TWO-SIDED MATCHING ALGORITHMS FOR DYNAMIC LABOR MARKETS

XINKAI XU

A THESIS

IN THE DEPARTMENT OF

CONCORDIA INSTITUTE FOR INFORMATION SYSTEMS ENGINEERING

PRESENTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF MASTER OF APPLIED SCIENCE(QUALITY SYSTEMS ENGINEERING)

CONCORDIA UNIVERSITY

MONTRÉAL, QUÉBEC, CANADA

AUGUST 2015

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Xinkai Xu**

Entitled: **Two-sided matching algorithms for dynamic labor markets**

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science(Quality Systems Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

Dr. A. Awasthi Chair

Dr. A. Ben Hamza CIISE Examiner

Dr. H. Ge External Examiner (BCEE)

Dr. C. Wang Supervisor

Dr. Y. Zeng Co-supervisor

Approved by _____

Chair of Department or Graduate Program Director

_____

Dean of Faculty

Date **August 10, 2015**

Abstract

Temporary recruitment is a fast growing labor market with candidates, usually professionals, looking for contract positions in today's fast evolving and dynamic economy. While two-sided matching algorithms have been applied to some labor markets during the past decades, they usually assume a static environment in which the strategic interactions between candidates and employers are designed without considering dynamic changes in market participants. In recent years, the advance of Internet and mobile technologies has enabled the implementation of large scale online labor markets which can capture the dynamics of the participants in a near real time manner.

In this thesis, I study how to improve the effectiveness and the efficiency of recruitment processes in the context of temporary labor markets. The key challenge is how to design two-sided matching algorithms to accommodate dynamic changes in market participants. To have a better understanding of the challenge and possible solutions, I first analyzed the problem using the Environment-Based Design (EBD) methodology. The analyzing results show that two aspects need to by improved in the proposed two-sided matching algorithm: the efficiency, which is the responsiveness of the algorithm, and the effectiveness, which is measured by the consistency of solutions generated by the algorithm when dynamics are introduced.

Based on the results derived from EBD analysis, I designed two algorithms to improve the efficiency and the effectiveness. For the efficiency, I designed a request-accumulated

deferred acceptance algorithm, which is a modification of the classical deferred acceptance algorithm. In addition, I designed a repair-based two-sided matching algorithm to improve solution consistency when changes are introduced to the market. The performance of the proposed algorithms are evaluated through a computational study. The results show that the efficiency and the effectiveness design requirements are satisfied.

Acknowledgments

I would like to express my deepest gratitude to my advisor, Dr. Chun Wang and Dr. Yong Zeng. Thank you for your numerous hours spent on teaching me, reviewing my papers, and correcting my theses, and for countless stimulating discussions.

I would also like to thank my colleagues.Thanks to Mr. Daniil Barklon and Ms. Yijing Zeng for their assistance of analyzing the critical conflict of the two-sided matching algorithm problem. Also, I would like to thank Mr. Yiqiao Wang for his help in polishing my thesis. During coding and writing, he spent countless hours proofreading and listening to me talk about my research. Zhijie Xie provided needed encouragement and insights.

Last but not least, I would like to thank my family. They were always supportive of me and encouraged me with their best wishes.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background and Motivation

Recruitment refers to the process of sourcing, screening, selecting, and contracting people for a job vacancy within an organization. The recruitment industry comprises of recruitment agencies. It provides services of recruiting staff to meet some employers' requirement. It also helps to find employers for staffs. The process of recruitment occurs in different sectors of business enterprises, such as IT service, consultant, translation, and transportation. However, recruitment agencies are facing some real problems. The first one is how to recommend the appropriate candidates to employers or recommend the appropriate employer to candidates in a quick and efficient manner. Usually, the situation is as follows: a manager of a recruitment agency spends a whole day contacting candidates to see if there is a candidate who is interested in the employer. The recruitment agency can prevent themselves from wasting time if the manager can directly pick up the appropriate candidate. The second problem is, because of the increasing growing market, more than half of industries provide temporary and contract staffing placement. Temporary recruitment agencies place workers in jobs on a contract or temporary basis. Temporary workers, also called contract employees, make the fastest growing segment of the global workforce.

Temporary recruitment needs employers and staff to participate in the market dynamically. A common situation is that the recruitment manager spends lots of time to find an appropriate employer or candidate, when suddenly, new members from two sides join in and ask for the recruitment service. The recruitment manager may find, for some original employer, the most appropriate candidate is now among the new bunch of candidates. The service markets whose participants vary constantly make it hard for the recruiting agency to find appropriate employers or candidates for their clients. Physical interview and manual assessment are traditional recruitment process. However, a large pool of candidates will cost the recruitment agency manager enormous amounts of time in interview and assessment. Moreover, the employers have various requirements, which mean managers need to change standards for each employer requirement. In the dynamic market, not only some new employers who provide job positions might leave the market, but also the existing employers may also change their preferences over time. Even if the manager has a name list containing various candidates for various employers, he may still meet a big problem on recruitment. Sometimes, the most appropriate candidate for some employers may not be appropriate to this candidate. Therefore, it is neither efficient nor effective to let manager assign the candidate to specific employer in matching a large pool of candidates. Technological advances have moved the job posting from paper media to Internet applications. Internet-based recruitment solutions can be rather efficient in dealing with fast responsiveness. However, due to the lack of a powerful matching algorithm, current online recruitment systems cannot match exact results effectively, which results in an increase of mismatching risks and costly human intervention. Moreover, this two-sided matching is a rather complex process mainly due to dealing with a dynamic recruitment requirements in a time-sensitive environment. Therefore, in this paper, we attempt to analyze the dynamic environment and formalize the challenges of recruitment in temporary/contract staffing, and to propose an effective approach to improve its effectiveness and efficiency based on these challenges.

In my research, Environment-based design (EBD) methodology [23, 24] will be used to establish such analysis and formalization. This will be explained in section 2.1

The objective of this research is to apply two-sided matching algorithm in improving the efficiency and effective of the recruitment for the recruitment agency. We focus our attention on the temporary labor market. We noticed that the temporary labor market is dynamic. The two-sided matching algorithm was initially designed for the marriage market. Then some person applied it in a specific labor market. The specific labor market is to assign medical interns to the hospitals. Both the marriage market and the medical labor market, they are relative static. In their assumption, they do not need to face the requirement for constantly joining or leaving members. However, in our temporary labor market, not only need we to fast response of the members changes happen, but we also try to main the consistency among the results. We may generate different results based on the dynamic market, our customers prefer their result being more consistency.

Therefore, One of our goals pertaining to the deferred acceptance algorithm in the market is to improve the responsiveness of the algorithm, leading to a reduction in the running time of the program in order to solve the problem of fast responsiveness

The other is to modify the algorithm to cater for the requirement of the dynamic market where variances occurring in the market may turn the previous stable match into an unstable one. The repair-based algorithm is designed for this kind of situation. It uses the idea that repairing the unstable match based on the previous stable one can solve the challenge caused by the dynamic market.

1.2 Scope and Approach

Our approach is to find the core conflicts by applying a design methodology of the environment based design (EBD). I propose a request accumulated deferred acceptance (RADA) algorithm to extend the classical two-sided matching algorithm, by decreasing its running

time through optimizing the operation execution sequence. The RADA algorithm can improve the responsiveness rate of the deferred acceptance algorithm.

In addition, I propose a repair-based algorithm, extending classical context to dynamic circumstance in order to fix the unstable match caused by the dynamic market. It can reduce the variance among results generate by the algorithm so that it can improve the consistency of the results, further to improve the satisfactory of customers.

1.3 Thesis Organization

This thesis is organized as follow: Chapter 2 introduces how to analyze recursively the problem via Environment based design (EBD). Chapter 3 reviews some papers about two-sided matching, from two aspects, including the two-sided matching model study and the two-sided matching algorithm study. Chapter 4 first presents two-sided matching market, defines the basic model based on the market. Then propose an request accumulated deferred acceptance algorithm based on the classical one proposed by [9]. Chapter 5 divides the market into static market and dynamic market and uses the repair-based algorithm to solve the dynamic problem of the temporary labor market. Chapter 6 presents some experiments to validate the algorithms' efficiency improvement. Chapter 7 draws some conclusions and proposes some related future work.

Chapter 2

Problem Analysis

2.1 Introduction

In this chapter, I am trying to use the environment based design (EBD) methodology to analysis the abstract problem we discuss on the background. I would like to get the critical conflict that the recruitment agency happened to.

2.2 Research Methodology

The EBD methodology focuses on the changes in the environment, which applies the recursive evolution in the processes of designing a problem. Three basic activities can be comprised into this design problem: environment analysis, conflict identification, and solution generation. The main purpose of this is through environment analysis, we can detect that what conflicts it has in the problem context. Based on this, it is reasonable to propose the corresponding solutions to solve the current conflict. However, the real problems is not always as easy to solve, as usually, it will produce new conflicts when the solutions have been used. Therefore, indefinite problems is hard. It gives us enough reasons to use EBD methodology to face such design problem. Using EBD methodology, allows for a

more logical direction to follow. Rely on the recursive object model (ROM) diagram, it can transfer the design problem description to graph, representing its linguistic structure and demonstrate the relationships among the objects. Then, dig into the design problem via asking two kinds of questions. One kind question is trying to complement information based on ROM diagram for designer. Opposite these generic questions, the other kinds questions are proposed based on the product life cycle, in order to get designer realize the potential requirements of the design. After several rounds of such two kinds of questions asking be run recursively, although the original environment is updated, the direction of design problem analysis remains the same, which is the critical conflict identification. The relationships among all iterations is called the performance network in EBD that is important, as in the following, the solution proposal are based on critical conflict that can be infer through performance networking.

2.3 1^{st} Round Analysis

For a given design problem, we can initially describe it using natural language following the template below.

Meta-verb (e.g. propose design, develop) product (S) to Verb.(I) noun (E)

According to the knowledge of designer, for example, the initial design problem in this case can be semantically described as follows:

"Propose an approach to improve effectiveness and efficiency of temporary and contract placement."

In above sentence, the "approach" is our goal. In order to implement this goal, it is critical to improve the effectiveness and efficiency in terms of temporary contract placement. Based on current knowledge we have, it is necessary to make clear about the issue on "placement" and the extent for "temporary", and then, to finally answer the question on the effectiveness and efficiency based on temporary and contract placement, the approach

to improve it can be proposed consequently.

Therefore, in the 1^{st} round of environment analysis, we can get some questions in the following table.

Table 1: 1^{st} round Generic question asking-answering

Questions	**Answers**
Who propose approach to “improve effectiveness and efficiency of temporary staffing service"?	The recruitment agencies propose an approach to “improve effectiveness and efficiency of temporary staffing service"
Why propose an approach to “improve effectiveness and efficiency of temporary staffing service"?	To make more profit by satisfying customers’ needs, the recruitment agencies propose an approach to “improve effectiveness and efficiency of temporary staffing service"
What is “temporary staffing service"?	Temporary staffing service is the match between highly skilled professionals and temporary and contractual positions. It includes four phases: sourcing, searching, matching and contracting.
What is “effectiveness"?	Effectiveness is the degree to which something is successful in producing a desired result.
What is “efficiency"?	Efficiency is the extent to resources is used for the intended.
What is “effectiveness and efficiency" in “temporary staffing service"?	Effectiveness of temporary staffing service is the degree to which temporary staffing service is successful in producing a desired result. Efficiency of temporary staffing service is the extent to which temporary staffing service is used for the intended task.
What is the meaning of improving effectiveness and efficiency in temporary staffing service?	Since the subject of the verb “improve" is the product “approach", this question is left for solution generation

After appending the original problem statement with these answers, it gets digged into

the problem. The updated problem statement is following:

Recruitment agency proposes an approach to improve degree of success to produce desired results and extent to use resource. The match includes sourcing, searching, matching, contracting between high skilled professionals and temporary contractual positions. The match produce desired results and use resources. To make more profit by satisfying customer's need, the recruitment agencies propose an approach to "improve effectiveness and efficiency of temporary staffing service"

Each action in the problem statement is called interaction, from the above updated problem statement. Identify interaction is a right way to understand their relationships. Because performance network, namely interaction relationship, can help us find which interaction is the root that constrain the others. The critical conflict of our problem is hidden in root interaction. The performance network table is below.

Table 2: Interactions and their description

Interaction	**Description**
I_1	Recruitment agency proposes an approach.
I_2	Approach improves degree.
I_3	Approach improves extent.
I_4	Match produces results.
I_5	Placement uses resources.
I_6	Recruitment agency satisfy customer.
I_7	Recruitment agency makes profit.

And the performance networking is below.

When converting these answers to interaction relationships into diagram, it is easy to find that the root interaction is. The interaction diagram is below:

In the "Reactive conflict and active conflict" Figure Figure 1, reactive conflict is the conflict caused by several interactions that are taking up identical resource, while active

Table 3: Performance networking 1^st interaction

Questions	**Answers**
What is the relationship between I_2 and I_3	There is no relationship between I_2 and I_3.
What is the relationship between I_4 and I_5	I_5 constrains I_4 .
What is the relationship between I_2 and I_6	I_2 constrains I_6.
What is the relationship between I_3 and I_6	I_3 constrains I_6.

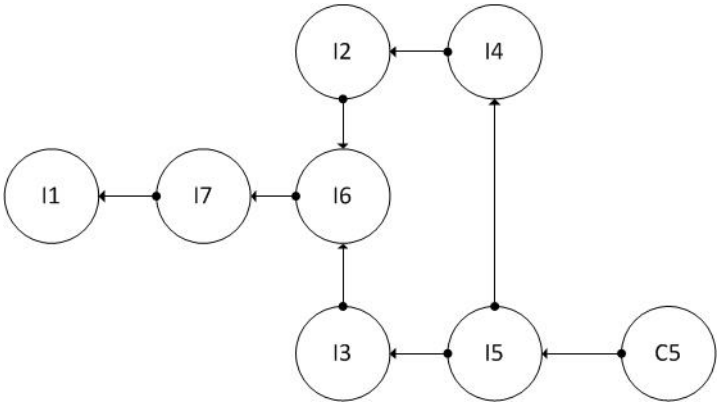

Figure 1: Reactive conflict and Active conflict 1st

conflict is caused by interactions that are short of resources. In my case, the active conflict is caused by Interaction I_5. It is “Placement uses resources". The active conflict is that the placement that match candidates and positions is short of resources.

As consequently, it is necessary to give more attention on Interaction I_5.

Table 4: Generic question asking-answering 1^st interaction

Questions	**Answers**
What are resources?	The resources refer to time, money, and customers. Customers are employers who provide temporary positions, and employees who get temporary positions.

2.4 2^{nd} Round Analysis

We have noticed that the above answer, explaining what resource may be consumed in my case, showing us new interactions. It is recursive processes by keeping ask and answer about the questions on case environment, leading us understand the case environment well. Therefore, the new interactions are added in the following Table 5.

Interaction I_8 and I_9 are the new added ones that reveal more information for us. Continue to asking their relationships and the answer Table 6 is below.

As result, the updated relationship among interactions in Figure 2 can show us I_8 and I_9 are the root for the whole interaction in current round.

2.5 3^{rd} Round Analysis

Aiming at the interactions I_8 and I_9, we can propose further questions in order to dig into the market environment, in the Table 7

Table 5: Interaction and description 2^{nd} interaction

Interaction	**Description**
I_1	Recruitment agency proposes an approach.
I_2	Approach improves degree.
I_3	Approach improves extent.
I_4	Placement produces results.
I_5	Placement uses resources.
I_6	Recruitment agency satisfies customer.
I_7	Recruitment agency makes profit.
I_8	Employers provide positions.
I_9	Employees get positions.

Table 6: Performance networking 2^{nd} interaction

Questions	**Answers**
What is the relationship between I_5 and I_8?	I_5 constrain I_8.
What is the relationship between I_5 and I_9?	I_5 constrain I_9.

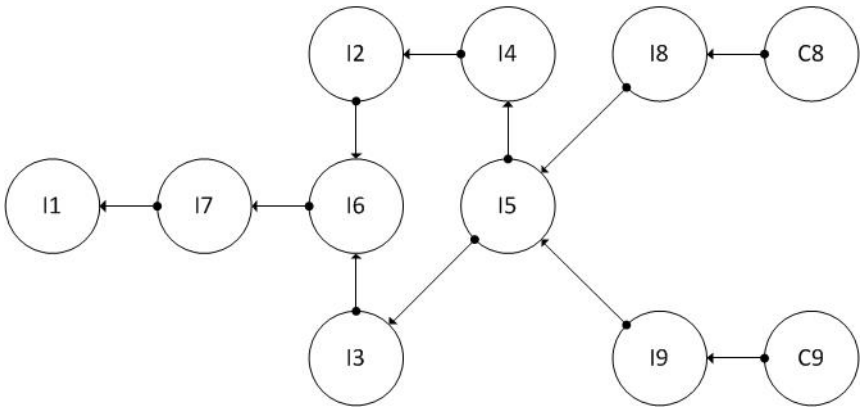

Figure 2: Reactive conflict and Active conflict 2^{nd} round

Table 7: Generic question asking-answering 3^{rd} interaction

Questions	**Answers**
Why the employers provide temporary contractual positions?	To find an appropriate employee for job opening, employers provide position. Job opening is a position available for employee.
Why the employees get temporary contractual positions?	To fulfill the employee's goal, employees get temporary contractual positions.

The above answer update interactions further by extract new interactions in the answers. The Table 8 shows us all the interactions.

Table 8: Interactions and description 3^{rd} interaction

Interaction	**Description**
I_1	Recruitment agency proposes an approach.
I_2	Approach improves degree.
I_3	Approach improves extent.
I_4	Placement produces results.
I_5	Placement uses resources.
I_6	Recruitment agency satisfies customer.
I_7	Recruitment agency makes profit.
I_8	Employers provide positions.
I_9	Employees get positions.
I_{10}	Employers find Employee.
I_{11}	Employees fulfill goal.

The interaction I_{10} and I_{11} are brought into, it is necessary to identify what relationships between new interactions and previous part. Table 9 show us performance network in 3^{rd} round.

From the Figure 3, we noticed that Interaction I_8 and I_9 are constraint by I_{10} and I_{11},

Table 9: Performance networking 3^{rd} interaction

Questions	**Answers**
What is the relationship between I_{10} and I_8 ?	I_{10} constrain I_8.
What is the relationship between I_{11} and I_9?	I_{11} constrain I_9.

respectively.

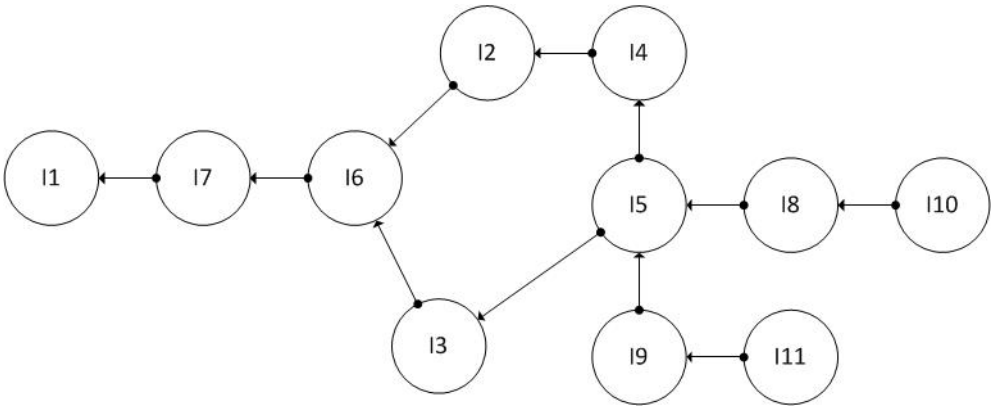

Figure 3: Reactive conflict and Active conflict 3^{rd} round

Based on this, we noticed that our problem environment actually can be converted into several interactions. In such that job market, the employers provide positions (I_8), employees get positions (I_9); employers find employees (I_{10}), employees fulfill goals (I_{11}).Our problem statement is " how to find employees for employers and how to let employees get job position in job market."

Finally, the original problem statement: *"Propose an approach to improve effectiveness and efficiency of temporary and contract placement"*, converting into the statement is below:

1. *Let an employee profile fit the job opening requirement in order for an employer to find an employee;*

Table 10: Generic question asking-answering 4^{th} interaction

Questions	**Answers**
How to find employees for employers ?	Job fit goals.
How to let employees get job position ?	Employees' Profile fit requirements.

2. *Let the job opening fit an employee goal in order for an employee to get a job opening.*

2.6 Summary

From above, we find that we need such a algorithm to address these sub-problems derived from the abstract problem, with which we can not only meet employees' requirement, but also cater for employers' requirement. The two-sided matching algorithm has such features. In the following, I introduce the classical two-sided matching algorithm in chapter 4. Through this classical algorithm, we can find it still has some issues that make this not suitable for our problem perfectly. For the sake of this, I introduce a dynamic algorithm for it in chapter 5. In chapter 6, I do some experiment to validate the effective.

Chapter 3

Literature Review

3.1 Introduction

So far, most people studied two-sided matching field from two aspects. Because two-sided matching is related to the game theory, many people try to study this from explore its model with the view of game theory, connecting it to market theory. Another aspect is try to find a more efficient way to implement it. My literature review is divided into two parts. section 3.2 collects several papers related the two-sided matching model study, while section 3.3 is mainly focuses on using various methods to improve the efficiency of their algorithm.

3.2 Two-Sided Matching Model Study

Gale and Shapley [9] proposed a paper that describes the marriage problem and college admissions problem in detail. The admissions problem is that universities select students who apply for them according to their own quota and their preferences. Simultaneously, students also decide which universities to apply for according to their preferences. The marriage problem is that men marry to women who also have preferences over man at

same time. The difference between the admissions problem and the marriage problem is that in the marriage problem, man can only marry to one woman, so can woman, while in the admissions problem, a college can admit many students, and student can only enroll into one college. Therefore, D. Gale and L.S. wanted to introduce a model solving two sides selection problem. Gale and Shapley introduced some basic concepts related to the criteria on how to assign applicant to the colleges. Gale and Shapley [9]thought that algorithm can assign the agents in one side to other side's. For example, algorithm assign applicants to appropriate college in order to let the matching including applicants and colleges keep stable. Based on this, they proposed two-sided matching algorithm that can generate a stable matching.

Dr. Kirby tried to apply the Gale and Shapley algorithm [9] into the matching between trip origin and trip destination. According to the paper, a person's preference between different modes of travel would be evaluated by associating a utility function with each mode [15].

Dr. Kelso and Dr. Crawford studied the labor market, in which firms and workers are two disjoint set. He viewed this as a resource allocation problem from the viewpoint of theory general economic equilibrium in market economies. He found that there was uncertainty on the part of participants existing in the labor market, which was a critical features for the market. Moreover, some conflicts limit workers be assigned to one job. Therefore, Dr. Kelso tried some other dimension: he studied the outcome of competitive sorting processes in markets. Dr. Crawford generalized Gale and Shapley's deferred-acceptance procedure [9]to the case where a money good is present. Their algorithm is "salary-adjustment process" that can be used to establish sufficient conditions for the existence of the core in some markets. The author argue that all workers are gross substitutes, although the workers' choice is discrete. The author also studied the equilibrium of the adjustment process. Dr. Kelso also thought that Gale and Shapley's result is equilibrium and biased for agents

on the side of the market that “sending request". In addition to this, the author studied the situation of adding agents to the market. Because new participants will affect the stability of the market, author discussed the effects of some new members add into the market [3].

Dr. Roth discussed this matching problem on standpoint of game-theoretic aspects [16]. He focused on the situation where whether participants from two disjoint sides would reveal honestly their preference. In the resulting non-cooperative game, it is a dominant strategy for each participant to reveal his preference honestly. In such a procedure, a member cannot get better outcome if he misrepresents his preference. And he drew the conclusion that matching procedure either yields a stable matching or give participants the incentive to reveal their true preference. Besides, he thought that there exists such matching procedure which generate a stable matching and also gives members from one of two sides of agents the incentive to reveal their true preference. Dr. Roth thought a market is completely centralized, in which all members reveal their preference. He proposed the results, because he thought that any possible incentive would misrepresent participants' true preference. Finally, Dr. Roth discussed the student-admission market and found that the student-optimal stable matching give students no incentive to misrepresent their preferences. The reason is that in that procedure, the matching generated by algorithm is a dominant strategy for student to reveal their preference. Although the potential misrepresentation would come from college side, colleges may have less scope for it, because of regulation and law, and incentive can affect only one side to reveal the true, which can also deduce stable match.

Dr. Peter extended the stable marriage problem by allowing individuals to provide weak ordering preference. Gale and Sharpley [9] proved that if the preference is strictly ordering, it can find a stable matching by their algorithm. Dr. Peter thought the strict preference assumption is not realistic. He mentioned in most applications it seems much more natural and less restrictive to rely on weak preference orders. Besides this, he proposed it

can expand the stability criterion of assignment. Based on this, he studied majority decision [10] and proved assignment derived from majority decision can also be stable, if it is a unique assignment which is preferred by a majority to all other assignments. And also he argued there is no assignment can be preferred by a majority to any stable assignment, if all preferences orders are strict. In his paper, Dr. Peter defined the concept of majority assignment, generally speaking, if there are more persons agree one assignment than another assignment, it means the fore assignment is preferred to the latter one. This definition was proposed based on voting theory. According to this definition, if there exists one assignment to which there no other assignment can be preferred, this assignment is a majority assignment. Moreover, if for any assignment, there exits one assignment that is preferred to others, the assignment is strong majority assignment. Dr. Peter found if the preferences are strict, then all stable assignments are majority assignments. And he also proved that any strong majority assignment is stable. Dr. Peter used a new concept which is choice number to look for the assignment. The choice number is the number of objects that rank before some specified object in a preference. Dr. Peter designed an approach to minimize the sum of choices number to get strong majority as well as stable matching.

Dr. Dolton [8] summarized the marriage game in terms of assignment game with indivisibilities, giving a new proof of the existence of the core for this game situation, analyzing the condition necessary for the core to be unique.

Dr. Roth [18] proposed an opposite opinion against the view trusted by most researchers who believe that “college admission problem" is similar to “marriage problem". In his view, there is no stable matching procedure in which it is a dominant strategy for colleges to reveal their true preference, maybe there are exist outcomes that all colleges strictly prefer to the College optimal stable outcome. Dr. Roth focused on the “marriage problem" two years ago, and proposed in “marriage problem" there is no such stable matching procedure that can make itself to be dominant strategy for all agent, which means there is no such

procedure that makes all agents in the market state honestly their true preference. [16] Dr. Roth summarized the Gale and Shapley algorithm [9] can generate the Man optimal stable matching, and found it is a dominant strategy for the male side in the marriage problem. However for the college admissions problem, Dr. Roth did not think it is a dominant strategy for college side. That is because he thought a college would not prefer one stable matching to another unless the college had already specified its preference over outcome. The outcome assigned to the college is not just one student but many students, which is not the same as marriage problem.

Dr. Roth [17] discussed the job-matching in the labor market. This market is similar to the marriage problem, but the difference is that each firm that can hire a set of employees and each worker can be accepted by a set of jobs. He found that the interests of the firms and workers are polarized over the set of stable matching. According to Gale and Shapley's algorithm, it can generate a stable matching that is firm-optimal stable matching, which means the best stable outcome for each firm. However, it is also means this result is the worst for every worker. Similarly, there is also a worker-optimal stable matching that can be generated by Gale and Shapley's algorithm in which the outcome is best for worker but worst for each firm. Based on this, Dr. Roth would like to study the cause for such polarization of interests. He drew the conclusion about the polarization of interests. He believed that these occurrences in the matching model do not depend on the many of the special assumptions made in the model. He cannot explain the reason, but he can prove the assumptions, like the preference restriction, is not the reason to generate the polarization of interests in the stable matching set. As we know, for a two-sided matching problem, we can get several matches that meet for the definition of "stable", but we can only obtain two optimal stable matches. They are optimal for one of sides, respectively. Making it clear on the structure of the set of stable matching is necessary, because we can analysis it to find if there exists a method that can find arbitrary stable matches or find all stable matches. Dr.

Roth is concerned the concrete conflict and coincidence of workers' and firms' interests of whole set of stable matching. But he also admits that there are still many open questions remaining.

Dr. Roth [1] guessed that market failures may be associated with instability of the outcomes generated by centralized deterministic matching procedures. But he also did not observed many entry-level labor markets and other two-sided matching situations which do not employ centralized matching procedures experience such failure. He tried to start from an arbitrary matching, the process of allowing randomly chosen blocking pairs to match will converge to a stable matching with probability one. At last, he found that there has no such procedure that can not only generate stable matching, but also make it be dominant strategy for all agents to state their true preference. He also thought that Gale and Shapley's algorithm can generate stable matches in terms of the true preference. This algorithm may have an incentive to misrepresent their preference.

Dr. Abeledo [11] studied characterized sets of mutually acceptable pairs of individuals in the stable matching problem for which a stable matching exists under all preference profiles. He also proposed a notion of acceptability graphs that can be viewed as a representation of the underlying social structure of mutually acceptable pairs. The acceptability graphs is bipartite. Dr. Abeledo found the number of men and the number of women is equal in the original marriage problem proposed by Gale ans Shapely. He thought the more realistic situation is one where the number of men and women are unequal, even where each man or woman holds the incomplete preference ordering, which indicate that the individuals outside that subset are unacceptable partners. Dr. Abeledo thought a man and a woman are mutually acceptable, if they are on each other's list of acceptable partners. In addition to this, he also introduced a acceptability graph while trying to solve the stable matching problem from the view of a graph theoretic aspect. The stable matching problem

consists of a bipartite graph and an ordering mapping. Dr. Abeledo treated the acceptability graph's vertex set as the member of market, and the edges of this graph as the mutually acceptable pairs. Dr. Abeledo tried to solve the stable matching problem by introducing the concept of acceptability graphs. Besides, he also checked the conditions for stability based on the acceptability graph.

Dr. Zhou [25] proved that even though agents reveal their their preferences strategically, the Gale-Shaley's algorithm still yields stable matching with respect to the true preferences. Dr. Zhou extended Roth's [17] result on if members in market revealing their true preferences.

Dr. Tamura [21] proposed a counterexample for D. E. Knuth's view that any matching can be transformed to any stable matching. This paper argue that it can only be transformed to some stable matching. He also give an algorithm to find either such stable matching. The transformation from unstable matching to other matching relies on b-interchange, which refers to a blocking math, assigning them their preferred choice respectively. Cyclically using this b-interchange, we can transform unstable one to some stable match.

Dr. Csima [14] proposed a timetabling algorithm using matrix terminology in his treatment. He assumed men and women only date with each other in unit time for constant time in marriage market. Therefore, the matching among men and women ca be treated as r-regular graph.

Dr. Manlove [5] focused on the issue that the preference list of the participants are not necessarily complete and not necessarily totally ordered. Based on this, the author proved the existence of a 2-approximation algorithm for the problem. He introduced the notion of "ties" to indicate the situation that some of individuals might be indifferent among some members of the opposite sex side. Therefore, if there involves tie in market, then the stability of the stable matching would be weak stability.

Dr. Dutta [4] proposed an assumption that the composition of one's co-worker can

affect the preference. Firstly under this assumption, the matching is nonempty. We first consider the case when subsets of the individuals are couples. We assume that each couple prefers a matching in which they are matched together with an institution rather than another one where they are paired with different institutions. There is a unanimous ranking of all workers, and any worker prefers to join a set of workers containing higher-ranked workers.

3.3 Two-Sided Matching Algorithm Study

"The stable marriage problem" [7] written by DG. McVitie, et, al. They indicated that an extension algorithm based Gale & Shapley's algorithm (G&S algorithm). The extend algorithm executes the G&S algorithm recursively in order to generate all the stable matches. Dr. Vitie analyzed G&S algorithm, finding that the G&S algorithm consist of two operations, which are "sending request" action and "decision making" action. These two operations are run alternatively to deduce the stable matching. Therefore, they tried to break up a pair of stable match manually, and called G&S algorithm is executed again to attain a different stable match. But abusing break stable matching randomly may leads the same stable solution being obtained many times. The author solved it by constraining the algorithm, setting it up two rules. First of all, order the agent in two side set. One of the rules is starting at a stable matching obtained by a successful breaking operation on agent i in one side, then break operation may only be performed on element $\geq$ i. The second rule is in a break operation starting on agent i on sending side in the two sides, two sides can be separate into sender side and receiver side. Agent from sender side can send request to opposite side, agent from receiver side only make decision with agents received. Only agent $\geq$ i may do "sending request action". With the help of two rules, author proved that when the break operation is applied to a set of stable match, it is terminated successfully with the result set of stable match. Besides, according to Gale and Shapley's algorithm [9], the

stable match derived from G&S algorithm is optimal for one side of two sides according to which side is set as sender side. Dr. Vitie thought that any stable solution, excluding one sided optimal stable matching, can be obtained from this optimal stable matching by successive application of the break operation. And the break operation with the two rules given above allows each stable matching to be obtained only once.

Dr. Wilson analyzed the stable matching assignment algorithm [22]. Dr. McVitie and Dr. Wilson proposed an algorithm call algorithm 411 [7]. This algorithm extended Gale and Sharpley's idea [9] to generate one stable matching by using two main operations including "sending request" action and "decision making" action. But Dr. Wilson made some changes in the execution sequence of the algorithm. In algorithm 411, they let the "sending request" action happen one by one rather than simultaneously do as in the G&S algorithm. He tried to calculate the executing time of "sending request" action, and found as the problem sizes (n) increases, the number of "sending request" actions, which is directly related to the number of Complexity is of the order of $nlogn$. The G&S algorithm is of the order $\cong n^{5/2}$.

Dr. McVitie and Dr. Wilson applied their algorithm into solving the problem of British university admissions. Moreover, they observed the effect of the large size of this problem [6]. In this case, the scale of the problem is determined mainly by the number of students applicants and the number of University courses offered. The number of students is about 110,000, while the number of university courses seems about 8000. Dr. McVitie considered two basic algorithms developed to apply into this large-scale university assignment problem. The Gale and Shapley algorithm [9] would let all applicants do the "sending request" actions simultaneously according to their own preferences. And the other side, namely the college side will receive these requests and makes decision. since there are 110,000 student applicants initially, it is a big challenge to the memory to hold such amounts of data. The reason why algorithm 411 [22] is better than G&S algorithm [9]

with respect to the aspect of large scale problem size is the former involve applicant to " send request" one by one while the latter loads all the initial data at once time, which is a burden for the computer's computing capability. However algorithm 411 involves a lot of searching for one applicant, which is an inefficient process because it needs more "deciding" actions executed by the other side. Dr. McVitie choose algorithm 411 and found that a small assignment of 30 courses and 800 students required 4-5 minutes run time. And 292 courses and 9000 students required 61 minutes. The considerable increases would occur, if we to use more of applicants and courses.

Dr. Itoga [12] published a paper in which they summarize the two algorithms proposed by Gale and McVitie, and discuss the upper bound for the stable marriage problem. When he got the optimal stable matching by following Gale and Shapley's algorithm, he proved that at most one girl ends up with her last choice as a partner, and the maximum number of stages for a problem is $n^2 - 2n + 2$ where size equals to n. The number of stages is derived by counting the number of "sending request" actions used at each stage. Therefore, Dr. Itoga took the number of stages required to solve the problem into account to measure the computational complexity of it, and found that in most complex, it $n^2 - 2n + 2$.

Dr. Itoga [13] provided an in-depth analysis of the work of stable marriage problem, he extended the preliminary set of participants to dynamical participants, and also allowed certain relative preference conditions to be maintained. Finally, he proposed several theorems, referring to extendable dynamical agent situation. He studied and found that the algorithm provided by Gale and Shapley [9] relying on static requirement of the matching problem. If the requirement of any participant is changed after the stable matching generation, the previous stable matching may not be stable any more. Therefore, he relaxed this requirement by allowing the additional participants to join after a stable matching has been found. He thought the method starting all over and applying the Gale and Shapley's algorithm to fix

the new stable matching is not efficient. To solve this problem, he introduced a ripple operation, instead of repeatedly applying Gale and Shapley's algorithm. The ripple operation consist of a sequence of steps. Based on previous stable matching, it allows new members to execute "sending request" action, which would lead corresponding opposite member to be rejected, lead further the rejected members to execute their "sending request". The process can be executed repeatedly, and it would be interrupted by one of two situations. The first situation is all members of the "sending request" side are on hold; the second one is some the member of the "sending request" side are rejected by all opposite side members. Dr. Itoga compared two kinds of algorithm in terms of complexity computation, and found the average number of "sending request" action of Gale and Shapley's algorithm is twice the corresponding result in his algorithm. The main reason to decrease the complexity is his algorithm generate result based previous stable matching, which means it can omit the first round "sending request" and set the corresponding pair in stable matching as hold rather than generate the same pair as ones in previous stable matching.

Dr. Irving [19] summarized that Gale and Shapley's algorithm could generate a stable matching that either man optimal stable matching or woman optimal stable matching in the marriage market. And he found that the man-optimal stable matching sacrifices woman's requirement to meet men requirement, leading all man in the context can not find better choice in other stable matches. Dr. Irving focused on problem of finding a stable matching more equitable for two side's requirements. He tried to design an algorithm to maximize the average satisfaction of members based on stable matching. In his opinion, he can measure the average satisfaction of members in the market via the position of partner in the preference list. Dr. Irving used graph-theoretic methods which is an $O(n^4)$ algorithm. Dr. Irving obtained the position number of each man's partner in corresponding man's preference list and the position number of each woman's partner, and then calculated the sum of position number. He thought a stable match is optimal stable match if it has minimum

possible value. After defining the criteria, Dr. Irving introduced a concept called “Rotations", which is a sequence of man/woman pairs where the woman in some pair is the favorite choice of the man in the pair, and the woman in next pair is the second choice of the man in last pair. Dr. Irving thought in the male optimal solution, assign the second choice of each man instead of the favorite one, leading another stable matching. Based on this, Dr. Irving proposed another algorithm based graph-theoretical algorithm, finding all the rotations; and then construct a graph of the weighted rotation poset, and then find the maximum-weight closed subset based on the poset, and then eliminate the rotations in that closed subset to obtain the optimal stable matching. Dr. Irving emphasized the main point in the algorithm is the maximum weight closed subset can be obtained from the graph.

Dr. Tseng [20]argued that the stable marriage problem can be solved by a divide-and-conquer approach. but it appears to be hard to obtain the average performance of their parallel algorithm. The parallel algorithm consists of two parts, the first one is dividing, which first divided the agents into two parts, and updated the preference matrix according to their different group. Recursively applied this algorithm, then applied merging algorithm to combine sub-result based on different group. In the process of the divide-and-conquer, Dr. Tseng first divided the problem into four sub-problems that contains only one man, if he wanted to obtain man-optimal stable matching. Then the approach begins to merge the sub-solution for each sub-problem into one solution. If different men pair with different women, it can combine two pairs simply. When different men choose the same woman, it should consider that woman’s preference to decide which man is held temporarily. Dr. Tseng also talked about the worst case of his parallel algorithm. According to Wilson [22], the maximal number of “sending request" action to obtain the man-optimal stable matching is $n^2 - n + 1$ for the size of n. Based on this the probability of worst case is less than $4.7 * 10^{-16}$ when size of n equal to 8.

3.4 Summary

From the literature review above, I notice that nowadays papers study this field, with a assumption that the market is relatively static. But the temporary labor market I face is such a market that lots of members join or leave so that I cannot wait for a time to execute algorithm to calculate results. In the following chapter, I first introduce the classical two-sided matching with its model, criteria and implementation in chapter 4. After that, We can tell more clearly about its limitation for our problem. Then I introduce two dynamic algorithms in chapter 5.

Chapter 4

Request-Accumulated Deferred Acceptance Algorithm

4.1 Introduction

We have already defined our problem. we also know that two-sided matching algorithm can solve the related problem. Therefore, I introduce related things in this chapter. In section 4.2, I first define the temporary labor market model and matching in this market. Then, I introduce the criteria of stable status for matching in the market. Lastly, I implement the classical deferred acceptance algorithm. However, I face the situation that a fast responsiveness is required in the dynamic market, I propose a request-accumulated deferred acceptance algorithm in order to solve this problem.

4.2 Classical Deferred Acceptance Algorithm

Gale and Shapley [9] proposed a paper that describes the marriage problem and the college admissions problem in detail. Actually, a similar approach for the temporary labor market can also be used. In that market, employers and candidates spend large amounts of time

matching, according to their own criteria. For example, a candidate who wants to work in a company near their apartment has limited working positions to choose from, while an employer who can provides position that is easily accessible areas may attract fewer candidates. When trying to improve the efficiency of matching a candidate and an employer, the first problem is how to satisfy both sides. For the sake of argument, various candidates have various requirements the employers' side. The difficulty in this circumstance is that for a given position some candidates may prefer it while others may not. Similarly, it is possible that some prospective candidates do not like to work for some employers while these employers want to hire them.

4.2.1 Temporary Job Market Model

In this temporary labor market, candidates want to find an appropriate position provided by employers, and employers also try to find optimal prospective candidates. Each candidate has their own preferences over all the working positions, and each employer has their own requirement, which form unique preferences over candidates. The whole market is dynamic, which means new candidates or employers join or leave the market at any time, therefore, other members need modify their preferences respectively. Also, the preferences of some existing agents (candidates and employers) may be modified due to some reasons. In one word, the changes happen on the market lead to modification for members' preferences of the market, which affect the accuracy of our results. In order to solve this, we need to define some concepts.

There are disjoint sets of candidates and employers, $C = \{c_1, c_2, \ldots, c_n\}$ and $E = \{e_1, e_2, \ldots, e_m\}$, for each element in these two sets, there are specific preferences. For example, the preference of c_1, $P(c_1)$, is an ordered list on the set $E \cup \{c_1\}$. It might be of the following equation(1)

$$P(c_1) = (e_i, e_j, \ldots, c_1, \ldots, e_k), i, j, k \in \{1, \ldots, m\} \tag{1}$$

The position of c_1 in this ordered list indicates c_1's preference. The elements before c_1 in terms of position in the ordered list represent employer whom c_1 find attractive, while those agents whose position in c_1's preference list are behind c_1's represent employers whom c_1 finds less preferable to not working at all. Moreover, the ranking of position in this preference ranking list indicates preference, for example, candidate #1 prefers job position provided by employer #i's to employer #j's.

Similarly, each employer also has its own preference ranking list. e_i has preference raking list(2) to indicate his preference over all tenants with the same form.

$$P(e_i) = (c_o, c_p, \ldots e_i, \ldots, c_q) o, p, q \in \{1, n\} \tag{2}$$

For employer #e_i, he would focus on the people whose position in his preference ranking list are ahead of his, because for those whose positions in the list are behind his indicate. Candidates who are in the top of the preference ranking list are the preferential choices of employer #e_i.

For candidates, when they find an appropriate job position, it is like matching them to the appropriate employer. For employers, when they find an appropriate candidate, it means that they agree to match to the candidate. The temporary labor market includes a set of candidates, a set of employers and their preference ranking lists.

In the context of markets, employers try to find prospective candidates to fill their job positions according to their own preference, while the candidates also try to find appropriate place to work based on their preference. Our goal is to improve the efficiency on establishing a stable matching between candidates and employers based on their own preferences and requirements.

4.2.2 Matching in Temporary Labor Market

A matching μ is a one to one correspondence from set $E \cup C$ onto itself of order two such that if $\mu(e_i) \neq e_i$, then $\mu(e_i) \in C$; if $\mu(c_i) \neq c_i$, then $\mu(c_i) \in E$. We can notice that so-called matching includes several pairs that consist of two elements coming from the set of employers and from the set of candidates. Moreover, two following conditions constrain the matching. That is, if one of the two elements comes from either the set of employers or the set of candidates, then the other only comes from the opposite set. [2]

The agents (employers or candidates) have their own preference ranking list, the position of the ordered list indicates the agent's preference. For example, if c_i is before c_j in e_k's preference ranking list, we can notice that e_k prefer c_i to c_j, or(3)

$$c_i > e_k^{c_j} \tag{3}$$

Similarly, if e_i is before e_j in c_k's preference, like(4)

$$e_i > c_k^{e_j} \tag{4}$$

Also indicating that tenant c_k prefer landlord e_i to landlord e_j.

4.2.3 Criteria for Stable Match

Agents from one of two sets find appropriate agents from the opposite set, which actually is a strategy to match to the preferential agent according to their own preference lists. Therefore, the existing pair of employers and candidates may be broken up because more attractive agents come to contact them. More specifically, the existing pair is to be broken up because of two kinds of situations.

Blocked By Individual

From the agent's preference ranking list, we can find some agents from opposite side whose position in its preference ranking list is behind the place of itself in the list. For example, if employer e_i prefers to hire no one rather than hire c_j to work for him, we can say in (5)

$$c_j < e_i^{e_i} \tag{5}$$

Therefore, at the beginning, if we assign e_i to c_j, the match will be broken up; the pair is blocked by individual employer c_j. If a matching includes such a pair, it means this matching can be improved by assigning other matches to employer e_i, we can save time by omitting the matches that contains such pairs blocked by an individual. [2]

Blocked By Pair

Another situation would also lead to the existing pairs to be broken up. Assume there exists a pair of employer e_i and candidate c_i. In this matching, there also exists another pair employer e_j, candidate c_j. But, according to the preferences of everyone, we find e_i prefers c_j to c_i, and at the same time, c_i prefers e_i to e_j. The two conditions can be expressed like:(6)

$$c_j > e_i^{c_i} and \quad e_i > c_j^{e_j} \tag{6}$$

In this situation, this matching can be improved by breaking the previous two pair in order to match c_j to e_i, because original matching is blocked by those pairs.

If a matching is not in both above two situations, we can notice that the current matching is a stable matching. We recommend stable matching for the each member of temporary labor market, as a result, an agent would be satisfied with their partner. This agent could not be matched with another preferential one, because there exists other agent from same

set who can replace it. [2]

4.2.4 Criteria for Optimal Match

Optimal matching is a special case of stable matching, we can get a number of stable matches via the above definition and constraint. The candidate-optimal matching is picking up one stable matches among all stable matching. In the candidate-optimal matching μ each candidate has a partner at least not worse than that provided from any other matching γ (7) [2]

$$\forall c \in C, \mu(c) \geqslant c^{\gamma(c)} \tag{7}$$

4.2.5 Deferred Acceptance Algorithm

Gale and Shapley [9] propose the so-called deferred acceptance algorithm, while trying to achieve an optimal matching. Assume that we want to find the candidate-optimal matching. According to the above optimal matching definition, for each candidate, we can intuitively assign their best choice according to their preference ranking list. As a result, some employers might receive requests from these candidates. The employers who receive requests can choose to hold them temporarily or release them. But these employers who received requests can only hold one request. The two operations are run repeatedly until all the candidates reach an agreement or one of them uses up their preference choices. This kind of strategy can let all candidates choose the best job position from those offered by the employers who are willing to accept their request. Accordingly, such an algorithm is divided into two operations: the first operation is a preference-based request sending from one side to the other side, while the second step is decision-making. As a result, assuming the candidates side is the side that sends a request to the employers' side, a candidate can match with employer to whom they preferred and this employer also agree to accept the

candidate.

In operation 1, each candidate sends one request to his favorite employer according to his preference. As a result, in the side of the employers, some employer may receive one or more requests from various potential candidates, while some employers may face the fact that no one sent the request to them. It means that they are not the first choice of any one of the candidates.

In operation 2, for each employer, when all of candidates end up sending operations, it means employers can decide on the requests they received. For those who receive nothing, they need to do nothing; for those who only receive one request, they can holding this request temporarily; for those who receive two or more requests, they need to make a decision on hold only one of them and at the same time releasing the rest. After all employers who receive two or more requests end up with decision-making, the released rejected candidates are free to send the request again to their preferred position as second choice.

The whole algorithm may run several rounds which involve alternating operations of sending-request and decision-making until all candidates match to corresponding positions or some non-matching candidates have already use up all of their choices. The flow chart of the proposed deferred acceptance algorithm is in Figure 4

The corresponding notations and functions are shown in Table 11

And the pseudo-code of the deferred acceptance algorithm proposed by Gale is shown in algorithm 2

1. At line 1-5, for each candidate, sends a request to their preferred employer in its preference set $P(t_i)$, then taking off this employer from current preference list, next, adds current candidate into the employer's received list, which means this employer has received a request from current candidate.

2. At line 6-15, all candidates have already sent requests to their favorite employer. For

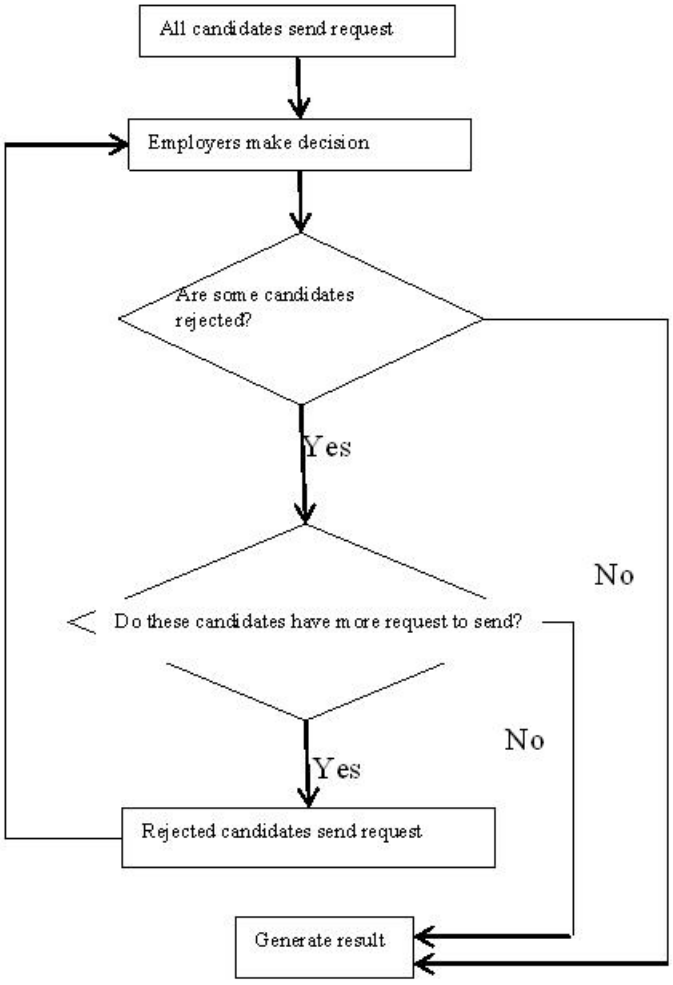

Figure 4: Deferred acceptance algorithm flow chart

Table 11: Explanations for the proposed deferred acceptance algorithm

Notation	**Meanings**
C	Candidates set
E	Employers set
c_i	Candidates c_i.
e_i	Employer e_i
RE_k	Employer e_i's list for request received from candidates
$Pair\_e_i$	It is a candidate matched to employer e_i
H_c_i	Candidate c_i status on if it is chosen
$P(c_i)$	Candidate c_i's preference ranking list
$P(e_i)$	Employer e_i's preference ranking list
$Send(P(t_i))$	Select favorite employer for candidate c_i and send request to the employer
$P(C)$	All candidates' preferences over all all employers
$P(E)$	All employers' preferences over all candidates
$M_s(C, E)$	stable matching contains key-pairs in which candidate in key set, employer in value set

Input: $P(C)$, $P(E)$
Output: $M_s(C, E)$ stable matching

for $\forall c_j \in C$ **do**
  $e_k = send(P(c_i))$;
  $P(c_i)$ delete e_k ;
  $RE_k \leftarrow c_i$;
end
for $\forall e_j \in L$ **do**
  for $\forall c_i \in RE_i$ **do**
    if e_i *prefer* c_i *to others* **then**
      H_c_i is true ;
      $Pair\_e_i = t_i$;
    end
    else
      H_c_i is False ;
    end
  end
end
while $P(t_i)$ *is null* || *all* H_c_i *are true* **do**
  for $\forall c_i \in T$ **do**
    if H_c_i *is true* **then**
      Continue ;
    end
    else
      $e_k = send(P(c_i))$;
      $P(t_i)$ delete e_k ;
      $RE_k \leftarrow t_i$
    end
  end
  for $\forall e_j \in E$ **do**
    for $\forall c_i \in RE_i$ **do**
      if e_i *prefer* c_i *to others* **then**
        H_c_i is true ;
        $Pair\_e_i = c_i$;
        M_sȧdd(c_i, e_i)
      end
      else
        H_c_i is false ;
      end
    end
  end
end
return $M_s(C, E)$;

Algorithm 1: Pseudo algorithm for the proposed deferred acceptance algorithm

each employer, selects its favorite candidate's request among all requests it received, and marks this candidate as true as well as the rest as false, which means it decides to hold that true candidate temporarily and reject the rest.

3. At line 16-35, the first round of operations has terminated. Moves to next round after judging the condition. When one candidate's preference ranking list is null, indicating all employers rejected him, therefore he has no employer to choose and send request. Operation ending up with current matching that is final result. If all candidates are held, the stable match is generated. Otherwise, we only need to focus on the rejected candidates. For those rejected candidates, because his previously selected employer has already been deleted from his preference ranking list, he selects his favorite employer based on revised preference raking list. After all rejected candidates finishing sending requests, it is the employer's turn to make decision to continue to hold current candidate or replace it via the new candidate from new received requests.

4.3 Request-Accumulated Deferred Acceptance Algorithm

4.3.1 Analysis on Deferred Acceptance Algorithm

The first operation of the deferred acceptance algorithm is to define which side of two sides as sender side, while the other side is automatically the receiver side. The result is sender optimal stable matching. In our paper, we define the candidate in temporary labor market as the sender side, and employer side as the receiver side. After making sure which side is sender or receiver, the algorithm will recursively run two operations, "sending request" and "making decision". Sender side first sends requests to their current best choice, and then removes the choice from their preference list, meaning it has no chance to resend the request to the same object again. And because of this, if the employer (request receiver)

declines it, the candidate has no choice but to send request to its next choice that is currently its best choice after it removes last employer from the preference list.

Therefore, the reason why this algorithm can find an Request accumulated matching is because sender side (candidates side) has chances to match to their best choice, and they will match to their best choice among all receivers (employers) who are also willing to match to them. Besides, receiver sides can only match to their best choice among all senders who already sent requests. The resulting match is stable match, because for any pair of candidate and employer in the stable match, the candidate in this pair can not find another receiver who not only receives request from the sender of the pair, but also likes to accept this request.

In summary, on one hand, senders (candidates side)in the market constantly send requests and adjust their preference list until the preference list is empty or the stable matching is generated. On the other hand, the receiver (employers side) constantly makes decisions on holding requests. They can only hold one request, which means they need to reevaluate the requests they receive according to their own preference list. For each round, free candidates come from the candidates who were rejected. Only rejected senders can send new requests again. But in the deferred acceptance algorithm, rejected senders can only do the operation after all of receivers have finished their decision making.

Through the analysis of the above algorithm, we can notice that the most time consuming part is the recursive execution of operations, including “sending request" by senders and “decision making" by receivers. In the same context , the result, sender (candidates side) optimal stable matching stays the same, which means total time of execution of “sending request" remains the same. For each sender, they will take fixed times of execution of operation “sending" no matter how many times they run the algorithm to obtain sender optimal stable matching. Only if the preferences stay the same, the “sending request" operation will be executed a fixed number of times. However, even if we can not decrease the number of

times of execution of “sending" for identical preferences from senders and receivers, we still can optimize the amount of times of execution of operation “decision making". If, in the market, sender s_1 is at last matched to receiver r_1 in the result of sender optimal stable matching, then the best way to optimize the execution times of operation “decision making" is let the s_1 send request to r_1 as soon as possible. We will validate this later.

4.3.2 Improving in Algorithm

Based on the idea above, I propose an request-accumulated algorithm based on the deferred acceptance algorithm. The following is its algorithm flow chart in Figure 5. The pseudo-code of the RADA algorithm is in algorithm 2

Input: $P(S)$, $P(R)$
Output: $Stable(S, R)$ stable matching
while $\forall s \in S$ **do**
    $r \leftarrow sending(s, S)$;
    $Stable(S, R) \leftarrow \{s, r\}$;
end
while $\forall r$ *has more than one request* **do**
    if $\forall s$ *be rejected by* R **then**
        generate result;
        break;
    end
    for $\forall \{s, r\} \in Stable(S, R)$ **do**
        $s \leftarrow deciding(r, R, Stable(S, R))$;
        $Rejected\_Set \leftarrow s$;
    end
    for $\forall s \in Rejected\_Set$ **do**
        $r \leftarrow sending(s, S)$;
    end
end
return $Stable(S, R)$;

Algorithm 2: Pseudo algorithm for the RADA algorithm

This main purpose of it is to let the sender side (candidates side) send requests more quickly, decreasing the number of times of execution of “decision making"

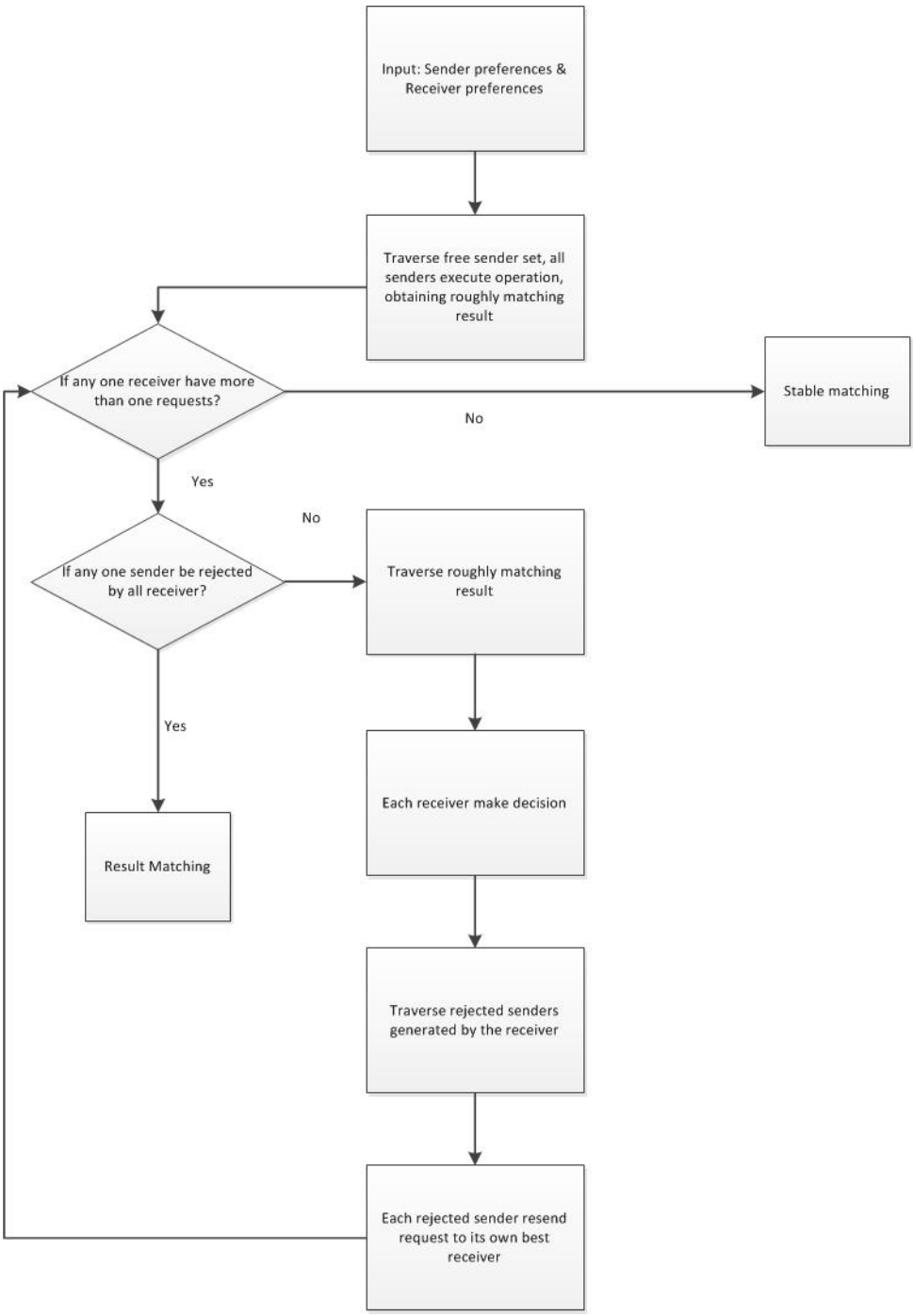

Figure 5: Request-accumulated deferred acceptance algorithm flow chart

4.3.3 Theorem for Stable Match

The RADA algorithm mentioned above can save more time in running, but I also need to proof that it can still generate the same result as classical algorithm do. In Gale and Shapley's paper, we can find that they had already proof some theorems using marriage market.

Theorem 1. *There is at least one stable matching for every marriage market. [2]*

Proof. Round 1: At begin, each agent from sender side send request to his best choice. Then each agent from receiver side hold one request and reject the rest.

Round K: All rejected at round k-1 agents from sender side send a new request to his best choice who hasn't yet rejected him. Then each agent from receiver side hold one request and reject the rest.

Stop condition: When one agent from sender side has been rejected by all receiver agents, or all sender side agents has been held by opposite side ones simultaneously.

We can notice that the algorithm terminates, because agents from sender side can not send the request to the same receiver side agent twice. The matching generated via this way is stable. It is not blocked by individual, because the sender would send the request to the acceptable receiver, and the receiver would hold the request from the acceptable sender. Besides, it is not blocked by pair, because if some sender would prefer to a receiver than his assigned in current match, he must have already proposed to this receiver, and this receiver has rejected him, which means this receiver prefer other sender than him. □

This proof proves that the deferred acceptance algorithm can generate a stable match. We can also prove that our RADA algorithm can also generate a stable match.

Theorem 2. *The RADA algorithm can generate a stable match.*

Proof. Initially, there is a circular linked list containing n receivers.

Step 1: At the beginning, each agent from the sender side sends a request to their best choice. The first agent from the receiver side makes a decision, holding one request while rejecting the rest.

Step K: Each sender sends a request to their best choice according to the receiver side decision from step K-1. The receiver following the receiver in step K-1, makes a decision based on all requests he holds currently, holding one request from one sender, and rejecting the rest.

Stop condition: Stop condition: When one agent from sender side has been rejected by all receiver agents, or all sender side agents has been held by opposite side ones simultaneously.

Based on the analysis above in Theorem 1, we can find the RADA algorithm still insists two rules. We assume all the receiver participate the algorithm is acceptable for all senders, which prevent the match from being blocked by individual. Sender sends its request as the sequence of its preference. It prevents the match from being blocked by pair. □

Through the proof, we can assure that it can also generate a stable matching like the classical algorithm. But it seems not appropriate to move this algorithm to temporary labor market. In the next chapter, I introduce the dynamic market to make it more suitable for the situation of temporary labor market.

4.4 Summary

In this chapter, I mainly introduced related materials with classical two-sided matching algorithm. Through this part, We can find that the classical two-sided matching algorithm was designed mainly for finding stable matching for a static market. It means the whole market cannot be changed while the algorithm is running to generate the results. However, we are facing a problem happen in such a market where the members constantly change

over time. We cannot guarantee the basic assumption for dynamic market so that this algorithm is not suitable for our problem where happen in the dynamic market. In the chapter 5 I propose two dynamic algorithms for the case of the dynamic market.

Chapter 5

Repair-Based Algorithm

5.1 Introduction

In this chapter, I propose two dynamic algorithms to solve the problem happen in the circumstance of dynamic market. I introduce the dynamic market and its differences to the static market. In a word, the dynamic market has more variability, comparing to the static market where it does not need to think over the situation of the market's changes happen. Then, I introduce two kinds of algorithms respectively, each of them can solve the problem, leading to a stable match. However, given the consistency of the stable match, the second algorithm, namely repair-based algorithm can lead to higher consistency of the result. Higher consistency of the result can lead to higher customer satisfaction, because less of the customer may be affected by the market's changes.

5.2 Dynamic in Temporary Labor Market

As we mentioned before, the temporary labor market is facing more variability in the members participating in market. The basic feature of two-sided matching is to try to match elements in two disjoints sets with each other. In the context of marriage problem, the number

of men and women remains relatively constant, compared to the temporary labor market. People in the marriage market do not have tendency to join or leave the market constantly in a short period of time. Things change when moving to the temporary labor market. In this context, new candidates and employers may join or leave market constantly with more possibility. Also, the stable concept is a relative one, which relies on the stability of the market. When a member of the market changes, the stability of the market will be interrupted. Existing two-sided matching models usually require stable agent population and their preferences, which is usually unattainable in large online markets. Dynamic changes are an essential characteristic of large online markets. The population of the agents, their status and preferences may change over time. There are continuously new members joining the market and members withdrawing from the market. Even the preferences of members are fine-tuned. The static market we talked before is static in a short period of time but it also can be treated as a dynamical market in long period of time. The market changes with time. Resulting to changes in the market context, leading to destabilizing of the previous stable match. In the following, it could discuss some possible changes happening in the temporary labor market.

5.2.1 New Agents Entering Market

Different from classical two-sided matching models, in continuous two-sided markets studied in this paper, there will be no fixed time frame restrictions on when the agents (providers or customers) have to enter into the market and when the market has to be cleared. New agents can join the market at any time, which will affect the stability of the existing matching. A new stable matching needs to be computed to regain the stability of the market.

5.2.2 Existing Agents Withdraw From Market

Both existing providers and customers can withdraw from the market at any time. However, a provider needs to reject the proposal she is holding before withdrawing. If she does not hold any proposal, she can withdraw freely. In the same way, if a customer is held by a provider, she can withdraw freely. However, if she is held by a provider, she will not be allowed to make a new proposal to another provider.

5.2.3 Preference Changes

Agents' preferences may change during the matching process. These type of changes are usually small. They could be a preference fine-tuning resulting from additional information obtained by the agents along the process of matching. However, in theory, preference changes will also affect the stability of the matching even though the changes are not drastic.

There can be a fourth dynamic change to the market: agents leave the market with agreements. If a provider is satisfied with the customer she is holding and does not want to wait any longer, she can accept the proposal of the customer. In this case, we say the provider and customer have reached an agreement. These agents will leave the market with a successful matching. Since we suppose that a stable matching has been generated, agents are already paired to each other meaning that there are no any two unpaired agents from each side of the market willing to break their current matching and be paired together. In this case, if two paired agents reach the agreement and leave the market, this event will not provide any more options for the remaining pairs. Therefore, the remaining pairs will not deviate from the remaining matching, in other words, the matching is still stable. Therefore, the leaving of successfully matched pairs will not affect the stability of the market. I believe that occasion happen when intermediary agency exists, because in this situation, two sided agents have no idea about the information on whether others agents join

or leave. The demand of matching is conducted by the “sending request" side. We define the market which has no intermediary agency involved as the centralized dynamical market. Similarly,the de-centralized dynamical market refers to one where there is intermediary agency.

5.3 Re-matching-Based Algorithm

5.3.1 Fixed Time Interval in Re-matching

As we know, The notion of stable matching was derived from static market settings. It is a relative one, which means, after a stable matching is reached, a market change could break the stability of the existing matching. In the setting of dynamic markets, we define stable matching as a time-related concept. When we say a matching is stable we mean the matching is stable at a particular time given the population of the agents and their preferences at that time. When the market is static, the agents who join the market stay the same, as a result, the market preferences sets still remains the same, a stable matching can be obtained via Gale and Shapley’s algorithm. However, when we thought over the dynamical market from the de-centralized aspect, we can extend the Gale and Shapley’s algorithm to meet for the changes. In the following section, we propose algorithms for obtaining time-related stable matching through re-matching

Notice that the dynamic market with respect to time. We can assume that there are several nodes in the time horizon where the market keeps static in any time interval between any two adjacent nodes. All modification occurs on the endpoint of a period of time. This assumption simulates the realistic occasion. Based on this assumption, we can re-run Gale and Shapley’s algorithm at each endpoint of a unit of time to obtain a revised stable matching after changes occur in the context of two-sided matching market. We just assume the duration of the period of time is fixed to get close to the realistic situation. Based on

this, we propose a revised algorithm, and the flowchart is in Figure 6

The corresponding notations and functions are shown in Table 12

Table 12: Explanation for the revised algorithm based on the fixed time interval

Notation	**Meanings**
C	Candidates set
E	Employers set
c_i	Candidates c_i.
e_i	Employer e_i
C_{ava}	Available candidates set
E_{ava}	Available employers set
Adjust()	Adjust preferences based on current available member
Availed(c_i)	Judge candidate c_i that whether it is available for matching

And the pseudo-code of the revised preference algorithm based on the fixed time interval is in algorithm 3

1. At line 1-5, for each candidate in the previous matching, judge its availability, if it is available for next matching operation, put it into available candidate set.

2. At line 6-8, for each candidate in new joined candidates set, put it into available candidate set.

3. At line 9-13, for each employer in previous matching, judge its availability, if it is available for next matching operation, put it into available employer set.

4. At line 14-16, for each employer in new joining employer set, put it into available employer set.

5. At line 17 generate new preference set based on current available members in market

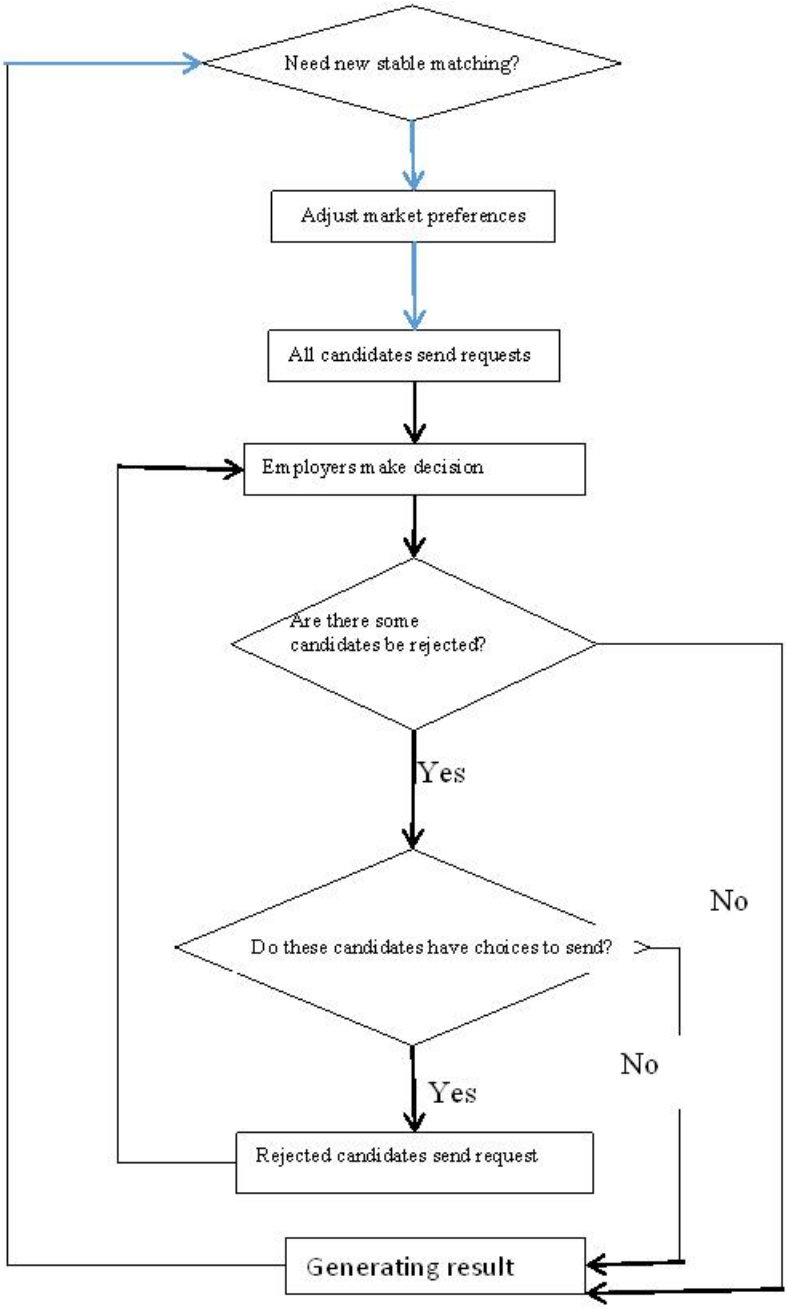

Figure 6: A revised algorithm based on the fixed time interval

Input: new candidatesC_n, new employersE_m
Output: market preference set

```
for ∀c_j ∈ C do
    if availed(c_i) = true then
        C_ava.add(c_j);
    end
end
for ∀c_i ∈ C_n do
    C_ava.add(c_j);
end
for ∀e_j ∈ E do
    if availed(c_i) = true then
        C_ava.add(c_j);
    end
end
for ∀e_i ∈ E_m do
    E_ava.add(e_i);
end
(P(C_ava), P(E_ava)) = adjust(C_ava, E_ava);
```

Algorithm 3: Pseudo algorithm for the revised preference algorithm based on the fixed time interval

After we obtained the revised preferences set via the above algorithm, we can call the Gale and Shapley algorithm to get new revised stable matching based on new context of market.

5.3.2 Flexible Time Interval in Re-matching

The previous algorithm simulates realistic situations approximately, as it assumes changes happen just on endpoints of a period of time, while changes may occurs at any time. Facing this assumption, we need to relax the restriction by changing the fixed time interval into the flexible time interval. The accuracy of the result affected by the restriction get the better if the time interval gets smaller. In the flexible time interval, the algorithm will be executed to generate revised stable matching when new changes occur. Its flow chart is in Figure 7

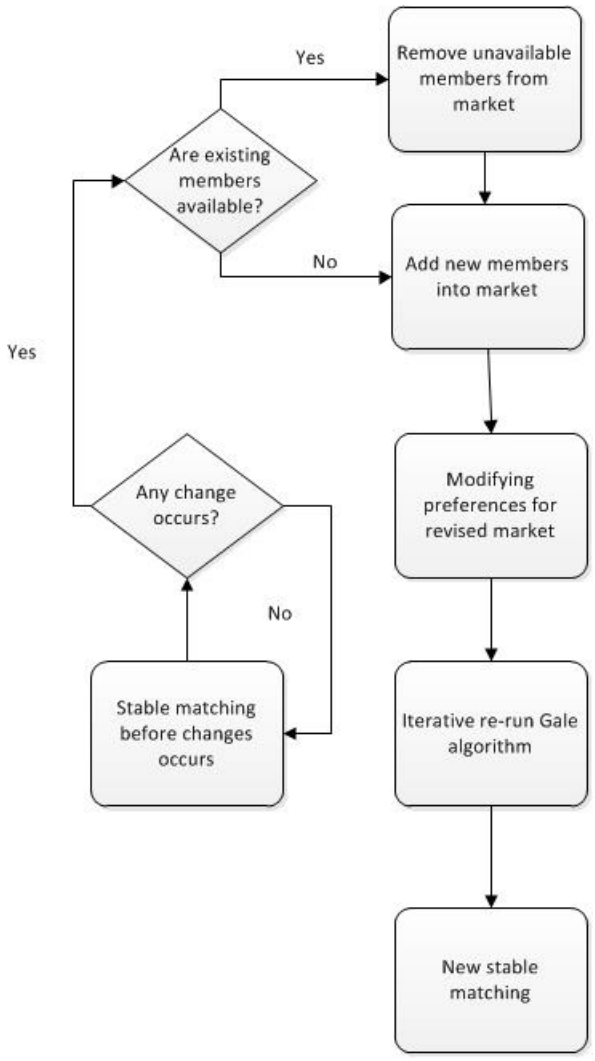

Figure 7: A revised algorithm based on flexible time interval

This algorithm from the flexible time interval aspect is similar to the previous one, the unique difference is the frequency of re-running Gale and Shapley's deferred acceptance algorithm to obtain a new stable matching when some changes occurs in the market leading to the original stable matching becoming unstable. We discuss the re-matching-based algorithm which makes use of the deferred acceptance algorithm periodically for dynamic market. Imagine that at the beginning, based on the current market, we can generate a stable matching. When detecting new changes, employers need to decide whether they want to accept the holding candidate. For those that decide to reach the agreement with their holding customers, both the providers and their customers will leave the market with agreements. Therefore they are unavailable for future rounds. Employers can, of cause, reject the holding candidate and withdraw from the market. In this situation, the candidate becomes a rejected one, and they need to decide whether they remain in the market for next round of matching. If they decide to leave, then they are unavailable and need to be remove from the market member set. At the end of each time interval, the remaining agents and newly entered agents form an updated market. At the beginning of the next interval, the agents in the new market will rematch according to the deferred acceptance algorithm, based on the updated agent population and preferences. The length of the interval can be adjusted to reflect the designer's preference over the responsiveness of the market to dynamic changes. An extreme case is that the deferred acceptance algorithm is rerun whenever a change occurs. In general, this re-matching-based algorithm does not obtain time-related stable matching in a real-time manner. However, it can approximate the solution concept by using a sufficient small time interval.

5.4 Repair-Based Algorithm

5.4.1 Result Consistency

The re-matching-based algorithm can approximate time-related stable matching. However, since it recalculates the whole process from beginning of each interval, the revised stable matching result is not a derived one based on the previous one. As a result, the re-matching-based algorithm could not generate a stable matching and also remain the consistency between the previous matching and the revised matching. In my opinion result consistency also plays an important role in evaluating the result besides its stability. For example, a stable matching with high consistency can interfere with less agents, otherwise, those agents may think the recommendation from the agency is volatile, which would affect their confidence for said agency. To measure it, we bring the notion of consistency to our model. Consistency can be measured by the difference between the current stable matching with the last stable matching. The reason that we should think about this is that high levels of consistency can improve customer satisfaction, because we can help them save time instead of beginning another contract with a new agent. High level of consistency ensures that the original agent's matching can be distributed as little as possible. But the disturbing due to the changes in the market can not be stopped completely. The new changes which occur in the market leads to changes in stable matching which can be treated as disturbing for original matching. The fewer members who change their matches, thus affecting the market, the more satisfied the customers will be due to a high degree of consistency. The following (8) shows how to calculate consistency.

$$Result\_consistency = \frac{num\_onverlapped\_pairs}{num\_pairs\_updated\_market} \tag{8}$$

Where $num\_onverlapped\_pairs$ is the number of same pairs between the original market and the revised market; the $num\_pairs\_updated\_market$ is the number of pairs of the

revised market after changes occurs.

5.4.2 Improving Result Consistency

It is unrealistic to rerun the algorithm constantly, because we need to think about such as the fact that while sometimes candidates and employers were pairs in the last stable matching, they have not made the decision to sign a contract and are still available. At that time, we need to make such pairs as consistent as possible. The improved way is to repair the unstable part after the preferences change rather than rerun the whole operation to obtain a new stable matching.

When the temporary labor market changes, the previous stable matching becomes unstable. It is critical to adjust the available members in the market and modify their corresponding preferences. After that, we can find which candidates need to execute the operation of “sending request" by analyzing previous stable matching structure instead of breaking up the original pair and force them to re-execute “sending request". It is obvious that there exists several pair candidates and employers who are still available for the next matching. At the same time, some agents (candidates or employers) may not be available and are going to be deleted from the available list. Besides, some new agents are going to join in the market; therefore, they will be added into available list. To repair this unstable matching, it is easy to divide all candidates into two parts, candidates from one part are already in pair with some employers, they are stable pairs from the previous matching. One thing we know is that the match, which consists of these pairs, is not stable, but close to. This is due to the fact that these pairs may not change, even if a new stable matching is generated by the re-matching algorithm. I think this part of candidates is held by corresponding landlord temporarily. The other part of the candidates, they are broken up and single. As the result, these parts of candidates are what we need to focus on. It can save more time to allow part of the candidates to send requests rather than allow all of them to

do so. Repairing unstable matching does not involving the operation again but to do the smallest possible operations.

5.4.3 De-centralized Dynamical Market

I propose a repair-based algorithm to calculate a new stable matching to meet for the dynamic changes occur in market. The algorithm will start repairing a disrupted stable matching whenever a dynamic change occurs. In this case, employers may have the tendency to hold the request from candidates side until receive better one. To be fair to the candidates, as a market rule, it is reasonable to request that the receiver side(employers) is obligated to either accept or reject its holding request from sender side(candidates). Since candidates held by employers are obligated to be paired with the holding employer if the employer accepts the request, in the algorithm design, we only allow rejected and newly entered candidates to initiate a proposal. The reason for this is because we assume there is intermediate agency in the market, which we called it “de-centralized dynamical market". In this kind of market, only the sender side’s requirement drive the execution of “sending request". We would discuss the difference between de-centralized dynamical market and centralized dynamical market in the latter part.

For the purpose of easy representation, without losing generality, we assume a discrete time line consists of small time intervals. The size of the interval can be as small as the designer wishes to capture real-time dynamic changes. Suppose that a set of dynamic changes occur at time interval *k*. We define S^{k-1} as the matching solution at interval *k-1*, New_e^k as the set of providers entered into the market at interval *k* and New_c^k as the set of candidates entered into the market at interval *k*. All employers without an agreement at *k-1* is denoted as NoA_e^{k-1}; all candidates rejected at *k-1* is denoted as $Rejected\_t^{k-1}$. Candidates are not held by employers are called *free candidates*, denoted as $Free\_c^k$. For a candidate *c*, if a employer has never rejected it, the employer belongs to the available

employers set of candidate *c*, denoted as $Avai\_e_c^k$. At each iteration of matching, *holding_-* c_e stores the most preferred proposal that is held by the employer *l*.

The flowchart Figure 8 show us how to adjust the changes of agents in the market. After that we use repair-based algorithmFigure 9 recursively to generate a revised stable matching with higher result consistency.

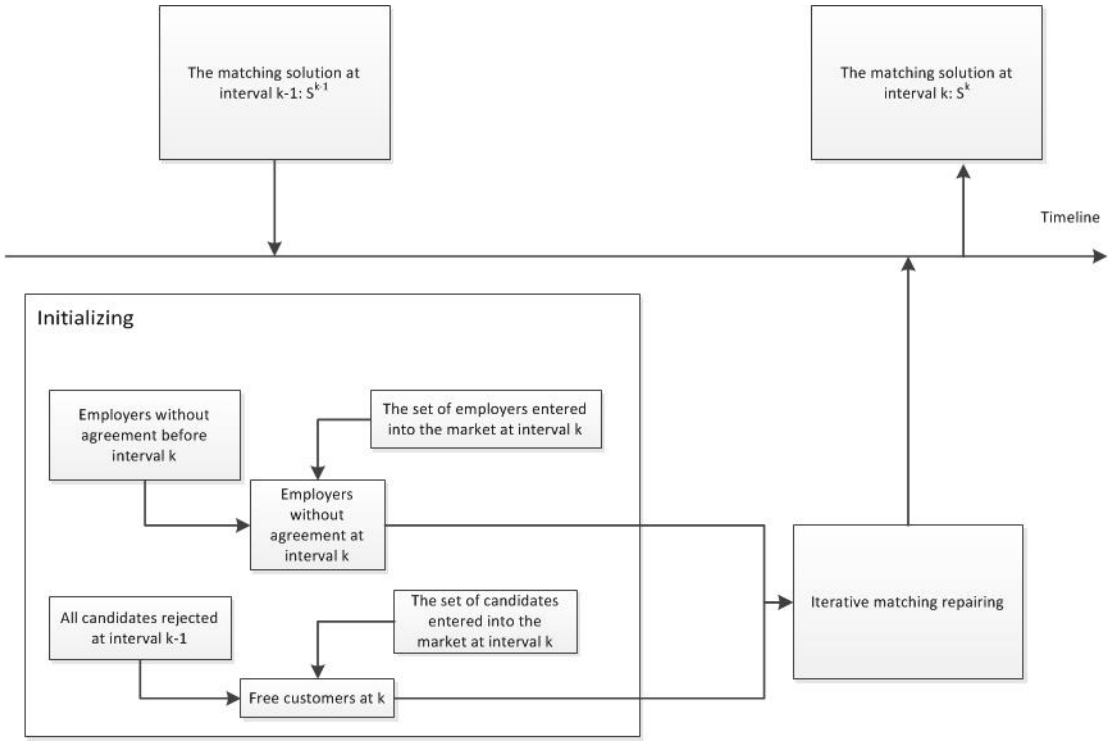

Figure 8: Repair-based algorithm flow chart 1

The flowing explanation can help explain the algorithm.

1. Line 1 is initialization, it can obtain those still available agent in interval k
2. Line 2 - 7: Adjusting available agents set by union added agents in interval *k*
3. Line 3: Obtaining the rejected candidates in interval *k-1*
4. Line 7: Free candidates on interval *k*

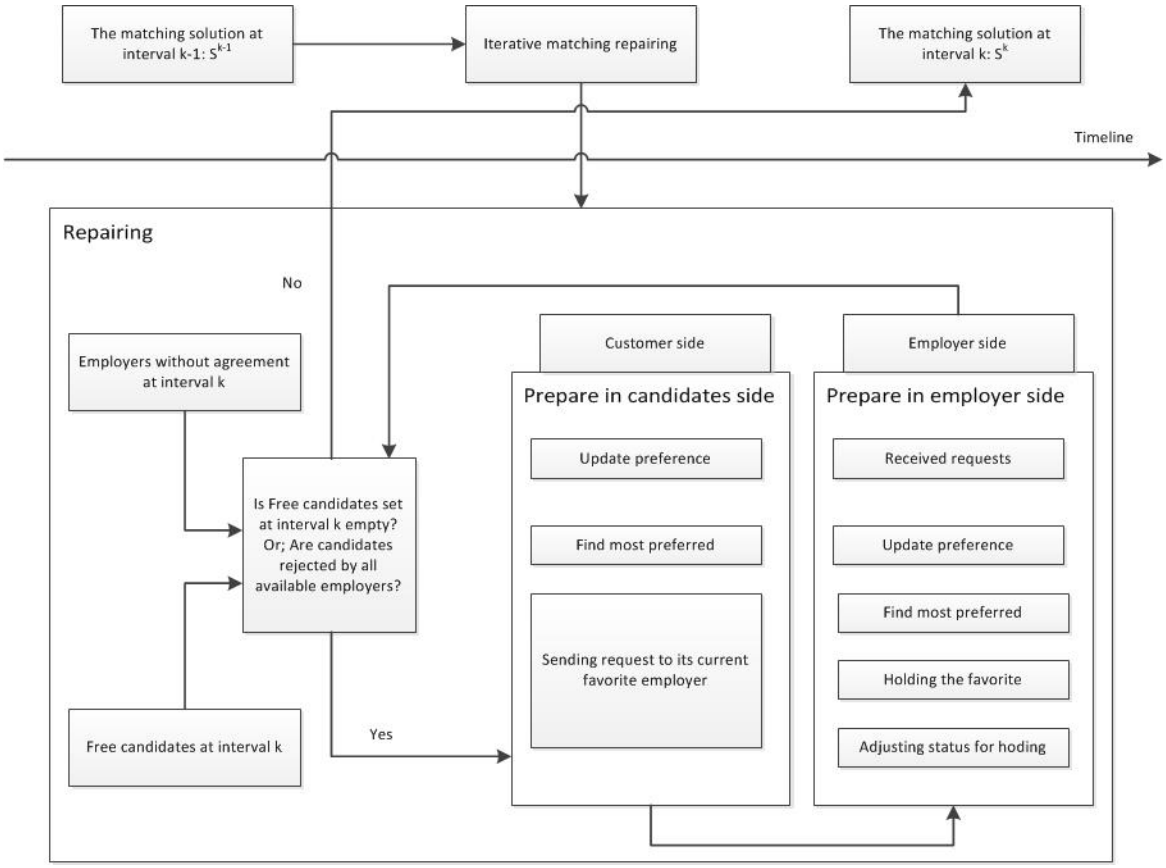

Figure 9: Repair-based algorithm flow chart 2

Input: new candidatesNew_c^k, new employersNew_e^k, S^{k-1}
Output: S^k

$NoA\_e^{k-1} = getNoAgreementProviders(S^{k-1})$;
$NoA\_e^k = NoA\_e^{k-1} \cup New\_e^k$;
$Rejected\_c^{k-1} = getRejectedcustomers(S^{k-1})$;
for $\forall c \in NoA\_c^{k-1}$ **do**
    $holding_{t_l} = getHoldingCustomer(c, S^{k-1})$;
end
$Free\_c^k = Rejected_c^{k-1} \cup New\_c^k$
for $\forall c \in Free\_e^k$ **do**
    $Ava\_e_c^k = NoA\_e^k$;
end
while $Free\_c^k = \varnothing or Avai\_e_c^k = \varnothing for \forall c \in Free_c^k$ **do**
    for $\forall c \in Free\_c^k$ **do**
        $update(Prefer(t))$;
        $find\_most\_prefered(Prefer(c), Avai\_c_t^k)$;
        $send\ request\ to\ most\ preferred\ e$;
        for $\forall e \in NoA_e^k$ **do**
            $Received\_c_e \longleftarrow \{receivedrequests\} \cup holding\_c_e$;
            $update(Preference_e)$;
            if $Received\_c_e \neq \varnothing$ **then**
                $find\_most\_prefered(Prefer(e), Received\_c_e)$;
                $holding\_c_e = mostpreferedc$;
                for $\forall c \in Received\_c_e - holding\_c_e$ **do**
                    $remove\ e\ from\ Avai\_e_t^k$;
                    $add\ c\ to\ Free\_c^k$;
                end
            end
        end
    end
end
Return$S^k = \{(e, holding\_c_e) | e \in NoA_e^k\}$;

Algorithm 4: Pseudo algorithm for the repair-based algorithm

5. Line 8-10: Initializing available employers for free candidates

6. Line 11-29: Iterative matching repairing

5.4.4 Centralized Dynamical Market

It is necessary to mention that the repair-based de-centralized dynamic matching algorithm does not accommodate preference changes from candidates' side. Once a candidate has submitted a proposal to a provider and the proposal is on hold (not rejected), the candidate is not free anymore. They cannot withdraw their proposal even their preference has changed. I made this assumption due to the fact that I think when there has intermediate agency in market, agency will not let the original stable pair break to contact with the new joined employers. The operation of "sending request" is conducted by free candidates. In special cases, only one candidate joins the de-centralized dynamic market will lead to the original stable matching breaking, because there is no free candidate existing in current market. However, things may change when it is conducted in centralized dynamic market. The centralized dynamic market refers to no intermediate agency present in the market. It means that the candidate side knows which new agents joined the other side, which is different from the mechanism in de-centralized dynamic market. The candidates side does not know the occasions of participation in de-centralized dynamical market where only intermediate agency understands these occasions from both two sides. Because of this, even only if one employer joins the market,some original stable pair still encounter the possibility to break up pair for the new employer, when the candidate prefers the new employer to the one assigned to it on the previous time interval by algorithm.

Therefore, the basic idea to adjust the free candidate list, when the preferences modification is finished. The adjustment process needs to make a judgment to see if the candidate prefers the new employers to its current partner. As a result, we can summarize that there are following ways to lead free candidates appear.

1. New candidate join into our market;

2. Existing employers in stable pair of previous stable matching leave our market;

3. New employers join into our market so that some candidate in one stable pair even prefer to them.

Based on above, I propose another algorithm to carter for this kinds of situation. The algorithm flow chart is in Figure 10Figure 11Figure 12

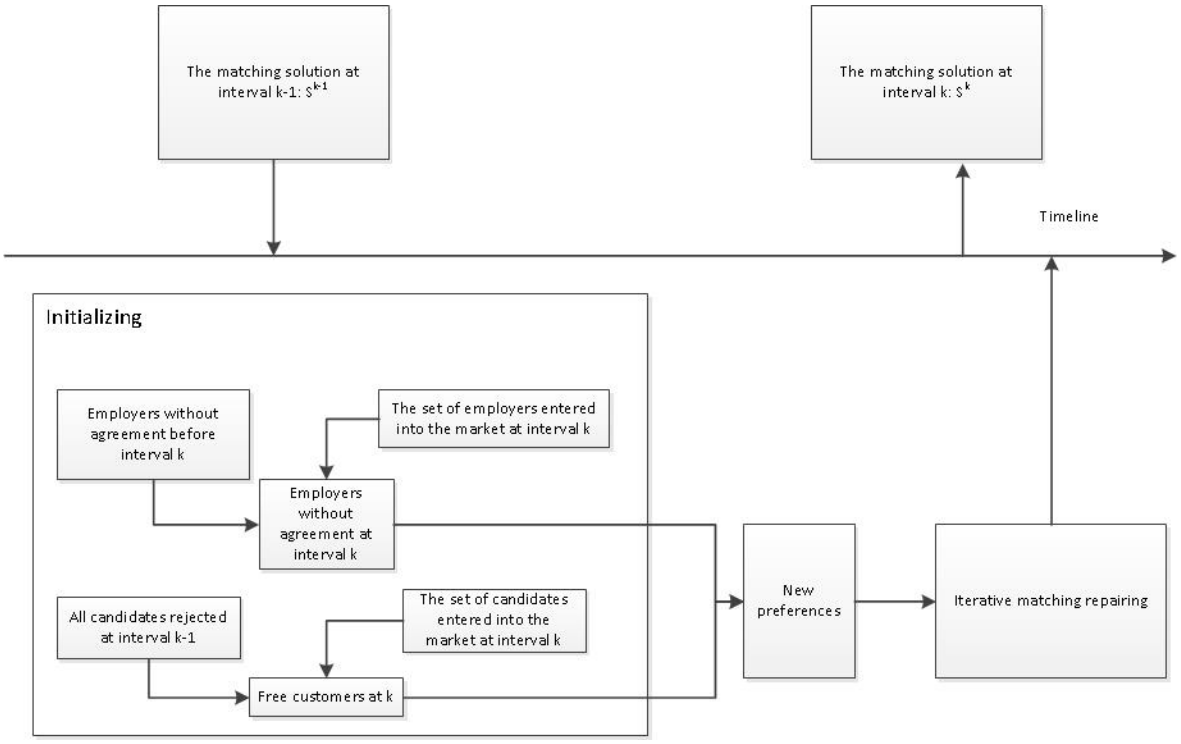

Figure 10: Repair-based algorithm flow chart for centralized dynamical market 1

In the centralized dynamic market, the issue we face is that any change occurring in the market also leads to the original stable matching breaking up. As we discuss before, if the dynamic market has the third part who plays a role as the intermediate agency, which means neither one of two sides does not know the changes of opposite side agents. At that time,

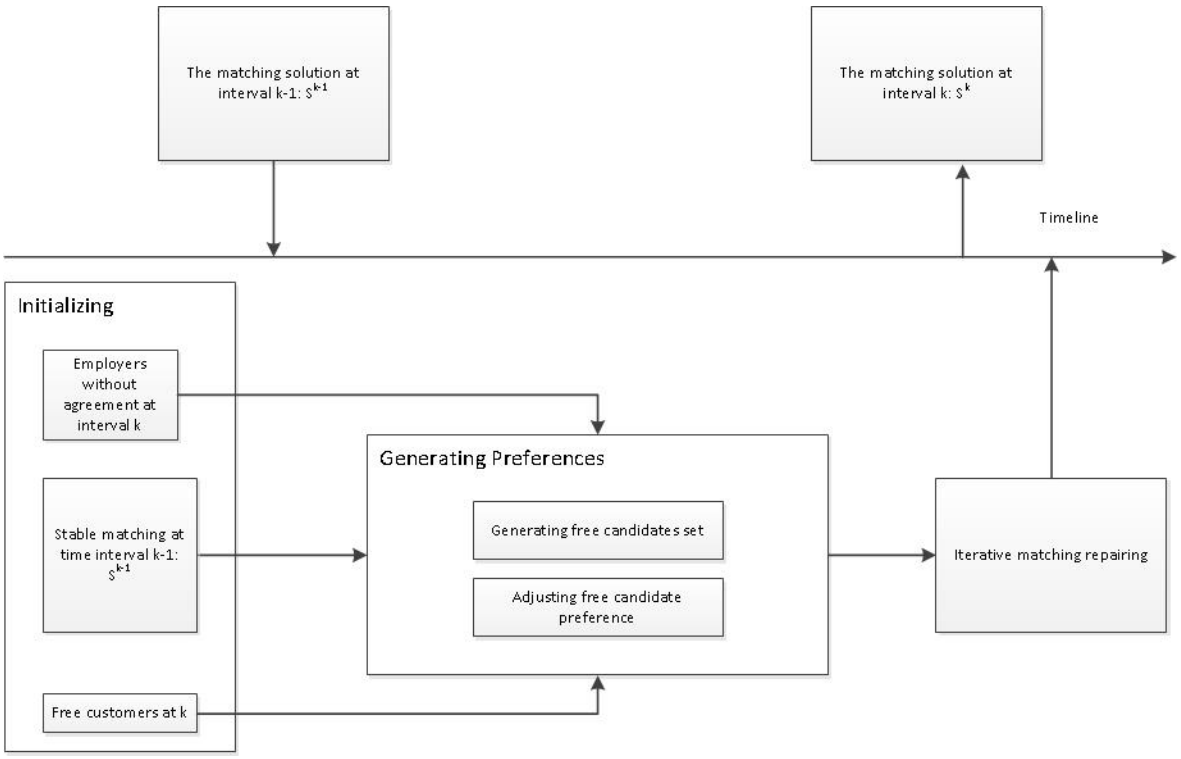

Figure 11: Repair-based algorithm flow chart for centralized dynamical market 2

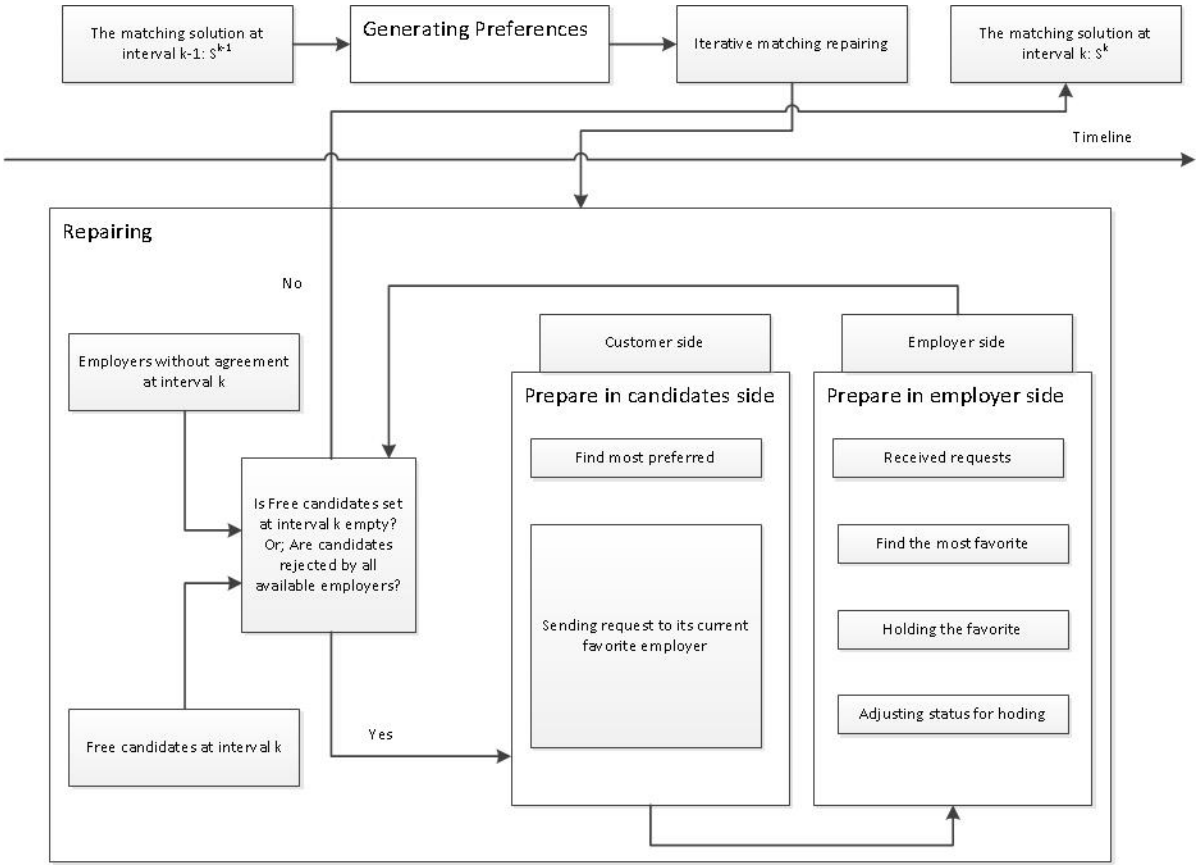

Figure 12: Repair-based algorithm flow chart for centralized dynamical market 3

the participation happen on employers side (requests receiver) does not lead to the original stable matching breaking up. However, in the centralized dynamic market, namely, there is not intermediate agency in it. As a result, we add the operation that compares current matching agents of candidates (request sender side) to new joined employers (request receiver side). If some candidate prefers one new employer to its current matching agent, the pair also needs to be broken up, to let the candidate have a chance to send the request to the new employer. This is different from the operation in de-centralized dynamical market.

5.4.5 Theorem

Theorem 3. *Given that a stable matching has been generated for a market, the repair-based matching algorithm is able to compute a new stable matching when new agents enter into the market, existing agents withdraw from the market and providers change their preferences.*

Proof. The theorem is proven under the following two cases:

Case#1: New agents enter the market and existing agents leave the market. In this case, the algorithm updates the set of all providers without an agreement, $NoA_e^k$and free customers who are not held by providers, $Free_e^k$Since the algorithm only allows free customers to initiate proposals, and the iterative matching module of the algorithm resembles the deferred acceptance algorithm, when the algorithm terminates, that is the set of free customers is empty or the set of available providers for any free customers is empty, there will be no more new proposals to be made. Therefore, the resulting matching is stable.

Case #2: Providers change their preferences. In the iterative repairing module of the algorithm, during each iteration, all providers have the opportunity to re-evaluate their current holding candidate and newly received customer proposals. Should an employer's preference change during any iteration, they can reject the current holding customer and switch

to a new one. After the switch, the rejected customer will be equivalent to a newly entered customer and go through the procedure described in Case#1 to find another provider. Therefore the process will terminate with a stable matching. □

5.5 Summary

In this chapter, I use two kinds of dynamic algorithms to solve the dynamic problem resulting from the dynamic market. One of them is more straightforward to propose. It is simple to keep repeating the same operation of the deferred acceptance algorithm. When some changes happen in the market. I called this method the re-matching algorithm that can react to generate a new stable match after some changes happen in the market. But it has an obvious shortcoming that those customers keep receiving some different recommendations from the system. These recommendations, since they may change over time, can be a disturbance to customers. This is due to the fact that some others' behavior may change the status of the current market, causing the original stable match to become unstable. The system repeatedly runs the algorithm to generate a new stable match and informs all of the customers. In such a circumstance, a large part of customers may be disturbed by the recommendations as a new stable match is generated. This, in turn, may affect the customers' satisfaction. Therefore, based on the re-matching-based algorithm, I propose the repair-based algorithm to try to increase the consistency of the stable match. This algorithm aims to further improve the customer satisfaction.

Next, we tackle to evaluate the effectiveness of the algorithms, by conducting some experiments to compare these algorithms in chapter 6.

Chapter 6

Performance Evaluation

6.1 Introduction

In this chapter, I first aim to compare the deferred acceptance algorithm with the request-accumulated deferred acceptance algorithm in terms of the running time and the number of times operations are executed. I then want to compare the re-matching-based algorithm with the repair-based algorithm in terms of the consistency of the result. Finally, I mention the best result consistency when I specified a concrete situation where We can find the result consistency in the repair-based algorithm is also close to the best situation in the same circumstance.

6.2 Deferred Acceptance Algorithm and RADA Algorithm

The RADA algorithm is employed to minimize the execution time of the algorithm. The advantage of this algorithm is that it can minimize the number of times of execution for the operation “decision making" without increasing the times of operation “sending request". we try to record the execution time of the program, and count the times of execution of operations.

6.2.1 Experiment 1

Assuming we have a marriage market, with 10 men and 10 women. For the sake of simplicity, we set the number of men and women equal to each other. Their preferences are in Table 13

Table 13: Preferences for 10 * 10 marriage model

Women ID	**Woman Preferences**	**Men ID**	**Man Preferences**
0	[8, 6, 7, 4, 2, 0, 9, 3, 1, 5]	0	[3, 4, 2, 6, 0, 7, 1, 8, 9, 5]
1	[5, 9, 2, 3, 4, 0, 7, 1, 8, 6]	1	[6, 4, 9, 2, 8, 3, 0, 5, 1, 7]
2	[2, 4, 0, 5, 3, 6, 8, 7, 1, 9]	2	[8, 3, 9, 6, 2, 4, 7, 0, 5, 1]
3	[8, 1, 9, 5, 6, 7, 3, 0, 4, 2]	3	[9, 3, 1, 2, 0, 8, 7, 5, 6, 4]
4	[1, 3, 8, 4, 7, 0, 2, 6, 5, 9]	4	[5, 9, 0, 1, 3, 2, 8, 7, 4, 6]
5	[3, 5, 2, 4, 9, 6, 7, 1, 8, 0]	5	[0, 8, 7, 4, 9, 2, 3, 1, 5, 6]
6	[2, 4, 6, 0, 1, 7, 9, 5, 3, 8]	6	[4, 6, 1, 7, 9, 2, 0, 5, 8, 3]
7	[4, 9, 5, 0, 7, 1, 8, 2, 3, 6]	7	[1, 4, 7, 8, 3, 5, 6, 9, 0, 2]
8	[5, 3, 7, 1, 2, 4, 8, 0, 9, 6]	8	[7, 0, 1, 6, 5, 2, 9, 8, 3, 4]
9	[5, 4, 9, 6, 0, 7, 3, 1, 8, 2]	9	[3, 2, 7, 1, 9, 6, 4, 8, 5, 0]

We run the deferred acceptance algorithm 10 times to obtain the average time of execution. The man-sided optimal stable matching for running the algorithm ten times is in Table 14

Table 14: 10 * 10 Man optimal stable matching

Woman ID	0	1	2	3	4	5	6	7	8	9
Man ID	5	7	0	9	1	4	6	8	2	3

The executing time for 10 times running of the algorithm is in Table 15

And then, we use the request-accumulated deferred acceptance algorithm to work with

Table 15: Running time for 10 * 10 Man optimal stable matching

Time (Milliseconds)	1.23	1.23	1.21	1.19	1.19	1.24	1.21	1.25	2.25	1.19

the same preferences in Table 13, and find that the result is the same, which because the request-accumulated deferred acceptance algorithm also can obtain male optimal stable matching, and the male optimal stable matching the unique for the same set of preferences. We also record the running time below.

Table 16: Request-accumulated algorithm running time for 10 * 10 Man optimal stable matching

Time (Milliseconds)	1.33	1.61	1.45	1.42	1.19	1.38	1.21	1.34	1.46	1.37

We can notice that the average running time for the request-accumulated deferred acceptance algorithm is *1.376* milliseconds while the one for deferred acceptance algorithm is *1.319* milliseconds. The running times are similar to each other. At the same time, the variance for RADA algorithm results is *0.32* and the variance for the normal algorithm is *0.12*. It means the RADA algorithm has more stability.

In addition to this, we count the times operations are executed, specially “sending request" and “decision making". The result is that the normal deferred acceptance algorithm executes the operation “sending request" 14 times, and “decision making" 50 times, while the RADA algorithm runs the same number of times on the operation “sending request" but only 4 times on the “decision making". Due to this, I infer that when the size gets bigger, the running time should be less via the RADA algorithm.

6.2.2 Experiment 2

We randomly expand the size the set of men and the set of women, n =30, 40, 50, 60. The corresponding preferences are in Table 17

The 30 * 30 male-sided optimal stable matching is in Table 21 and the contradistinction of two algorithms in terms of running time is in Table 22. From Table 22, we can notice that the average running time for the deferred acceptance algorithm is *52.61* milliseconds with stand deviation of *6.22*, while the RADA algorithm's running time is *8.45* milliseconds with standard deviation of *1.01*. The reason for this is that we need to execute "sending request" operation *45* times and "decision making" operation up to *2790* times via deferred acceptance algorithm, while the other algorithm executes "sending request" operation up to *45* times, too, but "decision making" operation only *111* times.

The 40 * 40 man optimal stable matching result is in Table 23 and the contradistinction of two algorithms in terms of running time is in Table 24. We can notice that the average running time for the deferred acceptance algorithm is *45.02* milliseconds with stand deviation of *5.88*, while the RADA algorithm's running time is *9.03* milliseconds with stand deviation of *1.01* The reason for this is that we need execute "sending request" operation for *214* times and "decision making" operation up to *2400* times via deferred acceptance algorithm, while other algorithm executes "sending request" operation up to *214* times, too, but "decision making" operation only *157* times.

The running time situations are similar, when size = 50, 60. When size = 50, the deferred acceptance algorithm averagely need *29.78* milliseconds, *163* times "sending request" and *1000* times "decision making" to get result, while the other need *5.73* milliseconds, *163* times "sending request" and *92* times "decision making" to run. When size = 60, the deferred acceptance algorithm needs, on average, *135.48* milliseconds, *360* times "sending request" and *7020* times "decision making" to get the result, while the other needs *17.39* milliseconds, *360* times "sending request" and *275* times "decision making" to run.

Table 17: Preferences for 30 * 30 marriage model

Women ID	**Woman Preferences**	**Men ID**	**Man Preferences**
0	[13, 23, 27, 8, 24, 7, 28, 22, 2, 25, 3, 12, 15, 16, 19, 9, 17, 20, 0, 29, 6, 14, 4, 11, 26, 21, 5, 1, 10, 18]	0	[26, 27, 8, 22, 13, 24, 0, 20, 25, 11, 4, 29, 18, 17, 19, 12, 1, 14, 2, 7, 9, 28, 21, 6, 3, 5, 16, 23, 10, 15]
1	[14, 5, 2, 21, 7, 1, 12, 28, 24, 17, 13, 15, 20, 26, 22, 8, 19, 29, 11, 23, 16, 10, 27, 6, 9, 25, 3, 0, 4, 18]	1	[22, 18, 23, 4, 13, 2, 19, 28, 17, 14, 15, 7, 10, 5, 29, 16, 25, 8, 21, 9, 11, 3, 1, 20, 27, 6, 12, 0, 24, 26]
2	[17, 23, 3, 15, 25, 14, 19, 12, 29, 0, 4, 22, 20, 10, 28, 18, 8, 16, 27, 13, 2, 26, 7, 21, 11, 5, 1, 24, 9, 6]	2	[9, 8, 6, 17, 0, 18, 10, 20, 26, 3, 27, 14, 29, 12, 4, 21, 13, 22, 25, 5, 7, 24, 2, 15, 1, 23, 11, 28, 16, 19]
3	[19, 18, 13, 20, 9, 22, 27, 29, 26, 24, 25, 10, 5, 6, 4, 3, 17, 8, 0, 16, 23, 7, 12, 2, 15, 21, 14, 1, 11, 28]	3	[14, 24, 16, 18, 20, 4, 7, 26, 11, 22, 23, 25, 28, 13, 2, 0, 3, 5, 17, 29, 15, 6, 10, 8, 21, 12, 9, 27, 1, 19]
4	[14, 12, 13, 18, 15, 17, 23, 26, 8, 29, 22, 5, 7, 3, 21, 0, 9, 4, 6, 2, 10, 16, 1, 19, 20, 27, 24, 11, 28, 25]	4	[24, 3, 17, 22, 23, 25, 2, 29, 6, 27, 12, 19, 8, 21, 15, 11, 14, 13, 10, 5, 9, 4, 18, 7, 26, 16, 1, 0, 28, 20]
5	[1, 18, 20, 10, 23, 16, 28, 26, 8, 21, 9, 7, 27, 5, 13, 11, 4, 15, 2, 24, 22, 29, 0, 6, 19, 25, 3, 14, 17, 12]	5	[16, 9, 23, 28, 4, 20, 21, 1, 19, 24, 14, 26, 25, 8, 6, 17, 2, 0, 5, 7, 3, 18, 11, 22, 12, 10, 13, 27, 15, 29]
.....	.....	.....	.....
27	[17, 29, 0, 20, 25, 10, 7, 9, 22, 13, 15, 11, 19, 27, 21, 14, 5, 28, 6, 4, 24, 3, 16, 26, 23, 8, 18, 2, 12, 1]	27	[23, 26, 27, 12, 18, 25, 17, 3, 11, 24, 29, 16, 4, 0, 2, 21, 28, 7, 13, 5, 8, 14, 9, 20, 19, 15, 6, 1, 22, 10]
28	[29, 14, 7, 13, 17, 1, 15, 5, 28, 4, 12, 16, 20, 21, 19, 26, 27, 6, 23, 24, 11, 9, 10, 18, 8, 3, 2, 25, 22, 0]	28	[22, 4, 24, 10, 6, 9, 8, 26, 17, 5, 11, 15, 13, 23, 14, 20, 21, 28, 0, 2, 16, 1, 7, 19, 18, 12, 3, 29, 25, 27]
29	[10, 24, 21, 12, 1, 28, 13, 5, 3, 0, 29, 16, 18, 7, 15, 25, 23, 26, 2, 19, 8, 11, 9, 14, 22, 27, 6, 17, 4, 20]	29	[17, 23, 15, 0, 29, 10, 28, 25, 1, 11, 21, 18, 27, 26, 13, 20, 14, 12, 16, 3, 7, 6, 2, 22, 5, 4, 8, 19, 9, 24]

Table 18: Preferences for 40 * 40 marriage model

Women ID	**Woman Preferences**	**Men ID**	**Man Preferences**
0	[4, 18, 35, 30, 38, 14, 19, 36, 15, 33, 13, 29, 3, 39, 10, 8, 9, 32, 24, 26, 0, 7, 2, 34, 5, 22, 11, 17, 1, 28, 37, 25, 16, 27, 23, 31, 6, 21, 20, 12]	0	[6, 7, 4, 38, 3, 18, 33, 39, 27, 36, 15, 20, 32, 28, 12, 17, 0, 31, 21, 30, 16, 34, 26, 8, 23, 22, 14, 24, 1, 9, 29, 37, 11, 10, 25, 13, 2, 5, 19, 35]
1	[11, 5, 30, 15, 22, 32, 9, 23, 7, 13, 20, 24, 16, 10, 33, 14, 28, 21, 38, 8, 3, 29, 19, 34, 36, 37, 31, 26, 2, 18, 27, 4, 17, 25, 6, 0, 35, 12, 39, 1]	1	[28, 37, 32, 14, 5, 3, 26, 16, 7, 21, 10, 31, 8, 27, 29, 15, 18, 35, 6, 2, 24, 17, 30, 4, 19, 34, 1, 38, 39, 25, 20, 13, 12, 22, 11, 33, 9, 23, 36, 0]
2	[15, 34, 27, 29, 36, 32, 18, 23, 0, 37, 19, 39, 8, 33, 5, 31, 17, 28, 25, 2, 20, 4, 10, 7, 24, 1, 30, 13, 12, 6, 38, 26, 21, 16, 35, 22, 11, 3, 14, 9]	2	[15, 34, 27, 29, 36, 32, 18, 23, 0, 37, 19, 39, 8, 33, 5, 31, 17, 28, 25, 2, 20, 4, 10, 7, 24, 1, 30, 13, 12, 6, 38, 26, 21, 16, 35, 22, 11, 3, 14, 9]
3	[14, 24, 22, 11, 30, 20, 31, 18, 16, 27, 2, 19, 3, 0, 15, 36, 10, 39, 1, 25, 21, 23, 4, 28, 38, 6, 26, 9, 17, 37, 34, 7, 33, 35, 13, 8, 5, 32, 12, 29]	3	[22, 9, 18, 2, 4, 15, 35, 36, 25, 31, 11, 34, 20, 5, 21, 39, 1, 24, 12, 30, 0, 29, 28, 32, 38, 23, 33, 26, 17, 37, 3, 19, 7, 27, 10, 16, 6, 13, 14, 8]
. . . . .	. . . . .	. . . . .	. . . . .
37	[9, 22, 0, 27, 20, 6, 10, 18, 8, 16, 25, 26, 3, 13, 39, 19, 31, 15, 7, 5, 32, 17, 1, 12, 37, 14, 35, 2, 29, 4, 21, 38, 23, 34, 30, 24, 11, 33, 28, 36]	37	[1, 10, 6, 37, 14, 31, 36, 24, 30, 21, 5, 15, 19, 18, 28, 7, 2, 27, 13, 0, 9, 3, 25, 32, 26, 34, 20, 23, 16, 12, 22, 39, 29, 35, 38, 11, 17, 8, 4, 33]
38	[3, 36, 21, 28, 31, 23, 24, 34, 18, 20, 2, 39, 25, 8, 33, 4, 35, 22, 5, 14, 0, 26, 9, 13, 30, 12, 7, 1, 15, 17, 27, 16, 37, 11, 10, 38, 32, 29, 6, 19]	38	[27, 11, 16, 8, 1, 31, 10, 20, 39, 24, 32, 2, 3, 5, 34, 12, 29, 37, 18, 7, 19, 14, 15, 13, 35, 22, 30, 23, 4, 17, 26, 36, 0, 25, 28, 33, 6, 9, 38, 21]
39	[7, 9, 27, 6, 16, 10, 0, 34, 3, 28, 29, 25, 13, 26, 11, 15, 20, 18, 38, 37, 30, 31, 1, 19, 2, 35, 33, 8, 24, 5, 17, 32, 39, 36, 21, 14, 4, 23, 22, 12]	39	[21, 20, 4, 17, 10, 14, 0, 28, 22, 38, 31, 18, 19, 33, 1, 29, 7, 11, 32, 6, 26, 24, 13, 12, 30, 5, 3, 27, 23, 37, 34, 8, 36, 25, 2, 15, 16, 39, 9, 35]

Table 19: Preferences for 50 * 50 marriage model

Women ID	**Woman Preferences**	**Men ID**	**Man Preferences**
0	[31, 43, 35, 23, 14, 0, 5, 28, 26, 36, 48, 27, 40, 19, 3, 41, 25, 12, 44, 8, 4, 22, 21, 39, 30, 46, 45, 13, 11, 38, 32, 47, 2, 6, 9, 42, 24, 17, 16, 49, 37, 15, 1, 34, 20, 29, 10, 18, 7, 33]	0	[18, 39, 16, 6, 22, 9, 36, 31, 11, 34, 37, 14, 44, 26, 23, 43, 30, 12, 38, 48, 15, 25, 42, 7, 29, 35, 45, 3, 49, 27, 8, 40, 46, 13, 4, 1, 47, 2, 28, 17, 41, 0, 20, 33, 21, 5, 19, 24, 10, 32]
1	[33, 43, 38, 9, 10, 41, 47, 30, 49, 32, 25, 13, 42, 0, 37, 31, 23, 1, 5, 21, 22, 8, 35, 44, 48, 6, 12, 36, 14, 45, 2, 34, 19, 40, 39, 16, 7, 28, 29, 26, 20, 24, 11, 17, 46, 15, 4, 18, 27, 3]	1	[47, 19, 12, 26, 30, 28, 38, 2, 33, 21, 9, 18, 40, 46, 49, 1, 42, 10, 29, 5, 41, 20, 3, 6, 11, 48, 44, 32, 27, 14, 34, 45, 23, 36, 15, 39, 0, 37, 16, 31, 13, 22, 7, 8, 24, 4, 35, 25, 43, 17]
2	[0, 14, 43, 23, 20, 17, 39, 1, 31, 10, 13, 12, 7, 25, 36, 49, 40, 16, 22, 46, 33, 38, 19, 35, 4, 18, 26, 15, 32, 3, 11, 6, 37, 48, 34, 42, 45, 2, 21, 27, 41, 47, 24, 29, 44, 28, 5, 8, 9, 30]	2	[13, 19, 31, 15, 5, 9, 1, 32, 48, 24, 36, 29, 34, 7, 23, 2, 42, 21, 35, 30, 3, 39, 33, 18, 45, 17, 25, 49, 8, 44, 28, 12, 22, 43, 27, 14, 38, 41, 26, 20, 4, 16, 40, 11, 46, 6, 10, 0, 47, 37]
.....	.....	.....	.....
48	[3, 43, 36, 21, 6, 42, 34, 29, 38, 41, 4, 39, 33, 17, 11, 10, 27, 26, 5, 25, 18, 40, 0, 37, 13, 28, 8, 20, 49, 44, 30, 48, 14, 22, 31, 32, 1, 16, 15, 24, 23, 2, 35, 47, 9, 12, 45, 7, 19, 46]	48	[28, 31, 37, 9, 14, 7, 22, 10, 42, 19, 33, 3, 6, 43, 46, 1, 39, 45, 11, 24, 15, 48, 16, 12, 40, 49, 17, 25, 21, 27, 29, 8, 26, 5, 35, 36, 38, 20, 4, 41, 44, 32, 0, 47, 2, 18, 30, 13, 34, 23]
49	[16, 24, 10, 31, 38, 9, 45, 12, 26, 39, 14, 22, 17, 23, 37, 1, 4, 19, 7, 20, 8, 30, 46, 28, 21, 47, 33, 18, 6, 27, 44, 49, 3, 41, 42, 32, 40, 35, 15, 34, 25, 36, 29, 48, 11, 5, 0, 43, 13, 2]	49	[21, 6, 32, 9, 4, 48, 38, 26, 3, 41, 46, 36, 17, 18, 13, 25, 14, 31, 29, 8, 20, 35, 5, 22, 23, 42, 44, 12, 27, 33, 15, 28, 30, 39, 45, 10, 43, 37, 19, 16, 2, 24, 0, 34, 7, 11, 1, 49, 40, 47]

Table 20: Preferences for 60 * 60 marriage model

Women ID	**Woman Preferences**	**Men ID**	**Man Preferences**
0	[39, 38, 34, 50, 44, 36, 48, 25, 13, 47, 18, 12, 51, 31, 10, 1, 41, 43, 23, 46, 24, 20, 42, 26, 22, 56, 16, 57, 19, 55, 21, 49, 17, 28, 40, 2, 27, 54, 32, 30, 53, 59, 15, 3, 45, 58, 5, 52, 29, 6, 9, 0, 33, 4, 35, 14, 8, 37, 7, 11]	0	[44, 27, 30, 46, 9, 18, 20, 6, 25, 34, 2, 54, 13, 17, 12, 48, 36, 52, 28, 14, 16, 59, 4, 11, 49, 29, 8, 7, 41, 33, 32, 55, 40, 31, 1, 24, 51, 57, 47, 45, 35, 37, 53, 56, 5, 39, 3, 38, 0, 19, 58, 23, 21, 15, 10, 43, 50, 26, 22, 42]
1	[32, 52, 16, 4, 13, 27, 20, 23, 1, 0, 24, 50, 56, 40, 8, 54, 10, 49, 53, 14, 9, 30, 19, 22, 15, 7, 26, 31, 36, 39, 2, 3, 29, 21, 33, 34, 28, 41, 11, 38, 35, 59, 42, 51, 58, 55, 46, 25, 17, 43, 44, 45, 48, 6, 18, 5, 57, 47, 37, 12]	1	[52, 22, 30, 10, 56, 48, 14, 33, 19, 18, 27, 23, 35, 41, 21, 37, 55, 49, 34, 39, 12, 13, 50, 29, 2, 6, 43, 45, 53, 8, 4, 46, 17, 31, 3, 9, 51, 25, 59, 58, 16, 40, 57, 7, 28, 20, 47, 38, 0, 1, 42, 11, 24, 5, 15, 44, 36, 32, 54, 26]
. . . . .	. . . . .	. . . . .	. . . . .
58	[25, 0, 41, 57, 9, 6, 47, 55, 34, 38, 29, 43, 12, 32, 30, 26, 17, 44, 56, 49, 50, 28, 33, 45, 11, 16, 15, 54, 13, 36, 10, 35, 21, 42, 31, 58, 23, 51, 48, 53, 2, 22, 24, 59, 14, 1, 20, 39, 3, 5, 40, 8, 4, 52, 7, 18, 37, 27, 46, 19]	58	[28, 5, 30, 17, 57, 52, 12, 31, 13, 51, 19, 47, 55, 41, 42, 11, 44, 49, 25, 23, 58, 15, 53, 21, 46, 20, 6, 39, 3, 32, 34, 29, 22, 2, 43, 38, 10, 18, 27, 8, 16, 0, 54, 59, 36, 48, 33, 40, 50, 35, 1, 26, 24, 4, 7, 45, 14, 9, 56, 37]
59	[48, 57, 25, 18, 22, 8, 21, 11, 41, 15, 54, 1, 50, 24, 17, 59, 2, 12, 3, 5, 43, 29, 37, 20, 35, 0, 6, 38, 45, 39, 51, 49, 4, 34, 7, 46, 52, 23, 40, 44, 31, 33, 55, 10, 42, 36, 9, 47, 19, 27, 58, 14, 28, 56, 53, 30, 13, 26, 16, 32]	59	[44, 27, 53, 50, 1, 7, 33, 35, 43, 32, 28, 59, 12, 11, 45, 18, 30, 52, 54, 36, 26, 0, 41, 16, 13, 31, 20, 40, 47, 38, 15, 4, 21, 9, 5, 14, 6, 2, 10, 39, 8, 49, 57, 29, 51, 34, 42, 3, 37, 56, 55, 19, 25, 17, 23, 22, 24, 48, 58, 46]

The results illustrate the RADA algorithm can save time by minimizing the running time of the operation “decision making" while keeping the running time of operation “sending request" the same.

Table 21: 30 * 30 Man optimal stable matching

Woman ID	0	1	2	3	4	5	6	7	8	9
Man ID	24	5	18	20	13	10	23	19	14	12
Woman ID	10	11	12	13	14	15	16	17	18	19
Man ID	28	9	6	7	21	11	22	17	3	26
Woman ID	20	21	22	23	24	25	26	27	28	29
Man ID	2	8	4	1	27	25	16	0	15	29

Table 22: Running time for 30 * 30 Man optimal stable matching (millisecond)

Deferred	59.60	45.44	60.89	56.08	43.92	47.88	46.79	52.39	58.03	55.12
operation	7.60	9.36	8.59	7.67	10.59	7.95	9.28	7.28	8.29	7.94

6.3 Re-matching-Based Algorithm and Repair Based Algorithm

Moving to the dynamic market, we discuss the temporary labor market. Facing the various agents in various time intervals in this market, we need to constantly update stable matching to meet different agents via the re-matching algorithm or the repair-based algorithm. Both algorithms can all find recursively stable matches for the market with different agents.

Table 23: 40 * 40 Man optimal stable matching

Woman ID	0	1	2	3	4	5	6	7	8	9
Man ID	18	32	6	22	28	1	21	35	21	3
Woman ID	10	11	12	13	14	15	16	17	18	19
Man ID	24	16	31	30	37	33	29	15	17	27
Woman ID	20	21	22	23	24	25	26	27	28	29
Man ID	39	19	11	26	13	9	36	25	4	12
Woman ID	30	31	32	33	34	35	36	37	38	39
Man ID	7	34	11	10	2	5	14	20	8	0

Table 24: Running time for 40 * 40 Man optimal stable matching (millisecond)

Deferred	56.45	41.99	48.44	35.59	48.82	39.72	45.73	40.22	46.69	46.61
operation	10.46	9.54	8.36	11.04	8.71	8.07	8.81	8.73	8.69	7.94

Both algorithms behave in terms of running time, but they are different in the aspect of result consistency. The concept aims to improve the satisfaction of agents in the market by protecting agents from variations resulting from each iteration of the re-matching algorithm or the repairing algorithm

6.3.1 Experiment 3

I designed a experiment to compare two algorithms in terms of result consistency. I obtain an optimal stable matching via the deferred acceptance algorithm and then take off 4 employers and 4 employees randomly, then recalculate twice through the re-matching-based algorithm and the repair-based algorithm to obtain two new stable matches. Based on this, we can calculate the result consistency for the two different algorithms. The result is in Table 25 Table 26

Table 25: Result consistency for size = 30 with 4 * 4 leaving

Algorithm	**Result Consistency**				
Repair	61.53%	69.23%	53.84%	73.07%	61.53%
Re-match	38.46%	50%	53.84%	61.53%	53.84%

Table 26: Result consistency for size = 40 with 4 * 4 leaving

Algorithm	**Result Consistency**				
Repair	80.55%	69.44%	83.33%	75%	77.78%
Re-match	75%	63.89%	52.77%	66.67%	61.11%

After comparing, I notice that the repair algorithm is more consistent than re-matching-based algorithm when 4 * 4 agents from both sides leave based on samples where agents number 30 * 30 and 40 * 40.

6.3.2 Experiment 4

To further analyze the algorithms, we execute them based on the same context with 8 * 8 agents from both sides leaving. The result is in Table 27. We can find in most situations,

Table 27: Result consistency for size = 30 with 8 * 8 leaving

Algorithm	**Result Consistency**				
Repair	59.09%	63.63%	59.09%	59.09%	59.09%
Re-match	50%	59.09%	63.63%	59.09%	40.90%

Table 28: Result consistency for size = 40 with 8 * 8 leaving

Algorithm	**Result Consistency**				
Repair	71.87%	68.75%	65.62%	43.75%	50%
Re-match	40.62%	59.37%	34.37%	50%	50%

the repair-based algorithm does better than re-matching algorithm. This is because the repair-based algorithm obtains the new result based on the last result before changes occur, while the re-matching-based computes a stable match based on new context completely. The re-matching-based can guarantee that each time the result is the sender (candidate) optimal stable matching, but the repair-based algorithm can not.

6.4 Result Consistency in All Stable Matching

When randomly remove 4 based one 30 * 30 preferences, we get 7262 stable matches, among them the highest result consistency is 14, while the repair-based algorithm can reach 12.

6.5 Summary

In this chapter, I mainly discussed these algorithms in the view of its efficiency. I simulate several possible situations to show their differences in the aspect of efficiency. All of the experiments can be divided into three kinds. The first kinds of experiments try to show the differences between the deferred acceptance algorithm and the request-accumulated deferred acceptance algorithm. The second kinds of the experiments mainly aim to show the variance of the re-matching-based algorithm and the repair-based algorithm from the view point of result consistency. The result consistency is positively related to the customer satisfactory. The final kinds of experiments are designed to show the distance of the repair based algorithm to the ideal situation. I can draw the conclusion that the repair-based algorithm can improve the efficiency and effectiveness in terms of dynamic market where the system need to assign the matching between agents from one side to the other side's.

Chapter 7

Conclusion and Future Work

This thesis solves the problem occurring in the temporary labor market for recruitment agencies. The main efforts addresses in proposing and implementing a two-sided matching algorithm. Such algorithm considers the characteristics in the dynamic temporary recruiting market. This chapter summarizes the main contributions of this work are summarized as below:

1. In the problem analysis section, we analyzed the abstract problem by using EBD methodology, through three rounds of digging. Also, we applied the two-sided matching algorithm to solve our problem.

2. A request-accumulated deferred acceptance algorithm is developed. This algorithm inherits the idea of classical deferred acceptance algorithm to get a stable match. The RADA algorithm retains the number of the times of the operation “sending request" while decreasing the number of the times of the operation “decision making". Compared to the deferred acceptance algorithm, our RADA algorithm can reduce to fulfill the design requirement of fast responsiveness in the dynamic market.

3. A repair-based algorithm is designed for the dynamic market. Not only can it renew its stable match as the changes happen in the market, but also it can also improve

the result's consistency, compared to the re-matching based algorithm. The repair-based algorithm fixes the unstable match caused by the market's changes, which is better than simply repeating the deferred acceptance algorithm to generate a new stable match. This algorithm makes good use of previous stable match, improving the result's consistency, further improving customer satisfaction.

The two-sided matching algorithms proposed in this thesis takes into account both static service market and continuous dynamic service markets. For static two-sided matching market, compared to the deferred acceptance algorithm, the RADA algorithm aims to minimize program running time by decreasing execution times of operation "decision making" while maintaining the execution times of operation "sending request". The RADA algorithm uses 90% less "decision making" operations, resulting in sharply reduced running time.

The algorithms for dynamic two-sided matching market are designed based on the re-matching-based and the repair-based matching models. The re-matching-based algorithm is straightforward and easy to implement. However, it does not maintain result consistency with the previous matching solution. The repair-based matching algorithm can generate a stable matching and, at the same time, maintains good matching consistency. We also designed the repair-based algorithms to accommodate preference changes from the sender (candidate). Also, we tried to improve matching consistency by exploring various repairing methods for dynamic two-sided matching markets.

For research direction, the RADA algorithm can be combined to the "re-matching-based" algorithm, catering for the requirement of distributed dynamic market using the "divide-conquer" concept. In this case, the two-sided matching algorithm can be applied in the field of big data. It can be explored in more broader market.

Bibliography

[1] JHV Vate AE Roth. Random paths to stability in two-sided matching. *Econometrica: Journal of the Econometric Society*, pages 1475–1480, 1990.

[2] MAO Sotomayor AE Roth. *Two-sided matching: A study in game-theoretic modeling and analysis*. Number 18. Cambridge University Press, 1992.

[3] VP Crawford AS Kelso Jr. Job matching, coalition formation, and gross substitutes. *Econometrica: Journal of the Econometric Society*, pages 1483–1504, 1982.

[4] J Massó B Dutta. Stability of matchings when individuals have preferences over colleagues. *Journal of Economic Theory*, 75(2):464–475, 1997.

[5] K Iwama S Miyazaki DF Manlove, RW Irving. Hard variants of stable marriage. *Theoretical Computer Science*, 276(1):261–279, 2002.

[6] LB Wilson DG McVitie. The application of the stable marriage assignment to university admissions. *Operational Research Quarterly*, pages 425–433, 1970.

[7] LB Wilson DG McVitie. The stable marriage problem. *Communications of the ACM*, 14(7):486–490, 1971.

[8] PJ Dolton. The âĂŸmarriage gameâĂŹ: An assignment problem with indivisibilities. *Theory and Decision*, 14(4):373–389, 1982.

[9] David Gale and Lloyd S Shapley. College admissions and the stability of marriage. *American mathematical monthly*, pages 9–15, 1962.

[10] P Gärdenfors. Match making: assignments based on bilateral preferences. *Behavioral Science*, 20(3):166–173, 1975.

[11] G Isaak H Abeledo. A characterization of graphs that ensure the existence of stable matchings. *Mathematical social sciences*, 22(1):93–96, 1991.

[12] SY Itoga. The upper bound for the stable marriage problem. *Journal of the Operational Research Society*, pages 811–814, 1978.

[13] SY Itoga. A generalization of the stable marriage problem. *Journal of the Operational Research Society*, pages 1069–1074, 1981.

[14] L Lovász J Csima. A matching algorithm for regular bipartite graphs. *Discrete applied mathematics*, 35(3):197–203, 1992.

[15] RF Kirby. A preferencing model for trip distribution. *Transportation Science*, 4(1):1–35, 1970.

[16] AE Roth. The economics of matching: Stability and incentives. *Mathematics of operations research*, 7(4):617–628, 1982.

[17] AE Roth. Stability and polarization of interests in job matching. *Econometrica: Journal of the Econometric Society*, pages 47–57, 1984.

[18] AE Roth. The college admissions problem is not equivalent to the marriage problem. *Journal of economic Theory*, 36(2):277–288, 1985.

[19] D Gusfield RW Irving, P Leather. An efficient algorithm for the âĂIJoptimalâĂİ stable marriage. *Journal of the ACM (JACM)*, 34(3):532–543, 1987.

[20] RCT Lee SS Tseng. A parallel algorithm to solve the stable marriage problem. *BIT Numerical Mathematics*, 24(3):308–316, 1984.

[21] A Tamura. Transformation from arbitrary matchings to stable matchings. *Journal of Combinatorial Theory, Series A*, 62(2):310–323, 1993.

[22] LB Wilson. An analysis of the stable marriage assignment algorithm. *BIT Numerical Mathematics*, 12(4):569–575, 1972.

[23] Y. Zeng. Axiomatic theory of design modeling. *Journal of Integrated Design and Process Science*, pages 1–28, 2002.

[24] Y. Zeng. Axiomatic theory of design modeling. *Environment-based formulation of design problem*, pages 45–63, 2004.

[25] L Zhou. Stable matchings and equilibrium outcomes of the gale-shapley's algorithm for the marriage problem. *Economics Letters*, 36(1):25–29, 1991.