OptOrder: optimum serial dictatorship in stochastic matching

Someone

Abstract

This paper studies a simplified version of the stochastic matching proposed in [Chen *et al.*, 2009]. Previous study want to find the optimal policy for a matching system to maximize the cardinality of a matching. In that setting, an agent may reject the assigned candidate (with a certain probability) and stay in the market or leave. Finding the optimum policy to match meets great difficulties mainly from two aspects. First, there is no evidence that the optimal policy can be expressed in polynomial space. Second, collecting the acceptance ratio of each pair usually incurs considerable costs.

To express a policy efficiently and solve the optimum solution efficiently, we choose serial dictatorship and consider the case where the acceptance ratios are identical for each pair. This paper aims to give an algorithm with a graph and the acceptance ratio as input and outputs a optimum order for serial dictatorship. We first describe a straightforward integer linear programming with $O(|E|^4)$ entries. Then, we reduce the number of entries to $O(|E|^2)$ by an important observation. Finally, in the experimental part, we compare methods to solve even larger graphs.

1 Introduction

Motivated by applications like kidney exchange and online dating, [Chen *et al.*, 2009] defines stochastic matching. The goal to maximize the cardinality of a stochastic matching. The problem can be modeled in a graph. Any edge has a probability to be fake. (In our work, we assume that these probabilities are identical.) When a fake edge is probed to match, it rejects the prob and the edge is removed from the graph. Otherwise, it accepts the prob and both ends of the edge are removed from the graph. They show a greedy algorithm with 0.25 approximation compared to the optimal algorithm. The approximation ratio was improved by [Adamczyk, 2011; Costello *et al.*, 2012]. When compared to the offline optimal, this problem is closely related to the randomized greedy matching problem. [Poloczek and Szegedy, 2012; Aronson *et al.*, 1995] show a greedy algorithm that can match more than one half agents and [Goel and Tripathi, 2012] improve the lower bound significantly.

The current setting faces two main drawbacks to prevent it from practical use.

First, to know the exact probability of a prob being accepted is difficult. Typically, in order to know the exact probabilities, the market needs to collect many data and conduct careful analyses on historical data. It is worthy in important markets like kidney exchange market [Dickerson *et al.*, 2013; Dickerson and Sandholm, 2015]. However, in many less important market, collecting acceptance ratios is quite costly and unworthy. Similar to online dating problem, let us consider a friend recommendation problem. Someone (called Alice) has many friends, x of them are single boys and y are single girls. Alice wants to introduce boys to girls for marriage. She wants to maximize the number of matched couples. Every boy or girl has some requirements that must be satisfied. However, even though a boy and a girl satisfy each other's requirements, they may still not be willing to match. In this case, it is hard and costly for Alice to survey the acceptance ratio of each pair. Similar problem also appear in other applications like roommate problem [Roth, 1982], but never serious considered as far as we know.

Second, as mentioned above, most study focus on giving worst case guarantees, compared both to the optimal solution and the offline optimum. The main difficulty preventing people to desire the optimum solution is that the optimal solution maybe not able to be expressed in polynomial space. To solve the stochastic matching problem in practical, there is no evidence that the existing algorithms can work well enough.

In this paper, a simplified setting of stochastic matching is considered, which can still be utilized to solve many real life applications. We assume that all the edges have the same acceptance ratio. This assumption avoid the difficulty in surveying the acceptance probabilities and provides an essential condition for our algorithm. As for the expressiveness of the solution, we improve it by serial dictatorship. Although, serial dictatorship may reduce the expected number of matched agents, our experimental results demonstrate that the influence is not big. With the solution, anyone can conduct the matching without any further help of a computer. One only needs $O(1)$ computation in average for each prob. We are going to release our work as an open source tool. With the help of this tool, anyone is able make precise decisions for stochastic matching problems around themselves.

The following part of this paper is organized as follows. Section 2 defines the serial dictatorship and show some fundamental good properties of it. Section 3 shows the construction of the integer linear programming step by step and prove its correctness. Section 4 explores methods to expand our algorithm to larger graphs. Section 5 shows our experimental results. Section 6 concludes this paper.

2 The settings

In this section, we will describe our model and the serial dictatorship. After that, some desired properties of serial dictatorship are demonstrated.

2.1 The model

Our problem is modeled in an undirected graph $G = (V, E)$, $V = \{v_1, v_2, \dots, v_n\}$ denotes the set of agents. $e_{ij} \in E$ denotes an edge between v_i and v_j. For any edge $e_{ij} \in E$, it has two states, *exist* or *fake*. The probability of "exist" is a constant p. The goal of the matching is to maximize cardinality. Each time, the system prob a pair of agents v_i and v_j with an edge e_{ij} connecting them.

- If e_{ij} exists, then v_i and v_j agree to match with each other, both of the two agents and the edges attached to them are removed from the graph. The number of matched agents increases by 2.
- If e_{ij} is fake, v_i and v_j refuses to match. The system removes e_{ij} from E.

2.2 ESD algorithm

To maximize the number of matched agents, we apply *edge serial dictatorship algorithm* (ESD). The outline is described In Algorithm 1.

Algorithm 1 ESD algorithm

Require: A sorted array of E, denoted by $S = s_1, s_2, \dots, s_m$ and the set of vertices V.
Ensure: A set of disjoint edges A, such that each two agents connected by an edge in A agree to match.

```
A = ∅
for do i = 1, 2, ..., m
    Let v_x and v_y be the two agents connected by s_i
    if then v_x ∈ V and v_y ∈ V
        Prob v_x and v_y whether they agree to match with each other.
        if B then oth v_x and v_y accept to match
            Put s_i into A
            Remove v_x and v_y from V
        end if
    end if
end for
Outputs A as the set of matched edges.
```

Algorithm 1 is a typical serial dictator algorithm. The system just picks edges one by one in some order (denoted by S) and try to match the current edge if both ends of the edge are still in the graph. Our goal is to maximize the expected size of the output set A among all S.

2.3 Desired properties

We restrict our focus on the serial dictatorship for many reasons.

Usability. The solution of ESP has a size of $O(|E|)$, which equals to the size of input. The executor of the matching only needs to use O(1) computation for each prob. These two properties make the solution perfect to be used as handouts. In other words, the order can be generated by a computer and any people can execute the matching without a computer.

Deterministic. A great advantage of ESD is that it outputs a deterministic solution. One can count the expected number of matched agents easily. Many previous algorithms lack this property. In our setting, given the serial S, the expected number of matched agents $c(S)$ can be counted by the following equation.

$$c(S) = 2 * \sum_{i=1}^{m} p(1-p)^{\sum_{j=1}^{i-1} \delta(s_j, s_i)} \tag{1}$$

In equation 1, $\delta(s_i, s_j)$ is an indicator of whether s_j appears before s_i and they have a common vertex. In other words, $\delta(s_j, s_i) = 1$ only when s_j is probed before s_i and they have a common vertex, otherwise $\delta(s_j, s_i) = 0$. In the integer linear programming, we will use variables δ_{ij}, which holds the same meaning as $\delta(s_i, s_j)$ here.

3 Optimizing the serial

Algorithm 1 has given a framework of serial dictatorship. The problem is how to find the serial S that maximizing the expected cardinality of the matching. In this section, we will introduce the integer linear program (ILP) we used to find the optimum S step by step.

3.1 Overview of the algorithm

The objective function of our ILP is a representation of Equation 1. As for the constraints, if we faithfully write the constrains according to definition of serial dictatorship and the objective function, we need $O(|E|^4)$ variables in the ILP, which is insufferable. We are not going to discuss in details for this idea.

To reduce the number of variables, we introduce a novel method. We first relax the solution space to obtain a relaxed-form ILP. Then, we show that the solution of the relaxed-form ILP can be mapped to the optimal serial S.

The relaxed-form ILP is based on the following observation. Given the optimal serial S, there is a determined order between any two edge. For example, for $e_{ij}, e_{ik} \in E$, given S, we know that e_{ij} is always probed before e_{ik} or e_{ik} is always before e_{ij}. Now, we are ready to relax the solution space. Rather than a total order, we only assume that there is a determined order between any two edges with a common vertex. We do not have any other constraints on the solution space. Then, we will introduce a method that maps the optimal solution of the relaxed-form ILP to the optimal serial S in ESD algorithm. The relaxed-form ILP only has $O(|E||V|)$ variables and $O(|E||V|)$ entries in total. So, it improves the performance significantly.

3.2 Variable Definition

Three types of variables are used in our ILP[1]. For a better presentation of our ILP, we give an alias for each element in E. We number all the edges in E as $e^1, e^2, \ldots, e^m$. This notation is not conflict with the previous definition. An edge e^i may denote the same meaning as e_{jk}. Now, we are ready to define the variables in our setting.

- $\delta_{ij}(i, j \in \{1, 2, \ldots, m\})$ denotes an indicator of e_i is probed before e_j and they have a common vertex (just as mentioned before). If both conditions are satisfied $\delta_{ij} = 1$, otherwise $\delta_{ij} = 0$. The value of δ_{ij} is selected from $\{0, 1\}$.
- $o_i(i \in \{1, 2, \ldots, m\})$ denotes the number of edges that are probed before e_i and have a common vertex as e_i. By definition, we have $o_i = \sum_{j=1}^{m} \delta_{ji}$. This equation is also used as a constraint in the ILP. The value of o_i is selected from nonnegative integers
- $l_{ij}(i, j \in \{1, 2, \ldots, m\})$ denotes an indicator for whether o_i is greater than or equal to j. This variable is used to express the objective function of ILP. Its value is selected from $\{0, 1\}$.

3.3 Objective function

The objective of the ILP is to maximize to total number of cardinality, which has been shown in Equation 1. To express it in the ILP fashion, we write it as follows. The index "2" is omitted, thus the objective function becomes the expected number of matched edges.

$$
\begin{aligned}
obj &= \sum_{i=1}^{m} p * (1-p)^{o_i} && (2)\\
&= p * \sum_{i=1}^{m}((1-p)^{o_i} - (1-p)^{o_i - 1} + (1-p)^{o_i-1} - \\
&\quad (1-p)^{o_i-2} + \ldots - (1-p) + (1-p) - 1 + 1) && (3)\\
&= mp + p * \sum_{i=1}^{m}\sum_{j=1}^{o_i}((1-p)^j - (1-p)^{j-1}) && (4)\\
&= mp + p * \sum_{i=1}^{m}\sum_{j=1}^{m}((1-p)^j - (1-p)^{j-1})l_{ij} && (5)
\end{aligned}
$$

In equation 5, the objective function becomes a linear function, such that it can be used in the ILP. The current problem is how to make l_{ij} an indicator for $o_i \geq j$.

As stated before, the solution of the ILP only make sense when the objective function obj has been maximized. So, we only need to add the following constraint to restrict l_{ij}.

$$o_i \leq j - 1 + l_{ij} * |E| \quad (6)$$

In our ILP, l_{ij} only appears in the constraint above and the objective function.

Lemma 1 *When obj has been maximized, $o_i \geq j$ if and only if $l_{ij} = 1$.*

[1]From here on, all the "ILP" refers to the relaxed-form ILP.

Proof: To prove this lemma, we only need to show that the following two cases are impossible when obj is maximized.

- $o_i < j$ and $l_{ij} = 1$.
- $o_i > j$ and $l_{ij} = 0$.

The second case is obviously impossible by Equation 6.

If the first case is satisfied and obj is maximized, set l_{ij} to be 0 will increase the obj and all the constraints are still satisfied. A contradiction. ■

3.4 The ILP

Now we are ready to conclude the ILP we used to solve the optimal serial as follows.

$$
\begin{aligned}
\text{Maximize:} \quad & mp + p * \sum_{i=1}^{m}\sum_{j=1}^{n_i}((1-p)^j - (1-p)^{j-1})l_{ij} && (7)\\
\text{Subject to:} \quad & 1.\ \delta_{ij} + \delta_{ji} = 1, \forall e_i, e_j \in E, e_i \cap e_j \neq \emptyset && (8)\\
& 2.\ o_i \leq j - 1 + l_{ij} * |E|, i = 1, 2, \ldots, m, j = 1, 2, \ldots, && (9)\\
& 3.\ o_i = \sum_{e_j \cap e_i \neq \emptyset, i \neq j} \delta_{ji}, \forall i = 1, 2, \ldots, m && (10)
\end{aligned}
$$

$e_i \cap e_j \neq \emptyset$ denotes that e_i and e_j have a vertex in common. In equation 9 and the objective function, n_i denotes the number of edges that have a common vertex as v_i. Introducing n_i decreases the number of variables we used. Further, we omitted the δ_{ij}'s for disjoint e_i and e_j. These two steps ensure the number of entries to be $O(|E||V|)$. We will discuss the complexity problem later.

3.5 Mapping to ESD

We note that when the objective function is not maximized, the values of variables correspond to nothing useful for serial dictatorship. However, when the objective function, things become different. Now, we are going to show how to map the solution of the ILP to the optimal sequence S for ESD. The method is as follows.

Mapping process: First, we define a function o satisfying $o(e_i) = o_i^*$. Note that we reuse the o for the same meaning. o_i^* denotes the value of o_i when the objective function in the ILP has been maximized. Then, we sort the E to be a sequence $S^* = s_1^*, s_2^*, \ldots, s_m^*$, satisfying $\forall i, j \in \{1, 2, \ldots, m\}$, $o(s_i) < o(s_j)$. The S is the optimal sequence for ESP.

3.6 A summary of our algorithm for optimum serial dictatorship

For a better understanding, our algorithm for optimum serial dictatorship is shown in Algorithm 2. The overall idea is to solve the ILP, map to a sequence and run ESD algorithm.

3.7 The correctness proof

Recall the definition of o_i, we just sort all the edges according to the number of neighboring edges probed before the edge. For short, we call "the number of neighboring edges probed before the edge" as the *order* of the edge. Now, we are going to demonstrate that S^* is the optimal sequence. The

Algorithm 2 Optimum serial dictatorship

Require: A graph $G = (V, E)$

Ensure: A set of disjoint edges A, such that each two agents connected by an edge in A agree to match and the expected size of A is maximized among solutions from edge serial dictatorship.

1: Run the ILP shown in Equations 7-10 on graph G. The value profile for the optimum solution is denoted by P.
2: Map P to a sequence S by mapping process.
3: Run Algorithm 1 on S, outputs the matched edges A as output.

rough idea is that the optimal solution of ILP is always implementable by ESD. The reason is that an edge always probed earlier than one with higher order. Otherwise, let lower-order edge be probed first can increase the value of the objective function.

Formally, it is stated as follows.

Theorem 1 *The output sequence S^* of our algorithm is the sequence that maximizes the cardinality in the ESD algorithm.*

Proof: The proof of this theorem consists of two parts. Case (1): We show that any sequence of edges corresponds to a point in the ILP's solution space. This property indicates that the maximum value of the ILP is no smaller than the maximum expected number of matched edges. Case (2): We show that the sequence S^* constructed by the mapping process corresponds to the optimal solution of the ILP. In other words, the expected matched edges of S^* is the same as the maximum value of the objective function. It indicates that the maximum value of the ILP is no bigger than the maximum expected number of matched edges.

Case (1): Given a sequence of edges $S = s_1, s_2, \dots, s_m$, we can construct the variables in the ILP as follows. For any $i < j$ and e_i and e_j have a common edge. $\delta_{ij} = 1$, otherwise $\delta_{ij} = 0$. Let o_i equal to $\sum_{j=1}^{m} \delta_{ji}$. $l_{ij} = 1$ if and only if $o_i \geq j$. We call this process as *reverse mapping process*. In this way, we have map S to a point in the solution space of the ILP. Further, the value of objective function under this construction is exactly the expected number of matched edges. So we conclude the maximum objective function value is greater than or equal to the maximum expected number of matched edges.

Case (2): We are going to show that S^* corresponds to the optimum solution of the ILP by reverse mapping. To prove this, we only need to prove that the values of all the variables are the same for 1. the optimum ILP solution where S^* is mapped from and 2. the variables generated by reverse mapping from S^*. Then, we show that we only need to compare the values of δ_{ij}'s.

- In the optimal solution, according to Lemma 1, l_{ij}'s are decided by o_i's. Further, o_i's are decided by δ_{ij}'s according to the equation 10. So, when δ_{ij}'s are fixed in the optimal solution, the values of all the other variables are fixed.
- For the variables generated by reverse mapping from S^*, l_{ij}'s, o_i's and δ_{ij}'s follow the same relationship as the former case. In details, $l_{ij} = 1$ if and only if $o_i \geq j$, and $o_i = \sum_{j=1}^{m} \delta_{ji}$. So if the two cases have the same δ_{ij}'s, all the variables are the same.

To prove that the two cases have the same δ_{ij}'s, we need the following lemma.

Lemma 2 *In the optimum solution of ILP, if $o_j \leq o_i$, $\delta_{ij} = 0$*

Proof: We prove this lemma by contradiction. By contradiction, we assume that in the optimum solution, there exist i and j satisfying $o_j \leq o_i$ and $\delta_{ij} = 1$. By Lemma 1 and Equations 2 to 5, in the optimum solution, the Equation 2 is equivalent to the objective function of the ILP. So, in the optimum solution, the value of the objective function is as follows.

$$obj_1 = \sum_{i=1}^{m} p(1-p)^{o_i} \tag{11}$$

Then, let $\delta_{ij} = 0$ and $\delta_{ji} = 1$ and keep the other δ's unchanged. We call the new values of o_i and o_j to be o'_i and o'_j. We have $o'_j = o_j - 1$ and $o'_i = o_i + 1$. Now, we consider the point in the feasible space where $l_{ij} = 1$ iff $o_i \geq j$. The value of the objective function at this point is as follows.

$$\begin{aligned} obj_2 \quad = \quad & obj_1 + p(1-p)^{o_j - 1} + p(1-p)^{o_i + 1} \\ & -p(1-p)^{o_j} - p(1-p)^{o_i} \end{aligned} \tag{12}$$

Thus, we have:

$$\begin{aligned} obj_2 - obj_1 \quad = \quad & p(1-p)^{o_j - 1} + p(1-p)^{o_i + 1} \\ & -p(1-p)^{o_j} - p(1-p)^{o_i} \\ = \quad & p(1-p)^{o_j - 1}(1 + (1-p)^{o_i - o_j + 2} \\ & -(1-p) - (1-p)^{o_i - o_j + 1}) \\ > \quad & 0 \end{aligned}$$

So, we find a feasible point whose objective function's value is larger than optimum. A contradiction. ■ Now, we consider a pair of edges e_i and e_j.

- If $\delta_{ij} = 1$, by Lemma 2, $o_i < o_j$. Then, in S^*, e_i is ahead of e_j. Then, after reverse mapping, as e_i and e_j have a common vertex and e_i is ahead of e_j, $\delta_{ij} = 1$.
- If e_i and e_j are disjoint, after mapping and reverse mapping, $\delta_{ij} = 0$
- If $\delta_{ij} = 0$ but e_i and e_j have a common vertex, e_j is ahead of e_i in S^*. After reverse mapping, δ_{ij} still equals to 0.

So, all the δ_{ij}'s are the same between the optimum solution where S^* is generated from and the feasible point S^* corresponds to in the ILP. Above all, we have finished the proof for the second case. ■

3.8 Complexity analysis

Even though solving ILP is an NP-Hard problem, instances at a small scale is solvable by many ILP solvers. We care about the number of entries in our ILP. By direct observation, we find the number of entries can be bounded by $O(|E|^2)$. However, with carefully analysis, we reduce it to $O(|V||E|)$.

Theorem 2 *The ILP only has $O(|V||E|)$ entries.*

Proof: In the ILP, each δ_{ij}, o_i or l_{ij} appears no more than 2 times. So, we only need to show that the number of variables is $O(|V||E|)$.

- The number of o_i's is $|E|$.
- For each pair of neighboring edges e_i and e_j, there are two corresponding variables δ_{ij} and δ_{ji}. The total number of pairs of neighboring edges can be counted as follows. For an arbitrary edge e_i if e_j and e_j has a common vertex. The common vertex has two alternatives and the other end of e_j has $|E|-2$ alternatives. So, the number of δ's is $O(|V||E|)$.
- The number of l_{ij}'s is exactly the sum of n_i's. Each pair of neighboring edges increases $\sum_{i=1}^{m} n_i$ by 2. So, the number of l_{ij}'s is $O(|V||E|)$.

■

4 Extensions of our algorithm

Our algorithm can perform well on small graphs. However, for larger graphs, ILP is too slow. As our algorithm is the fastest way to get the optimum sequence for serial dictatorship to the best of our knowledge, we need find methods to approximate the optimum sequence for larger graphs.

Further, ESD is a non-adaptive algorithm. In other words, during the executing of ESD algorithm, the order stays unchanged without using considering the results of the previous probings. However, based on the ILP, we can also extend the algorithm to an adaptive algorithm.

4.1 Force mapping

When the size of the graph is big enough, the ILP can not be solved in an acceptable time limit. However, our mapping process can always get a feasible solution.

The rough idea of the force mapping is as follows. We set a maximum acceptable time T. Terminate the ILP at time T. At that time, the *variable profile* (the values of all the variables) is denoted by P and the corresponding objective value is denoted by obj. Then, map P to a sequence S by the mapping process. S is used as the sequence for ESD algorithm.

4.2 Improvements on optimum serial dictatorship

Actually, the payoff of Algorithm 2 can be further improved. However, such improvements breaks some great properties ESD holds, such as offline executable and deterministic. At the same time, these improvements induce more computation.

Adaptive ESD

To improve Algorithm 2 on payoff, we can use adaptive ESD. The rough idea is that only use Algorithm 2 to decide the next k probed edges, and repeat this process. It is formally stated in Algorithm 3.

When k is small, the algorithm 3 tends to output a better solution but costs more time to compute.

Theorem 3 *For any positive integer k, the payoff of Algorithm 3 is same as or better than the payoff of Algorithm 2*

Algorithm 3 Optimum serial dictatorship

Require: A graph $G=(V,E)$ and a positive integer k.
Ensure: A set of disjoint edges A, such that every edge in A exists.

$A=\emptyset$
while G is non-empty **do**
  Run the ILP shown in Equations 7-10 on graph G. The variable profile for the optimum solution is denoted by P.
  Map P to a sequence $S=s_1,s_2,\ldots$ by mapping process.
  for $i=1,2,\ldots,min(k,length(S))$ **do**
    if Both ends of s_i are still in G **then**
      Prob s_i to match
      if s_i is proved to be exist. **then**
        Remove the two ends of s_i and all edges attached to them from G.
        Add s_i to A
      else
        Remove s_i from G
      end if
    end if
  end for
end while
Output A

The rough idea of the proof is as follows. We prove this by induction.

The drawback of 3 is that it needs interaction with computer. While Algorithm 2 can be totally offline with a more clear scheme.

Skip ratio

[Bansal *et al.*, 2012] gets some bounds on weighted stochastic matching. A technique they used to get bounds is adding a skip rate when matching following a sequence.

we can also add a skip ratio $r(0 \le r < 1)$ to algorithm 2. When running the ESD algorithm, each time, we skip each edge with probability r. In this way, it is possible to get a higher payoff.

If we directly add r to our ILP, it will incurs great difficulty for solving it. So, we only add the r on the optimum sequence. Then the value of the payoff on a sequence S is as follows.

$$obj(s)=\sum_{i=1}^{m}(1-r)p(1-(1-r)p)^{\sum_{j=1}^{i}\delta(s_j,s_j)} \tag{13}$$

In practice, this methods only have little effect on the payoff. Further, this method is not deterministic.

References

[Adamczyk, 2011] Marek Adamczyk. Improved analysis of the greedy algorithm for stochastic matching. *Information Processing Letters*, 111(15):731–737, 2011.

[Aronson *et al.*, 1995] Jonathan Aronson, Martin Dyer, Alan Frieze, and Stephen Suen. Randomized greedy matching. ii. *Random Structures & Algorithms*, 6(1):55–73, 1995.

[Bansal *et al.*, 2012] Nikhil Bansal, Anupam Gupta, Jian Li, Julián Mestre, Viswanath Nagarajan, and Atri Rudra. When lp is the cure for your matching woes: Improved bounds for stochastic matchings. *Algorithmica*, 63(4):733–762, 2012.

[Chen *et al.*, 2009] Ning Chen, Nicole Immorlica, Anna R Karlin, Mohammad Mahdian, and Atri Rudra. Approximating matches made in heaven. *Automata, Languages and Programming*, pages 266–278, 2009.

[Costello *et al.*, 2012] Kevin P Costello, Prasad Tetali, and Pushkar Tripathi. Stochastic matching with commitment. In *Automata, Languages, and Programming*, pages 822–833. Springer, 2012.

[Dickerson and Sandholm, 2015] John P Dickerson and Tuomas Sandholm. Futurematch: Combining human value judgments and machine learning to match in dynamic environments. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2015.

[Dickerson *et al.*, 2013] John P Dickerson, Ariel D Procaccia, and Tuomas Sandholm. Failure-aware kidney exchange. In *Proceedings of the fourteenth ACM conference on Electronic commerce*, pages 323–340. ACM, 2013.

[Goel and Tripathi, 2012] Gagan Goel and Pushkar Tripathi. Matching with our eyes closed. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 718–727. IEEE, 2012.

[Poloczek and Szegedy, 2012] Matthias Poloczek and Mario Szegedy. Randomized greedy algorithms for the maximum matching problem with new analysis. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 708–717. IEEE, 2012.

[Roth, 1982] Alvin E Roth. Incentive compatibility in a market with indivisible goods. *Economics letters*, 9(2):127–132, 1982.