CS 333 – Algorithm Analysis

Spring 2020

Project Report

Group members: Can Yılankıran, Gökberk Aslantaş, Kıvanç Yılmaz,

Title: Clearing Algorithms for Barter Exchange Markets: Enabling Nationwide Kidney Exchanges

ABSTRACT

The problem addressed in this article is about the patients who have kidney disease, need treatment like kidney transplant, but there is a long waiting list for those kidneys and the demand for kidneys surpasses the supply. For instance, 4,052 people lost their lives in that waiting process in the United States in 2005. To overcome this problem, patients may choose to find a living donor who volunteers to denote her/his kidney. The issue with those donations, some of the potential donors and his/her kidney may be incompatible with the patient in terms of blood-type or tissue-type. This issue lowers the number of live donations, however these days patients with the kidney disease swaps their incompatible donors to obtain the kidney which is compatible for them. This exchange process can be named as a barter exchange, agents try to swap their items with each other,to increase their productivity. In our case, agents are patients and items are incompatible donors. These exchange processes include cycles of patients(agents), each of them obtaining the next incompatible donor(item) of the next patient in the cycle. Developing an optimized clearing algorithm for barter exchange markets to get a national kidney-exchange market, will reduce deaths which are caused by kidney disease, but in order to get that solving the clearing problem is necessary. Clearing problem can be explained as while the maximum length of a cycle is fixed, the goal is trying to find a social welfare which has maximum exchange. The reason that long cycles is prohibited, all transplant operations in the cycle must be done simultaneously. The main challenge appears when we try to build a national kidney exchange organization, we acknowledged that this clearing problem with cycle-length restriction is NP-hard. While constructing this algorithm the key is using incremental problem formulation. Adapting two models to the algorithm which are constraint generation and column generation for the task. To improve runtime and memory usage, developed methods are used for each model. After the tests, we obtained that column generation scales strongly better than constraint generation.

1. Introduction

Kidney failure is one of the most important diseases seen in our age. Kidney failure is a loss of function of the kidneys for various reasons. In addition to diseases like diabetes, hypertension, drugs and alcohol also result in kidney failure. There are two remedies for kidney failure. The first is that the patient enters the dialysis machine 2 or 3 times a week. Dialysis machine is a machine developed for patients suffering from kidney failure, which allows the dirty blood taken from the patient to be cleaned and returned to the patient. When this is said, even though this machine looks like an angel of goodness, it actually has many difficulties. One of the biggest challenges is that the patient spends a long time in the dialysis machine and repeats it several days a week.In fact many patients prefer to withdraw from dialysis because the quality of life on dialysis can be highly low, leading to a natural death. Only 12% of dialysis patients survive 10 years [6].

The second is to transfer a healthy kidney from a suitable donor to the sick person. Kidney transplants are by far the most common transplant. Unfortunately, the demand for kidneys far outstrips supply. For example in the United States in 2005, 4,052 people died waiting for a life-saving kidney transplant. During this time, almost 30,000 people were added to the national waiting list, while only 9,913 people left the list after receiving a deceased donor kidney. The waiting list currently has over 70,000 people, and the median waiting time ranges from 2 to 5 years, depending on blood type. This number has increased as we come to 2011. Kidney transplants can come from both deceased and living donors. Deceased donor kidneys are allocated to patients by means of a waiting list, which in the US currently contains 108,571 patients and has an average waiting time of 4 years [5].Unfortunately, the number of kidneys available for transplantation is still largely insufficient to meet demand: only about 17,000 US patients can receive a transplant each year [5].

For many patients with kidney disease, the best preference is to find a living donor, that is, a healthy person willing to donate one of his/her two kidneys. Although there are marketplaces for buying and selling living-donor kidneys, the commercialization of human organs is almost universally regarded as unethical, and the practice is often explicitly illegal, such as in the US. However, in most countries, live donation is legal, provided it occurs as a gift with no financial compensation. In 2005, there were 6,563 live donations in the US [1]. This number of donations was not enough at the time, and is still not enough today. In 2014, 17,107 kidney transplants took place in the US. Of these, 11,570 came from deceased donors and 5,537 came from living donors[7]. As seen in the examples, while the number of live donors should increase, unfortunately it has decreased over the years.

Now let's focus on barter exchange markets. Agents (patients) want to exchange their items (incompatible donors) with each other. These exchange processes include cycles of patients (agents), each of them obtaining the next incompatible donor (item) of the next patient in the cycle. These exchanges can be seen in every field of life. To give some examples there are some websites to trade books PaperBackSwap [2], and BookMooch[3]. Another website for trading is TradeStuff [4], which allows agents to swap items for their own desire.

While encoding a barter exchange market we used directed graph as $G = (V, E)$. For each agent construct one vertex. If v_x seek to get the item of v_y, add a weighted edge e from agent v_x to v_y. This weighted edge shows the benefit to v_x of getting the v_y's item. If there is a cycle c occur in the graph, that means a possible swap opportunity has been formed. In this cycle each agent gets the next agents' item. The sum of edge weights in the cycle c is represented as w_c. The collection of disjoint cycles is an exchange. The sum of its cycle weights is the weight of an exchange.

At figure 1 we showed an example of a market with 5 agents, $\{v_1, v_2, v_3, ..., v_5\}$, and all edges $\{e_1, e_2, e_3, ..., e_8\}$ have weight 1. This market example has 4 cycles, $c_1 = (v_1, v_2)$, $c_2 = (v_2, v_3)$, $c_1 = (v_3, v_4)$ and $c_4 = (v_1, v_2, v_3, v_4, v_5)$, and there is two maximal exchanges, which are $M_1 = \{c_4\}$ and $M_2 = \{c_1, c_3\}$. In this figure, M_1 includes the most edges.

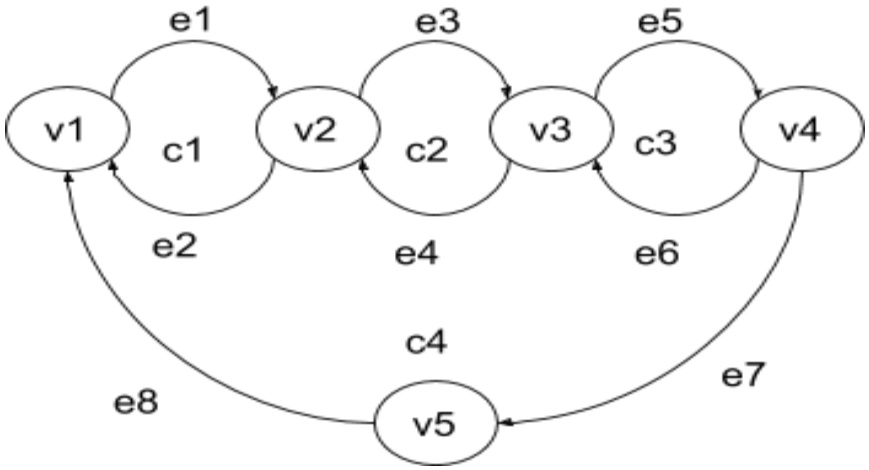

Figure 1: An example of barter exchange market.

Now let's move on to the clearing problem, to goal is finding a maximum-weight exchange consisting of cycles with length at most some small constant L. This cycle-length limitation increases for several reasons. In our case, in a kidney exchange, performing the operations in a cycle needs to be done simultaneously. That means a

donor might back down from after his incompatible partner has received a kidney. That's why these operations need to be done at the same time. Even if all the donors are operated at the same time, a k-cycle requires between $3k$ and $6k$ doctors, around $4k$ nurses, and $2k$ operating rooms.

This kind of resource limitation, this national kidney exchange market will allow only cycles of length 2 and 3. If we use these short cycles there are some advantages as well. When there is a failure fewer patients (agents) are affected. There can be a case like before the operation of the transplant, in the last-minute testing can reveal that new incompatibility that was not detected in the initial testing. To make it a more general expression, a patient (agent) might back down from out of a cycle when his/her preferences have changed. Considering the trading website which we gave as an example of a barter exchange market, if an agent forgets to send the item which is offered to be transferred to the other agent, that will drop him/her out of the cycle.

We will also show why is the clearing algorithm NP-complete for $L \geq 3$ Section 3. This algorithm is based on an integer-linear program(ILP) formulation, which is solved using specialized tree search, for 2 reasons.

1) First, if there is a loss of optimality that could lead to unnecessary patient deaths.

2) Second, an attractive feature of using an ILP formulation is that it allows one to easily model a number of variations on the objective, and to add additional constraints to the problem. For example, if 3-cycles are believed to be more likely to fail than 2-cycles, then one can simply give them a weight that is appropriately lower than 3/2 the weight of a 2-cycle. Or, if for various (e.g., ethical) reasons one requires a maximum cardinality exchange, one can at least in a second pass find the solution (out of all maximum cardinality solutions) that has the fewest 3-cycles. Other variations one can solve for include finding various forms of “fault tolerant” (non-disjoint) collections of cycles in the event that certain pairs that were thought to be compatible turn out to be incompatible after all.[1, p 2]

Further in this paper, we will see the details of this algorithm which is capable of clearing these markets. While constructing this algorithm the key is using incremental problem formulation. Adapting two models to the algorithm which are constraint generation and column generation for the task. To improve runtime and memory usage, developed methods are used for each model.

1.1 Paper Outline

Section 2 shows why the market clearing decision problem is NP-complete and proves that. Sections 3 and 4 each contain an ILP formulation of the clearing problem.

2. Background

Lets prove that the market clearing problem with short cycles is NP-complete. We need to give some explanation about NP-hard and NP to understand NP-complete problems.A problem is assigned to the NP class if it is solvable in polynomial time by a nondeterministic Turing machine[10].A problem H is NP-hard when every problem L in NP can be reduced in polynomial time to H.Let's assume that problem H's solutions takes 1 unit time, H's solution can be used to solve L in polynomial time[8]. Now we can define what NP-complete problems means. If a problem is both in NP and NP-hard, this problem is NP-complete[9]. In D. J. Abraham, A. Blum, and T. Sandholm [1, p 4] used this theorem.

Theorem 1. Given a graph $G = (V, E)$ and a integer $L \geq 3$, the problem of deciding if G admits a perfect cycle cover containing cycles of length at most L is NP-complete.

Proof. They claimed that this problem is clearly in NP. When it comes to proving this problem is NP-hard, they tried to reduce from 3D-Matching, which is a NP-complete problem of, given disjoint sets X, Y , Z of size q , and a set of triples $T \subseteq X \times Y \times Z$, deciding if there is a disjoint subset M of T size q .

The first idea was to construct a tripartite graph with vertex sets $X \cup Y \cup Z$ and directed edges $(x_a, y_b), (y_b, z_c)$ and (z_c, x_a) for each triple $t_i = \{x_a, y_b, z_c\} \varepsilon T$. But they realized that this encoding fails because a perfect cover might include a cycle without corresponding triple.

Then they decided to use the following reduction. Given an instance of 3D-Matching, add one vertex for each element in X, Y and Z. For each triple, $t_i = \{x_a, y_b, z_c\} \varepsilon T$ gadget in Figure 2. They showed that the gadgets intersect only on vertices $X \cup Y \cup Z$. This construction can be done in polynomial time.

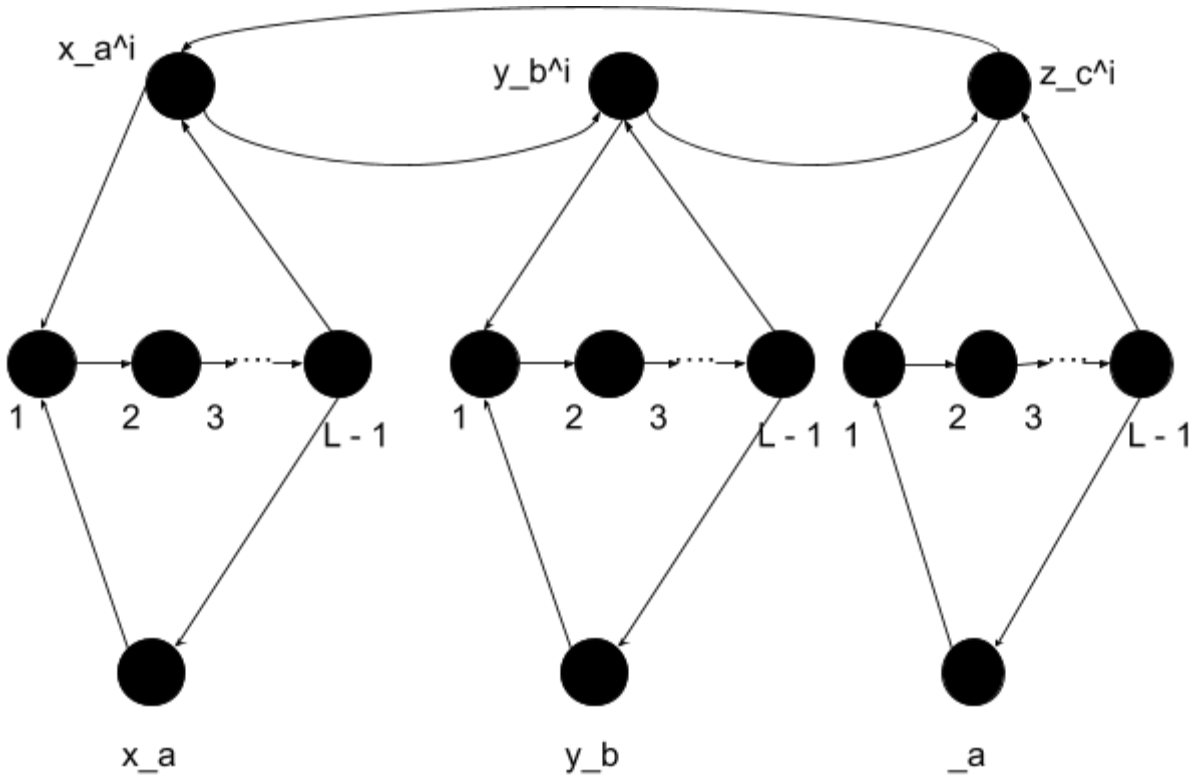

Figure 2 : NP-completeness gadget for triple t_i and maximum cycle length L.

3. SOLUTION BASED ON AN EDGE FORMULATION

In this section to get a solution, we used ILP (integer linear programming) formulation of the clearing problem with one variable for each edge. As we did at section 1, let assume that given a market $G = (V, E)$, now our goal is to get a perfect matching in a bipartite graph. To achieve that we are adding an edge between the patient (agent) and with his own incompatible donor (item). Now we have got a perfect matching. Let's add edge e with w_e between agent v_i and the item which he/she seeks to obtain. We can call it as v_j's item. When v_j's item is taken by v_i, v_j need to obtain another patients' (agents) item. Solving his unlimited clearing problem is possible now by finding a maximum-weight perfect matching. Figure 3 is the bipartite graph encoding of Figure 1. At Figure 3 the dashed lines illustrate edges with zero weight. Bold edges indicates the swapping process.

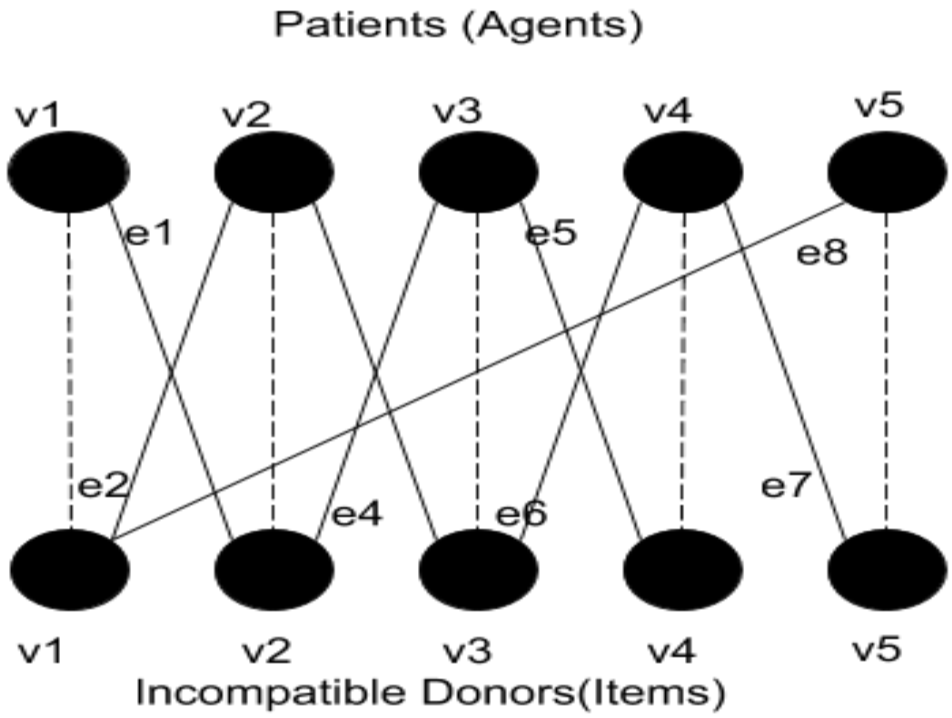

Figure 3 : Perfect matching encoding of the market in Figure 1.

There is an alternative way to solve the problem which is encoding it as an ILP like we did at first but we are going to use original market graph G, to deal with cycle length limitations.

$$max \sum_{e \varepsilon E} w_e e$$

When we consider the process of the swapping items between agents, the edges going outside from each agent to the other agent needs to be equal to edges which are coming in from the other agent.

$$\sum_{e_{out}=(v_i,v_j)} e_{out} - \sum_{e_{in}=(v_j,v_i)} e_{in} = 0$$

And an agent can't do 2 swapping in our case.

$$\sum_{e_{out}=(v_i,v_j)} e_{out} \leq 1$$

When the cycles are allowed to have length at most T, adding constraint $e_{p1} + e_{p2} + ... + e_{pL} \leq L - 1$, for each length-L path $p = \left(e_{p1}, e_{p2}, ... , e_{pL}\right)$ will do the work. But this approach fails, in a market of only 1000 patients, the number of length-3 paths exceeds 400 million, that means this ILP formulation cannot even be constructed without running out of memory.

//TREE SEARCH EXPLANATİON

3.1 Constraint Seeder

// Constraint Seeder explanation

3.2 Constraint Generation

// Constraint Generation explanation

4. SOLUTION BASED ON A CYCLE FORMULATION

In this section we are using ILP formulation as we did in section 3, but the only difference is instead of edge we will use cycles. Like we did at section 3, consider a market $G = (V, E)$, we only take cycle c of length 2 and using them to make a graph. Therefore, the market clearing problem with $L =$ 2 can be solved in polynomial time by finding a maximum-weight matching.

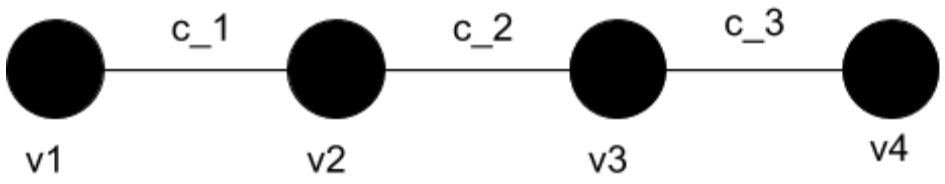

Figure 4 : Maximum weight matching encoding of the Figure 1.

With an arbitrary L to generalize this encoding, let $C(L)$ be the set of all cycles of G with length at most L, this ILP finds the maximum-weight cycle in the $C(L)$.

$$max \sum_{c \varepsilon C(L)} w_c c$$

$$subject\ to \sum_{c: v_i \varepsilon c} c \leq 1 \quad \forall v_i \varepsilon V$$

$$with\ c\ \varepsilon\ \{0,1\} \quad \forall c\ \varepsilon\ C(L)$$

4.1 Column Generation for the LP

A corresponding LP with $L = 2$ in figure 1.

$$\begin{aligned} max \quad & 2c_1 + 2c_2 + 2c_3 \\ subject\ to \quad & c_1 \leq 1 \\ & c_1 + c_2 \leq 1 \\ & + c_2 + c_3 \leq 1 \\ & c_3 \leq 1 \\ with \quad & c_1, c_2, c_3 \geq 0 \end{aligned}$$

Figure 5: LP formulation // I will solve this LP in following days

5. Discussion

Analysis about runtime, correctness and any comparisons will be in this section. Write the main proofs in your own words and show that you really understand the content. If you did an implementation and made an experimental comparison of runtimes, it should appear in this section.

6. Conclusion and Future Work

Briefly summarize the main aim of the paper and list any open questions and possible future work.

7. References

[1]D. Abraham, A. Blum, and T. Sandholm. Clearing algorithms for barter exchange markets: enabling nationwide kidney exchanges. ACM EC, (07), 2007.
[2] PaperBackSwap. https://www.paperbackswap.com/index.php
[3] BookMooch. http://bookmooch.com/
[4] TradeStuff. https://www.tradestuff.com/
[5] SRTR. https://www.srtr.org/
[6] United States Renal Data System (USRDS). http://www.usrds.org/
[7] HRSA. http://optn.transplant.hrsa.gov/
[8] NP-hard https://en.wikipedia.org/wiki/NP-hardness
[9] NP-complete https://en.wikipedia.org/wiki/NP-completeness
[10]NP-problem https://mathworld.wolfram.com/NP-Problem.html