Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia - Social and Behavioral Sciences 54 (2012) 1168 – 1175

Procedia
Social and Behavioral Sciences

EWGT 2012

15th meeting of the EURO Working Group on Transportation

Flow Expansion on Transportation Networks with Budget Constraints

Amir Elalouf [a,*], Ron Adany[b], Avishai (Avi) Ceder[c]

[a]*Department of Management, Bar-Ilan University, Ramat Gan, 52900, Israel*
[b]*Department of Computer Science, Bar-Ilan University, Ramat Gan, 52900, Israel*
[c]*Department of Civil and Environmental Engineering, University of Auckland, Auckland 1142, New Zealand*

Abstract

This study considers the Budgeted Flow Expansion (*BFE*) problem on transportation network. The problem input includes a given budget and a transportation network, i.e. a directed graph with edges' capacities. In addition, each edge is associated with possible expansion capacity and the expansion cost. For instance, given a transportation network connecting two cities and possible options of expanding existing roads and/or constructing new roads, the objective is to efficiently utilize a given budget to maximize the flow between the cities. The *BFE* problem is NP-hard for the general case where the expansion options are all-or-nothing, i.e., expand by utilizing the entire expansion capacity or do not expand at all. Nonetheless, in this study we consider a special case in which any integral amount of expansion capacity can be utilize; for this case a polynomial algorithm is proposed. The algorithm iteratively expands the flow by one unit by one unit as long as the resultant cost is within the budget constraint. In each iteration, the maximum flow is found using the known Ford-Fulkerson algorithm. Based on the residual network, combined with the possible expansion of edges, the cheapest path for expanding the flow is selected. The method described can be used as an efficient tool for decision makers to attain the best improvements of transportation networks when a limited budget is available. The methodology and algorithms can be applied to a real-world road network including the exhibition and interpretation of the unique features used and the benefits expected.

Keywords: Flows in graphs; Flow control; Optimization; Transportation.

1. Introduction

In the max-flow problem, one considers a transportation network, i.e. a directed graph, $G(V,E)$, with a source, s, and a sink, t. Each edge, (u,v), has a capacity, $c(u,v)$, which is the maximum flow that can traverse

* Corresponding author. Tel.: +972-3-5317128; fax: +972-3-5422904.
E-mail address: Amir.Elalouf@biu.ac.il

 doi:10.1016/j.sbspro.2012.09.831

from u to v. The flow can be associated with a flow of people, vehicles, trains, bits of information, water, etc. An $s-t$ flow is the amount of flow that can be transformed from source s to sink t. The flow of an edge (u,v) is denoted as $f(u,v)$. There are two basic properties in the transportation network: (i) the sum of the flow entering each vertex is equal to the sum of the flow exiting it, and (ii) the flow at each edge does not exceed its capacity. The max-flow problem can be mathematically formulated accordingly:

$$\begin{aligned} Max \quad & \sum_{(u,v)\in E} f(u,v) \\ s.t. \quad & \sum_{v\in V|(u,v)\in E} f(u,v) = \sum_{k\in V|(k,u)\in E} f(k,v) \quad \forall u \in V, u \neq s,t \qquad (1) \\ & f(u,v) \le c(u,v) \qquad \forall (u,v) \in E \end{aligned}$$

Since almost any kind of real network is dynamically expanded from time to time, decision makers regularly face such problems when they need to choose the best approach to improve the graph's flow. For example, road networks frequently need to be expanded in order to cope the growing transportation. The problem appears also in other domains such as in cellular and Internet networks that need to be dynamically expanded in order to deal with market changes and to improve the quality of service.

In the current work we analyze the Budgeted Flow Expansion (*BFE*) problem, i.e., the problem of expanding the maximum flow in a given network with budget constraints. The budget constraints make the problem more interesting and harder compared to the typical max-flow problem.

The rest of this paper is organized as follows. In Section 2 we provide an overview of max-flow problems. We formally describe the problem and its mathematical model in Section 3. We present our proposed algorithms in Section 4 and a run of the algorithms on a small sample in Section 5. Finally we summarize in Section 6.

2. Related Work

The max-flow problem was first solved by Ford and Fulkerson (FF) [1, 2]. The main idea of their algorithm is to iteratively push a flow from the source to the sink as long as there is a path, denoted as the augmenting-path, connecting them. The algorithm complexity is $O(f \mid E \mid)$, depending on the maximum flow of the graph, f, and the number of edges, $\mid E \mid$. A well-known implementation of the same idea with the complexity depending only on the input size was proposed as the Edmonds-Karp algorithm [3]. In this implementation the augmenting-paths are found using Breadth-First Search algorithm, outcome with a complexity of $O(\mid V \mid \mid E \mid^2)$.

The minimum-cost flow problem is similar to the max-flow problem [4], whereby extra inputs are given, a target flow value and the cost of sending a flow over each of the edges. The objective is to find the cheapest way to send the target flow from the source vertex to the sink vertex where the cost of sending a flow over an edge is proportional to the amount of the flow. The problem can be solved by successive shortest path and capacity scaling algorithms.

In [5] a similar approach was applied to solve the constraint maximum flow problem. The objective is to maximize the $s-t$ flow within a given budget where the cost of sending a flow over an edge is proportional to the amount of the flow, i.e. a linear cost function. This problem is different from *BFE* since the cost model is different as described in section 3.

The budged restricted maximum flow problem is discussed in [6], where the objective is to maximize the $s-t$ flow within a given budget. In this type of problem, a fixed cost is assigned to each edge which is incurred as soon as any positive flow is sent through that edge, i.e. the problem if all-or-nothing type. Since Knapsack is a special case of this problem the problem is NP-hard. Similarly, the general case of the *BFE* problem, where the expansion cost model is all-or-nothing, is also NP-hard.

Applications of the max-flow FF algorithm in relation to the connectivity of Public-Transport (PT) services can be found in [7]. Expansion of the connectivity-based network of PT services can be interpreted by means of

specific projects with natural budget constraints. For instance, Ceder in [7] describes such expansion by creating more capacity in terms of passenger-hours required for transferring between two buses, or between bus, rail and ferry. This expansion can take place by changing the location of stations, adding escalators, and even decreasing the transfer time using a better information system.

3. Problem Definition

The Budgeted Flow Expansion (*BFE*) problem input includes a directed graph $G(V,E)$; a source, s; a sink, t; and an expansion budget, B. In addition, each edge, (u,v), is associated with an integrated capacity, $c(u,v)$; an expansion integrated capacity, $e(u,c)$; and the expansion cost per one unit of flow, $p(u,v)$. The objective is to efficiently utilize the budget such that the $s-t$ flow will be maximized. Note that we can extend the problem to a multi-source multi-destination graph by adding a global source, s', connected to all sources and a global destination, t', connected to all destinations.

There are two types of decision variables in the problem. The first type of variable is the flow values per edge, as in the original max-flow problem, denoted as $f(u,v)$. The second type of variable is the expansion values per edge, denoted as $x(u,v)$. There is a connection between the variables as $f(u,v) \le c(u,v) + x(u,v)$, while the possible capacity values are:

$$\overbrace{\underbrace{0,1,\cdots,c(u,v)}_{\text{Original capacity}},\underbrace{,\cdots,c(u,v)+e(u,v)}_{\text{Optional expansion range}}}^{\text{Possible capacity range (flow range)}}$$

Summary of notation:

- $c(u,v)$, the capacity of the edge.
- $e(u,v)$, the maximum possible edge capacity expansion.
- $p(u,v)$, cost of expansion of the edge capacity by one unit.
- $f(u,v)$, the flow over the edge.
- $x(u,v)$, the expansion of edge, $0 \le x(u,v) \le e(u,v)$.

The *BFE* problem is mathematicaly formulated as:

$$\begin{aligned}
Max \quad & \sum_{(u,v)\in E} f(u,v) \\
s.t. \quad & \sum_{(u,v)\in E} x(u,v)\,p(u,v) \le B \\
& \sum_{v\in V|(u,v)\in E} f(u,v) = \sum_{k\in V|(k,u)\in E} f(k,v) \quad \forall u \in V, u \ne s,t \\
& 0 \le x(u,v) \le e(u,v) \quad \forall (u,v) \in E \\
& 0 \le f(u,v) \le c(u,v) + x(u,v) \quad \forall (u,v) \in E
\end{aligned} \tag{2}$$

4. Algorithms

We propose a method for solving the *BFE* problem in polynomial time. Our main idea is to iteratively increase the flow in the network using the cheapest path from the source vertex to the sink vertex. The remaining budget is updated in each iteration and the process stops when the entire buget has been utilized. The solution is divided into two algorithms. Algorithm 1 returns the edges that need to be expanded in order to expand the flow by one unit at minimum cost. Algorithm 2 iteratively call Algorithm 1 and keeps track of the budget limit B.

In Algorithm 1, the maximum flow is calculated using the max-flow FF algorithm. Then, based on the residual graph $G_f(V,E_f)$, which is given by the max-flow FF algorithm, a new graph is built. The new graph, G', is similar to the residual graph, except for the values associated with the edges. Each edge in G' is associated with a value that represents the cost of pushing one flow unit over it. The edge cost values in G', denoted $w'(u,v)$, depend on the flow values in the residual graph and the expansion value of each edge, i.e. $x(u,v)$.

If the flow pushed over the edge is smaller than its original capacity $c(u,v)$, then it is possible to push more flow over the edge without incurring a cost, thus the cost is set to 0 (line 6).

If the flow pushed over the edge is equal or larger than the original capacity, it may or may not be possible to expand it and push more flow through it. Three different cases are considered as described below.

The first case, handled in line 8, is where the flow is equal to the original capacity, i.e., all the original capacity is used and yet no expansion has been used. In this case the cost of the edge (u,v) is set to $p(u,v)$, which is the expansion cost per one flow unit. In addition, in order to allow the flow cancellation of the edge (u,v), the cost of the opposite edge (v,u) is set to 0, i.e. $w'(v,u)=0$.

The second case, handled in line 10, is where the flow is larger than the original capacity but lower than the maximum possible capacity including the expansion capacity, In othere words in this case some, but not all, of the expansion value has already been used. In this case the cost of the edge (u,v) is set to $p(u,v)$. To allow cancellation of the expansion of the edge (u,v), the cost of the opposite edge, (v,u), is set to $-p(u,v)$, i.e. $w'(v,u)=-p(u,v)$.

The third case, handled in line 12, is where the flow is equal to the maximum possible capacity including the expansion capacity, i.e. all of the expansion value has been used. In this case it is not possible to push more flow over the edge (u,v) and the edge cost is set to $+\infty$. As in the second case, to allow cancellation of the expansion of the edge (u,v), the cost of the opposite edge, (v,u), is set to $-p(u,v)$, i.e. $w'(v,u)=-p(u,v)$.

Algorithm 1: Single Expansion

1: Find the maximum flow in G using the max-flow FF algorithm.

2: Denote the last residual graph $G_f(V,E_f)$

3: Build $G'(V',E')$ such that $V'=V, E'=E_f$

4: **for** each $(u,v)\in E'$ **do**

5: **if** $c_f(u,v)>0$ and $c_f(v,u)>0$ **then**

6: Set $w'(u,v)=0$ and $w'(v,u)=0$

7: **else** { $c_f(u,v)=0$ or $c_f(v,u)=0$, w.l.o.g. assume $c_f(u,v)=0$ }

8: **if** $x(u,v)=0$ **then**

9: Set $w'(u,v)=p(u,v)$ and $w'(v,u)=0$

else if $0 < x(u,v) < e(u,v)$

Set $w'(u,v) = p(u,v)$ and $w'(v,u) = -p(u,v)$

else { $x(u,v) = e(u,v)$ }

Set $w'(u,v) = +\infty$ and $w'(v,u) = -p(u,v)$

end if

end if

end for

Find the shortest path, P, in G' regarding w' values

for each $(u,v) \in P$ **do**

if $w'(u,v) > 0$ **then**

$x(u,v) = x(u,v) + 1$

else if $w'(u,v) < 0$ **then**

$x(u,v) = x(u,v) - 1$

end if

end for

return The expansion values $x(u,v)$, and P's cost, i.e. $\sum_{(u,v)\in p} w'(u,v)$

Algorithm 2: Max Expansion

Run Algorithm 1 to attain the cheapest expansion of one flow unit, i.e. $x(u,v)$ values, and its cost

While $(B - ExpansionCost > 0)$ **do**

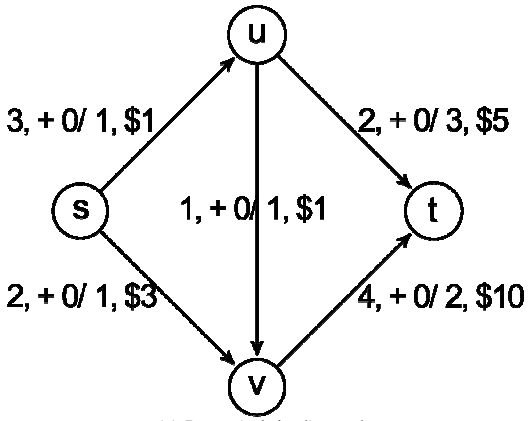
Update capacity values in G according to expansion values $x(u,v)$

Update remain capacity, i.e. $B = B - ExpansionCost$

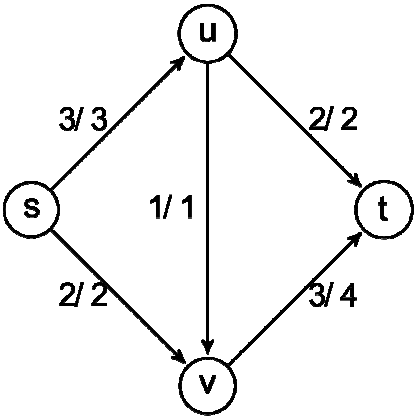
Run again Algorithm 1

end while

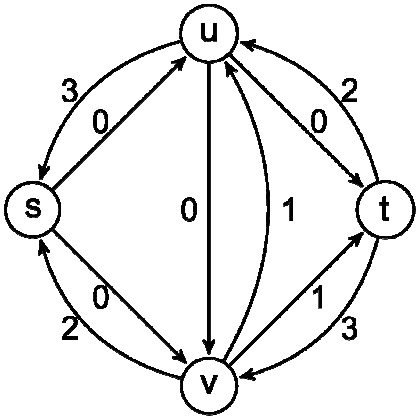
return The expansion values $x(u,v)$



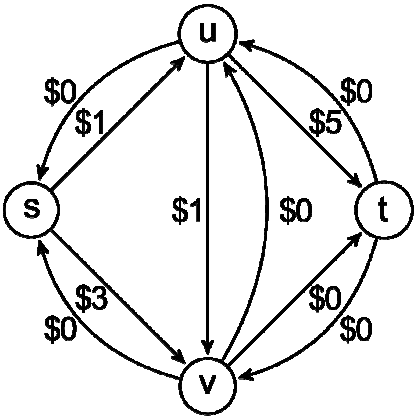
(a) Input (original) graph.



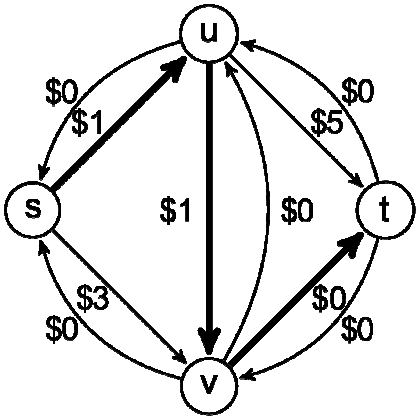
(b) Maximum flow: 5 units flow.



(c) Residual graph.

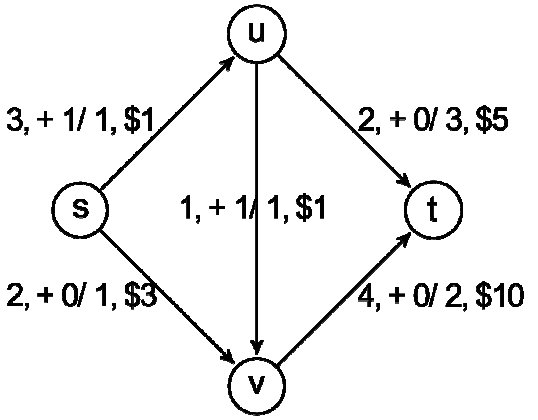


(d) G' graph.

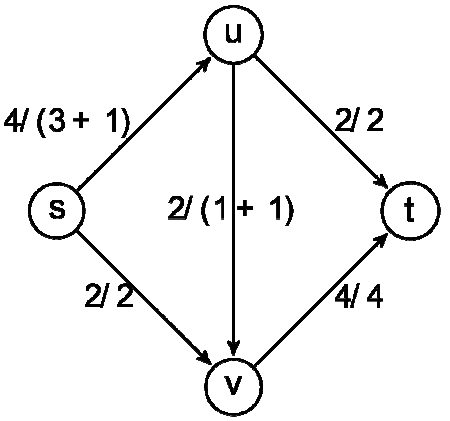


(e) Shortest path in G' at cost 2$.

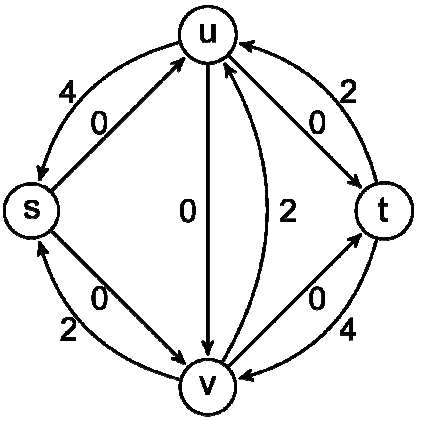
Fig. 1. First call to Singel Expansion (Algorithm 1).



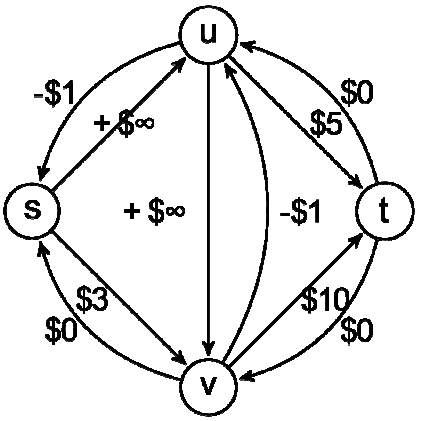
(a) Input graph.



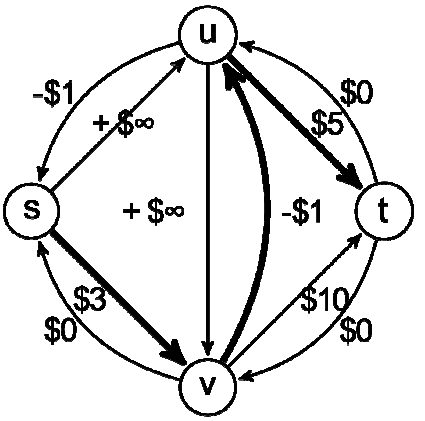
(b) Maximum flow: 6 units flow.



(c) Residual graph.



(d) G' graph.



(e) Shortest path in G' at cost 7$.

Fig. 2. Second call to Single Expansion (Algorithm 1).

5. Example

A numerical example with a budget of \$9 is illustrated in the figures. Fig. 1 represents the first iteration of Algorithm 1 and Fig. 2 represents the second iteration.

The first graph (1a) is the original graph, where the edge label format is (capacity, +current expansion / max possible expansion, expansion cost per unit), i.e., $(c,+x/e,\$p)$. Since this is the first iteration, the $x(u,v)$ values are all zero. The maximum flow is presented in graph (1b), where the edge label format is (flow/capacity), i.e. (f/c). The corresponding residual graph is presented in graph (1c). The graph constructed by Algorithm 1, G', is presented in (1d) and the cheapest path from s to t is marked in graph (1e). As a result of the first iteration, the flow is increased by expanding the edges (s,u) and (u,v) each by one unit incurring a total cost of $1+1=\$2$.

Since the remaining budget is $\$7>0$, another iteration of Algorithm 1 is executed and presented in Fig. 2. Graph (2b) shows the maximum flow where the capacities considered are updated and include the expansion from the previous step, e.g., the capacity of (s,u) is $(3+1)$ which signifies the original capacity, 3, and the expansion, 1. The cheapest path from s to t is marked in graph (2e). This path contains the edge (v,u) which is associated with a negative cost that signifies the payback for canceling the expansion of edge (u,v). As a result, the flow is increased by canceling one unit of expansion of edge (u,v) and expanding both edges, (s,v) and (u,t), by one unit at a total cost of $(-1)+3+5=\$7$.

The entire budget, $B=9$, was spent, $\$2+\$7$, and the algorithm terminates. The total flow expansion was 2 flow units.

6. Summary

This study considers the Budgeted Flow Expansion (*BFE*) problem for expanding the maximum flow of a transportation network given a budget. This problem arises in cases where decision makers need to dynamically improve the network flow. The general case is NP-hard where the expansion options are all-or-nothing, i.e. expand by utilizing the entire expansion capacity or do not expand at all. In this paper we consider a special case in which any integral amount of expansion capacity can be utilized. We have presented a polynomial algorithm to solve it. Future work will consider the general case with a possible approximation algorithm.

References

[1] L. Ford, D. Fulkerson, Maximal flow through a network, Canadian Journal of Mathematics 8 (3) (1956) 399–404.
[2] L. R. Ford, D. R. Fulkerson, Flows in networks, Princeton University Press, 1962.
[3] J. Edmonds, R. Karp, Theoretical improvements in algorithmic efficiency for network flow problems, Journal of the ACM (JACM) 19 (2) (1972) 248–264.
[4] T. L. M. Ravindra K. Ahuja, J. B. Orlin, Network Flows: Theory, Algorithms, and Applications, Prentice-Hall, 1993.
[5] R. Ahuja, J. Orlin, A capacity scaling algorithm for the constrained maximum flow problem, Networks 25 (2) (1995) 89–98.
[6] H. Eiselt, H. von Frajer, On the budget-restricted max flow problem, OR Spectrum 3 (4) (1982) 225–231.
[7] A. Ceder, Public transit planning and operation: theory, modeling and practice, Elsevier, Butterworth-Heinemann, Oxford, UK, 2007.