Algorithmic Game Theory Project
Note for Market Equilibrium via a Primal-Dual-Type Algorithm

林超超 ID: 1120191287

2021 年 11 月 19 日

Abstract

This paper[1] proposes the first polynomial time linear utility market equilibrium model. The author uses the methods of duality and graph theory, mainly maximum flow and minimum cut, to establish a basic algorithm for calculating market equilibrium prices. Then optimize the basic algorithm by finding high-surplus buyers, and obtain a polynomial time market equilibrium model. I restated the author's thoughts based on my own understanding. But there may be some flaws.

Problem

The author wants to solve an old problem defined by Irving Fisher in 1981[3]. This question assumes a market of buyers and goods. The goal is to calculate the price of goods so that there is no deficiency or surplus of any goods, i.e. the market clears.

The market has two specified assumptions.

1. The first is the money and the amount of each good owned by the buyer.
2. The second is the linear utility function is of buyers.

Express the above description as mathematical language. The author defines a A set of n goods and a B set of n' buyers, that is

$$|A| = n, |B| = n'$$

For each buyer i, the author specifies that she possesses the amount e_i of money and for each good j, the amount b_j of this good. In addition, each buyer i has a utility u_{ij} for each good j, and it is assumed that the utility is a linear function.

The goods has been given a price $(p_1, p_2, ..., p_n)$. If after each buyer is assigned such a basket of goods, there is no surplus or deficiency of any of the goods. The current price $(p_1, p_2, ..., p_n)$ is the market clearing price. The problem is to find such a price in polynomial time.

In order to make the author's process clearer, I drew a market case diagram as Figure 1. There are 3 buyers and 4 goods in the example market. The buyer's money is (e_1, e_2, e_3). The price of the goods is (p_1, p_2, p_3, p_4).

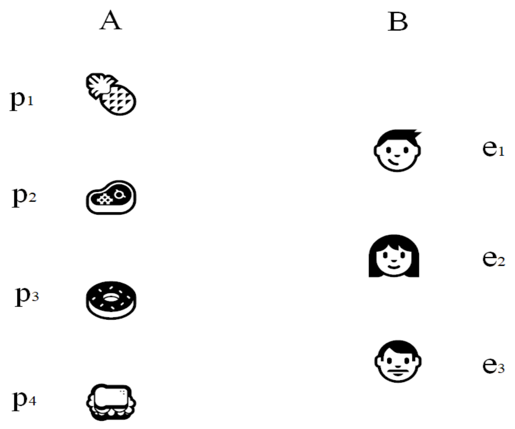

Figure 1: An example of market

Idea

For this problem, the author's idea is to construct a bipartite graph called an equality subgraph and its edges are called equality edges. Author let $\mathbf{p} = (p_1, p_2, ..., p_n)$ denote a vector of price. If at these prices buyer i is given good j, she derives u_{ij}/p_j amount of utility per unit amount of money spent. She will feel happiest when she is allocated a good that maximize this ratio (If there are several goods maximizing this ratio, she is equally happy with any combination of these goods). So the author defines her *bang per buck* to be $\alpha_i = \max\{u_{ij}/p_j\}$. Obviously, for each $j \in A$ and $i \in B$, $\alpha_i \geq u_{ij}/p_j$.

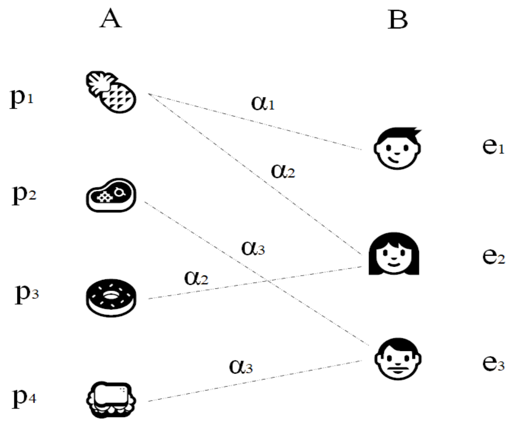

Figure 2: An example of equality subgraph

Therefore, the author constructed a bipartite graph G of market relations. Its bipartition is (A, B) and for $i \in B$ and $j \in A$, the edge (i, j) in G means $\alpha_i = u_{ij}/p_j$. Having an edge (i, j) in the graph G means that getting good j will make buyer i the happiest, ralative to the current price. This is the definition of equality subgraph, like Figure 2 in the example market.

In order to computing the largest amount of goods that can be sold along the edges of the equality subgraph, without exceeding the budgets of buyers or the amount of goods available, the author introduced the following network based on the equality subgraph.

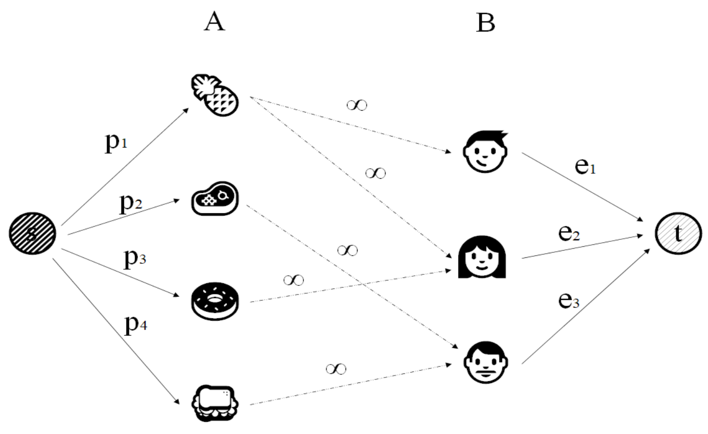

Figure 3: An example of the network

Like the Figure 3, the steps to construct the author's network are as follows.

1. Firstly, direct edges of G from A to B and assign a capacity of infinity to all these edges
2. Secondly, introduce source vertex s and a directed from s to each vertex $j \in A$ with a capacity of p_j.
3. Thirdly, introduce sink vertex t and a directed from each vertex $i \in B$ to t with a capacity of e_i.

So the problem can be transformed into a max-flow problem in the above network which is clearly a function of the current prices $\mathbf{p}$ and will be denoted $\mathbf{N(p)}$.

In order to ensure that all goods are sold at the current price, the **Invariant** is the cut $(s, A \cup B \cup t)$ is a min-cut in $\mathbf{N(p)}$. That is the sum of the capacity of the red edges in the Figure 4.

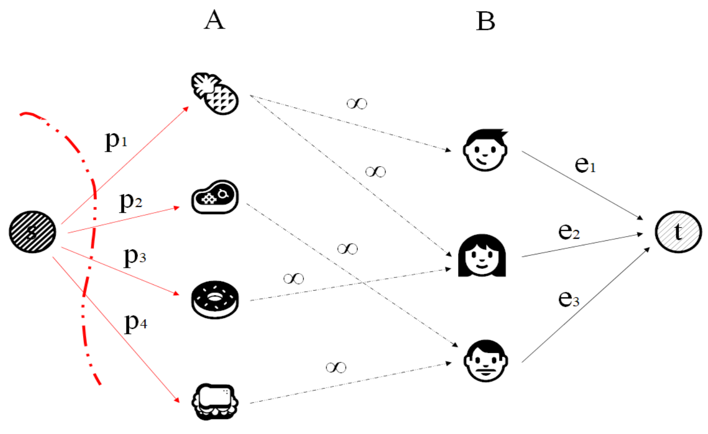

Figure 4: The min-cut of example network

That is the cut $(s, A \cup B \cup t) \geq$ the cut $(s \cup A \cup B, t)$ (the sum of the capacity of the red edges in the Figure 5). Otherwise the goods will not be sold out if the buyers do not have enough money to buy all the goods. So, the only eventuality is that buyers may be left with surplus money.

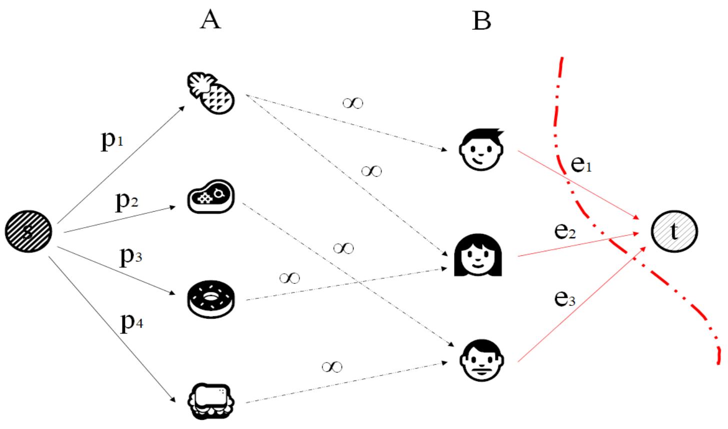

Figure 5: The cut of example network

When we raises the prices systematically and always maintaining the **Invariant**, the surplus money with buyers will keep decreasing. Then, if the surplus vanishes, the current prices is the market clearing prices which we want. At this time, the cut $(s \cup A \cup B, t)$ is also a min-cut in $\mathbf{N(p)}$, that is, the max-flow in $\mathbf{N(p)}$ equals the total amount of money possessed by the buyers(Max-flow min-cut theorem[2]).

Therefore, the market equilibrium problem is transformed into an **optimization problem**: find prices $\mathbf{p}$ under which network $\mathbf{N(p)}$ supports maximum flow. The author's algorithm is just like the above thought.

Basic Algorithm

Initialization

In order to guarantee the **Invariant** at the start of the algorithm, the author proposes two condition.

1. The first is to fix the initial price at $1/n$.
2. The second is to compute α_i for each buyer i at the prices fixed in the previous step and compute the equality subgraph. And if good j has no edge incident, reduce its price to

$$p_j = \max_i \{\frac{u_{ij}}{\alpha_i}\}$$

The first step ensures that the price is low enough that each buyer can afford all the goods, because the goods together cost one unit and the amount of money e_i possessed by each buyer i is integral.

The second step ensures that each good j has an interested buyer at least, that is, each vertex j has an edge at the equality subgraph. And we reduce the price of good if it has no edge incident by

Algorithm 1 the basic algorithm

```
1: // Initialization
2: $\forall j \in A$, $p_j \leftarrow 1/n$;
3: $\forall i \in B$, $\alpha_i \leftarrow \min_j u_{ij}/p_j$;
4: Compute equality subgraph $G$
5: if $\forall j \in A$, $degree_G(j) = 0$ then
6:     $p_j \leftarrow \max_i u_{ij}/\alpha_i$;
7: end if
8: Recompute $G$
9: $(F, F') \leftarrow (\emptyset, \emptyset)$; // the frozen subgraph
10: $(H, H') \leftarrow (A, B)$; // the active subgraph
11: while $H \neq \emptyset$ do
12:     $x \leftarrow 1$;
13:     Define $\forall j \in H$, price of $j$ to be $p_j x$;
14:     Raise x continuously until one of two events happens:
15:     if $S \subseteq H$ becomes tight then
16:         Move $(S, \Gamma(S))$ from $(H, H')$ to $(F, F')$;
17:         Remove all edges from $F'$ to $H$;
18:     end if
19:     if an edge $(i, j)$, $i \in H'$, $j \in F$ attains equality, $\alpha_i = u_{ij}/p_j$ then
20:         Add $(i, j)$ to $G$;
21:         Move connected component of $j$ from $(F, F')$ to $(H, H')$;
22:     end if
23: end while
```

the given equation. At this time, buyer i will be happiest to get this good j, that is, adding an edge (i, j) from vertex $j \in A$ to vertex $i \in B$.

Iteratively Increasing Prices

The key point of the so-called primal-dual-type algorithm is the next steps. The iterative improvement steps follow the spirit of the primal-dual schema: The "primal" variables are the flows in the edges of $\mathbf{N(p)}$ and the "dual" variables are the current prices. The current flow suggests how to improve the prices and vice versa.

Then, let's focus on the definition of a few symbols and the key concept of the algorithm.

symbol

For $S \subseteq A$, define its neighborhood $\Gamma(S)$ in $\mathbf{N(p)}$

$$\Gamma(S) = \{j \in B | \exists i \in S \; with(i, j) \in G\}$$

For example, in the Figure 3, there are 4 goods, i.e. pineapple, meat, donut, sandwiches and 3 buyers, i.e. boy, woman, man. Assuming a set S of the boy and the woman, its neighborhood $\Gamma(S)$ is the pineapple and the donut.

In addition, define the symbol of money $m(S)$. For $S \subseteq B$, define its money $m(S) = \sum_{i \in B} e_i$. For $S \subseteq A$, define its money $m(S) = \sum_{j \in A} p_j$.

concept

The one key concept is **tight set**. If there is a unique maximal set $S \subseteq A$ such that $m(S) = m(\Gamma(S))$, this is a tight set. According to the example above, in the example network, $S = \{boy, woman\}$ and $\Gamma(S) = \{pineapple, donut\}$. So, assuming $p_1 + p_3 = e_1 + e_2$, i.e. $m(S) = m(\Gamma(S))$, we call this set a tight set.

The another is the **active subgraph** and the **frozen subgraph**. According to the definition of tight set above, the prices of goods in the tight set cannot be increased without violating the **Invariant**. So we call $(S, \Gamma(S))$ frozen subgraph and $(A - S, B - \Gamma(S))$. The subgraph in the example consisting of boy, woman, pineapple, donut is a frozen subgraph and the active subgraph is composed of the man, meat and sandwich.

Lemma 2 For given prices **p** network $\mathbf{N(p)}$ satisfies the **Invariant** iff

$$\forall S \subseteq A : m(S) \leq m(\Gamma(S)).$$

In addition, the author propose **Lemma 2** before the first key concept above. He used the method of contradiction to prove the lemma. Maybe I can explain it more simply. We also assume that $m(S) > \Gamma(S)$. Because $S \subseteq A$, obviously, in this condition, the buyers of $\Gamma(S)$ cannot afford the goods of S, which means that the prices is higher and the buyer owns less money. Well, a cut can be found that break the **Invariant**.

Lemma 3 If the **Invariant** holds and $S \subseteq A$ is the tight set, then each good $j \in (A - S)$ has an edge , in the equality subgraph to some buyer $i \in (B - \Gamma(S))$.

After that, the author also proposed **Lemma 3**, which was also proved by contradiction, using the conclusion of **Lemma 2**. This lemma can also be considered more simple. Some examples are given above. You can see that there is no edge between the subgraph formed by tight set S with the corresponding buyers $\Gamma(S)$ and its complement subgraph $(A - S, B - \Gamma(S))$. These goods $A - S$ will definitely be sold to the buyers in $B - \Gamma(S)$, that is, there must be an edge between S and $\Gamma(S)$, otherwise, it will violate invariant.

algorithm key

According to some of the above concepts and theorems, the author established the key part of the algorithm, that is, raise prices of goods in the active subgraph in such a way that retain the equality edges. Therefore, the algorithm uses the multiplying prices by x and gradually increasing x, starting with $x = 1$, until one of the following condition happens:

1. A set $R \neq \emptyset$ goes tight in the active subgraph.

2. An edge (i, j) with $i \in (B - \Gamma(S))$ and $j \in S$ becomes an equality edge.

If the first event happens, the author redefine the active subgraph to be $(A-(S\cup R), B-\Gamma(S\cup R))$ and proceed with next iteration.

If the second event happens, the author move (S_l, T_l) into the active subgraph, because for the new equality edge (i, j), $\Gamma(S_l) = T_l \cup i$, S_l is not tight anymore.

We use the maximum flow to find the tight set. And the author gave a proof of why, summarized as follows.

At first, define

$$x^* = \min_{\emptyset\neq S\subseteq A} \frac{m(\Gamma(S))}{m(S)},$$

the value of x at which nonempty set goes tight. Let S^* denote the tight set at prices $x^* \cdot \mathbf{p}$. Assuming that $A_1, A_2 \subseteq A$ and $B_1, B_2 \subseteq B$, $(s \cup A_1 \cup B_1, A_2 \cup B_2 \cup t)$ is a cut in the network.

If $x \leq x^*$, it is obvious that

$$\forall S \subseteq A : x \cdot m(S) \leq m(\Gamma(S)).$$

Therefore by **Lemma 2**, the **Invariant** hold, i.e. $(s, A \cup B \cup t)$ is a min-cut.

If $x > x^*$, the cut $(s \cup S^* \cup \Gamma(S^*), t)$ will smaller than the cut $(s, A \cup B \cup t)$, because $x \cdot m(S^*) > x^* \cdot m(S^*) = m(\Gamma(S^*))$. Therefore, $(s, A \cup B \cup t)$ is not a min-cut in this condition.

Then, the author proves that $S^* \subseteq A_1$. If you want to prove this, then just prove that the intersection of S^* and A_2 is empty. So, he let $S^* \cap A_2 = S_2$ and $S^* - S_2 = S_1$ and prove by contradiction, that is, assuming $S_2 \neq \emptyset$. Obviously $\Gamma(S_1) \subseteq B_1$, otherwise it will cause the cut to have infinite capacity. There are two contradiction.

1. At first, if $m(\Gamma(S_2) \cap B_2) < x \cdot m(S_2)$, that is the sum prices of goods in S_2 would exceed the money the buyers $\Gamma(S_2) \cap B_2$ had. Therefore, if we move S_2 and $\Gamma(S_2)$ to the s side, we can get a smaller cut. This is a contradiction. So, $m(\Gamma(S_2) \cap B_2) \geq x \cdot m(S_2) > x^* m(S_2)$.

2. Then, if $S_2 = S^*$, it will also lead to contradiction. Because $m(\Gamma(S_2) \cap B_2) + m(\Gamma(S_2)) \leq x^*(m(S_2) + m(S_1))$, it can be obtained from the above inequality, $m(\Gamma(S_1))/m(S_1) < x^*$. This contradicts the definition of x^*.

Therefore, $S_2 = \emptyset$ and $S_* \subseteq A_1$. Furthermore, if $x = x^*$, the min-cuts are obtained by moving a bunch of connected components of $(S^*, \Gamma(S^*))$ to the s-side of the cut $(s, A \cup B \cup t)$.

Finally, let $x = m(B)/m(A)$. If $x \geq x^*$, $(s, A \cup B \cup t)$ is a min-cut in $\mathbf{N(x\cdot\ p)}$, then $x^* = x$ and $S^* = A$. Otherwise, if $(s \cup A_1 \cup B_1, A_2 \cup B_2 \cup t)$ is a min-cut in $\mathbf{N(x\cdot\ p)}$. We know that if $A_1 = A$, then $B_1 = B$. The latter will be not a min-cut. Hence, $S^* \subseteq A_1 \subset A$. Therefore, it is sufficient to recurse on the smaller graph $(A_1, \Gamma(A_1))$, i.e. x^* and S^* can be found using n max-flow computation.

termination

Next, it is necessary to prove that this algorithm can terminate at the market clearing price. So let's assume some symbols. The paper let M be the total money possessed by buyers and let f be the max-flow computed in network $\mathbf{N(p)}$ at current prices $\mathbf{p}$. Thus, $M - f$ is the surplus money with the buyers. Then, let $U = \max_{i \in B, j \in A}\{u_{ij}\}$ and $\Delta = nU^n$.

Let S be the newly tight set and consider the equality subgraph induced on the bipartition $(S, \Gamma(S))$. The author pick a subgraph in which $j \in S$ can reach all other vertices $j' \in S$ and propose that $p_{j'} = ap_j/b$ where a and b are products of l untility parameters $(u'_{ik}s)$ each. So, $p_j = m(\Gamma(S))d/c$, where $c \leq \Delta$. Thus, at the termination of a phase, the prices of goods in the newly tight set must be rational number with denominator $\leq \Delta$. More simply, in the tight set, the denominator of the price should $\leq \Delta$. Otherwise, the buyers in $\Gamma(s)$ will have surplus money after getting the goods in S. It is contradict to the definition of tight set. Since then, the author has proved that the algorithm will end at the market clearing price.

running time

The author mentioned the complexity of Algorithm 1 in this paper. The running time of launching the maximum flow calculation is $O(Mn^2\Delta^2)$.

The Algorithm 1 terminates with market clearing prices in at most $M\Delta^2$ phase. At First, let the price of j at the end of two phases P and P' be p/q and r/s respectively. Clearly, the later price of j, $r/s > p/q$. Then, $q \leq \Delta$ and $s \leq \Delta$. Thus, the increase in the price of j, going from P to P', is $\geq 1/\Delta^2$, because

$$\frac{r}{s} - \frac{p}{q} = \frac{rq - sp}{sq} \geq \frac{1}{\Delta^2}.$$

Clearly, each phase can lead to a increase of the price of j. So, after k phases, the increase $\geq k/\Delta^2$, which lead to $f \geq k/\Delta^2$. By the lemma of $f \geq k/\Delta^2$, then the phases times $k \leq f\Delta^2$ and if the surplus vanishes, the cut $(s \cup A \cup B, t)$ is also a min-cut in $\mathbf{N(p)}$, that is the max-flow f is equal to the total amount of money M possessed by the buyers. So, the phases times $k \leq M\Delta^2$. Then, the author proposes a loose the complexity of Algorithm 1 $O(Mn^2\Delta^2)$. But remark it is easy to shave off a factor of n by giving a tighter version. For each phases, its iteration bringing goods from the tight set to the active subgraph, this cannot happen more than n times without a set going tight. (This means that the more tight complexity of this algorithm should be $O(Mn\Delta^2)$?)

Polynomial Time Algorithm

idea

Based on the above basic algorithm, the author proposes an accelerated version Algorithm 2, by only increasing the prices of goods adjacent to buyers with high surplus money. For example, in the Figure 3, if the boy is the high-surplus buyer, then increase the price of the pineapple.

Here are some new symbol definitions. At first, restrict a specific flow so that the surplus of a buyer is well-defined. Then, define the surplus of buyer i of the given flow f in the network $\mathbf{N(p)}$, $\gamma_i(\mathbf{p}, f)$. It is the residual of the edge (i, j) with respect of f, which is equal to the e_i minus the flow sent through the edge (i, t). Let $R(\mathbf{p}, f)$ be the residual network of $\mathbf{N(p)}$ with the flow f (I drew an example to explain this network in the Figure 6 and Figure 7). The surplus vector $\gamma(\mathbf{p}, f) = (\gamma_1(\mathbf{p}, f), \gamma_2(\mathbf{p}, f), ..., \gamma_n(\mathbf{p}, f))$. Let $||v||$ denote the l_2 norm of vector, that is $||v|| = \sqrt{\sum_{i=1}^{n} v_i^2}$. Define **Balanced flow** is a maximum flow that minimizes $||\gamma(\mathbf{p}, f)||$ for given $\mathbf{p}$. δ is a surplus (a deeper explanation in the further proof.).

Algorithm 2 the polynomial time algorithm

```
1: // Initialization
2: $\forall j \in A$, $p_j \leftarrow 1/n$; $\forall i \in B$, $\alpha_i \leftarrow \min_j u_{ij}/p_j$;
3: Define $G(A, B, E)$ with $(i, j) \in E$ iff $\alpha_i = u_{ij}/p_j$
4: if $\forall j \in A$, $degree_G(j) = 0$ then
5:     $p_j \leftarrow \max_i u_{ij}/\alpha_i$;
6: end if
7: Recompute $G$; $\delta = M$;
8: repeat
9:     Compute a balanced flow $f$ in $G$;
10:    Define $\delta$ to be the maximum surplus in $B$;
11:    Define $H$ to be the set of buyers with surplus $\delta$;
12:    repeat
13:        Let $H'$ be the set of neighbors of $H$ in $A$;
14:        Remove all edges from $B \backslash H$ to $H'$;
15:        $x \leftarrow 1$;
16:        Define $\forall j \in H$, price of $j$ to be $p_j x$;
17:        Raise x continuously until one of two events happens:
18:        if an edge $(i, j)$,$i \in H$,$j \in A \backslash H'$ attains equality, $\alpha_i = u_{ij}/p_j$) then
19:            Add $(i, j)$ to $G$; Recompute $f$;
20:            In the residual network corresponding to $f$ in $G$, define $I$ to be the set of buyers that
21:            can reach $H$, $H \leftarrow H \cup I$;
22:        end if
23:        if $S \subseteq H$ becomes tight then
24:            Move $(S, \Gamma(S))$ from $(H, H')$ to $(F, F')$;
25:            Remove all edges from $F'$ to $H$;
26:        end if
27:    until some subset $S \subseteq H$ is tight
28: until A is tight
```

The optimized algorithm and the basic algorithm are consistent in the initialization part. The two types of events are also similar, and the difference lies mainly in the key cycle. Next, the author proposes the effectiveness of the optimization algorithm through the **balanced flow**.

algorithm key

The key point of the optimized algorithm is to find the high-surplus buyers in the balanced flow and increase the prices of their goods.

Let f and f' be any two maximum flows in $\mathbf{N(p)}$. If $\gamma_i(\mathbf{p}, f') < \gamma_i(\mathbf{p}, f)$, there is a positive flow along the edge (i, t) in the flow $f' - f$. So, in the residual network, there is a positive flow from t to j and then to i. Then, there exist a $j \in B$ such that $\gamma_j(\mathbf{p}, f') < \gamma_j(\mathbf{p}, f)$. There is a path from j to i in $R(\mathbf{p}, f)\backslash\{s, t\}$ and a path from i to j in $R(\mathbf{p}, f')\backslash\{s, t\}$.

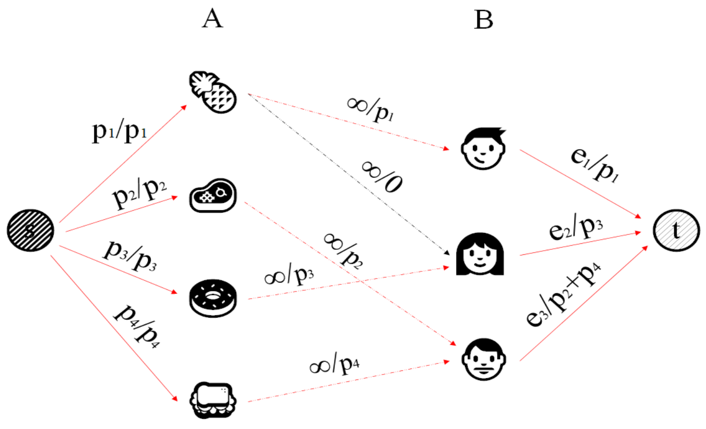

Figure 6: An example of the max-flow in the network sample

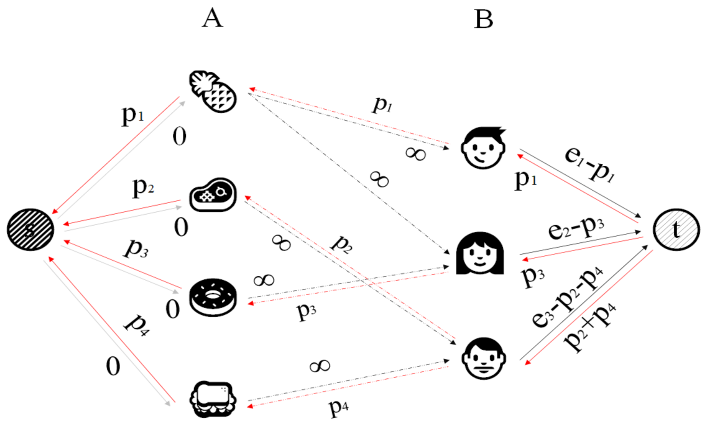

Figure 7: The residual network of the example max-flow

Then, the author proposed that a maximum-flow f is balanced iff there is no path from $i \in B$ to node $j \in B$ in $R(\mathbf{p}, f)$ if surplus of i is more than surplus of j in $N(\mathbf{p}, f)$. Otherwise, there will be another max-flow f' which is more balanced, $||\gamma(\mathbf{p}, f')|| < ||\gamma(\mathbf{p}, f)||$, leading to a contradiction.

Next, the author derives the upper bound on the $l_2 - norm$ of the surplus vector of buyers at the end of each phases. For a feasible and a balanced flow in $\mathbf{N(p)}$ and for some $i \in B, \delta > 0$, $\gamma_i(f) = \gamma_i(f^*) + \delta$, there will be a equation $||\gamma(\mathbf{p}, f)||^2 \geq ||\gamma(\mathbf{p}, f^*)||^2 + \delta^2$.

running time

The main idea of the Algorithm 2 is trying to reduce $||\gamma(\mathbf{p}, f)||$ in every phase. The method used by the author is finding a set of high-surplus buyers in the balanced flow and increasing the prices of goods in their basket.

First, let δ be the maximum surplus in B. At begin, obviously, $||\gamma(\mathbf{p})||^2 \leq M^2$. The number of iterations executed in a phase is at most n. Moreover, in every phase, there is an iteration in which surplus of at least one of the vertices is reduced by at least δ/n. Then, $||\gamma(p^*)||^2 \leq ||\gamma(p_0)||^2(1 - \frac{1}{n^3})$. After $O(n^2)$ phases, the value of $||\gamma(\mathbf{p})||^2 \leq 1/\Delta^2$. The iteration times of each phases is $O(n^2 \log(\Delta^4 M^2))$. And each iteration requires at most $O(n)$ max-flow computations.

After the above optimization, Algorithm 2 executes at most $O(n^4(\log n + n \log U + \log M))$ max-flow computations and finds market clearing prices.

Summary

The polynomial time market equilibrium model proposed in this paper mainly based on the ideas of maximum flow and minimum cut. The flow of the algorithm is roughly understood. What is not well understood is mainly a series of proofs of the algorithm optimized by high-surplus. For example, the proof of time complexity, why can we take the logarithm? The intuitive feeling when doing a project is that it is a difficult job to fully understand a paper. I tried to understand the main process and part of the proof of the model and it was quite rewarding. Thanks for your criticism and correction.

Reference

[1] Nikhil R Devanur, Christos H Papadimitriou, Amin Saberi, and Vijay V Vazirani. Market equilibrium via a primal-dual-type algorithm. In *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.*, pages 389–395. IEEE, 2002.

[2] Lester Randolph Ford and Delbert R Fulkerson. Maximal flow through a network. *Canadian journal of Mathematics*, 8:399–404, 1956.

[3] Herbert E Scarf and Terje Hansen. *The computation of economic equilibria.* Number 24. Yale University Press, 1973.