Information Processing Letters 116 (2016) 560–563

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

Improved balanced flow computation using parametric flow

Omar Darwish *, Kurt Mehlhorn

Max-Planck-Institute für Informatik, Saarbrücken, Germany

ARTICLE INFO

Article history:
Received 18 December 2015
Received in revised form 15 April 2016
Accepted 18 April 2016
Available online 22 April 2016
Communicated by Ł. Kowalik

Keywords:
Algorithms
Network flow
Market equilibrium
Balanced flow
Parametric flow

ABSTRACT

We present a new algorithm for computing balanced flows in equality networks arising in market equilibrium computations. The current best time bound for computing balanced flows in such networks requires $O(n)$ maxflow computations, where n is the number of nodes in the network [1]. Our algorithm requires only a single parametric flow computation. The best algorithm for computing parametric flows [3] is only by a logarithmic factor slower than the best algorithms for computing maxflows. Hence, the running time of the algorithms in Devanur et al. [1] and Duan and Mehlhorn [2] for computing market equilibria in linear Fisher and Arrow–Debreu markets improve by almost a factor of n.

1. Introduction

Balanced flows are an important ingredient in equilibrium computations for the linear Fisher and Arrow–Debreu markets. In the Arrow–Debreu market model, we have n agents, where each agent owns some goods and a utility function over the existing set of goods. The goal is to assign prices to all the goods such that the market clears, meaning that all the goods are sold and each agent spends all of his budget on only goods with maximum utility. These prices are called the equilibrium prices and the state of the market with such prices is the equilibrium state we want to reach. Under some assumptions, it was shown in [7] that such prices exist. The Fisher model is a simplification of the Arrow–Debreu model, where each agent comes with a budget instead of the endowment of goods [8]. The linear Arrow–Debreu market is a special case where the utility functions are linear, and the same applies to the linear Fisher model. Later on, we will refer to agents as buyers.

The problem of finding the equilibrium prices in these market models has been studied for a long time, see [9] and [10] for surveys. The concept of a balanced flow was introduced in the first combinatorial algorithm [1] for finding the equilibrium prices in the linear Fisher market. Later, balanced flows were used in [2] and [11] for combinatorial algorithms for computing equilibrium prices in the linear Arrow–Debreu market.

The equilibrium algorithms mentioned above compute balanced flows in a special type of flow network, known as *equality network*. An *equality network* is a bipartite flow network where we have a set of buyers B adjacent to the source node s and a set of goods C adjacent to the sink node t. Hence, the vertex set of the network is $\{s,t\}\cup B\cup C$. Also, we assume that the number of buyers is equal to the number of goods. Each buyer b_i has a positive budget e_i and each good c_j has a positive price p_j. The edge set is defined as follows:

1. An edge (s,b_i) with capacity e_i for each $b_i \in B$.
2. An edge (c_i,t) with capacity p_i for each $c_i \in C$.
3. All edges running between B and C have infinite capacity. Also, we assume that each buyer is connected to at least one good and similarly each good is connected to at least one buyer.

* Corresponding author.
E-mail addresses: odarwish@mpi-inf.mpg.de (O. Darwish), mehlhorn@mpi-inf.mpg.de (K. Mehlhorn).

http://dx.doi.org/10.1016/j.ipl.2016.04.008
0020-0190/

A flow in this network corresponds to the money flow from buyers to goods. So, an amount of x units flowing from buyer b_i to good c_j indicates that b_i spends x units on good c_j. We refrain from relating the equality network to the market equilibrium computation as this is not important for understanding our algorithm.

Balanced flows are defined for equality networks in [1]. We denote the set of buyers by $B = \{b_1, b_2, ..., b_n\}$. The surplus $r(b_i)$ of the buyer b_i with respect to a maximum flow f is the residual capacity of the edge (s, b_i) (capacity of this edge minus the flow running through it). The surplus vector is the vector of the surpluses of all buyers $(r(b_1), r(b_2), ..., r(b_n))$. The balanced flow is the maximum flow that minimizes the two-norm of the surplus vector among all maximum flows of the network. In [1], it is shown that a balanced flow can be computed with $O(n)$ maxflow computations. This is the best time bound achieved up till now for computing balanced flows.

The name "balanced flow" comes from the fact that it is the maximum flow in the equality network that balances the surpluses as much as possible. Additionally, it is shown in [1] that the surplus vector is unique for all balanced flows in a given equality network. From this fact and other properties shown in [1] and [2], we can deduce the following characterization of balanced flows in equality networks:

1. Buyers can be partitioned into maximal disjoint blocks based on their surpluses, $B_1, B_2, ..., B_h$ having surpluses $r_1 > r_2 > ... > r_h \geq 0$. Buyers in block B_i have the surplus r_i. The uniqueness of this partition follows from the uniqueness of the surplus vector of the buyers.
2. We can assume without loss of generality that $r_h = 0$.
3. Define C_i to be the set of goods to which money flows from B_i. For $i < h$, all goods in C_i are completely sold, i.e., the edges from C_i to t are saturated. There is no flow from B_i to C_j for $j < i$. Also, there are no edges from B_i to C_j for $j > i$.

We show how to compute balanced flows in equality networks by a single parametric flow computation, which improves the running time of the algorithms in [1] and [2] for computing market equilibria in linear Fisher and Arrow–Debreu markets by almost a factor of n. Our algorithm adds to the applications of parametric flows mentioned in [3].

In Section 2, we give an introduction to parametric network flows. In Section 3, we state our algorithm for computing balanced flows, analyze its running time, and prove its correctness.

2. Parametric network flows

The parametric network flow problem [3] is a generalization of the standard network flow problem, in which the arc capacities are not fixed, but are functions of a real valued single parameter λ. More specifically, we consider the following parametric problem:

1. The arc (s, b_i) has capacity $\max(0, e_i - \lambda)$.
2. The capacity of all other arcs is constant.
3. The parameter λ decreases from a large value down to zero.

We define the *min-cut capacity function* $\kappa(\lambda)$ as the capacity of a minimum cut in the network as a function of the parameter λ [3]. Among the cuts of capacity $\kappa(\lambda)$, let $(X(\lambda), \overline{X(\lambda)})$ be the cut with the smallest sink side[1] $\overline{X(\lambda)}$. It is shown in [3] that $X(\lambda) \subseteq X(\lambda')$ for $\lambda \geq \lambda'$.

This implies that we have at most $n - 1$ distinct minimum cuts. Under the assumption of linearity of the arc capacity functions of λ, $\kappa(\lambda)$ is a piecewise-linear concave function with at most $n - 2$ breakpoints. A breakpoint is a value of λ where the slope of $\kappa(\lambda)$ changes.

As long as the minimum cut is the same, decreasing λ will only result in increasing the flow linearly in the current segment of $\kappa(\lambda)$. [3] gives an algorithm for computing all breakpoints of $\kappa(\lambda)$ which runs in $O(nm \log(n^2/m))$ worst-case time complexity, for a network with n nodes and m edges; they call it the *breakpoint algorithm*. Additionally, they state that this algorithm can be easily augmented to store for each vertex $v \notin \{s, t\}$ the breakpoint at which v moves from the sink side to the source side of a minimum cut of minimum sink size without altering the time bound. We will refer to the breakpoint algorithm with the stated augmentation as the *augmented breakpoint algorithm*. We will use this *augmented breakpoint algorithm* in our algorithm for computing balanced flows.

The algorithm in [3] extends the standard preflow algorithm by Goldberg and Tarjan [12] for solving the maximum flow problem. Its running time is only by the factor $\log(n^2/m)$ slower than the running time of the best maxflow algorithms [4–6].

3. Improved algorithm for balanced flow computation

In this section, we describe our algorithm for computing a balanced flow in a given equality network N and prove its correctness. The intuition behind our algorithm is as follows. For $\lambda = \infty$, all edges out of s have capacity zero. Since all edges into t have positive capacity, the minimum cut will have all buyers on the sink side. Then, step by step, at λ values corresponding to $r_1, ..., r_h$, the blocks $B_1, ..., B_h$ of buyers will move from the sink side of the minimum cut to the source side. The behavior of the function $\kappa(\lambda)$ changes at these breakpoints, since at these surpluses some buyers will not be able to push more flow to the goods side. Hence the flow will stop at this point at these buyers, while continuing to increase for other buyers, and so on. An example for the evolution of the minimum cut is shown in Fig. 1. We will next give the details.

[1] With respect to set inclusion, there is a unique minimum cut of capacity $\kappa(\lambda)$ with smallest sink side.