INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 11(1):113-125, February 2016.

A Preflow Approach to Maximum Flows in Parametric Networks Applied to Assessing the Legal Information

L. Sângeorzan, M. Parpalea, M.M. Parpalea,
A. Repanovici, R. Matefi, I. Nicolae

Livia Sângeorzan*
Transilvania University of Brasov
Faculty of Mathematics and Computer Science
25, Eroilor, Blvd, 500030, Brasov, Romania
*Corresponding author:sangeorzan@unitbv.ro

Mircea Parpalea
National College Andrei Şaguna of Brasov
1, Şaguna, street, 590000, Brasov, Romania
parpalea@gmail.com

Mihaela Marinela Parpalea
Transilvania University of Brasov
Faculty of Letters
25, Eroilor, Blvd, 500030, Brasov, Romania
mihaela.parpalea@unitbv.ro

Angela Repanovici
Transilvania University of Brasov
Faculty of Product Design and Environment
25, Eroilor, Blvd, 500030, Brasov, Romania
repa@unitbv.ro

Roxana Matefi, Ioana Nicolae
Transilvania University of Brasov
Faculty of Law
25, Eroilor, Blvd, 500030, Brasov, Romania
roxana.matefi@unitbv.ro, ioana.nicolae@unitbv.ro

Abstract: The present paper proposes a fractal-like approach to the parametric flow problem, derived from the rules and recursion of generative linguistics. In the same manner in which any sentence can be analysed in terms of "theme" of the sentence (that which is spoken about in the sentence) and "rheme" of the sentence (that which is said about the theme in the sentence), in the proposed parametric preflow-push algorithm, a "partitioning push" (a non-cancelling push of flow in the parametric residual network) might leave the node unbalanced for only a subinterval of the range of parameter values. This will lead to separating the problem for the two disjoints subintervals, which allows the algorithm to continue after the same rules, independently, on each of the partitioned subintervals. The algorithm runs as the template-like structure of a dialogue act which reveals a design where information about the items (part-of-speech) is a two sections vector with one segment for each of the used part of speech categories. The article also proposes a possible application of the algorithm in assessing legal information and in semantic evaluation of legislation.
Keywords: Parametric preflow algorithm, partitioning technique, generative linguistics, legal information.

1 Introduction

Computing maximum flows in various networks is an important problem not only because the results are applied directly to traffic analysis of communication networks, but also because some efficient algorithms are often employed as sub-problems in other general network problems. Consequently, fundamental algorithms for computing maximum network flows were designed and efficient algorithms exist [1] to solve different instances [7] of the problem. A natural generalization of this problem can be obtained by considering that the capacities of some arcs are functions of a real parameter. The obtained parametric maximum flow problem is to compute all maximum flows for every possible value of the parameter.

For the maximum flow problem in parametric networks with zero lower bounds and linear capacity functions of a single parameter, Hamacher and Foulds [3] investigated a "horizontal" approach, based on augmenting directed paths algorithm. In each of its iterations, the algorithm computes an improvement of the flow defined on the whole interval of the parameter. For the same problem, Ruhe [8,9] proposed a "piece-by-piece", "vertical" approach. The main idea of his algorithm is to consider that the parametric maximum flow is known up to a parameter value λ_k and to compute a flow augmentation that assures the optimality of the flow for parameter values $\lambda \geq \lambda_k$ as well as the parameter maximum value λ_{k+1} up to which the computed flow remains a parametric maximum one. Each of the algorithm's iterations represents a non-parametric maximum flow sub-problem which can be solved via one of the classic approaches for the maximum flow problem.

Partitioning technique in networks has been, in the latest years, a more and more active research topic in both engineering and theoretical research. The reason why the problem under consideration is of genuine practical and theoretical interest lies in that graph partitioning applications are described on a wide variety of subjects, such as: data distribution in parallel-computing, VLSI circuit design, image processing, computer vision, route planning, air traffic control, mobile networks, social networks, etc. [2]. Unfortunately, graph partitioning is an NP-hard problem, and therefore all known algorithms for generating partitions merely return approximations to the optimal solution. Partitioning algorithms for the parametric maximum flow problem can be developed starting from any of the three classic approaches for the maximum flow problem: flow augmenting directed paths algorithms, preflow algorithms or min-max algorithm. The approach based on flow augmenting directed paths [4], makes use of the concept of shortest conditional augmenting directed path. In order to avoid working with piecewise linear functions, the approach uses a series of parametric residual networks defined for successive subintervals of the parameter values where the parametric residual capacities of all arcs remain linear functions. The approach based on preflow algorithms will further be presented in detail in this paper while the parametric min-max approach [6] is not itself an algorithm but rather a method of obtaining a parametric minimum flow using any algorithm for obtaining a maximum parametric flow and vice versa.

The idea of the partitioning approach based on preflows, which is used in this paper, derives from the rules and recursion of generative linguistics [10]. The concept which has proved to be the most useful in the description of German word order has become known under the name of Functional Sentence Perspective (FSP). Its main idea is that information is not transmitted in random order but the speaker seeks to give his information to his interlocutor in portions, normally starting from what he assumes is common to both and proceeding to what he regards as important new information [5]. In the same manner in which any sentence can be analysed in terms of "theme" of the sentence (that which is spoken about in the sentence) and "rheme" of the sentence (that which is said about the theme in the sentence), the proposed algorithm for the parametric maximum flow problem uses a fractal-like approach. In the proposed parametric

preflow-push algorithm, a 'partitioning push' (a non-cancelling push of flow in the parametric residual network) might leave the node unbalanced for only a subinterval of the range of the values of the parameter. Like in all fractal approaches a partitioning push is followed by separating the problem into disjoints subintervals, allowing the algorithm to continue after the same rules independently on each of the partitioned subintervals. Making use of the partitioning approach, the present paper proposes an original algorithm for computing the maximum flow in parametric networks with linear upper bound (capacity) functions. Although both the algorithm presented in [10] and the one that is proposed in this article deals with the same kind of approach of the parametric flow, they are totally different algorithms and they fundamentally differ because of their purpose: the first computes the minimum flow in networks with constant capacities and linear lower bound functions while the latter computes the maximum flow in networks without lower bounds and with linear capacity functions.

Further on, this paper is organized as follows. Section 2 presents the parametric maximum flow problem and the basic parametric network flow terminology and results which are used in the rest of the paper. Most of the definitions in this section are straightforward modifications of those in [4] and [6], adapted for the maximum flow problem in parametric networks with zero lower bounds. More specialized terminology and further details on the notions, definitions and main results within this section can be found in [1], [4] and [6]. In Section 3 we present our algorithm for solving the parametric maximum flow problem, called "Highest label partitioning push (HLPP) algorithm". In the same section there are also presented the corresponding theorems of correctness and of complexity of the algorithm. Using a multithread based parallel implementation, the complexity of the algorithm is linearly dependent on the number of breakpoints. Section 4 deals with a possible application of the proposed algorithm to assessing the legal information. Finally, Section 4 presents some conclusions regarding to our contribution to the topic presented in the article. In the presentation to follow, some familiarity with flow algorithms is assumed and many details are omitted, since they are straightforward modifications of known results.

2 Flows in parametric networks

The parametric maximum flow problem is an extension of the classical maximum flow problem [1] in which the capacities of certain arcs are functions of a parameter λ. The parametric maximum flow problem is to compute all maximum flows for every possible value of the parameter. For the case of linear capacity functions, the maximum flow value function in a parametric network is a continuous piecewise linear function of the parameter. Each linear segment of the maximum flow value function between the two breakpoints λ_k and λ_{k+1} corresponds to a cut that remains a minimum cut for any $\lambda_k < \lambda \leq \lambda_{k+1}$. The approach presented in this article refers to the maximum flow problem in a network with linear capacity functions of a single parameter.

2.1 Terminology and preliminaries

Let $G = (N, A, u, s, t)$ be a capacitated network with $n = |N|$ and $m = |A|$, $N = \{\dots, i, \dots\}$ being the set of nodes i and $A = \{\dots, a, \dots\}$ being the set of arcs a, so that for every arc in A holds that $a = (i, j)$ with $i, j \in N$. The capacity (*upper bound*) function is a nonnegative function $u(a)$ associated with each arc $a \in A$. The network has two special nodes: a source node s and a sink node t. A *flow* is a function $f : A \to \mathbb{R}^+$ satisfying the next conditions:

$$\sum_{j|(i,j)\in A} f(i,j) - \sum_{j|(j,i)\in A} f(j,i) = \begin{cases} v, & i = s \\ 0, & i \neq s, t \\ -v, & i = t \end{cases} \tag{1}$$

for some $v \geq 0$, where v is referred to as the value of the flow f. Any flow on a directed network satisfying the flow bound constraints:

$$0 \leq f(i,j) \leq u(i,j), \qquad \forall (i,j) \in A \tag{2}$$

for every arc $(i,j) \in A$ is referred to as a *feasible flow*. A *cut* is a partition of the node set N into two subsets S and $T = N - S$, denoted by $[S,T]$. A cut is nontrivial if both S and T are nonempty. An arc $(i,j) \in A$ with $i \in S$ and $j \in T$ is referred to as a *forward arc* of the cut while an arc $(i,j) \in A$ with $i \in T$ and $j \in S$ as a *backward arc* of the cut. Let (S,T) denote the set of forward arcs in the cut and let (T,S) denote the set of backward arcs. A cut $[S,T]$ is an $s-t$ cut if $s \in S$ and $t \in T$. The *maximum flow problem* is to determine a flow $\tilde{f}$ for which v is maximized

2.2 The parametric maximum flow

The parametric flow problem consists in generalising the classic problem of flows in networks by transforming the upper bounds of some arcs $(i,j) \in A$ of the network $G = (N, A, u, s, t)$ in linear functions of a real parameter.

Definition 1. *[4]* A directed network $G = (N, A, u, s, t)$ for which the upper bounds u of some arcs $(i,j) \in A$ are functions of a real parameter λ is referred to as a *parametric network* and is denoted by $\bar{G} = (N, A, \bar{u}, s, t)$.

For a parametric network $\bar{G}$, the *parametric upper bound* (*capacity*) function $\bar{u} : A \times [0, \Lambda] \to \mathbb{R}^+$ associates to each arc $(i,j) \in A$, for each of the parameter values $\lambda \in [0, \Lambda]$, the real number $\bar{u}(i,j;\lambda)$, referred to as the parametric upper bound of arc (i,j):

$$\bar{u}(i,j;\lambda) = u_0(i,j) + \lambda \cdot U(i,j), \qquad (i,j) \in A, \tag{3}$$

where $U : A \to \mathbb{R}$ is a real valued function associating to each arc $(i,j) \in A$ the real number $U(i,j)$ referred to as the *parametric part of the upper bound* of the arc (i,j). The nonnegative value $u_0(i,j)$ is the upper bound of the arc (i,j) for $\lambda = 0$, i.e. : $\bar{u}(i,j;0) = u_0(i,j)$ with $0 \leq u_0(i,j)$. For the problem to be correctly formulated, the upper bound function of every arc $(i,j) \in A$ must respect the condition $0 \leq \bar{u}(i,j;\lambda)$ for the entire interval of the parameter values, i.e. $\forall (i,j) \in A$ and $\forall \lambda \in [0, \Lambda]$. It follows that the parametric part of the upper bounds $U(i,j)$ must satisfy the constraint: $U(i,j) \geq -u_0(i,j)/\Lambda$, $\forall (i,j) \in A$. The *parametric flow value function* $\bar{v} : N \times [0, \Lambda] \to \mathbb{R}$ associates to each of the nodes $i \in N$ a real number $\bar{v}(i;\lambda)$ referred to as the value of node i for each of the parameter λ values.

Definition 2. *[4]* A feasible flow in the parametric network $\bar{G} = (N, A, \bar{u}, s, t)$ is called a *parametric flow*, $\bar{f} : A \times [0, \Lambda] \to \mathbb{R}^+$ satisfying the following constraints:

$$\sum_{j|(i,j)\in A} \bar{f}(i,j;\lambda) - \sum_{j|(j,i)\in A} \bar{f}(j,i;\lambda) = \bar{v}(i;\lambda), \qquad \forall i \in N,\ \forall \lambda \in [0, \Lambda], \tag{4}$$

$$0 \leq \bar{f}(i,j;\lambda) \leq \bar{u}(i,j;\lambda), \qquad \forall (i,j) \in A,\ \forall \lambda \in [0, \Lambda], \tag{5}$$

where $\sum_{i \in N} \bar{v}(i;\lambda) = 0, \qquad \forall \lambda \in [0, \Lambda]$.

The parametric maximum flow (PMF) problem is to compute all maximum flows for every possible value of λ in $[0, \Lambda]$:

$$maximize \quad \bar{v}(\lambda) \quad for\ all \quad \lambda \in [0, \Lambda], \tag{6}$$

$$\sum_{j|(i,j)\in A} \bar{f}(i,j;\lambda) - \sum_{j|(j,i)\in A} \bar{f}(j,i;\lambda) = \begin{cases} \bar{v}(\lambda), & i = s \\ 0, & i \neq s,t \\ -\bar{v}(\lambda), & i = t \end{cases} \tag{7}$$

$$0 \leq \bar{f}(i,j;\lambda) \leq \bar{u}(i,j;\lambda), \qquad \forall (i,j) \in A. \tag{8}$$

This problem looks like a classic maximum flow problem with the decisive difference that the variables $\bar{f}(i,j;\lambda)$ of this problem are piecewise linear functions instead of real numbers and that the upper bounds $\bar{u}(i,j;\lambda)$ are linear functions instead of constants.

Definition 3. *[6]* A *parametric $s-t$ cut partitioning* denoted by $[S_k; J_k]$, $k = 0, \ldots, K$, is defined as a finite set of cuts $[S_k, T_k]$, $k = 0, \ldots, K$, together with a partitioning of the interval $[0, \Lambda]$ of the parameter in disjoints subintervals J_k, $k = 0, \ldots, K$, so that $J_0 \cup \cdots \cup J_K = [0, \Lambda]$.

Definition 4. *[4]* For the parametric maximum flow problem, the *capacity $\tilde{\bar{c}}[S_k; J_k]$ of a parametric $s-t$ cut partitioning* is a linear function on every subinterval J_k, $k = 0, \ldots, K$, defined as:

$$\tilde{\bar{c}}[S_k; J_k] = \sum_{(i,j)\in(S_k,T_k)} \bar{u}(i,j;\lambda), \quad k = 0, \ldots, K. \tag{9}$$

Definition 5. *[4]* A parametric $s-t$ cut partitioning $[S_k; J_k]$ with the subintervals J_k assuring that every cut is a minimum cut $[\tilde{S}_k, \tilde{T}_k]$ within the subinterval $[\lambda_k, \lambda_{k+1}]$ is referred to as a *parametric minimum $s-t$ cut* and is denoted by $[\tilde{S}_k; J_k]$, $k = 0, \ldots, K$.

Theorem 6. (Parametric Max-Flow Min-Cut Theorem [6]) *If there is a feasible flow in the parametric network $\bar{G}$, the value function $\tilde{\bar{v}}$ of the parametric maximum flow $\tilde{\bar{f}}$ from a source s to a sink t equals the capacity $\tilde{\bar{c}}$ of the parametric minimum $s-t$ cut $[\tilde{S}_k; J_k]$, $k = 0, \ldots, K$.*

Let $\bar{f} = (\ldots \bar{f}(i,j;\lambda), \ldots)_{(i,j)\in A}$ be a vector of feasible flow functions defined on the interval $[0, \Lambda]$. Supposing that an arc $(i,j) \in A$ carries a flow $\bar{f}(i,j;\lambda)$, the existing flow can be increased either by sending the additional flow (*pushing*) $\bar{u}(i,j;\lambda) - \bar{f}(i,j;\lambda)$ from node i to node j over the arc (i,j) or by cancelling the flow $\bar{f}(j,i;\lambda)$ from node j to node i over the arc (j,i) which is equivalent to *pulling* the flow from node i to node j over the arc (j,i). These flows are computed as differences between piecewise linear functions of λ.

Definition 7. *[4]* For the parametric maximum flow problem, the *parametric residual capacity* $\tilde{\bar{r}}(i,j;\lambda)$ of any of the arcs $(i,j) \in A$, with respect to a given parametric flow $\bar{f}$, represents the maximum additional flow that can be sent from node i to node j over the arcs (i,j) and (j,i) and is given by:

$$\tilde{\bar{r}}(i,j;\lambda) = \bar{u}(i,j;\lambda) - \bar{f}(i,j;\lambda) + \bar{f}(j,i;\lambda). \tag{10}$$

Definition 8. *[4]* The subintervals $\tilde{I}(i,j) \subseteq [0, \Lambda]$ where an augmentation of the flow $\bar{f}(i,j;\lambda)$ is possible along the arc (i,j) are defined as follows:

$$\tilde{I}(i,j) = \{\lambda | \tilde{\bar{r}}(i,j;\lambda) > 0\}, \quad (i,j) \in A. \tag{11}$$

Definition 9. *[4]* Given a feasible flow $\bar{f}$ in the parametric network $\bar{G}$, the network denoted by $\tilde{\bar{G}}(\bar{f}) = (N, \tilde{A}(\bar{f}))$, with $\tilde{A}(\bar{f}) = \{(i,j) | (i,j) \in A,\ \tilde{I}(i,j) \neq \emptyset\}$ being the set consisting only of arcs with positive parametric residual capacities, is referred to as the *parametric residual network* with respect to the given flow $\bar{f}$ for the parametric maximum flow problem.

If an arc $(i,j) \in A$ does not belong to $\tilde{\bar{G}}(\bar{f})$ then $\tilde{I}(i,j) := \emptyset$ is set.

Definition 10. The parametric excess of a node $i \in N$ is defined as:

$$\tilde{e}(i;\lambda) = \sum_{j|(j,i)\in A} \bar{f}(j,i;\lambda) - \sum_{j|(i,j)\in A} \bar{f}(i,j;\lambda). \tag{12}$$

Definition 11. *[4]* The subintervals $\tilde{I}(i) \subseteq [0,\Lambda]$ where the excess of node i is positive are defined as follows:

$$\tilde{I}(i) = \{\lambda | \tilde{e}(i;\lambda) > 0\}, \quad i \in N - \{s,t\}. \tag{13}$$

In the residual network $\tilde{\bar{G}}(\bar{f})$ the *distance function* $\tilde{d} : N \to \aleph$ is a function from the set of nodes to the nonnegative integers. A distance function is said to be *valid* if it satisfies the following conditions: $\tilde{d}(t) = 0$ and $\tilde{d}(i) \leq \tilde{d}(j) + 1$, $\forall (i,j) \in \tilde{A}$.

Definition 12. *[4]* An arc $(i,j) \in \tilde{A}$ in the parametric residual network $\tilde{\bar{G}}(\bar{f})$ is referred to as *conditionally admissible* if both $\tilde{d}(i) = \tilde{d}(j) + 1$ and $\tilde{I}(i,j) \cap \tilde{I}(i) \neq \emptyset$; otherwise it is *conditionally inadmissible.*

3 Partitioning push algorithm

3.1 Highest-label partitioning-push algorithm

The *Highest-label partitioning-push* (HLPP) algorithm maintains a set L of active nodes organised as a priority queue. In the initialisation step of the algorithm, all the nodes $i \in N$ with $(s,i) \in \tilde{A}$ will gain a positive excess, becoming thus active nodes, by setting the flow to the upper bound value $\bar{f}(s,i;\lambda) := \bar{u}(s,i;\lambda)$ for every arc (s,i). Consequently, they are added to the priority queue L and then removed one by one, in the descending order of their priorities $\tilde{d}(i)$. For an active node $i \in \tilde{\bar{G}}(\bar{f})$, if there exists an conditionally admissible arc (i,j), the flow will be pushed over this arc and node j will be added to the priority queue L with the priority $\tilde{d}(j)$; otherwise node i will be relabelled so that at least one conditionally admissible arc to be created and node i is added to L with its new priority $\tilde{d}(i)$. The algorithm terminates when the queue of active nodes is empty. A push of flow from node i to node j is referred to as a *cancelling push* if it deletes the arc (i,j) from the residual network; otherwise it is a *non-cancelling push.*

Algorithm 1 Highest-label partitioning-push (HLPP) algorithm

```
1: procedure HLPP
2:     J ← [0, Λ]   L ← ∅   f̄ ← 0   compute d̃(·) in G̃(f̄)
3:     for all (s, i) ∈ Ã do
4:         f̄(s, i; λ) ← ū(s, i; λ)   ẽ(i; λ) ← ū(s, i; λ)
5:         if ẽ(i; λ) > 0 and i ≠ t then
6:             add i with priority d̃(i) to L
7:         end if
8:     end for
9:     d̃(s) ← n
10:    PP(L J)
11: end procedure
```

For any node $i \in \tilde{\bar{G}}(\bar{f})$, the expressions: *active node* and *balanced node* holds only for subintervals of the parameter values. While both the parametric residual capacity $\tilde{\bar{r}}(i,j;\lambda)$ of any

Algorithm 2 Partitioning push (PP) procedure

12: **procedure** PP(L J_p)
13: **if** $L \neq \emptyset$ *and* $J_p \neq \emptyset$ **then**
14: remove the first node i from L
15: **if** $\nexists$ an admissible arc (i,j) **then**
16: $\tilde{d}(i) \leftarrow min\{\tilde{d}(j)|(i,j) \in \tilde{A}\} + 1$
17: add i with priority $\tilde{d}(i)$ to L
18: PP(L J_p)
19: **else**
20: select an admissible arc (i,j)
21: push $\tilde{g}(i,j;\lambda) \leftarrow min\{\tilde{e}(i;\lambda), \tilde{\bar{r}}(i,j;\lambda)\}$ over (i,j)
22: **if** $j \notin L$ *and* $j \neq s$ *and* $j \neq t$ **then** add j to L
23: **end if**
24: $J_{p_1} \leftarrow \{\lambda|\tilde{e}(i;\lambda) \leq \tilde{\bar{r}}(i,j;\lambda)\}$
25: $J_{p_2} \leftarrow J_p - J_{p_1}$
26: $L_{p_1} \leftarrow L \quad L_{p_2} \leftarrow L$
27: add i with priority $\tilde{d}(i)$ to L_{p_2}
28: **do** in parallel
29: PP($L_{p_1} J_{p_1}$)
30: PP($L_{p_2} J_{p_2}$)
31: **end do**
32: **end if**
33: **end if**
34: **end procedure**

arc $(i,j) \in \tilde{A}$ and the parametric excess $\tilde{e}(i;\lambda)$ of any node $i \in N - \{s,t\}$ are linear functions of the parameter values, cancelling or non-cancelling pushes are defined only for certain subintervals of the parameter values.

A non-cancelling push of flow from a node $i \in N - \{s,t\}$ along an arc $(i,j) \in \tilde{A}$ in a subinterval $J_p = (\lambda_p, \lambda_{p+1}] \subseteq [0, \Lambda]$ which leaves the node i unbalanced is referred to as a *partitioning push.* Whenever the algorithm performs a partitioning push in $\tilde{\bar{G}}_p(\bar{f})$, i.e. the network $\tilde{\bar{G}}(\bar{f})$ defined for the subinterval J_p, a new partitioning of the interval J_p in the two subintervals J_{p1} and J_{p2}, with $J_{p1} \cup J_{p2} = J_p$ and $J_{p1} \cap J_{p2} = \emptyset$ will take place. Let J_{p1} be the subinterval within which the partitioning push balances the node i, i.e. $J_{p1} = \{\lambda|\tilde{e}(i;\lambda) \leq \tilde{\bar{r}}(i,j;\lambda)\}$. If $J_{p2} \neq \emptyset$ then, as on every subinterval J_p both $\tilde{\bar{r}}(i,j;\lambda)$ and $\tilde{e}(i;\lambda)$ are linear functions of λ, the partitioning push generates two parametric residual networks: $\tilde{\bar{G}}_{p1}(\bar{f})$ for $\lambda \in J_{p1}$ and $\tilde{\bar{G}}_{p2}(\bar{f})$ for $\lambda \in J_{p2}$, so that node i is balanced in $\tilde{\bar{G}}_{p1}(\bar{f})$ and active in $\tilde{\bar{G}}_{p2}(\bar{f})$ while arc (i,j) will not belong to $\tilde{\bar{G}}_{p2}(\bar{f})$, since $\tilde{\bar{r}}(i,j;\lambda) = 0$ after the partitioning push. The algorithm will then continue separately in each of the parametric residual networks and for each of the two subintervals. Under these observations, the push/relabel procedure from the non-parametric *Highest label* preflow algorithm is replaced with a recursive call of a *partitioning push*(L,J) procedure.

Theorem 13. (Theorem of correctness) *Highest-label partitioning-push algorithm computes correctly a maximum flow in the parametric network* $\bar{G} = (N, A, \bar{u}, s, t)$.

Proof: The proof of the theorem follows from the correctness of the general HL preflow algorithm for each of the subintervals of the parameter values. When the algorithm terminates, let S_p be the set of all nodes which are reachable from the source node within the subinterval J_p. For the resulting cuts $[S_p, T_p]$ and the intervals $J_p = (\lambda_p, \lambda_{p+1}]$, $p = 1, \ldots, K$, the following observations

hold:
(i) If $i \in S_p$, $j \in T_p$ and $(i,j) \in A$ then $J_p \cap \tilde{I}(i,j) = \emptyset$ for otherwise node j could be reached from s in $\tilde{\bar{G}}(\bar{f})$. Hence, by the definition of $\tilde{I}(i,j)$, $\tilde{\bar{f}}(i,j;\lambda) = \bar{u}(i,j;\lambda)$, $\forall \lambda \in J_p$;
(ii) If $i \in T_p$, $j \in S_p$ and $(i,j) \in A$ then $J_p \cap \tilde{I}(j,i) = \emptyset$ for otherwise node i could be reached from s in $\tilde{\bar{G}}(\bar{f})$. Hence, $\tilde{\bar{f}}(i,j;\lambda) = 0$, $\forall \lambda \in J_p$.
Summarizing, $\sum_{(i,j)\in(S_p,T_p)} \tilde{\bar{f}}(i,j;\lambda) = \sum_{(i,j)\in(S_p,T_p)} \bar{u}(i,j;\lambda)$ and $\sum_{(i,j)\in(T_p,S_p)} \tilde{\bar{f}}(i,j;\lambda) = 0$, thus the obtained flow is a maximum parametric flow which equals the capacity of the minimum $s-t$ parametric cut. □

3.2 Complexity issues

A breakpoint is a place where the slope of the piecewise linear maximum flow value function is changing. In the worst case the number of breakpoints may be exponential in the size of the problem. The example originates from the pathological graph of Zadeh [8]. The Highest-label partitioning-push algorithm overcomes this inconvenient by using the multi-thread parallel implementation of a non-parametric algorithm [12]. The main idea of this implementation is to assign a processor to each newly generated subinterval Jp which will carry out the problem forward from the current configuration of the problem. For each of the newly generated subintervals, a copy of the current distance labels values is generated so that they can be independently modified in the further parallel evolution of the algorithm.

Theorem 14. (Theorem of complexity) *The parallel implementation of the Highest-label partitioning-push algorithm solves the parametric maximum flow problem in* $O(n^2m^{1/2} + Kn)$ *time.*

Proof: The complexity of the non-parametric HL preflow algorithm [1] is $O(n^2m^{1/2})$. The HL partitioning-push algorithm generates new copies of distance label values (one for each of the two new threads which will further run in parallel) every time a breakpoint occurs, i.e. copying distance labels takes $O(Kn)$ time where K is the number of breakpoints. Thus, the total complexity of the algorithm is $O(n^2m^{1/2} + Kn)$. □

4 Application to assessing the legal information

To write an appropriate, objective and specific rule for every imaginable situation is an impossible thing to do. "Given the numberless potential variations, foreseeable and unforeseeable, in motives and circumstances, there can, probably, be no end to the possible specific scenarios - and thus no limit on the number of rules that would result from trying to write an appropriate one for each possible, distinct fact situation." [11] The solution of many complex decision problems, such as assessing legal information, involves combinatorial optimization, i.e. obtaining the optimal solution among a finite set of alternatives. Such optimization problems are notoriously difficult to solve. One of the primary reasons is that in most applications the number of alternatives is extremely large and only a fraction of them can be considered within a reasonable amount of time. As a result, heuristic algorithms, such as evolutionary algorithms are often applied in combinatorial optimization but their major problem consists in their high complexity.
In our opinion, the proposed algorithm can successfully be used in assessing legal information, in their semantic evaluation or in generating files or legal information recording models and subsequently transmitting them to the members of the information society.

4.1 Concepts association network

Regarding the law as a mathematical phenomenon, the legislation represents a logical structure. Any law can be considered as a collection of fractal scenarios, made up of legal stipulations and exceptions. In order to enhance assessing features, large collections of legal information must be organized in hierarchical structures based on key concepts. Generally, there are two approaches to hierarchical structures generation: the query-independent approach, offering a consistent view of the whole corpus, and the query-dependent approach which allows concepts to be organized differently depending on the query. The approach presented in our paper is a query-dependent one, based on statistical co-occurrence in determining the relationship between concepts. The algorithm builds a hierarchical structure where, for a certain topic, each of the main concepts is related to distinct sub-concepts with different degrees of association values, described by the parameter values.
The first stage in generating a concept hierarchy consists in extracting a set of main concepts, related to the topic, and computing the degree of co-occurrences in relation with other sub-concepts. Once the set of main concepts is extracted and their degrees of co-occurrences are calculated, the concepts association network, covering all concepts, is constructed. The network is built as a directed graph with weighted arcs, where each of the concepts represents a node and the source node represents the topic of the query-dependent structure. The directed graph contains an arc between two concepts only if those concepts co-occur in at least one legal stipulation.

Definition 15. The *degree of co-occurrence* is computed as the number of legal articles (stipulation) that contain (refer to) both concepts.

Definition 16. The *strength* of a concept, representing that concept's importance, is the sum of all its co-occurrences.

The capacities of the arcs (s, i) are set to $u(s, i) := strength(i)$ while all the arcs (j, t) have no upper value limit.

Definition 17. The parametric upper bound $\bar{u}(i, j; \lambda)$ of an arc (i, j) is computed as:

$$\bar{u}(i, j; \lambda) = \frac{strength(i)}{externaldegree(i)} + \lambda \cdot co - occurrence(i, j). \tag{14}$$

Finally, based on the computed parametric maximum flow in the concepts association network, the sets of concepts defined by the parametric cut partitioning group the concepts in classes which are ordered in hierarchies. As long as the sink node is reached in the concepts association network, any of the directed paths starting from the source node represents a legal demarch. In the case that for various reasons the sink node can not be reached, the meaning of this fact is that some legal provisions are contradictory, inconsistent or ambiguous.

4.2 Theoretical legal aspects

Let us analyze, for example, the general provision of the Romanian Constitution, namely Article 21, first alignment "any person can appeal to the court for protecting its legitimate interests and liberties" and the second alignment "no law can restrict the exercise of these rights". Among the main concepts with which the lawmaker operates, we can name: the free access to justice, rights, liberties, legitimate interests. These concepts are in close connection to other sub concepts which are not currently found in laws. Thus, according to Article 192 of the Civil Procedure Code, "in order to protect one's legitimate interests and rights, any person can

appeal to justice by approaching the competent court of law".
However, there are some exceptions to the general rule stated above, namely Article 193, the first alignment of the Civil Procedure Code, according to which "approaching the court is an action which can occur only after a preliminary procedure is completed if the law is clear on that matter. The proof of completing this procedure will be attached to the summon", as well as the second alignment of the same law, according to which "failure to complete the preliminary procedure can only be invoked by the defendant in his response". To complete these provisions, we have the first alignment of Article 7 of Law no 554/2004 regarding the administrative procedures which state the following "before addressing the competent legal court, the person who claims that it has suffered an injustice through an unilateral administrative act must first ask the public authority, within 30 days, to revoke that certain document or some parts of it (...)".

4.3 Legal case study

First Court solution

To demonstrate the theoretical aspects discussed above, we will use the following example: By their petition addressed to the Galati County Appeal Court, the plaintiffs R.I., B.S., G.M., S.C., G.V., S.G., R.V., S.C., A.C.E., G.C., S.A., B.I., E.A., G.G., G.G versus the Romanian Government, The Galati County Pension Institution and the Romanian Council for Fighting Discrimination have asked the annulment of the Government's Decision no 737/2010 regarding the method used for recalculation the amount of retirement money for the categories stated in Article 1, letters c to h of Law no 119/2010 regarding the establishing of some measures for maintaining the amount of retirement money, published in the Official Bulletin no 528 of June 29th, 2010, which the plaintiff was receiving at the date the document whose annulment is asked came into force. By the same petition, they asked, according to the provisions of Article 15 of Law no 554/2001 the suspension of the Government's Decision no 737/2010 until this matter is resolved. By its response, the Romanian Government claimed the petition was inadmissible as it did not complete the preliminary procedure, according to Article 7 of Law no 554/2004. It further asked for the petition to be dismissed as it is not insubstantial. The Romanian Ministry for Work, Family and Social Protection intervened on behalf of the defendant - the Romanian Government. By its decision no 52 of February 22nd, 2011, the Galati County Appeal Court, dismissed the plaintiffs petition as inadmissible and allowed the Ministry to intervene on behalf of the Romanian Government.
In order to reach this decision, the court stated that the provisions of Article 7, the first alignment claim that the legal proceedings can only occur after the administrative authorities are given the opportunity to revoke their document or amend it. In the matter at hand, although it was claimed that the Government's Decision no 737/2001 regarding the method used for recalculating the amount of retirement money for the categories stated in Article 1, letters c to h of Law no 119/2010 regarding the establishing of some measures for maintaining the amount of retirement money, should be annulled, there was no proof that a preliminary proceeding existed, as this was a legal condition for the admissibility of the petition, according to Article 109, second alignment of the Civil Procedure Code.

Appeal Court solution

The plaintiffs appealed this decision, by criticizing it for being illegal and unfounded and claiming that, in the matter at hand, the preliminary procedure can be completed at any time and this condition was met. The High Court of Justice, by analyzing the works and documents of this case, ruled that the first court was correct to dismiss the plaintiff's petition as inadmissible,

as it does not respect the provisions of the first alignment of Article 7 of Law no 554/2004. According to these provisions "before addressing the court, the person who claims one of its rights was disrespected must first ask the public authority who created the document, within 30 days of that date, to revoked that document. The petition can also be addressed to the superior hierarchical organ, if such an organ exists". Respecting the terms and conditions stated by law for the preliminary procedure is a special demand of the law and the disrespecting of this demand causes the plaintiffs inability to appeal the courts. In the matter at hand, the plaintiffs asked for the annulment of Government's Decision no 737/2010 without proving they have completed the preliminary proceedings. This is similar to not respecting the legal obligation enforced by the first alignment of Article 7 of Law no 554/2004. Indeed, according to the provisions of Article 7, the first alignment of Law no 554/2004, "in case of an administrative act, the preliminary procedure can be completed at any time", as the Government's Decision no 737/2010 is considered to be an administrative act with power of the law.

However, these legal provisions must be interpreted in close connection with those of Article 11 of the Administrative Proceedings Law which describes the term to formulate such a demand, as well as with the provisions of Article 7, the first alignment of the same law. The phrase "at any time" allows for the possibility of formulating such a petition and for completing the preliminary proceedings without respecting the term stated in Article 11 of Law no 554/2010, except for "ordinances of their dispositions which are considered to be unconstitutional, as well as administrative documents with power of the law which are considered to be illegal", as this course of action is regulated by the fourth alignment of this Article.

The preliminary procedure regulated by Article 7, alignments (1) and (11) of Law no 554/2004, is a condition for the admissibility of the petition, according to Article 109, second alignment of the Civil Procedure Code, as it is previous to appealing the court. Considering all aspects mentioned above and seeing that there are no reasons to annul the first court's decision, the High Court will dismiss the appeal as unfounded [13].

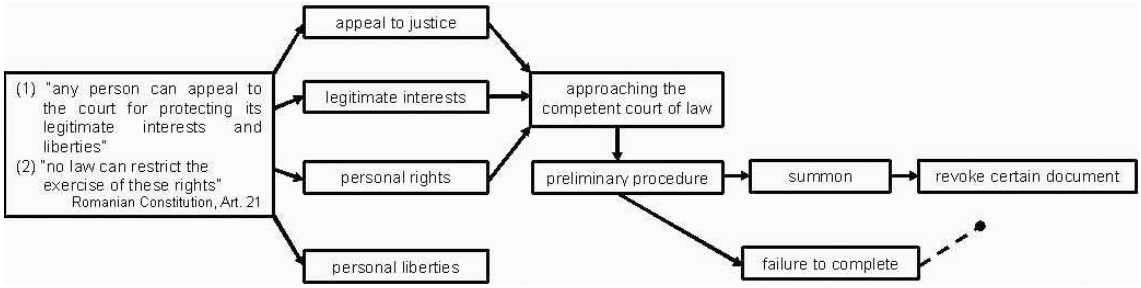

Figure 1: Concepts association network for the presented legal example.

As can be easily seen in the concepts association network presented above in Fig.1, by following the directed path which does not include the preliminary procedure attached to a summon, the sink node can never be reached, revealing the fact that some legal provisions are contradictory, inconsistent or ambiguous.

Further developing the above presented model, based on an appropriate corpus (database) of legal stipulations which is generated by our algorithm in the way it has previously been presented above already, a computer application can be developed so that to any online transmitted legal question, a solicitor could receive the adequate legal information.

5 Conclusions

The maximum flow problem in parametric networks turns out to be an important scenario in practice since the complexity of its solving algorithm was reduced to a linear dependency of the number of breakpoints. The present article presents the state-of-the-art of the approaches for solving the parametric maximum flow problem and presents an original parametric preflow algorithm, based on network partitioning technique. After presenting the basic parametric network flow terminology adapted for the parametric network with linear capacity functions and zero lower bounds, the proposed *highest label partitioning push (HLPP) algorithm* is described in details, being accompanied by the corresponding theorems of correctness and of complexity of the algorithm. In one of its final sections, the article also proposes a way of implementation of our algorithm in the legislation domain. The Definitions (12-14) and interpretations contained in this section are also original contributions of the authors.
Moreover, the given example shows the way the proposed algorithm progressively generates hierarchical structures of legal articles (stipulations), gathered according to their relevance in explaining a general legal topic (or legal problem). Using suitable file structures for legal information recording, these sets of legal stipulations can be transmitted to any user (or solicitor).

Bibliography

[1] Ahuja, R., Magnanti, T. and Orlin, J.(1993); *Network Flows. Theory, algorithms and applications*, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1993.

[2] Bichot, C-E., Siarry, P. (2011), *Graph Partitioning: Optimisation and Applications*, ISTE Wiley.

[3] Hamacher, H.W. and Foulds, L.R. (1989); Algorithms for flows with parametric capacities, *ZOR - Methods and Models of Operations Research*, 33:21–37.

[4] Parpalea, M., Ciurea, E.(2013); Partitioning Algorithm for the Parametric Maximum Flow, *Applied Mathematics (Special Issue on Computer Mathematics)*, 4(10A):3–10.

[5] Parpalea, M.M. (2009);German Word Order, *Bulletin of the Transilvania University of Brasov*, Series IV, 2(51): 175–182.

[6] Parpalea, M. (2010); Min-Max algorithm for the parametric flow problem, *Bulletin of the Transilvania University of Brasov*, Series III: Mathematics, Informatics, Physics, 3(52):191–198.

[7] Parvan, M., Ghionea, F., Flaut, C. (2008); A mathematical model of the relevant request determination in the transport problem, Mathematics and Computer in Business and Economics, *Proc.of the 9th WSEAS Intl. Conf. on Mathematics and Computer in Business and Economics (MCBE'08)*, Bucureşti, 24-26 iunie 2008, 105–111.

[8] Ruhe, G. (1988); Complexity Results for Multicriterial and Parametric Network Flows Using a Pathological Graph of Zadeh, *Zeitschrift fur Oper. Res.*, 32: 9–27.

[9] Ruhe, G. (1985); Characterization of all optimal solutions and parametric maximal flows in networks, *Optimization*, 16(1): 51–61.

[10] Sângeorzan, L., Parpalea, M., et al. (2010); Partitioning Preflow-pull Algorithm for the Parametric Net-work Flow Problem - a Linguistic Rule-based Constraints Optimisation

Approach, Recent Advances in Neural Networks, Fuzzy Systems and Evolutionary Computing, *Proc.of the 11th WSEAS International Conference on Neural Networks*, Iasi, Romania, 111–116.

[11] Stumpff, A.M. (2013); The Law is a Fractal: The Attempt to Anticipate Everything, *Loyola University Chicago Law Journal*, 44.

[12] Yong, C., Chang-le, Lu (2006); Using Multi-Thread Technology Realize Most Short-Path Parallel Algorithm, *World Academy of Science, Engineering and Technology*, 15: 11–13.

[13] *High Court's Decision no 656/2012*, given in a public hearing on February 9th, 2012, in the file no 1259/44/2010, http://www.iccj.ro/cautare.php?id=68146, accessed September 11th, 2014, at 13,40.