OptOrder: optimum serial dictatorship in stochastic matching

Someone

Abstract

This paper studies a simplified version of 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 difficulty 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. 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]. [Goel and Tripathi, 2012] [Feldman *et al.*, 2009]

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 setting

Our problem can be modeled in an undirected graph $G = (V, E)$, $V = \{v_1, v_2, \ldots, v_n\}$ denotes the set of agents. $e_{ij} \in E$ denotes an edge between v_i and v_j. 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. (1) If v_i and v_j accept to match with each other with p probability, both of the two agents are removed from the market. (2) If v_i and v_j do not accept with $1 - p$ probability, the system knows that they cannot match and removes e_{ij} from E.

3 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 $S = s_1, s_2, \ldots, s_m$ 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 doi = 1, 2, ..., m
    Let v_x and v_y be the two agents connected by s_i
    if thenv_x ∈ V and v_y ∈ V
        Prob v_x and v_y whether they agree to match with each other.
        if B thenoth 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 quite straightforward. The system just picks edges one by one in some order (denoted by S) and try to match the current edge. Our goal is to maximize the expected size of the output set A.

We use ESD for the following reasons.

1. 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. This property makes ESP quite practical. For example, when matching roommates among a class of students, the teacher can input a graph showing the compatibilities between students and an estimated acceptance ratio into the program. The program outputs an order of probs. The teacher can just follow the order to query pairs of students.
2. The optimal solution can be solved by Optimization 1 in small graphs and can be approximated in larger graphs.

In Algorithm 1, The generating of array S is not mentioned. Even though we find the S that can maximize the expected size of A, the current matching policy may not be optimal. However, ESD has the following advantages that motivates us to find a good policy based on it.

The most important advantage is that given S the expected size of A can be solved. It is not natural for other policies for optimizing stochastic matching. The expected cardinality of the matching can be solved as follows.

$$E_S(|A|) = 2\sum_{i=1}^{m} p * \prod_{j=1}^{i-1}(1-p)^{\delta(s_i,s_j)} \tag{1}$$

$\delta(s_i, s_j)$ is an indicator, which equals to 1 if s_i and s_j are disjoint, otherwise equals to 0.

3.1 Optimizing the order

A natural question is how to decide S to maximize cardinality. The most straightforward way is by enumerating all the permutations and find one permutation with the best payoff (the expected number of matched agents). In this way, the computational complexity is $O(|E|! * |E|^2)$. It can only handle a graph with up to 12 or 13 edges.

To solve the optimal order of edges for lager graphs, we model it as an integer linear program (ILP). With the ILP we proposed, we can easily handle a graph with more than 20 edges. For larger cases, we can approximate the optimal payoff by restricting the maximum iteration of the ILP.

The ILP problem is listed as Optimization 1. Renumber all the elements in E as $e_1, e_2, \ldots, e_m$.

x_{ij} is the indicator for whether s_i is the j-th edge in .

Optimization 1

$$\text{Maximize} p_i \tag{2}$$

References

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

[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.

[Feldman *et al.*, 2009] Jon Feldman, Aranyak Mehta, Vahab Mirrokni, and S Muthukrishnan. Online stochastic matching: Beating 1-1/e. In *Foundations of Computer Science, 2009. FOCS'09. 50th Annual IEEE Symposium on*, pages 117–126. IEEE, 2009.

[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.

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