position and the acceleration reaches the highest position (other black "X" symbol) while the opposite-side foot hits the ground. On the other hand, the backward-swing phase starts when the observed hand starts moving backward from the front of the waist while the acceleration decreases from the maximum value and the

opposite-side foot touches the ground. Acceleration in the backward-swing phase decreases to the smallest value (other pink cycle) in the stance phase, and then increases to the highest value (other black "X" symbol) when the hand is at the back of the user's body. Meanwhile, a same-side foot hits the ground. Therefore, there are two steps in a complete arm-swinging cycle.

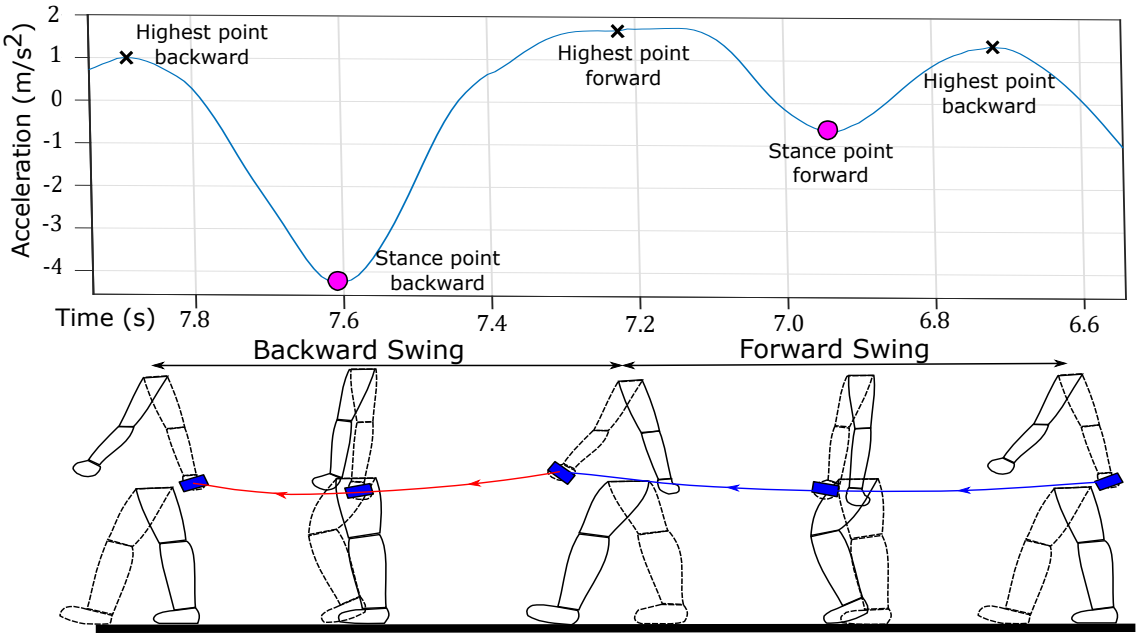


Figure 4: Arm-swing cycle analysis

5 Adaptive step-length estimation

A distance estimation method for pedestrian walking based on vertical acceleration was proposed by Weinberg [18]. His well-known formula is expressed as follows:

$$L_w \approx K_w \times \sqrt[4]{A_{max} - A_{min}}, \tag{3}$$

where, L_w is the step length; K_w is a constant; and A_{max}, A_{min} are the highest and smallest vertical acceleration values in a single step, respectively. Based on the Weinberg's formula, we propose a K-factor adapted along steps as a polynomial function of step velocity.

The adaptive K-factor for each step is considered as a general regression model:

$$K = f(V, \boldsymbol{\beta}) + \boldsymbol{e}, i = 1, 2, ..., n, \tag{4}$$

where

$$V = \begin{bmatrix} 1 & \overline{v_1} & \cdots & \overline{v_1}^p \\ 1 & \overline{v_2} & \cdots & \overline{v_2}^p \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \overline{v_n} & \cdots & \overline{v_n}^p \end{bmatrix}, \tag{5}$$

$$K = (K_1, K_2, ..., K_n)^T, \tag{6}$$

$$\boldsymbol{\beta} = (\beta_0, \beta_1, ..., \beta_p)^T, \tag{7}$$

$$\boldsymbol{e} = (e_1, e_2, ..., e_n)^T. \tag{8}$$

It should be noted that $\overline{v}$ is the magnitude of the average velocities on x, y, and z-axes in each step; $\boldsymbol{e}$ and n are the noise or measurement error and the number of observations, respectively. We assume the noise $\boldsymbol{e}$ is uncorrelated, mean zeros, and random variables. Then, the locally weighted polynomial regression is obtained through the solution of the weighted least squares problem:

$$\widehat{\beta} = \underset{\beta}{\operatorname{argmin}}(K - V\beta)^T W(K - V\beta), \tag{9}$$

where, W is a diagonal matrix with the Gaussian weights which can achieve a more accurate local approximation model [4]. The solution respecting to the coefficient β is

$$\widehat{\beta} = (V^T W V)^{-1}(V^T W K). \tag{10}$$

Additionally, the K-factor is obtained as

$$K_i = \beta_0 + \overline{v_i}\beta_1 + ... + \overline{v_i}^p\beta_p. \tag{11}$$

The proposed adaptive step-length estimation equation is

$$L_{step_i} = (\beta_0 + \overline{v_i}\beta_1 + ... + \overline{v_i}^p\beta_p) \times \sqrt[4]{A_{max} - A_{min}}. \tag{12}$$

The walking distance within each trial is calculated by summing the distance traveled of all the steps.

$$D = \sum_{i=1}^{N} L_{step}(i), \tag{13}$$

where, N is the number of walking steps in each trial.

6 Experimental results

In our experiments, three participants were required to walk forward 15 m, and swing their hands in their casual manner. We collected 300 samples of the acceleration and gyroscope signals at the sampling rate of 50Hz. To validate the performance of a proposed method and selecting the degree of a polynomial function, we applied a K-fold Cross-Validation ($K = 6$) on the dataset. We divided the dataset into 6 roughly equal-sized parts. We, then, took alternately 5 parts for the training set and used the remaining part as the test set. Figure 5 shows the mean squared errors versus degrees of freedom [9] of the training set for different degrees of the polynomial function ($K = 2, 3, ..., 6$). We selected the degree of the polynomial equal to 6 to minimize both errors of the training set and test set.

We compared the estimation results of our proposed method with Tian et al. [16] and Weinberg et al. [18] methods. Figure 6 shows the accuracy rates of three methods versus three speed levels, i.e., low, normal, and high speed. Note that the speed levels are obtained based on our previous approach [8]. We defined the speed levels in the experimental samples based on a normal distribution of average walking velocities $\overline{v}$ and velocity deviation σ. Thus, the ranges of the low, normal, and high speeds are $v \leq \overline{v} - \sigma$, $\overline{v} - \sigma < v < \overline{v} + \sigma$, and $v \geq \overline{v} + \sigma$, respectively. Table 1 describes the average results of error rates and standard deviations (*Std.*) of three methods.

The proposed method achieved a higher accuracy and a lower standard deviation in comparison with the two reference methods. In particular, the proposed method obtained the average

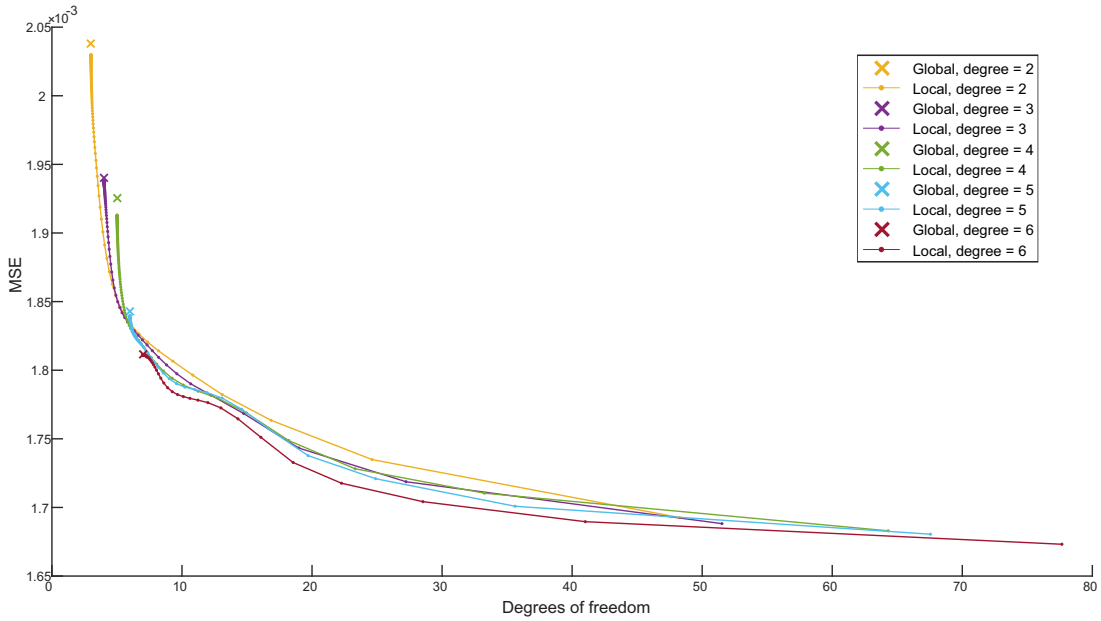


Figure 5: Mean squared error (MSE) versus polynomial degree

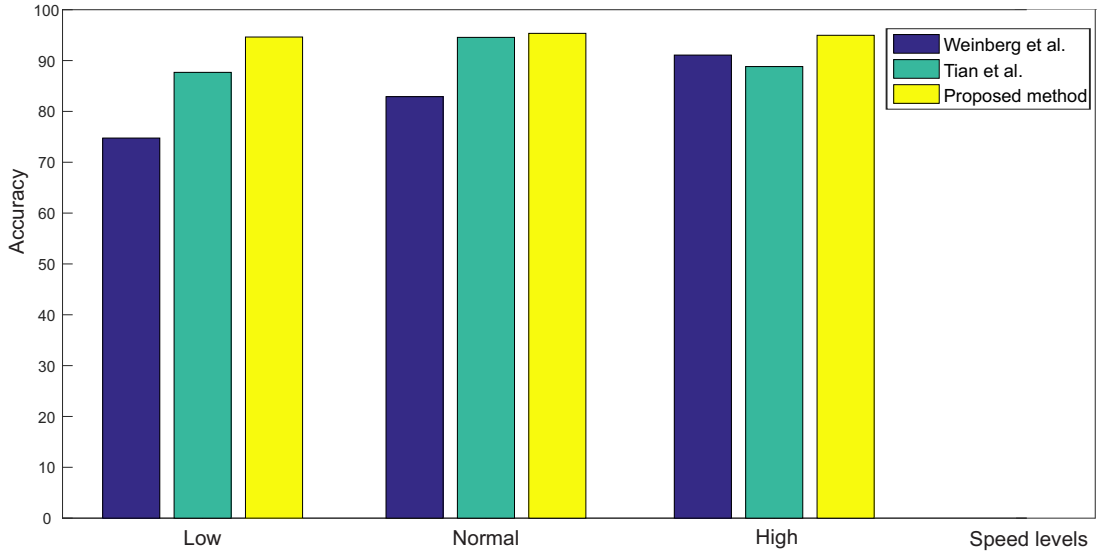


Figure 6: Estimation accuracy of the three methods

Table 1: Comparison of estimation results of the proposed method with reference methods

Speed level		Low		Normal		High	
		Error (%)	Std (m)	Error (%)	Std (m)	Error (%)	Std (m)
Weinberg [18]		25.24	0.74	17.10	0.89	8.92	0.67
Tian [16]		12.32	1.02	5.44	1.16	11.18	0.51
Proposed method	Training	3.87	0.56	4.02	0.54	4.09	0.65
	Testing	5.37	0.54	4.65	0.68	5.03	0.52

error rate and average standard deviation of 5.01% and 0.56 m in the testing phase, respec-

tively, which significantly surpassed the reference methods' results. Between the two reference methods, the method proposed by Tian et al. shows lower error rates for the slow and normal walking speeds (12.32% and 5.44%, respectively), but suffers a higher error for the high speed level (11.18%). On the contrary, the average standard deviation of Weinberg et al.'s method was better than Tian et al.'s method. Generally, the application of these two methods into walking distance estimation for the arm-swing activities is a case-by-case issue. Contrarily, the proposed method was well-performed for all three cases of walking speeds.

7 Conclusion

In this paper, we proposed an adaptive distance estimation method for arm-swinging activities at the three levels of speed, i.e., low, normal, and high speed. We analyzed the phases of the arm-swinging activities to determine the number of steps in each sample. The degree of the polynomial function was selected by analyzing the mean squared error versus the degrees of freedom of each polynomial value using the cross-validation technique. The results revealed that the proposed method estimated the distance traveled better than the conventional methods in terms of the accuracy and standard deviation of the results.

Acknowledgment

This research was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF- 2015R1D1A1A01 060917), the Korean Government (MSIP)(NRF-2016R1A5A1012966), and Global Scholarship Program for Foreign Graduate Student at Kookmin University in Korea.

Bibliography

[1] Alvarez, D.; Gonzalez, R. C.; Lopez, A.; Alvarez, J. C. (2006); Comparison of step length estimators from wearable accelerometer devices, *IEEE Engineering in Medicine and Biology Society*, 5964-5967, 2006.

[2] Capurso, N.; Song, T.; Cheng, W.; Yu, J.; Cheng, X. (2016); An Android-based mechanism for energy efficient localization depending on indoor/outdoor context, *IEEE Internet of Things Journal*, 4(2), 299-307, 2016.

[3] Chen, K.Y.; Bassett, D.R. (2005); The technology of accelerometry-based activity monitors: current and future, *Medicine and science in sports and exercise*, 37(11), S490-S500, 2005.

[4] Cleveland, W. S.; Loader, C. (1996); Smoothing by local regression: Principles and methods, *Physica-Verlag HD*, 10-49, 1996.

[5] Cleveland, W. S.; Devlin, S. J. (1988); Locally weighted regression: An approach to regression analysis by local fitting, *J. Am. Stat. Assoc.*, 83(403), 596-610, 1988.

[6] Donoso, Y.; Montoya, G. A.; Solano, F. (2015); An Energy-Efficient and Routing Approach for Position Estimation using Kalman Filter Techniques in Mobile WSNs, *International Journal of Computers Communications & Control*, 10(4), 500-507, 2015.

[7] Foxlin, E. (2005); Pedestrian tracking with shoe-mounted inertial sensors, *IEEE Comput. Graph. Appl*, 25, 38-46, 2005.

[8] Ho, N.-H.; Truong, P. H.; Jeong, G.-M. (2016); Step-detection and adaptive step-length estimation for pedestrian dead-reckoning at various walking speeds using a smartphone, *Sensors*, 16(9), 14-23, 2016.

[9] Jekabsons, G. (2016); Locally weighted polynomials toolbox for Matlab/Octave, 2016.

[10] Lall, U.; Moon, Y.-I.; Kwon, H.-H.; Bosworth, K. (2006); Locally weighted polynomial regression: Parameter choice and application to forecasts of the Great Salt Lake, *Water Resour. Res.*, 42, W05422, 2006.

[11] Liu, Y.; Chen, Y.; Shi, L.; Tian, Z.; Zhou, M.; Li, L. (2015); Accelerometer based joint step detection and adaptive step length estimation algorithm using handheld devices, *Journal of Communications*, 10(7), 520-525, 2015.

[12] Palais, B.; Palais, R.; Rodi, S. (2009); A disorienting look at Euler's theorem on the axis of a rotation, *American Mathematical Monthly*, 116(10), 892-909, 2009.

[13] Skog, I.; Handel, P.; Nilsson, J.-O.; Rantakokko, J. (2010); Zero-velocity detection−An algorithm evaluation, *IEEE Trans. Biomed. Eng*, 57, 2657-2665, 2010.

[14] Suh, Y. S.; Nemati, E.; Sarrafzadeh, M. (2016); Kalman-filter-based walking distance estimation for a smart-watch, *IEEE First Conference on Connected Health: Applications, Systems and Engineering Technologies*, 150-156, 2016.

[15] Susi, M.; Renaudin, V.; Lachapelle, G. (2013); Motion mode recognition and step detection algorithms for mobile phone users, *Sensors*, 13, 1539-1562, 2013.

[16] Tian, Q.; Salcic, Z.; Wang, K.; Pan, Y. (2016); A multi-mode dead reckoning system for pedestrian tracking using smartphones, *IEEE Sensors Journal*, 16(7), 2079-2093, 2016.

[17] Truong, P. H.; Lee, J.; Kwon, A. R.; Jeong, G.-M. (2016); Stride counting in human walking and walking distance estimation using insole sensors, *Sensors*, 16(6), 8-23, 2016.

[18] Weinberg, H. (2002); Using the ADXL202 in pedometer and personal navigation applications, *Analog Devices*, 2002.

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 13(4), 574-589, August 2018.

Arithmetic Operations with Spiking Neural P Systems with Rules and Weights on Synapses

H.F. Wang, K. Zhou, G.X. Zhang

Huifang Wang, Kang Zhou*
Department of Math and Computer
Wuhan Polytechnic University,
430023 Wuhan, Hubei, China
*Corresponding author: zhoukang65@whpu.edu.cn
13667286986@139.com

Gexiang Zhang
1. School of Electrical Engineering
Southwest Jiaotong University, Chengdu, China
2. Robotics Research Center
Xihua University, Chengdu, China
zhgxdylan@126.com

Abstract: The application of spiking neural P systems with rules and weights on synapses to arithmetic operations is discussed in this paper. We design specific spiking neural P systems with rules and weights on synapses for successfully performing addition, multiplication and the greatest common divisor. This is the first attempt to discuss the application of the new variant of spiking neural P systems, spiking neural P systems with rules and weights on synapses, and especially the use of spiking neural P systems to perform the greatest common divisor. Comparing with the results reported in the literature, smaller number of neurons are required to fulfill the arithmetic operations.
Keywords: SN P systems, rules and weights on synapses, addition, multiplication, the greatest common divisor.

1 Introduction

Membrane computing was initiated in [12]. The distributed and parallel computing devices in membrane computing are called P systems [13]. Some applications of P systems were reported in [2, 28, 29]. Especially, applications of P systems to arithmetic operations were discussed in [1, 2, 4, 31]. Some views on membrane computing were given in [3, 14]. For more information about membrane computing, one can refer to [15, 33]. Spiking neural P systems (in short, SN P systems) are a variant of P systems, which were introduced in [5] as a class of parallel and distributed computing models.

In recent years, SN P systems have been widely investigated in theory and applications. For instance, various variants of SN P systems were constructed by considering different biological sources [9,11,17,19,20,22,27] and the computing power/computational efficiency of some variants was discussed [10,18,26]. A wide range of applications of SN P systems was also presented, such as optimization [30], knowledge representation [23], fault diagnosis [16,24], and logical gates and circuits [6, 21]. SN P systems have been also used to perform arithmetic operations [8, 25, 32]. In order to answer the open problem in [8], SN P systems with a single input neuron for the addition and multiplication were constructed in [32]. These studies indicate that arithmetic/logic operations are promising applications of SN P systems.

In this paper, the application of SN P systems with rules and weights on synapses (RWSN P systems) is discussed to perform arithmetic operations. Smaller number of neurons are required to

construct the SN P systems for addition of n natural numbers and multiplication of two arbitrary natural numbers compared with the ones in [32]. This is the first attempt to discuss the use of spiking neural P systems to perform the greatest common divisor. We divide the solution for the greatest common divisor into several individual modules, and the binary representation method in [34] is used here. The inputs and outouts of these systems are natural numbers expressed in binary form, which are encoded as appropriate sequences of spikes.

This paper is organized as follows. Section 2 briefly introduces the SN P systems with rules and weights on synapses. In Section 3, SN P systems with rules and weights on synapses are used to perform arithmetic operations. Conclusions and future work are presented in Section 4.

2 Spiking neural P Systems with rules and weights on synapses

In this section, SN P systems with rules and weights on synapses are briefly described. More details can be referred to [7].

Such a system of degree $m \geq 1$ is a construct of the form:

$$\prod = (O, \sigma_1, \sigma_2, \ldots, \sigma_m, syn, \sigma_{in}, \sigma_{out}),$$

where:

- $O = \{a\}$ is the singleton alphabet (a is called spike);
- $\sigma_1, \sigma_2, \ldots, \sigma_m$ are neurons of the form $\sigma_i = (n_i)$, with $1 \leq i \leq m$, where n_i is initial number of spikes in neuron σ_i;
- syn is the set of synapses; each element in syn is a pair of the form $((i,j), w, R_{(i,j)})$, where (i,j) indicates that there is a synapse connecting neurons σ_i and σ_j, with $i, j \in \{1, 2, \ldots, m\}, i \neq j$; $w \in N(w \neq 0)$ is the weight on synapse (i,j); $R_{(i,j)}$ is a finite set of rules of the following two forms:
  - (1) $E/a^c \to a; d$, where E is a regular expression over O, $c \geq 1$ and $d \geq 0$;
  - (2) $a^s \to \lambda$, for some $s \geq 1$, with the restriction that $a^s \notin L(E)$ for any rule $E/a^c \to a; d$ from any $R_{(i,j)}$;
- σ_{in} and σ_{out} indicates the input neuron and the output neuron, respectively.

A rule $E/a^c \to a; d$ is called a standard firing rule; if $L(E) = \{a^c\}$, then the rule can be written in the simplified form $a^c \to a; d$. A rule of the form $a^s \to \lambda$ is called a forgetting rule. The firing rules are applied as follows. If $E/a^c \to a; d \in R_{(i,j)}$, and neuron σ_i contains k spikes such that $a^k \in L(E)$, $k \geq c$, then the rule is enabled. This means consuming (removing) c spikes (thus only $k-c$ remain in σ_i) from neuron σ_i. Neuron σ_i is fired, and it produces one spike. This one spike is multiplied w times by the weight w of the synapse in the process of transmission, and then reaches neuron σ_j after d time units (that is, w spikes arrive at neuron σ_j). When neuron σ_i contains exactly s spikes, then forgetting rule $a^s \to \lambda \in R_{(i,j)}$ is enabled. By using it, $s \geq 1$ spikes are removed from the neuron σ_i.

In order to express the synapse between the output neuron and the environment, the environment is represented by the letter e, and the synapse between the output neuron and the environment can be expressed as (out, e).

3 Arithmetic operations with SN P systems with rules and weights on synapses

In this section, SN P systems with rules and weights on synapses are used to design specific systems to perform three types of arithmetic operations, addition, multiplication and the greatest common divisor.

3.1 Addition

In this subsection, we describe an SN P system with rules and weights on synapses that performs the addition of n natural numbers. Such a system is called the SN P system with rules and weights on synapses for $n - addition$. The system has only one input neuron and can be used to perform the sum of arbitrary n natural numbers with k binary bits, where $n \geq 2$, $k \geq 1$. Compared with the standard SN P system, this type of SN P systems can reduce the number of neurons.

As shown in Fig. 1, we construct an SN P system $\prod_{Add}(n, k)$ with rules and weights on synapses to solve the $n - addition$ problem. Its formal definition is omitted.

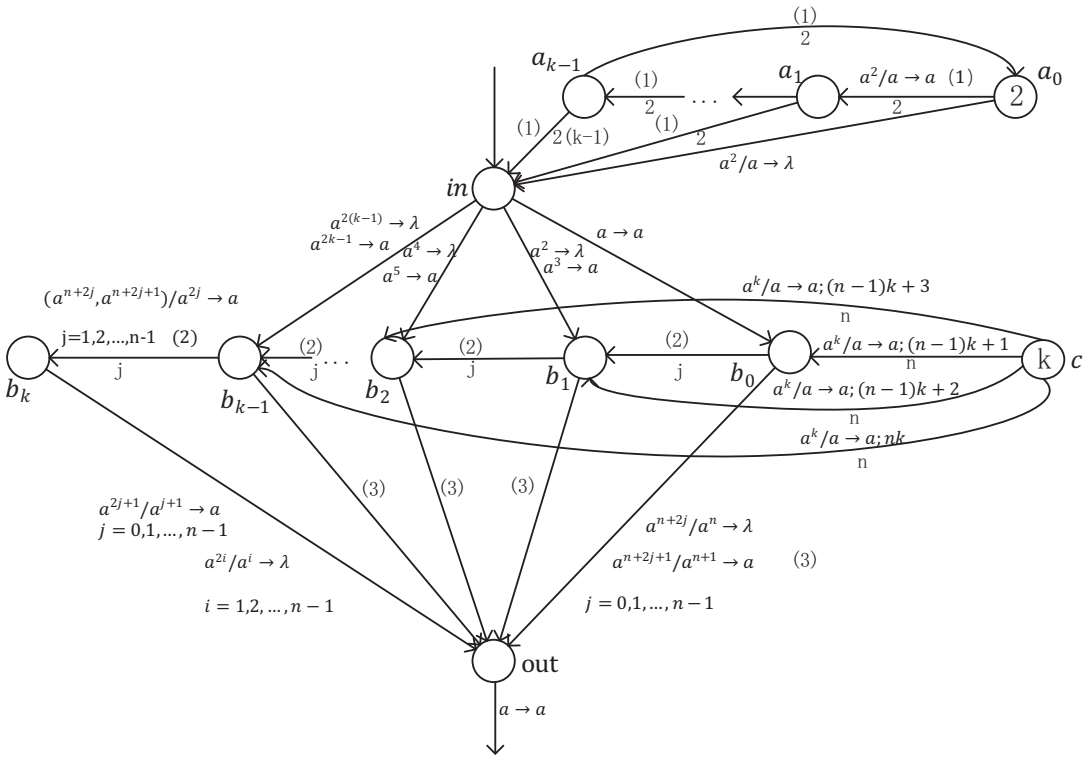


Figure 1: The SN P system $\prod_{Add}(n, k)$ with rules and weights on synapses for addition

The SN P system with rules and weights on synapses for $n - addition$ outputs the sum in binary form of n natural numbers with k binary bits, provided that the neuron σ_{in} is in binary form.

Let t denote the current time step. In the initial configuration ($t = 0$), only the neuron σ_{a_0} in the system contains 2 spikes and the neuron σ_c contains k spikes. The computing procedure of the SN P system with rules and weights on synapses in Fig. 1 is composed of the following steps:

1. Preprocessing of inputs. At time $t = 0$, with two spikes in the auxiliary neuron σ_{a_0}, the rules on synapses (a_0, in) and (a_0, a_1) can be applied. One of the two spikes is consumed by the firing rule $a^2/a \to a$ on synapse (a_0, a_1), and one new spike is produced. Then this new spike is multiplied by the weight 2 on synapse (a_0, a_1) during the transfer. That is, at the next time, two spikes will arrive at neuron σ_{a_1}, so that the rules on the two synapses emitted by neuron σ_{a_1} will be enabled and applied. The other spike is consumed by the forgetting rule $a^2/a \to \lambda$ on synapse (a_0, in), so the number of spikes that neuron σ_{in} receives from the auxiliary neuron σ_{a_0} is 0 at $t = 1$.

Similarly, the above process is repeated and then we can get that when time $t = 2$, neuron σ_{in} receives two spikes from the auxiliary neuron σ_{a_1}(the weight on synapse (a_1, in) is 2); until time $t = k$, neuron σ_{in} receives $2(k-1)$ spikes from neuron $\sigma_{a_{k-1}}$ (the weight on synapse (a_{k-1}, in) is $2(k-1)$). Then the auxiliary neurons send spikes to σ_{in} again, from σ_{a_0} to $\sigma_{a_{k-1}}$, until the computation ends.

2. Input and store the first n-1 numbers. At time $t = 1$, the digit which is associated with the power 2^0 in the binary representation of the first natural number is provided to the input neuron σ_{in}, while the auxiliary neuron σ_{a_0} sends 0 spikes to σ_{in}. At this time, the number of spikes that neuron σ_{in} contains may be 0 or 1. If there is no spike and no rule is activated, then 0 spike will be sent to neuron σ_{b_0} at the next time. If there is one spike, then the firing rule $a \to a$ on synapse (in, b_0) is triggered. This means that neuron σ_{b_0} can receive one spike at the next time. In this way, we can store the lowest bit of the first natural number into the neuron σ_{b_0}.

Similarly, we can store the digits which are associated with the power 2^1, 2^2, ..., 2^{k-1} of the first natural number into σ_{b_1}, σ_{b_2} , ..., $\sigma_{b_{k-1}}$, respectively. At the next time, the second natural number will begin to be input to the system, and the input process is same as that of the first number.

3. Input the last natural number and calculate the sum of each bit. At $t = (n-1)k + 2$, the spikes of the lowest bit of the last natural number arrive in σ_{b_0}, and at the same time, n spikes transmitted by synapse (c, b_0) also arrive at σ_{b_0} after $(n-1)k + 1$ steps of delay. The rules on synapses (b_0, out) and (b_0, b_1) can be applied when the number of spikes in σ_{b_0} is at least n and $n + 2$, so the system starts to calculate the sum and binary carry at this step, which corresponds to the power 2^0 of the n natural numbers. Then the sum is sent to the output neuron σ_{out}, and the carry is sent to the next neuron σ_{b_1}.

By the same token, the binary digits corresponding to the power 2^i of the last natural number and the n spikes transmitted by synapse (c, b_i) arrive at σ_{b_i} at the same time. Differing from σ_{b_0}, the carry from $\sigma_{b_{i-1}}$ also arrives at σ_{b_i} at this time, where $i = 1, 2, \ldots, k-1$, and the neuron σ_{b_k} only receives the carry from $\sigma_{b_{k-1}}$.

In neuron σ_{b_i}, we assume that the number of spikes receiving from the input neuron σ_{in} is p, and the number of spikes that receives from the previous neuron $\sigma_{b_{i-1}}$ is q, then the neuron σ_{b_i} contains $n + p + q$ spikes. The rules on the synapse (b_i, out) can be divided into two cases, where $i = 0, 1, \ldots, k-1$.

- If the value of $p + q$ is odd, that is $p + q = 2j + 1$, where $j = 0, 1, \ldots, n-1$, the rule $a^{n+2j+1}/a^{n+1} \to a$ on synapse (b_i, out) is applied and consumes $n+1$ spikes in neuron σ_{b_i}, then one spike is sent to the output neuron σ_{out}. The output neuron will send this one spike to the environment at next time, that is, the corresponding binary bit of sum is 1.

- If the value of $p + q$ is even, that is $p + q = 2j$, where $j = 0, 1, \ldots, n-1$, then n spikes in neuron σ_{b_i} are forgotten by the rule $a^{n+2j}/a^n \to \lambda$ on synapse (b_i, out), and no new spikes are produced. That is, the corresponding binary bit of sum is 0.

At the same time of producing each bit of the sum, it is possible to generate carry. When $j \neq 0$ (without 0 or 1 spike), the rule $(a^{n+2j}, a^{n+2j+1})/a^{2j} \to a$ on synapse (b_i, b_{i+1}) is applied and produces one spike. Then j spikes are sent to the next neuron (the weight on synapse (b_i, b_{i+1}) is j), where $j = 1, 2, \ldots, n-1$. It should be noted that the neuron σ_{b_k} only receives the carry spikes from the previous neuron $\sigma_{b_{k-1}}$.

4. Output the result. Each of the binary bits of the sum can be sent to the output neuron σ_{out} at different steps, and σ_{out} uses the rule $a \to a$ to output the result to the environment based on the number 0 or 1 of spikes in σ_{out}.

Based on the above description, the system shown in Fig. 1 can calculate the sum of the n natural numbers with k binary bits, where $n \geq 2$, $k \geq 1$, and send the result to the environment.

As an example, let us consider the addition $110_2 + 011_2 = 1001_2$, that is, $n = 2$, $k = 3$. Fig. 2 depicts the SN P system $\prod_{Add}(2, 3)$ with rules and weights on synapses. Table 1 reports the spikes contained in each neuron of $\prod_{Add}(2, 3)$, as well as the number of spikes sent to the environment, at each step during the computation. The input and the output sequences are written in bold. Note that the first instance of time for which the output is valid is $t = 7$.

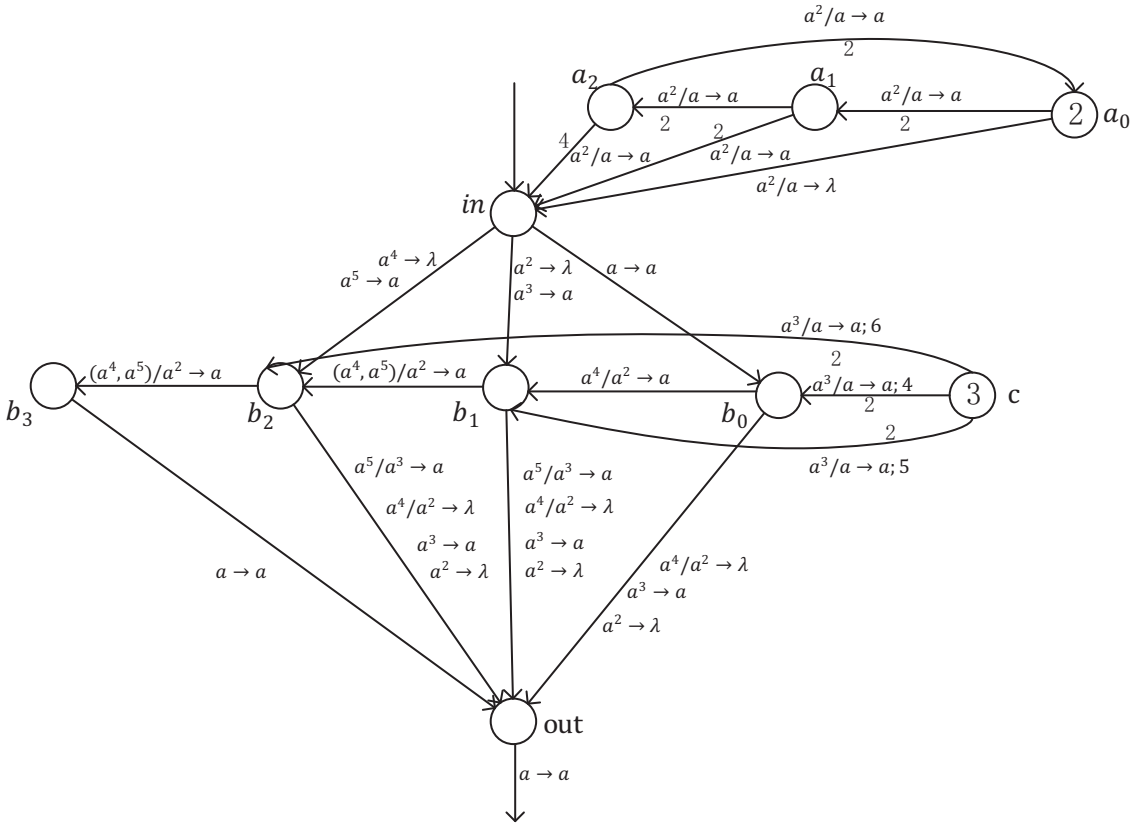


Figure 2: The SN P system $\prod_{Add}(2, 3)$ with rules and weights on synapses for addition

Compared with the example in [32], when $k = 3$, the number of neurons required to solve this problem by using the standard SN P system is 14; as shown in Fig. 2, the number of neurons required in SN P system with rules and weights on synapses is 10, which effectively reduces the number of neurons.

Table 1: The configurations and outputs of $\prod_{Add}(2,3)$ at each time step during the computation of the addition $110_2 + 011_2 = 1001_2$

t	in	a_0	a_1	a_2	c	b_0	b_1	b_2	b_3	out	$output$
0	0	2	0	0	3	0	0	0	0	0	0
1	**0**	0	2	0	$O(4)O(5)O(6)$	0	0	0	0	0	0
2	**1**+2	0	0	2	$O(3)O(4)O(5)$	0	0	0	0	0	0
3	**1**+4	2	0	0	$O(2)O(3)O(4)$	0	1	0	0	0	0
4	**1**	0	2	0	$O(1)O(2)O(3)$	0	1	1	0	0	0
5	**1**+2	0	0	2	$O(1)O(2)$	3	1	1	0	0	0
6	**0**+4	2	0	0	$O(1)$	0	4	1	0	1	0
7	0	0	2	0	0	0	0	4	0	0	**1**
8	2	0	0	2	0	0	0	0	1	0	**0**
9	4	2	0	0	0	0	0	0	0	1	**0**
10	0	0	2	0	0	0	0	0	0	0	**1**

3.2 Multiplication

Multiplication of two arbitrary natural numbers is also called the general binary multiplication, which has been performed on an SN P system in [32]. In order to reduce the number of neurons in the system, for two binary numbers m and n with k binary bits, we construct the SN P system $\prod_{Multiple}(k)$ with rules and weights on synapses as shown in Fig. 3 to perform the general binary multiplication, where $k \geq 1$, $m \geq 0$, $n \geq 0$.

The SN P system with rules and weights on synapses for the general binary multiplication of Fig. 3 outputs the product $m \times n$ in binary form of two natural numbers m and n with k binary bits, provided that the neuron σ_{in} is in binary form. Assuming that m and n are two natural numbers with k binary bits, and m and n are rewritten as follows:

$$m = \sum_{i=0}^{k-1} m_i 2^i, n = \sum_{j=0}^{k-1} n_j 2^j, \text{then:}$$

$$m \times n = n_0 m_0 2^0 + (n_0 m_1 + n_1 m_0)2^1 + \cdots + (n_1 m_{k-1} + n_2 m_{k-2} + \cdots + n_{k-1} m_1)2^k + \cdots + n_{k-1} m_{k-1} 2^{2k-2}$$

From the above expression we can see that the product of m and n can be decomposed into the sum of $2k-1$ terms.

Let t denote the current time step. In the initial configuration ($t = 0$), the neuron σ_{a_0} contains 2 spikes and σ_{g_i} contains k spikes, where $i = 2, 3, \ldots, k$. Fig. 3 shows the SN P system with rules and weights on synapses for the general binary multiplication, and the operation process consists of the following steps:

1. Preprocessing of input. It is similar to the input preprocessing part of the system $\prod_{Add}(n,k)$, so we only describe the differences.

The auxiliary neuron σ_{a_i} only acts on the natural number m, which is the first input to the system, where $i = 0, 1, \ldots, k-1$. When each of the corresponding binary bits of m is input to

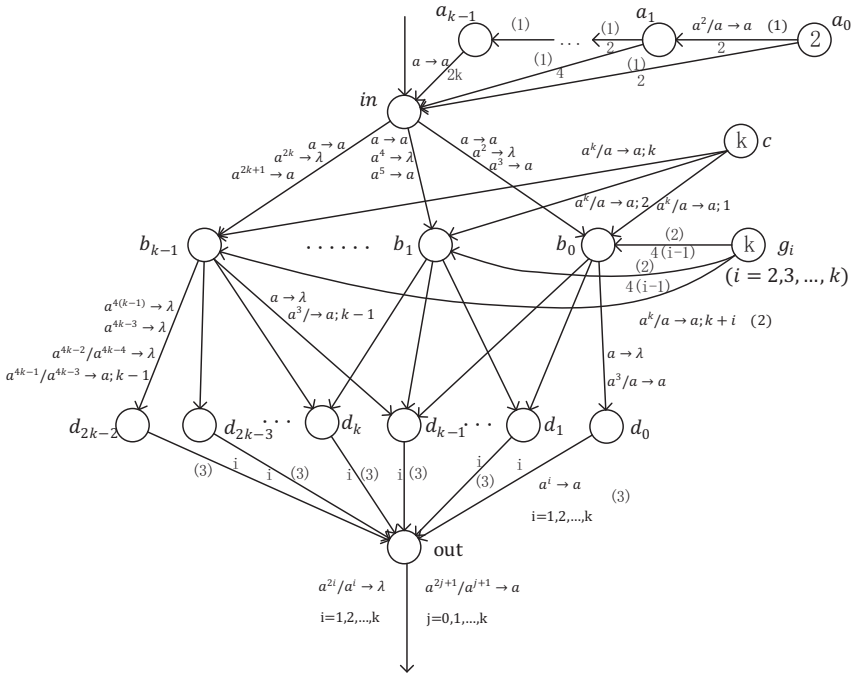


Figure 3: The SN P system $\prod_{Multiple}(k)$ with rules and weights on synapses for multiplication

σ_{in}, the auxiliary neurons σ_{a_0}, σ_{a_1}, ..., $\sigma_{a_{k-1}}$ send 2, 4 ,..., $2k$ spikes to σ_{in}, respectively.

2. Input and store the binary bits of the first natural number m**.** At $t = 1$, the digit which is associated with the power 2^0 in the binary representation of m has been provided to σ_{in}, while the auxiliary neuron σ_{a_0} sends 2 spikes to σ_{in}. So the number of spikes that neuron σ_{in} contains may be 2 or 3. If there are 2 spikes, then the rule $a^2 \to \lambda$ on synapse (in, b_0) can be applied, so that the 2 spikes are consumed and no spike is sent out. At $t = 2$, the neuron σ_{b_0} receives only one spike from synapse (c, b_0) (after one step delay), and this spike will be forgotten by rule $a \to \lambda$ on synapse (b_0, d_0) at the next step. If there are 3 spikes, where two of them come from σ_{a_0} and the other one from the environment, then the rule $a^3 \to a$ on synapse (in, b_0) is triggered; as a consequence, one spike is sent out. Then at $t = 2$, there are two spikes placed in neuron σ_{b_0}, where one of them comes from synapse (in, b_0) and the other one from synapse (c, b_0). These two spikes remain in σ_{b_0} for the next steps.

Thus, k binary bits of m can be stored in the neurons σ_{b_0}, σ_{b_1}, ..., $\sigma_{b_{k-1}}$ by repeating the above operations. If there are 2 spikes, the $i-th$ binary bit of m is 1; if there is no spike, the $i-th$ binary bit of m is 0, where $i = 0, 1, \ldots, k-1$.

3. Each binary number of the natural number m **multiplies all binary numbers of** n**, which can produce all possible products of binary numbers of** m **and** n**.** At $t = k+1$, the digit which is associated with the power 2^0 in the binary representation of n has been provided to the input neuron σ_{in}. If the digit is 1, then each rule $a \to a$ on synapses $(in, b_0), (in, b_1), \ldots, (in, b_{k-1})$ is triggered at the same time, and each of neurons σ_{b_0}, σ_{b_1}, ..., $\sigma_{b_{k-1}}$ will receive one spike at next step. Otherwise, no spike will be sent to neurons σ_{b_0}, σ_{b_1}, ..., $\sigma_{b_{k-1}}$.

Now let us focus on the neuron σ_{b_i}, where $i = 0, 1, \ldots, k-1$. At $t = k+2$, the number of spikes in σ_{b_i} may be 0,1,2,3, and we can thus consider the following four cases.

- If σ_{b_i} contains 0 spikes, then no rule can be applied, thus no spike is sent out. This encodes the operation $n_0 m_i = 0 \times 0 = 0$.

- If σ_{b_i} contains 1 spike, then it comes from σ_{in}. The rule $a \to \lambda$ on synapse (b_i, d_i) is triggered, so that one spike is consumed and no spike is sent out. This encodes the operation $n_0 m_i = 1 \times 0 = 0$.

- If σ_{b_i} contains 2 spikes, then they are all remained from the previous step. No rule can be applied. This encodes the operation $n_0 m_i = 0 \times 1 = 0$.

- If σ_{b_i} contains 3 spikes, then two of them are remained from the previous step and the other one comes from σ_{in} at this step. The rule $a^3/a \to a; i$ on synapse (b_i, d_i) is triggered; as a consequence, one spike reaches the neuron σ_{d_i} after i steps of delay and one spike is consumed, thus two spikes in σ_{b_i} are left for the next step. This encodes the operation $n_0 m_i = 1 \times 1 = 1$.

When n_1 reaches neuron σ_{b_i}, σ_{b_i} also receives four spikes from neuron σ_{g_2} (after $k+2$ steps delay). Therefore, the number of spikes contained in neuron σ_{b_i} at this time may be 4, 5, 6, and 7, respectively, corresponding to the four cases 0, 1, 2, and 3 mentioned above, and the rules on synapses (b_i, d_{i+1}) can be applied and send the result $n_1 m_i$ to neuron $\sigma_{d_{i+1}}$, where $i = 0, 1, \ldots, k-1$.

Similarly, we can get $n_2 m_i, \ldots, n_{k-1} m_i$ in the same way, and $n_j m_i$ is sent to neuron $\sigma_{d_{i+j}}$, where $0 \le i \le k-1$, $0 \le j \le k-1$.

4. The sum of terms. At $t = k+3$, $n_0 m_0$ arrives at neuron σ_{d_0} through synapse (b_0, d_0). ... At $t = 3k+1$, $n_{k-1} m_{k-1}$ arrive at $\sigma_{d_{2k-2}}$. The number of spikes contained in neuron σ_{d_i} above corresponds to the binary bit of $m \times n$, where $i = 0, 1, \ldots, 2k-2$.

5. Output the result. The spikes in neurons σ_{d_0}, σ_{d_1}, ..., $\sigma_{d_{2k-2}}$ above are sent to the output neuron σ_{out}. According to the number of spikes in the output neuron σ_{out}, we consider the following two cases.

- If the number of spikes is odd, the rule $a^{2j+1}/a^{j+1} \to a$ on synapse (out, e) is applied, so that $j+1$ spikes are consumed and one spike is sent to the environment, and j spikes remain in σ_{out} for the next step.

- If the number of spikes is even, the rule $a^{2j}/a^j \to \lambda$ on synapse (out, e) is applied, so that j spikes are consumed and no spike is sent to the environment, and j spikes remain in σ_{out} for the next step.

From time $t = k+5$, the system begins to have output. Therefore, the system shown in Fig. 3 can perform the general binary multiplication, and output the result to the environment.

As an example, let us consider the multiplication $11_2 \times 11_2 = 1001_2$, that is $k = 2$. Fig. 4 depicts the SN P system $\prod_{Multiple}(2)$ with rules and weights on synapses. Table 2 reports the spikes contained in each neuron of $\prod_{Multiple}(2)$, as well as the number of spikes sent to the environment, at each step during the computation. Note that the time for the first instance that the output is valid is $t = 7$.

When $k = 2$, the number of neurons required in the multiplication SN P system is 17 in [32]; as shown in Fig. 4, the number of neurons required in SN P system with rules and weights on synapses is 11, which effectively reduces 6 neurons.

3.3 The greatest common divisor

In this subsection, the greatest common divisor of two natural numbers is performed on SN P systems with rules and weights on synapses. We divide the solution for the greatest common divisor into several individual modules, and then call the different modules according to the

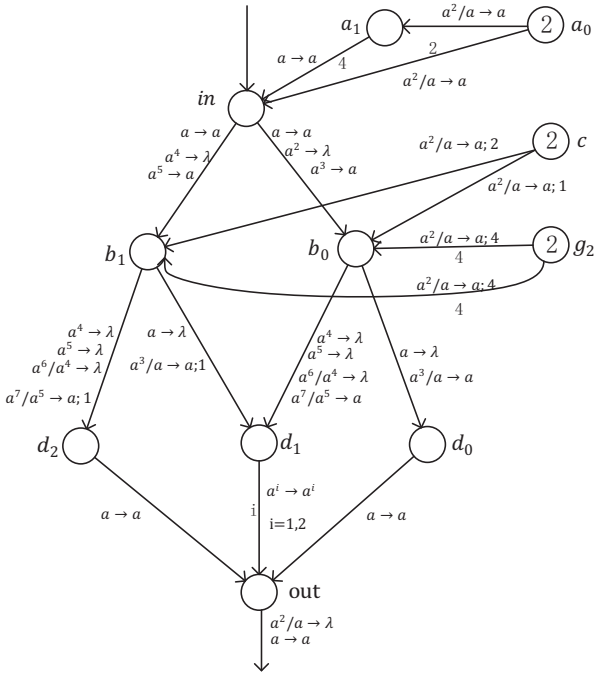


Figure 4: The SN P system $\prod_{Multiple}(2)$ with rules and weights on synapses for multiplication

method steps. The method for solving the greatest common divisor that we use is the binary method. Please refer to [34] to learn more details on the binary method.

Each of the modules required for the above method is given below. These modules are achieved by SN P systems with rules and weights on synapses.

Module 1: *To determine whether the natural number is even*

If a number is even, then the digit which is associated with the power 2^0 in the binary representation is 0; otherwise, this digit is 1. So we can only use the digit which is associated with the power 2^0 in the binary representation of the number to judge whether it is even or odd.

For a natural number, we construct the SN P system $\prod_{Even}$ with rules and weights on synapses as shown in Fig. 5 to determine whether it is even or not. If the input is even, the number of spikes that output to the environment is 0, otherwise, the output of this system is not 0. The time that the output is valid for the first instance is $t = 3$.

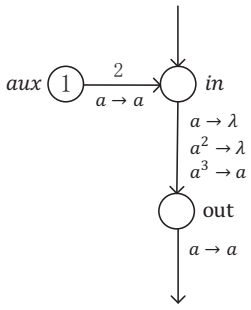


Figure 5: The SN P system $\prod_{Even}$ with rules and weights on synapses

Module 2: *To divide the number by 2 (to shift one bit to the right)*

The binary even number is divided by 2, which means that the last bit 0 of this binary number is removed. So if the length of binary bits of the input is k, then the length of binary

Table 2: The configurations and outputs of $\prod_{Multiple}(2)$ at each time step during the computation of the multiplication $11_2 \times 11_2 = 1001_2$

t	in	a_0	a_1	c	g_2	b_0	b_1	d_0	d_1	d_2	out	$output$
0	0	2	0	2	2	0	0	0	0	0	0	0
1	**1**+2	0	1	$O(1)O(2)$	$O(4)$	0	0	0	0	0	0	0
2	**1**+4	0	0	$O(1)$	$O(3)$	2	0	0	0	0	0	0
3	**1**	0	0	0	$O(2)$	2	2	0	0	0	0	0
4	**1**	0	0	0	$O(1)$	3	3	0	0	0	0	0
5	0	0	0	0	0	7	7	1	0	0	0	0
6	0	0	0	0	0	2	2	0	2	0	1	0
7	0	0	0	0	0	2	2	0	0	1	2	**1**
8	0	0	0	0	0	2	2	0	0	0	2	**0**
9	0	0	0	0	0	2	2	0	0	0	1	**0**
10	0	0	0	0	0	2	2	0	0	0	0	**1**

bits of the output is $k-1$.

For an even number, we construct the SN P system $\prod_{Divide2}$ with rules and weights on synapses as shown in Fig. 6 to divide it by 2. The time that the output is valid for the first instance is $t = 4$.

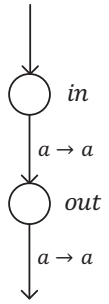


Figure 6: The SN P system $\prod_{Divide2}$ with rules and weights on synapses

Module 3: *To add 1 to the number*

For a natural number, we construct the SN P system $\prod_{Add1}$ with rules and weights on synapses as shown in Fig. 7 to add 1 to it. The first time step that the output starts to be emitted by the system is $t = 3$.

Module 4: *To compare the value of two natural numbers*

We construct the SN P system $\prod_{Comp}$ with rules and weights on synapses as shown in Fig. 8 to compare two natural numbers. The inputs of this system are two appropriate sequences of spikes of two natural numbers in binary form, but what is different from the other modules is that the input here starts from the highest bit to the lower.

For two natural numbers in binary form with the same length of binary bits, we compare

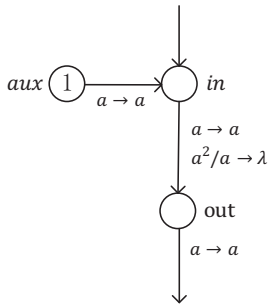


Figure 7: The SN P system $\prod_{Add1}$ with rules and weights on synapses

them from the highest bit of each number to the lower order, until the system outputs 2 or 1. The result is the last bit of the output. If the result is 0, then $in_1 = in_2$; if the result is 1, then $in_1 < in_2$; if the result is 2, then $in_1 > in_2$.

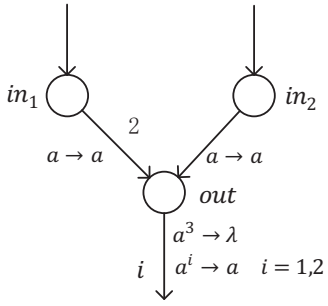


Figure 8: The SN P system $\prod_{Comp}$ with rules and weights on synapses

Module 5: *Subtraction*

The design principle of the subtraction module is referred to [8]. For two natural numbers, we construct the SN P system $\prod_{Sub}$ with rules and weights on synapses as shown in Fig. 9 to perform subtraction. The time that the output is valid for the first instance is $t = 4$.

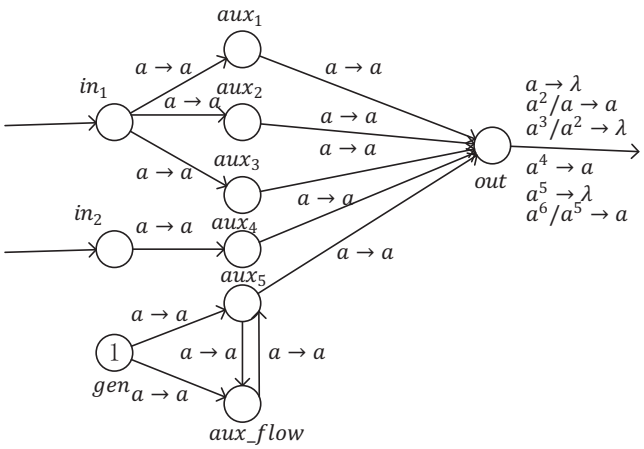


Figure 9: The SN P system $\prod_{Sub}$ with rules and weights on synapses

Module 6: *To calculate the result (to multiply by 2 with given times)*

By calling the above five modules, the numbers d and g are obtained. This module uses d and g to calculate the greatest common divisor $g \times 2^d$. That is, move g to the left by d bits,

Table 3: The configurations and outputs of $\prod_{Even}$ at each time step during the process of judging whether $a = 01100_2$ and $b = 10010_2$ is even

t	$in(a)$	$in(b)$	aux	out	$output$
0	0	0	1	0	0
1	**0**+2	**0**+2	0	0	0
2	**0**	**1**	0	0	0
3	**1**	**0**	0	0	**0**
4	**1**	**0**	0	0	**0**
5	**0**	**1**	0	0	**0**
6	0	0	0	0	**0**
7	0	0	0	0	**0**

where d represents the number of times that all the two numbers are even.

We construct the SN P system $\prod_{Multiply2}$ with rules and weights on synapses as shown in Fig. 10 to calculate the result. The input is the number g in binary form. The first time step that the output starts to be emitted by the system is $t = 3$.

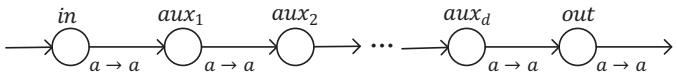


Figure 10: The SN P system $\prod_{Multiply2}$ with rules and weights on synapses

By calling above modules, we can get the greatest common divisor of two natural numbers. In what follows, an example is provided to detail the process.

As an example, let us consider the greatest common divisor of the numbers $a = 01100_2$ and $b = 10010_2$. The calling process of modules to calculate $gcd(01100_2, 10010_2)$ is as follows. In the initial time, we assume that $g = 0$ and $d = 0$.

Step 1. *Call the module 1 to determine whether a and b are even*

When the input of system $\prod_{Even}$ is $a = 01100_2$ or $b = 10010_2$, the spikes contained in each neuron of $\prod_{Even}$, as well as the number of spikes sent to the environment at each step during the process that judge whether a or b is even, are reported in Table 3.

From Table 3, we can know that the output of $\prod_{Even}$ is 0, so $a = 01100_2$ is an even number, and b=10010_2 is an even number, too.

Step 2. *Call the module 2 to divide a and b by 2*

When the input of system $\prod_{Divide2}$ is $a = 01100_2$, the output is 0110. When the input of system $\prod_{Divide2}$ is $b = 10010_2$, the output is 1001. So, after dividing a and b by 2, we can get $a = 0110_2$ and $b = 1001_2$.

Step 3. *Call the module 3 to add 1 to d*

In the initial time, $d = 0$, so the input of system $\prod_{Add1}$ is 0. After adding 1 to d, we can get $d = 1$.

Step 4. *Call the module 1 to determine whether a and b are even*

Table 4: The configurations and outputs of $\prod_{Comp}$ at each time step during the process of comparing the value of $a = 0011_2$ and $b = 1001_2$

t	in_1	in_2	out	$output$
0	0	0	0	0
1	**0**	**1**	0	0
2	**0**	**0**	1	0
3	**1**	**0**	0	0
4	**1**	**1**	2	**1**

When the input of system $\prod_{Even}$ is $a = 0110_2$, the output is 0, so $a = 0110_2$ is an even number. When the input of system $\prod_{Even}$ is $b = 1001_2$, the output is 1, so $b = 1001_2$ is odd.

Step 5. *Call the module 2 to divide a by 2*

The input of system $\prod_{Divide2}$ is $a = 0110_2$. After dividing a by 2, we get $a = 011_2$.

Step 6. *Call the module 1 to determine whether a is even*

The input of system $\prod_{Even}$ is $a = 011_2$, and then the output of $\prod_{Even}$ is 1, so $a = 011_2$ is an odd number.

Step 7. *Call the module 4 to compare the value of a and b*

In the system $\prod_{Comp}$, $in_1 = a = 0011_2$ and $in_2 = b = 1001_2$. Table 4 reports the spikes contained in each neuron of $\prod_{Comp}$, as well as the number of spikes sent to the environment at each step during the process of comparing the value of $a = 0011_2$ and $b = 1001_2$. The output of the system is 1, so $a < b$.

Step 8. *Call the module 5 and module 2 to calculate* $(b - a)/2$

Because $a < b$, we can get $in_1 = b = 1001_2$ and $in_2 = a = 0011_2$ in the system $\prod_{Sub}$. After computing in $\prod_{Sub}$, we get $b - a = 1001_2 - 0011_2 = 0110_2$.

Then the input of system $\prod_{Divide2}$ is 0110_2. After dividing it by 2, we can get $b = (b-a)/2 = 0110_2 \div 10_2 = 011_2$.

Step 9. *Call the module 4 to compare the value of a and b*

In the system $\prod_{Comp}$, $in_1 = a = 011_2$ and $in_2 = b = 011_2$. The output of the system is 0, so $a = b$.

Step 10. *Call the module 6 to calculate the result*

From the above steps, we can get $g = a = b = 011_2$ and $d = 1$. So the input of system $\prod_{Multiply2}$ is $g = 011_2$, and the spikes contained in each neuron of $\prod_{Multiply2}$, as well as the number of spikes sent to the environment at each step during computation of the multiplication $011_2 \times 2^1 = 110_2$ are reported in Table 5.

So the greatest common divisor of 01100_2 and 10010_2 is $g \times 2^d = 011_2 \times 2^1 = 110_2$.

Through the above analysis we can see that the greatest common divisor of two arbitrary natural numbers is available by calling the corresponding modules of method.

3.4 Discussion

For addition and multiplication, both the systems constructed in [32] and the ones in this paper have only one input neuron, and the integers encoded in binary form should be input in order. The system in [32] to perform the addition of n natural numbers with 3 binary bits

Table 5: The configurations and outputs of $\prod_{Multiply2}$ at each time step during the computation of the multiplication $011_2 \times 2^1 = 110_2$

t	in	aux_1	out	$output$
0	0	0	0	0
1	**1**	0	0	0
2	**1**	1	0	0
3	**0**	1	1	**0**
4	0	0	1	**1**
5	0	0	0	**1**
6	0	0	0	**0**

requires 14 neurons, while 10 neurons are enough by using SN P systems with rules and weights on synapses. When the multiplication of two natural numbers with 2 binary bits is performed, the system designed in [32] requires 17 neurons, while in this paper, we use only 11 neurons. Thus, we can conclude that the systems with a single input neuron for addition and multiplication in [32] are improved by using the SN P systems with rules and weights on synapses. It is worth pointing out that the designed SN P systems in this paper use larger number of synapses and delays than the systems in [32]. For addition, due to the constraints of input conditions, the systems need to calculate the sum of each bit until the last number is input, which makes the addition system relatively complicated. In addition, this study is the first attempt to design an SN P system to calculate the greatest common divisor of two natural numbers.

4 Conclusions and future work

The use of the SN P systems with rules and weights on synapses to design specific systems for fulfilling the arithmetic operations is focused in this paper, including addition, multiplication and the greatest common divisor. Comparing with the results reported in [32], smaller number of neurons are required to perform the addition and multiplication. Calculating the greatest common divisor of two natural numbers is not an easy task. This paper achieved a good design by using several modules. Following this work, we will consider how to use the SN P system with rules and weights on synapses to perform the division operation.

Acknowledgments

This work was supported by National Natural Science Foundation of China (61179032, 61672437, 61702428), the Special Scientific Research Fund of Food Public Welfare Profession of China (201513004-3), the Guiding Scientific Research Project of Hubei Provincial Education Department (2017078), the Humanities and Social Sciences Fund Project of Hubei Provincial Education Department (17Y071) and Sichuan Science and Technology Program (2018GZ0185, 2018GZ0085, 2017GZ0159).

Bibliography

[1] Atanasiu, A.; Martinvide, C. (2001); Arithmetic with membranes, *Romanian Journal of Information Science and Technology*, 4(12), 5–20, 2001.

[2] Ciobanu, G.; Păun, Gh.; Pérez-Jiménez, M.J. (2006); *Applications of membrane computing*, Springer, 2006.

[3] Dzitac, I. (2015); Impact of membrane computing and P systems in ISI WoS. Celebrating the 65th Birthday of Gheorghe Păun, *International Journal of Computers Communications & Control*, 10(5), 617–626, 2015.

[4] Guo, P.; Chen, H.; Zhang, H. (2015); An integrated P system for arithmetic operations, *Journal of Computational & Theoretical Nanoscience*, 12(10), 3346–3356, 2015.

[5] Ionescu, M.; Păun, Gh.; Yokomori, T. (2006); Spiking neural P systems, *Fundamenta Informaticae*, 71, 279–308, 2006.

[6] Ionescu, M.; Sburlan, D. (2012); Some applications of spiking neural P systems, *Computing and Informatics*, 27(3), 515–528, 2012.

[7] Kong, Y. (2005); *Study on the computational power of spiking neural P system based on different biological background*, Huazhong University of Science and Technology, 2005.

[8] Naranjo, M.A.G.; Leporati, A.; Pan, L. (2009); Performing arithmetic operations with spiking neural P systems, *Proc. of the Seventh Brainstorming Week on Membrane Computing*, 11(4), 181–198, 2009.

[9] Pan, L.; Păun, Gh. (2009); Spiking neural P systems with anti-spikes, *International Journal of Computers Communications & Control*, 4(3), 273–282, 2009.

[10] Pan, L.; Păun, Gh.; Zhang, G.; Neri, F. (2017); Spiking neural P systems with communication on request, *International Journal of Neural Systems*, 27(8), 2017.

[11] Pan, L.; Zeng, X.; Zhang, X.; Jiang, X. (2012); Spiking neural P systems with weighted synapses, *Neural Processing Letters*, 35(1), 13–27, 2012.

[12] Păun, Gh. (2000); Computing with membranes, *Journal of Computer and System Sciences*, 61(1), 108–143, 2000.

[13] Păun, Gh. (2002); Membrane computing: an introduction, *Theoretical Computer Science*, 287(1), 73–100, 2002.

[14] Păun, Gh. (2016); Membrane computing and economics: A general view, *International Journal of Computers Communications & Control*, 11(1), 105–112, 2016.

[15] Păun, Gh.; Rozenberg, G.; Salomaa, A. (2010); *The oxford handbook of membrane computing*, Oxford University Press, 2010.

[16] Peng, H.; Wang, J.; Pérez-Jiménez, M.J.; Wang, H.; Shao, J.; Wang, T. (2013); Fuzzy reasoning spiking neural P system for fault diagnosis, *Information Sciences*, 235(6), 106–116, 2013.

[17] Song, T.; Gong, F.; Liu, X.; Zhao, Y.; Zhang, X. (2016); Spiking neural P systems with white hole neurons, *IEEE Transactions on Nanobioscience*, 15(7), 666–673, 2016.

[18] Song, T.; Pan, L. (2015); Spiking neural P systems with rules on synapses working in maximum spikes consumption strategy, *IEEE Transactions on NanoBioscience*, 14(1), 38–44, 2015.

[19] Song, T.; Pan, L.; Păun, Gh. (2014); Spiking neural P systems with rules on synapses, *Theoretical Computer Science*, 529, 82–95, 2014.

[20] Song, T.; Rodríguez-Patón, A.; Zheng, P.; Zeng, X. (2017); Spiking neural P systems with colored spikes, *IEEE Transactions on Cognitive & Developmental Systems*, doi: 10.1109/TCDS.2017.2785332, 2017.

[21] Song, T.; Zheng, P.; Wong, M.L.D.; Wang, X. (2016); Design of logic gates using spiking neural P systems with homogeneous neurons and astrocytes-like control, *Information Sciences*, 372, 380–391, 2016.

[22] Wang, J.; Hoogeboom, H.J.; Pan, L.; Păun, Gh.; Pérez-Jiménez, M.J. (2010); Spiking neural P systems with weights, *Neural Computation*, 22(10), 2615–2646, 2010.

[23] Wang, J.; Shi, P.; Peng, H.; Pérez-Jiménez, M.J.; Wang, T. (2013); Weighted fuzzy spiking neural P systems, *IEEE Transactions on Fuzzy Systems*, 21(2), 209–220, 2013.

[24] Wang, T.; Zhang, G.; Zhao, J.; He, Z.; Wang, J.; Pérez-Jiménez, M.J. (2015); Fault diagnosis of electric power systems based on fuzzy reasoning spiking neural P systems, *IEEE Transactions on Power Systems*, 30(3), 1182–1194, 2015.

[25] Zeng, X.; Song, T.; Zhang, X.; Pan, L. (2012); Performing four basic arithmetic operations with spiking neural P systems, *IEEE Transactions on Nanobioscience*, 11(4), 366–374, 2012.

[26] Zeng, X.; Xu, L.; Liu, X.; Pan, L. (2014); On languages generated by spiking neural P systems with weights, *Information Sciences*, 278(10), 423–433, 2014.

[27] Zeng, X.; Zhang, X.; Song, T.; Pan, L. (2014); Spiking neural P systems with thresholds, *Neural Computation*, 26(7), 1340–1361, 2014.

[28] Zhang, G.; Cheng, J.; Gheorghe, M.; Meng, Q. (2013); A hybrid approach based on differential evolution and tissue membrane systems for solving constrained manufacturing parameter optimization problems, *Applied Soft Computing*, 13(3), 1528–1542, 2013.

[29] Zhang, G.; Pérez-Jiménez, M.J.; Gheorghe, M. (2017); *Real-life applications with membrane computing*, Springer International Publishing, 2017.

[30] Zhang, G.; Rong, H.; Neri, F.; Pérez-Jiménez, M.J. (2014); An optimization spiking neural P system for approximately solving combinatorial optimization problems, *International Journal of Neural Systems*, 24(5), 450–3642, 2014.

[31] Zhang, G.; Rong, H.; Ou, Z.; Pérez-Jiménez, M.J.; Gheorghe, M. (2014); Automatic design of deterministic and non-halting membrane systems by tuning syntactical ingredients, *IEEE Transactions on NanoBioscience*, 13(3), 363–371, 2014.

[32] Zhang, X.; Zeng, X.; Pan, L.; Luo, B. (2009); A spiking neural P system for performing multiplication of two arbitrary natural numbers, *Chinese Journal of Computers*, 32(12): 2362–2372, 2009.

[33] [Online]. Available: ppage.psystems.eu.

[34] [Online]. Available: en.wikipedia.org/wiki/Greatest_common_divisor.

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 13(4), 590-601, August 2018.

An ABC Algorithm with Recombination

X. You, Y. Ma, Z. Liu, M. Xie

Xuemei You*, Yinghong Ma, Zhiyuan Liu, Mingzhao Xie
Business School,
Shandong Normal University,
Shandong, 250014, P. R. China
*Corresponding author: sdxmyou@126.com yinghongma71@163.com
liuzhiyuan@sdnu.edu.cn

Abstract: Artificial bee colony (ABC) is an efficient swarm intelligence algorithm, which has shown good exploration ability. However, its exploitation capacity needs to be improved. In this paper, a novel ABC variant with recombination (called RABC) is proposed to enhance the exploitation. RABC firstly employs a new search model inspired by the updating equation of particle swarm optimization (PSO). Then, both the new search model and the original ABC model are recombined to build a hybrid search model. The effectiveness of the proposed RABC is validated on ten famous benchmark optimization problems. Experimental results show RABC can significantly improve the quality of solutions and accelerate the convergence speed.
Keywords: Artificial bee colony (ABC), recombination, hybrid search model, global optimization.

1 Introduction

In recent years, swarm intelligence (SI) has become a research focus in optimization field. The SI refers to establish mathematical model to simulate the social behaviors from nature. In the past decades, different SI algorithms have been proposed, including ant colony optimization (ACO) [8, 17, 25], particle swarm optimization (PSO) [22], artificial bee colony (ABC) [9], firefly algorithm (FA) [20], hybrid algorithm (HA) [14], bat algorithm (BA) [2], and cuckoo search (CS) [5, 30].

Except PSO and ACO, ABC can be regarded as the most popular SI algorithm. The main reason contains: 1) ABC has less control parameters than other SI algorithms; and 2) ABC has powerful exploration ability. Due to the superiority of ABC, it has received much attention. In the original ABC, individuals are divided into three types: i.e., employed bees, onlooker bees and scouts. During the search, all individuals (bees) fly in the search space and try to improve the food sources (find better solutions). Compared to other SI algorithms, ABC uses a different search equation to generate new solutions. For the current solution, an individual (employed bee or onlooker bee) randomly chooses a different solution in the population and uses their difference to obtain a new solution by modifying one dimension. Based on this search mechanism, ABC shows slow convergence speed and exploitation ability.

To strengthen the exploitation capacity, this paper presents a new ABC variant (called RABC), which employs a recombination method between the original ABC search model and a modified search model. For the latter model, it is inspired by the updating equation of PSO. By combining the search information of the global best solution and previous best, RABC can improve the exploitation ability. Experiment is validated on ten well-known benchmark problems. Simulation results of RABC are compared with the original ABC and two improved ABC variants.

The rest of this paper is organized as follows. The original ABC is briefly introduced in Section 2. A short review of recent progress on ABC is given in Section 3. In Section 4, our

proposed RABC is described. Section 5 presents the simulation results and discussions. Finally, conclusion and future work are summarized in Section 6.

2 Artificial Bee Colony (ABC)

There are two famous SI optimization algorithms: PSO and ACO, which were developed in the 1990s. Recently, different SI algorithms were proposed. Among these algorithms, ABC becomes popular because of its superiorities [10]. In the original ABC, it consists of three types of individuals: employed bees, onlooker bees and scouts. For all food sources (solutions), the employed bees conduct the first round search around food sources and try to find new better solutions. The onlooker bees conduct the second round search around some selected better food sources. For a food source, if the above bees cannot find a better one to replace it after some rounds search, the scout will randomly find a food source to replace it.

2.1 Population initialization

In the following, we will describe the original ABC in details. First, a initial population consists of N food sources (solutions) $\{X_i|i = 1, 2, ..., N\}$, where X_i is the ith food source and N is the population size. The initial food sources are randomly generated as follows.

$$x_{ij} = low_j + rand_j \cdot (up_j - low_j) \tag{1}$$

where x_{ij} denotes the jth component for ith food source X_i, $i = 1, 2, ..., N$, $j = 1, 2, ..., D$; D is the dimensional size; $rand_j$ is a real random number between 0 and 1; up_j and low_j are the boundaries for the jth dimension.

2.2 Employed bee phase

For each food source X_i in the current population, each employed bee flies to its neighborhood and tries to find a new solution V_i. This process can be described as below [9].

$$v_{ij}(t) = x_{ij}(t) + \phi_{ij}(x_{ij}(t) - x_{kj}(t)) \tag{2}$$

where $j\epsilon[1, D]$ is a random index of the population; X_k is a randomly selected food source and it is mutually different with X_i; t represents the iteration index; ϕ_{ij} is a real random number in the range [-1, 1].

The ABC employs a greedy selection method to determine whether the new candidate solution (food source) V_i should be entered in the next generation. If V_i is better than X_i, then X_i is updated by V_i; otherwise X_i is unchangeable. The greedy selection can accelerate the population convergence.

2.3 Onlooker bee phase

The onlooker bees only fly to some selected food sources and search their neighborhood to find new food sources. The selection of each food source X_i is related to its fitness quality. A better food source has a higher selection probability. In the original ABC, the selection probability p_i for each food source X_i is defined by [9]:

$$p_i = \frac{fit_i}{\sum_{i=1}^{N} fit_i} \tag{3}$$

where fit_i is the fitness value of X_i. When X_i is selected, the onlooker bee searches the neighborhood of X_i and obtain a new solution V_i according to Eq.(2).

Similar to the employed bees, the same greedy selection is utilized by the onlooker bees to determine whether the new solution V_i should be entered in the next generation.

2.4 Scout bee phase

If the employed or onlooker bees cannot improve the quality of X_i in limit iterations, X_i may be stagnated. Then, a scout bee re-initializes X_i according to Eq.(1).

3 A brief review of ABC

Although the original ABC has shown good performance, it still has some drawbacks. To tackle these issues, many improved ABCs have been proposed. In this section, a brief review of recent advance on ABC is presented.

Karaboga and Akay [10] compared ABC with some evolutionary algorithms on a number of benchmark problems. Simulation results demonstrate ABC is competitive to those compared algorithms. Zhu and Kwong [32] proposed a new ABC called GABC , which introduced the global best individual into the original solution search equation. Akay and Karaboga [1] designed a new parameter MR to adjust the probability of dimension perturbation. Results demonstrate the modified approach can accelerate the search. Wang et al. [23] combined multi-strategy ensemble learning and ABC. It aims that multiple strategies can effectively balance the global and local search. Inspired by Gaussian DE [21], Zhou et al. [31] introduced Gaussian sampling into ABC to obtain good performance. Cui et al. [4] used a ranking method to choose the parent solutions when generating new solutions. Simulation results show the ranking based ABC is very effective. In [6], Cui et al. developed another version of ABC, which employs a dynamic population mechanism. During the search, the population size is not fixed and dynamically updated. In [3], Chen et al. combined teaching learning based optimization into ABC, and proposed an improved ABC variant to optimize the parameters estimation of photovoltaic.

To improve the exploitation, Xiang et al. [26] used a grey relational model to choose the neighbor individuals. Then, the DE mutation operator is employed to generate new candidate solutions. In [27], Xiang et al. proposed another ABC variant based on the cosine similarity. To select some good neighbor individuals, a novel solution search model is designed. The frequency of parameters perturbation is also modified to share more search information between different solutions. Simulation results on twenty-four benchmark functions show that the proposed Cos-ABC is competitive. Yaghoobi and Esmaeili [29] designed an improved ABC by using three new strategies: chaos theory, multiple searches, and modified perturbation. Song et al. [18] presented an improved ABC based on objective function value information, which employs two modified solution search models. The objective function value is incorporated to adjust the step size. Experiments on thirty test functions show the proposed approach is better than other six ABCs. Li et al. [13] designed a new gene recombination operation to accelerate the convergence of ABC. Some good solutions in the population are chosen to generate offspring through the gene recombination. Results demonstrate the gene recombination operation can effectively strengthen the exploitation capacity of ABC. Kong et al. [12] presented a new ABC variant with two strategies: elite group guidance and combined breadth-depth search. The proposed algorithm was verified on twenty-two benchmark functions. Sulaiman et al. [19] presented a robust ABC to balance exploitation and exploration. Experiments on 27 test functions and economic environmental dispatch (EED) problems show the effectiveness of the approach.

4 ABC with recombination (RABC)

In Section 2, it can be seen that ABC mainly uses Eq.(2) to find new food sources (solutions) during the iterations. From Eq.(2), the new solution V_i is very similar to its parent solution X_i, because they are different on only one dimension. Based this search mechanism, ABC shows powerful exploration ability, but its convergence speed is slow. To improve this case, some researchers introduced some good search experiences into ABC to accelerate the search. In [32], the global best solution is used to improve the search equation. Results show the modification can significantly improve the performance.

PSO is a successful SI optimization algorithm. The original PSO references has more than 100,000 citations. The main advantages of PSO focus on fast convergence speed and strong search ability. In PSO, each individual (also called particle) flies to its previous best (pbest) and the global best (*gbest*) found so far. So, PSO takes full advantage of the search experiences of those best individuals. The PSO updating model is defined by [11].

$$v_{ij}(t+1) = w \cdot v_{ij}(t) + c_1 r_1(pbest_{ij}(t) - x_{ij}(t)) + c_2 r_2(gbest_j(t) - x_{ij}(t)) \tag{4}$$

where V_i and X_i are the velocity and position, respectively; $w\epsilon[0,1]$ is inertia weight; c_1 and c_2 are two learning factors; $r_1, r_2\epsilon[0,1]$ are two random numbers.

In this paper, a new solution updating equation is proposed inspired by PSO. However, this paper is not the first time to introduce PSO model into ABC. In [32], a gbest guided ABC (GABC) was proposed. In [28], Xiang et al. used the gbest and an elitist method to modify the solution search model. Liu [16] used pbest to update the employed bees and gbest to the onlooker bees. In [15], Li et al. defined a modified model as follows.

$$v_{ij}(t) = w \cdot x_{ij}(t) + 2\phi_{ij}(x_{ij}(t) - x_{kj}(t))\Phi_1 + \varphi_{ij}(gbest_j(t) - x_{kj}(t))\Phi_2 \tag{5}$$

where Φ_1 and Φ_2 are two positive values; ϕ_{ij} and φ_{ij} are two random numbers in [0,1].

Differs from these existing models, we design a new one by taking full advantage of *pbest* and *gbest*. The detailed model is defined as follows.

$$v_{ij}(t) = w \cdot x_{ij}(t) + r_1(pbest_{ij}(t) - x_{ij}(t)) + r_2(gbest_j(t) - x_{ij}(t)) \tag{6}$$

where j is a randomly selected dimension index, $r_1, r_2\epsilon[0,1]$ are two random numbers, and the weight factor $w\epsilon[0,1]$.

Based on Eq.(6), we propose an alternative model by combining the original ABC model. The recombined model is described as below.

$$v_{ij}(t) = \begin{cases} x_{ij}(t) + \phi_{ij}(x_{ij}(t) - x_{kj}(t)), & if rand(0,1) < pr \\ w \cdot x_{ij}(t) + r_1(pbest_{ij}(t) - x_{ij}(t)) + r_2(gbest_j(t) - x_{ij}(t)), & Otherwise. \end{cases} \tag{7}$$

where $rand(0,1)\epsilon[0,1]$ is a random value, and $pr\epsilon[0,1]$ is the probability rate.

In our approach RABC, both employed and onlooker bees use Eq. 6 to generate new solutions during iterations. Like the original ABC, a greedy selection method is also utilized to accelerate the search.

In comparison to the original ABC, RABC adds a probability parameter pr to control the usage of *pbest* and *gbest*. When pr is large, bees mainly use the original ABC model to generate new solutions. Then, RABC is similar to ABC. When pr is small, bees mainly employ the modified model to generate new solutions. Then, more search experiences of *pbest* and *gbest* are utilized in the search process. Therefore, pr plays an significant role in balancing exploration and

exploitation in RABC. Moreover, RABC only modifies the search model, and does not employ other operations. So, ABC and RABC have the same computational time complexity.

5 Experimental study

5.1 Test functions

In this paper, ten famous classical benchmark functions are utilized to validate the performance of RABC. These functions were early used in many optimization papers [24]. The dimension D is set to 30 in the experiments. Table 1 briefly describes the ten benchmark problems. All functions should be minimized and their mathematical definitions are listed as below.

f_1: Sphere

$$f_1(x) = \sum_{i=1}^{D} x_i^2$$

f_2: Schwefel 2.22

$$f_2(x) = \sum_{i=1}^{D} |x_i| + \prod_{i=1}^{D} x_i$$

f_3: Schwefel 1.2

$$f_3(x) = \sum_{i=1}^{D} (\sum_{j=1}^{i} x_j)^2$$

f_4: Schwefel 2.21

$$f_4(x) = max_i(|x_i|, 1 \leq i \leq D)$$

f_5: Rosenbrock

$$f_5(x) = \sum_{i=1}^{D-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$$

f_6: Step

$$f_6(x) = \sum_{i=1}^{D} (\lfloor x_i + 0.5 \rfloor)^2$$

f_7: Quartic with noise

$$f_7(x) = \sum_{i=1}^{D} i x_i^4 + rand(0, 1)$$

f_8: Schwefel 2.26

$$f_8(x) = \sum_{i=1}^{D} -x_i \sin(\sqrt{|x_i|})$$

f_9: Rastrigin

$$f_9(x) = \sum_{i=1}^{D} [x_i^2 - 10\cos(2\pi x_i) + 10]$$

f_{10}: Ackley

$$f_{10}(x) = -20 \cdot exp(-0.2 \cdot \sqrt{\frac{1}{D} \sum_{i=1}^{D} x_i^2}) - exp(\frac{1}{D} \sum_{i=1}^{D} cos(2\pi x_i)) + 20 + e$$

Table 1: Search range and global optimum for the benchmark problems.

Functions	Search range	Global optimum
f_1	[-100,100]	0
f_2	[-10,10]	0
f_3	[-100,100]	0
f_4	[-100,100]	0
f_5	[-30,30]	0
f_6	[-100,100]	0
f_7	[-1.28,1.28]	0
f_8	[-500,500]	$-418.98 \cdot D$
f_9	[-5.12,5.12]	0
f_{10}	[-32,32]	0

Table 2: PComputational results of RABC with different pr values.

Functions	$pr = 0.1$	$pr = 0.3$	$pr = 0.5$	$pr = 0.7$	$pr = 0.9$
	Mean	Mean	Mean	Mean	Mean
f_1	**5.94E-75**	6.51E-57	6.75E-48	7.46E-40	4.35E-33
f_2	**8.78E-36**	9.61E-33	1.54E-26	8.44E-20	5.65E-14
f_3	5.87E+03	6.40E+03	6.32E+03	**5.56E+03**	5.83E+03
f_4	**1.47E+00**	3.37E+00	9.89E+00	1.84E+01	3.32E+01
f_5	2.74E+01	2.54E+01	2.31E+01	1.61E+01	**1.43E-01**
f_6	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_7	2.79E-02	**2.25E-02**	3.59E-02	4.57E-02	1.06E-01
f_8	-10925.2	-12545.9	**-12569.5**	**-12569.5**	**-12569.5**
f_9	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_{10}	2.90E-14	3.26E-14	**2.55E-14**	2.89E-14	1.73E-12

Sensitivity analysis of the parameter pr

According to Eq.(7), a new parameter pr is introduced in our approach RABC. As mentioned in Section 4, pr is beneficial for balancing the exploration and exploitation. A large pr is good for exploration and a small pr is helpful for exploitation. So, how to choose a suitable pr is worthy to be investigated.

To analyze the parameter pr, we try to test different pr on the benchmark set. The parameter pr is set to 0.1, 0.3, 0.5, 0.7, and 0.9, respectively. For other parameters N and $limit$, they are set to 50 and 100, respectively. The termination criterion for running an algorithm is maximum number of function evaluations ($MaxFEs$). When the number of function evaluations reaches to $MaxFEs$, the algorithm is stopped. According to the literature [23], $MaxFEs$ is equal to 1.5E+05. For each parameter pr, RABC is run 25 trials.

Table 2 displays the results of RABC with different pr, where "$Mean$" is the mean best function value. As seen, when $pr = 0.1$, RABC outperforms other pr values on two functions f_1 and f_2. For functions $f_3 - f_5$, all pr values cannot help RABC find good solutions. on function f_3, $pr = 0.7$ is slightly better than other pr values, and $pr = 0.1$ is better on function f_4. For function f_5, $pr = 0.9$ can find reasonable solution, while other pr values fail. All pr values and find the global optimum on f_6 and f_9. When $pr > 0.3$, RABC can converge to the global optimum, but RABC with $pr = 0.1$ and 0.3 falls into local minima. For function f_7, RABC with $pr = 0.2$ is better than other pr values. When $pr = 0.5$, RABC achieves better results on function f_10. From the above analysis, RABC with a fixed pr value cannot obtain better results than other pr values. So, it is not easy to select which pr is suitable for the benchmark set.

In order to choose the relatively best pr, Friedman test is used to calculate the mean rank of each pr on the benchmark set. Table 3 shows the mean rank values of RABC with different pr values. As shown, RABC with $pr = 0.1$ achieves the best mean rank. It demonstrates that $pr = 0.1$ is the relatively best choice among five different pr values. In the following experiment, $pr = 0.1$ is used for RABC.

Table 3: Mean rank of RABC with different pr values.

RABC	Mean rank
$pr = 0.1$	**2.70**
$pr = 0.3$	3.00
$pr = 0.5$	2.80
$pr = 0.7$	2.90
$pr = 0.9$	3.60

Comparison of RABC and other well-known ABC algorithms

In this section, we compare RABC with the standard ABC and two other well-known ABCs. The compared algorithms are listed as below.

- ABC [9]
- GABC GABC [32]
- MABC [7]
- Our approach RABC

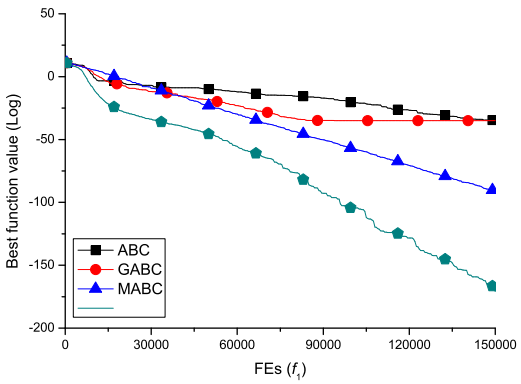
In the following experiments, all algorithms use the same population size and stopping condition. Both N and $limit$ are set to 100. $MaxFEs$ is equal to 1.5E+05. The parameter C is equal to 1.5 in GABC [32]. In MABC, the probability P is set to 0.7 by the suggestions of [7]. The parameter pr in RABC is equal to 0.1 based on experimental study. Each algorithm is run 25 trial on each function.

Table 4 presents the results among RABC, ABC, GABC, and MABC on the ten benchmark functions, where "$Mean$" indicates the mean best function value. From the table, RABC outperforms the standard ABC on 7 functions, while ABC is better than RABC on f_5 and f_8. For the last function f_6, all ABCs can find the global optimum zero. Compared to GABC, RABC achieves better solutions on 5 functions. GABC performs better than RABC on 3 functions. Besides the function f_6, RABC, GABC, MABC can converge to the global minima on f_9. MABC is better than RABC on 2 functions, but RABC outperforms MABC on six functions.

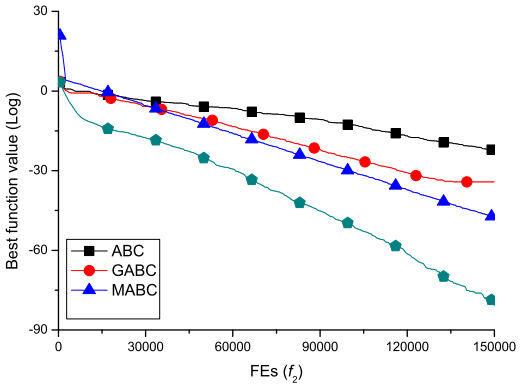
Table 4: Comparison results among RABC, ABC, GABC and MABC.

Functions	ABC	GABC	MABC	RABC
	Mean	Mean	Mean	Mean
f_1	9.67E-16	6.86E-16	2.98E-40	**5.94E-75**
f_2	2.36E-10	1.39E-15	2.13E-21	**8.78E-36**
f_3	9.21E+03	**4.29E+03**	1.01E+04	5.87E+03
f_4	3.73E+01	1.91E+01	5.71E+00	**1.47E+00**
f_5	1.21E+00	6.77E-01	**2.27E-01**	2.74E+01
f_6	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_7	1.68E-01	7.98E-02	4.02E-02	**2.79E-02**
f_8	-12332.6	**-12569.5**	**-12569.5**	-10925.2
f_9	5.33E-14	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_{10}	1.65E-09	3.97E-14	3.26E-14	**2.90E-14**

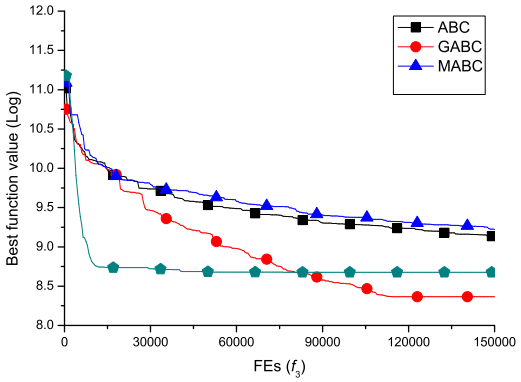
Fig.1 lists the convergence graphs of RABC and three other ABCs on six test functions. As shown, RABC converges faster than MABC, GABC, and ABC on f_1, f_2, f_4, f_9 and f_{10} on the whole search process. For function f_3, RABC is faster than other three algorithms at the beginning of the search. GABC is faster than RABC at the last stage. Because RABC converges to a local minima when FEs reaches to 3.0E+04. For function f_{10}, RABC converges much faster than other three algorithms at the beginning and middle stages. The convergence curves of RABC, MABC, and GABC are similar at the last stage of the search.



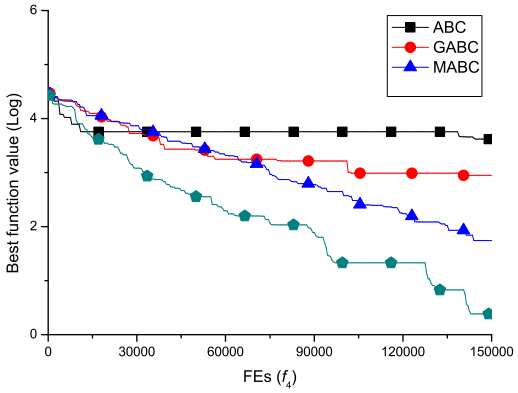
f_1



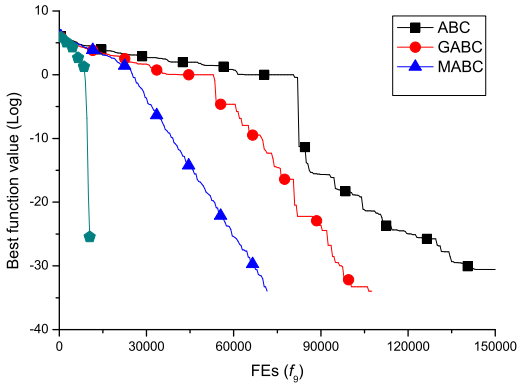
f_2



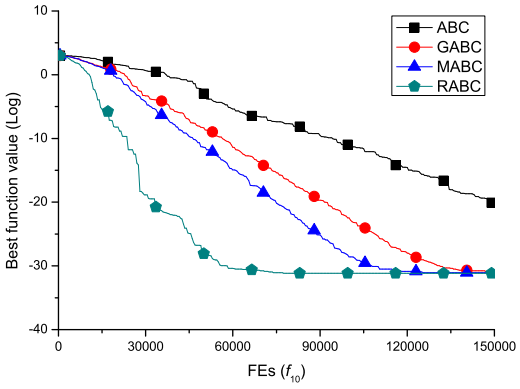
f_3



f_4



f_9



f_{10}

Figure 1: The convergence graphs of four ABCs on six selected functions.

6 Conclusion

In order to strengthen the exploitation ability of ABC, a new ABC variant with recombination (called RABC) is proposed in this paper. RABC firstly employs a new search model inspired by the updating equation of PSO. Then, both the new search model and the original ABC model are recombined to build a hybrid search model. A set of ten famous benchmark optimization problems are tested in the experiments. Results show RABC performs better than ABC, MABC, and GABC on most test functions.

The parameter *pr* aims to control the frequency of using *pbest* and *gbest*. How to choose the best *pr* is studied in the experiments. Results show that RABC with a fixed *pr* value cannot obtain better results than other *pr* values. The statistical test demonstrates that $pr = 0.1$ is the relatively best choice among five different *pr* values. However, a fixed *pr* is not a good choice. A dynamic *pr* may be more suitable. This will be studied in the future work.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Nos. 71701115, 71471106, 61502284, and 71704098), the Natural Science Foundation of Shandong Province (Nos. ZR2017MF058 and ZR2016GQ03), and the Higher School Science and Technology Foundation of Shandong Province (No. J17KA172).

Bibliography

[1] Akay, B.; Karaboga, D. (2012); A modified Artificial bee colony algorithm for real-parameter optimization, *Information Sciences*, 192, 120-142, 2012.

[2] Cai, X.; Wang, H.; Cui, Z.; Cai, J.; Xue, Y.; Wang, L.(2018); Bat algorithm with triangle-flipping strategy for numerical optimization, *International Journal of Machine Learning and Cybernetics*, 9(2), 199-215, 2018.

[3] Chen, X.; Xu, B.; Mei, C.; Ding, Y.; Li, K. (2018); Teaching Clearning Cbased artificial bee colony for solar photovoltaic parameter estimation, *Applied Energy*, 212, 1578-1588, 2018.

[4] Cui, L.; Li, G.; Wang, Z.; Lin, Q.; Chen, J.; Lu, N.; Lu, J. (2017); A ranking-based adaptive artificial bee colony algorithm for global numerical optimization, *Information Sciences*, 417, 169-185, 2017.

[5] Cui, Z.H.; Sun, B.; Wang, G.G.; Xue, Y.; Chen, J.J. (2017); A novel oriented cuckoo search algorithm to improve DV-Hop performance for cyber-physical systems, *Journal of Parallel and Distributed Computing*, 103, 42-52, 2017.

[6] Cui, L.; Li, G.; Zhu, Z.; Lin, Q.; Chen, J. (2017); A novel artificial bee colony algorithm with an adaptive population size for numerical function optimization, *Information Sciences*, 414, 53-67, 2017.

[7] Gao, W.; Liu, S. (2012); A modified artificial bee colony algorithm, *Computers & Operations Research*, 39, 687-697, 2012.

[8] Huang, P.; Lin, F.; Xu, L.J.; Kang, Z.L.; Zhou, J.L.; Yu, J.S. (2017); Improved ACO-bsed seep coverage scheme considering data delivery, *International Journal of Simulation Modelling*, 16(2), 289-301, 2017.

[9] Karaboga, D. (2005); An idea based on honey bee swarm for numerical optimization, *Technical Report-TR06, Erciyes University, engineering Faculty, Computer Engineering Department*, 2005.

[10] Karaboga, D.; Akay, B. (2009); A comparative study of artificial bee colony algorithm, *Applied Mathematics and Computation*, 214, 108-132, 2009.

[11] Kennedy, J.; Eberhart, R. (1995); Particle swarm optimization, *Proceedings of IEEE International Conference on Neural Networks*, 1942-1948, 1995.

[12] Kong, D.; Chang, T.; Dai, W.; Wang, Q.; Sun, H. (2018); An improved artificial bee colony algorithm based on elite group guidance and combined breadth-depth search strategy, *Information Sciences*, 442-443, 54-71, 2018.

[13] Li, G.; Cui, L.; Fu, X.; Wen, Z.; Lua, N.; Lu, J. (2017); Artificial bee colony algorithm with gene recombination for numerical function optimization, *Applied Soft Computing*, 52, 146-159, 2017.

[14] Li, J.; Pan, Q.; Xie, S.; Wang, S. (2011); A Hybrid Artificial Bee Colony Algorithm for Flexible Job Shop Scheduling Problems, *International Jotrnal of Computers Communications & Control*, 6(2), 286-296, 2011.

[15] Li, G.; Niu, P.; Xiao, X. (2012); Development and investigation of efficient artificial bee colony algorithm for numerical function optimization, *Applied Soft Computing*, 12(1), 320-332, 2012.

[16] Liu, J.J.; Zhu, H.Q.; Ma, Q.; Zhang, L.L.; Xu, H.L. (2015); An artificial bee colony algorithm with guide of global & local optima and asynchronous scaling factors for numerical optimization, *Soft Computing*, 37, 608-618, 2015.

[17] Rajput, U.; Kumari, M. (2017); Mobile robot path planning with modified ant colony optimisation, *International Journal of Bio-Inspired Computation*, 9(2), 106-113, 2017.

[18] Song, X.; Yan, Q.; Zhao, M. (2017); An adaptive artificial bee colony algorithm based on objective function value information, *Applied Soft Computing*, 55, 384-401, 2017.

[19] Sulaiman, N.; Mohamad-Saleh, J.; Abro, A.G. (2017); Robust variant of artificial bee colony (JA-ABC4b) algorithm, *International Journal of Bio-Inspired Computation*, 10(2), 99-108, 2017.

[20] Wang, H.;Wang, W.; H. Sun, H.; Rahnamayan, S. (2016); Firefly algorithm with random attraction, *International Journal of Bio-Inspired Computation*, 8(1), 33-41, 2016.

[21] Wang, H.; Rahnamayan, S.; Sun, H.; Omran, M.G.H. (2013); Gaussian bare-bones differential evolution, *IEEE Transactions on Cybernetics*, 43(2), 634-647, 2013.

[22] Wang, H.; Wu, Z.; Rahnamayan, S.; Liu, Y.; Ventresca, M. (2011); Enhancing particle swarm optimization using generalized opposition-based learning, *Information Sciences*, 181(20), 4699-4714, 2011.

[23] Wang, H.; Wu, Z.J.;Rahnamayan, S.; Sun, H.; Liu, Y.; Pan, J.S. (2014); Multi-strategy ensemble artificial bee colony algorithm, *Information Sciences*, 279, 587-603, 2014.

[24] H. Wang; H. Sun; C, Li; S. Rahnamayan; J.S. Pan; Diversity enhanced particle swarm optimization with neighborhood search, *Information Sciences*, 223, 119-135, 2013.

[25] Wu, J.; Wu, G.D.; Wang, J.J. (2017); Flexible job-shop scheduling problem based on hybrid ACO algorithm, *International Journal of Simulation Modelling*, 16(3), 497-505, 2017.

[26] Xiang, W.; Li, Y.; Meng, X.; Zhang, C.; An, M. (2017); A grey artificial bee colony algorithm, *Applied Soft Computing*, 60, 1-17, 2017.

[27] Xiang, W.; Li, Y.; He, R.; Gao, M.; An, M. (2018); A novel artificial bee colony algorithm based on the cosine similarity, *Computers & Industrial Engineering*, 115, 54-68, 2018.

[28] Xiang, Y.; Peng, Y.M.; Zhong, Y.B.; Chen, Z.Y.; Lu, X.W.; Zhong, X.J. (2014); A particle swarm inspired multi-elite artificial bee colony algorithm for real-parameter optimization, *Computational Optimization and Applications*, 57, 493-516, 2014.

[29] Yaghoobi, T.; Esmaeili, E. (2017); An improved artificial bee colony algorithm for global numerical optimisation, *International Journal of Bio-Inspired Computation*, 9(4), 251-258, 2017.

[30] Zhang, M.; Wang, H.; Cui, Z.; Chen, J. (2017); Hybrid Multi-objective cuckoo search with dynamical local search, *Memetic Computing*, doi: 10.1007/s12293-017-0237-2, 2017.

[31] Zhou, X.; Wu, Z.; Wang, H.; Rahnamayan, S. (2016); Gaussian bare-bones artificial bee colony algorithm[J], *Soft Computing*, 20(3), 907-924, 2016.

[32] Zhu, G.; Kwong, S. (2010); Gbest-guided artificial bee colony algorithm for numerical function optimization, *Applied Mathematics and Computation*, 217, 3166-3173, 2010.

Author index

Calle, E., 537

Dai, Y., 465

Ge, M., 521

Ho, N.H., 566
Huang, C., 477

Jeong, G.-M., 566
Jiang, W., 477

Kang, U., 492
Kooij, R.E., 537

Lee, B.M., 492
Lee, Y., 492
Lim, K., 492
Liu, Z., 590

Ma, Y., 590
Mao, Z., 465
Mironescu, I.D., 503

Nguyen, N.D., 566

Prus, P., 550

Rong, H., 521
Rueda, D.F., 537

Schoeneich, R.O., 550

Truong, P.H., 566

Vinţan, L., 503

Wang, H.F., 574
Wang, X., 537
Wu, W., 465

Xie, M., 590

You, X., 590

Zhang, G., 521
Zhang, G.X., 574
Zhou, H., 465
Zhou, K., 574
Zhu, M., 521
Zhu, X., 465