國立臺灣大學電機資訊學院電機工程學系

碩士論文

Department of Electrical Engineering
College of Electrical Engineering and Computer Science
National Taiwan University

Master Thesis

# 一個基於超級帳本區塊鏈之時間銀行系統及其動態加厚信息配對方法

# A Hyperledger Fabric based Time Bank and Its Thickness Dynamic Matching Method

林展嘉
Jhan-Jia Lin

指導教授：劉志文博士，吳家麟博士
Advisor:Chih-Wen Liu, Ph.D. , Ja-Lin Wu, Ph.D

中華民國 108年 6月
June, 2019

doi:10.6342/NTU201902682

# 誌謝

感謝帶領我的指導教授吳家麟老師，在接受吳老師的指導下，從新開始新的研究新的領域，老師的帶給學生的收穫良多，吳老師的用心指導與關心使學生我能在充滿知識與熱情的地方接受灌溉，無論是開會時的討論或是老師在會中的學術新知分享，還有老師直接的教課傳授，使學生不僅對於自身的研究有學習外，在密碼學，機器學習等等的知識有更多的了解，也啟發學生我對於研究應該要有的熱情，使得自身的研究能按部就班的完成。同時也謝謝電機所劉主任，在學生需要幫助時給予意見與協助，讓學生我十分感激。

再來謝謝陪伴我一起奮鬥一起研究的同學們育澤、雅文、品君、昌第、承億、承廷，一路走來和大家討論研究上的問題，互相激發想法，甚至有活動時也一起同樂，尤其感謝從碩一以來就一直是好同學好伙伴的育澤，在學習新的研究領域上，一起努力，同時我也要謝謝我的女友與家人一路的陪伴與包容，有你們才有現在的我，謝謝。



doi:10.6342/NTU201902682

# 摘要

本文基於厚度動態匹配市場算法，提出了一種區塊鏈上時間銀行 (Time Bank) 的實現方法。論文由兩部分組成，其一是在 Hyperledger Fabric（Fabric）平台上實現 Time Bank 系統，Fabric 是眾多許可式 (Permissioned) 區塊鏈平台之一。提供身份證書機制和可擴展的網絡結構是 Fabric 區塊鏈平台的特色。時間銀行系統的成員需要進行身份審查，以使系統更加安全。通過時間銀行交換服務也會增強社區成員之間的互動。故 Fabric 被視為是實現時間銀行系統適合的區塊鏈平台。其次，我們認為利用等待時間可以加厚動態匹配市場，並利用這一概念設計出動態調整策略（Dynamic Tuning Strategy: DTS）。根據市場規模，DTS 將決定每次匹配應執行或等待。實驗證實較厚的交易市場可使節點具有更多鏈接並且更容易匹配。



doi:10.6342/NTU201902682

# Abstract

This thesis presents a blockchain based Time Bank realization using thickness dynamic matching algorithm. This work consists of the following two parts: (1) A realization of the Time Bank System on the Hyperledger Fabric (or simply called Fabric) platform. Fabric is one of the well-known permissioned blockchain platforms. Fabric provides the identity certification mechanism and has an extendable network structure under the blockchain. Members of a Time Bank system have to go through an identity checking process for making the personnel of the system more secure. The exchange of services through Time Bank systems also warms the members' communities. (2) On the basis of using waiting time to thicken the dynamic matching market, a Dynamic Tuning Strategy (DTS) for enhancing matching performance, is proposed. According to the market size, DTS helps decide to do a match or to wait for a later chance in each run. Experimental results show that thicker market makes on-chain nodes have more links and is easier to find a match.

**Keywords:** Time Bank, Blockchain, Hyperledger Fabric, Dynamic Matching, Thickness



doi:10.6342/NTU201902682

# Contents





doi:10.6342/NTU201902682

doi:10.6342/NTU201902682

# List of Figures





doi:10.6342/NTU201902682

doi:10.6342/NTU201902682

# List of Tables





doi:10.6342/NTU201902682

# Chapter 1

# Introduction

In this thesis, a Blockchain-based Time Bank service-exchange system is proposed. Time Bank is a reciprocity-based service trading system in which the amounts of time-spent (i.e., seconds, minutes, or hours) are the statutory currencies. With time banking, a person with one skill set can bank and trade hours of work for equal hours of work in another skill set instead of paying or being paid for services. The hours banked are always traded equally regardless of the services rendered. This equality is intended to foster ties in communities and, by making all contributions valued equally, encourage equality in the communities themselves. The concept of service-exchange in time banking is the same as the mutual sharing in economy. Users in decentralized markets create alternative socio-economic systems through resources sharing, such as service, material, and knowledge. However, as shown in the left-side of Fig. 1.1, today's shared economic model tends to rely on a centralized organization to act as a broker or an arbitrator, merely connecting two parties on a supply-and-demand basis. Although such an architecture has succeeded in creating a huge economic market that didn't previously exist, it comes at a cost for users. First, users who use the platforms (e.g., Uber or Airbnb) will be charged by the company, which also makes the intermediary (centralized) broker gaining huge profits without providing fair services or creating respectable values. Second, the company controls the user's information so that users can only communicate via their In-App messaging platforms, which disallows users to connect to their preferred platforms so as to waive transaction fees. Third, a centralized company can hide information that is not conducive to the system



doi:10.6342/NTU201902682

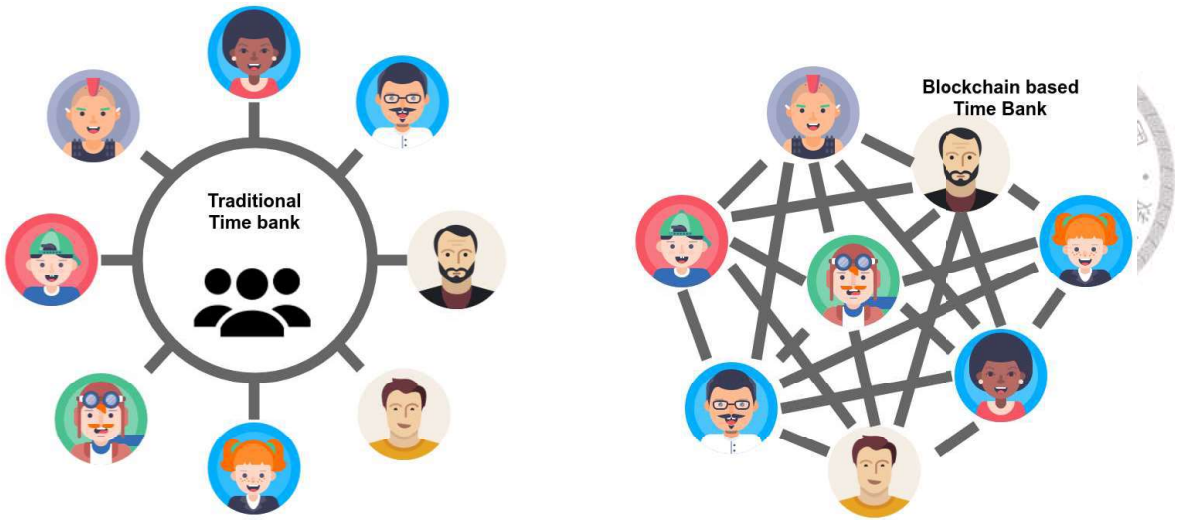



Figure 1.1: Traditional Centralized Time bank and Blockchain based Decentralized Time bank

and thus cause information inequality between users and the company. Clearly, companies who collected lots of valuable customer data could also lead to information monopoly, helping them easily deploy new products into an existing pool of potential customers.

Blockchain technology makes it possible to cancel intermediate Brokers and/or Arbitrators. Blockchain technology is a technical solution that does not rely on third parties to store, verify, and transmit messages through its own decentralized nodes. In a decentralized blockchain network, as shown in the right-side of Fig. 1.1, every node is connected in a peer-to-peer manner and can trade without mutual trusts, which makes the existence of a system manager redundant. Peer-to-peer connections of the blockchain enable users connecting and deciding for themselves to share the services or values of the exchange. No additional fees will be charged for the use of smart contract's fair execution agreements. In addition, the blockchain has several specific features that are suitable for integrating with the concept of a shared economy. The first is transparency. All members in the blockchain network can see each. The user' s data, as well as the evaluation of the previous service, that is, unfavorable information cannot be hidden. The second is that the recorded data cannot be modified. Once the information is written into the ledger, it cannot be changed, in other words, the information on the ledger is trustworthy.

In the application of Time Banking, identity authentication is a must. The exchange of services needs to be carried out face to face. Therefore, it is necessary to review the



doi:10.6342/NTU201902682

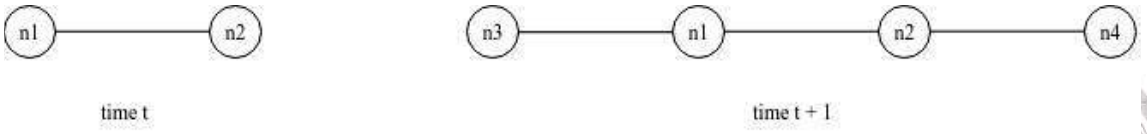



Figure 1.2: Illustrates the effect of "matching-option-expansion" by using the strategy of "waiting for match".

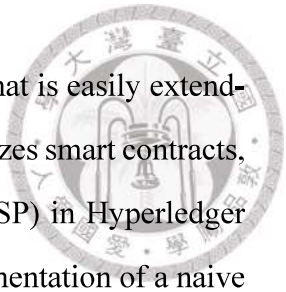
identity and confirm the security of involved members. This is the main reason why we use Fabric blockchain. Even though we don't need an intermediate manager to help us pair two sides with matching conditions, a special organization (Certificate Authority: CA) is required to validate the identities of on-chain members in the network. A verified member will get a digital certificate to prove his or her identity, and when a smart contract is invoked, the authentication is restored to evidence the corresponding membership. In a licensed blockchain, the on-chain information can only be queried by legal members and only valid members can invoke smart contracts. If a permissionless blockchain platform is used, everyone can join the blockchain network, and the safety of on-chain members cannot be guaranteed. In the application scenario of a Time Bank system, the institution responsible for issuing the certificate is, usually, a social welfare organization, who are not interested in infringing the security of system users.

In this thesis, we implemented a dynamic matching algorithm on smart contracts to ensure the transparency and correctness (or fairness) of the matching result. However, the high complexity of the dynamic matching algorithm handicaps its realization directly on a blockchain platform without a centralized server. Thus, we locally accumulate the supply-and-demand inquiries until the market size (i.e., the number of to-be-matched supply-and-demand pairs) reaches an economic scale so as to make the complexity per transaction and/or the probability of matching become reasonable for being conducted on a blockchain based platform. Furthermore, a dynamic tuning method is also proposed to reduce the waiting time for reaching a match. In other words, if the market size is large (or equivalently, the market density per unit inquiry is thick) enough, our system users may not need to wait and can get a match and leave the market, immediately.



doi:10.6342/NTU201902682

## 1.1 Contributions



The implemented system provides a dynamic matching mechanism that is easily extendable. Our system is a timing-based dynamic matching system that utilizes smart contracts, called chaincodes, and a modular Membership Service Provider (MSP) in Hyperledger Fabric to achieve the goals of Time Banking. However, direct implementation of a naive dynamic matching algorithm on blockchain, the corresponding complexity is too high to be used in practice. Moreover, in blockchain, there is no central server for providing global computing facility. Therefore, in this study, we conduct the matching algorithm in each validated node (it is said a local algorithm is used), until the thicken (or size) of the market reaches an economical scale. Then, a newly proposed dynamic tuning method is applied to reduce the waiting time without increasing any extra overhead. To maintain data transparency and fairness, all matching records and service materials are publicly recorded on the blockchain. When users execute the matching algorithm through smart contracts, they vote for the matching situation dynamically. If the market is thick enough, users need not to wait any longer, just conducting the match and leaving the market, as soon as possible. Congruent to the goals of sharing economy, this realization supports a decentralized matching mechanism for Time Bank systems without a centralized broker, which demonstrates another successful integration of blockchain technology into the daily lives of human beings.



doi:10.6342/NTU201902682

# Chapter 2

# Background and Related Work

Up to now, the time banking system has not been well developed even though its ideas have been widely accepted. The availability of time currency is also restricted by few communities and is not easy to circulate. The issues of fairness and transparency in such a sharing economy environment has also been questioned [4]. Fortunately, the goals of Fairness, transparency and high scalability can be achieved through the usage of blockchain technology. Nevertheless, realizing a blockchain-based Time Bank system also faces certain challenges. Among all of them, it is our belief that "how to build an efficient dynamic matching mechanism on the blockchain without a global planner" is on the top of the candidate list.

## 2.1 Blockchain

The first blockchain was conceptualized by Satoshi Nakamoto in 2008. Satoshi presented the design of using secure hash function for chaining blocks into the first blockchain platform, Bitcoin [1], which provides the most successful decentralized electronic cash system. Bitcoin, a digital ledger-based system, using proof-of-work (PoW) consensus algorithm and a few other technologies to help verify transactions and create a global peer-to-peer (p2p), decentralized cryptocurrency. In 2014, a new blockchain platform was born called Ethereum. Ethereum is proposed by Vitalik Buterin [2]. Vitalik suggested the usage of smart contracts for providing programming capability to blockchains.



doi:10.6342/NTU201902682

- Information is transparent. Data transparency in a Time Bank increases the social interaction between community members.

- The strategy of pairing is fair. Autonomous program running on smart contracts make the tampering and/or biasing the matching processes very difficult.

- The recorded data is trustworthy and safe. Time tokens are stored in a decentralized ledger and will be well protected and hard to modified.

## 2.2 Hyperledger Fabric

Hyperledger Fabric (Fabric for short) [5] is one of the permissioned blockchain platforms for building distributed ledger solutions. Fabric delivers high degrees of confidentiality, flexibility, resiliency, and scalability. This makes the solution developed on the basis of Fabric suitable for lots of industries.

Fabric leverages container technology to host smart contracts called "chaincodes", which fit different kinds of business rules of different applications. nd it is designed to support various pluggable components and to accommodate the complexity existing across the entire economy.

Membership Service in Fabric's Permissioned Model can be integrated with various standard identity management systems. To support this flexibility, Fabric adopted a novel architectural approach and improved the way the blockchain responds to non-determinism, resource exhaustion and performance attacks.

Fabric can also create channels enabling a group of participants to create separate transaction ledgers. This is especially important for networks where some participants may not want to have a competitor for each transaction - for example, a special price that is offered to some participants of the network but not to all members. If a group of participants forms a channel in Fabric, then only those participants, and no one else, have legal ledger copies of the channel.



doi:10.6342/NTU201902682

## 2.3 Dynamic Matching

As for the dynamic matching, the node's preferences are founded based on condition's or attributes compatibility. Therefore, "kidney-exchange process" is the most closely related literature in our studying about dynamic supply-and-demand matching on a network. M. Akbarpour [6] considered a model where compatibilities are based on a random graph model. Researches indicating that waiting for thickening the market size will yield large gains if the planner knows the departure time. There are other studies about the Imbalanced Markets [7][8], in which the exchanges of markets are conducted by easy-to-match and hard-to-match agents.

Kurino [9] and Bloch and Houy [10] study an overlapping generations model of the housing market. In their models, agents have deterministic arrivals and departures. In addition, the housing side of the market is infinitely durable and static, and houses do not have preferences over agents.

As pre-described, this thesis applies the strategy of market size thickening to the operations of a blockchain-based Time Bank system. To simplify the analyses, we treat the tasks of balancing supply-and-demand in a Time Bank as finding a match in a two-sided market.

## 2.4 Stable Marriage Problem

Another study related to our research is Stable Marriage Problem (simply called SMP). In mathematics, economics, and computer science, the SMP is the problem of finding a stable matching between two equally sized sets of elements given an ordering of preferences of each element. In 1962, David Gale and Lloyd Shapley proved the Gale-Shapley algorithm [**?**] to solve this problem. There are also related papers to discuss the SMP issue in different scenarios [**?**]. The biggest difference between SMP and our research is that the SMP problem is a preference based matching issue and our study focus on the compatibility-based matching issue. In future work, our system will integrate the preference-based matching mechanism.When each service has more than one choices, how to choose a better service



doi:10.6342/NTU201902682

will be an issue that needs to be improved.



doi:10.6342/NTU201902682

# Chapter 3

# System Overview

## 3.1 Application Overview

This section introduces the application scenarios of our system first, then a quick overview about the service-exchange process of a Time Bank system is given. The later chapters will explain each system module in more details. Currently, the most important business of a Time Bank is to provide services in exchange of time credits, where time credits can be used to exchange for other people's services. Before introducing the service-exchange process of a time bank, four settings need to be addressed first:

- "Service" represents a collective term of different useful and desirable skills for disabled and/or aged people, such as cleaning the house, repairing the machine, driving to hospital, and so on.

- The value of the time credit is measured according to the time-spent of the service, regardless the substances of the service. For example, cleaning a house for one hour gets the same time credit for taking care of an elderly citizen for one hour.

- Members of the system can be divided into two clusters: the Service Providers (SPs) and the Service Receivers (SRs). SPs provide services in their spare time, while SRs seek for assistance from others in their required instance. A system member can play the role of either SP or SR, but not at the same time.



doi:10.6342/NTU201902682

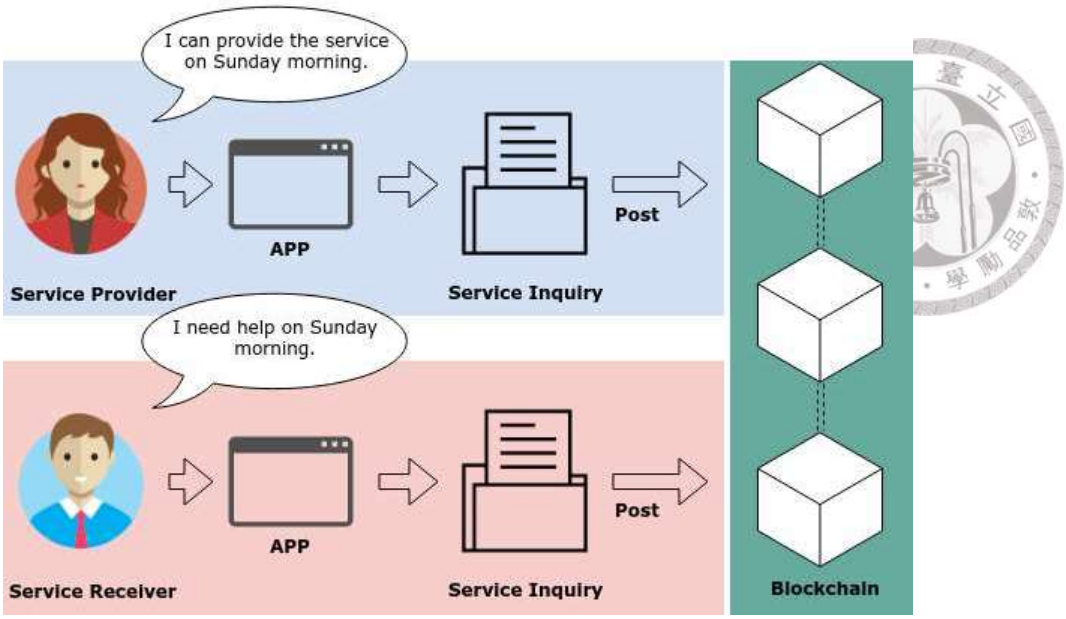



Figure 3.1: Posting service inquiries process

- The provision of services is not immediate. For example, Uber is a kind of immediate service provider. However, in Time Bank, if Bob needs someone to help baby-sit his child on Saturday, Bob has to issue the request a few days before.

In our Time Bank system, SPs and SRs match the supply-and-demand services on Fabric blockchain. The service-exchange process of our Time Bank system consists of the following three main steps:

1. Members need to post their service inquiries to the blockchain, where the service inquiries have two types. SPs post the supply inquiry, and SRs post the demand inquiry. Then, the service inquiry reveals the user's specific content of the supply or demand service and it will be posted to the blockchain through the client-site application software, as shown in Fig. 3.1

2. The supply-and-demand service inquiry keeps looking for candidates who match up with the matching criteria until a matching strategy is pursued. In this work, a market thickness based dynamic tuning matching strategy is adopted. In a nutshell, each service inquiry decides when to match with others, according to the thickness of the market.



doi:10.6342/NTU201902682

3. Before SP providing the pre-negotiated service to SR, Time Bank system will help SP check SR's account balance for ensuring SR has enough time credit for paying the required service charge. Similarly, Time Bank system will assist SR to send the pre-negotiated time credit to SP after SP did complete the service.

This section only overviews the information flow of the proposed system, while the detailed processes will be explained in the following Chapters.

## 3.2 Network Structure

Our system is based on the Hyperledger Fabric blockchain network. Before introducing the network structure, this section needs to introduce the involved members in the network and how to become a member. In our application scenarios, there are SPs, SRs, and a neutral certificate authority (CA), who's role is usually played by a social welfare agency, or a government-authorized organization. The neutral CA reviews the applicants who want to join the network and sends the identity certificate after validation. According to the issued identity certificate, users' footprints can be traced if they are doing something illegal. In addition, the service-exchange has to be conducted face to face between SRs and SPs, so the safety of members is the top priority. CA can also revoke the identity certificate for the member who had menaced to other members.

After the enrollment, valid users get permission to invoke chaincodes and query the on-chain data. Figure. 3.2 shows that there are two channels in our Time Bank networks:

- Service Channel (SC): Processes in SC managing all service inquiries and handling the matching process. Users evoke SC for posting their inquiries then match their supply-and-demand attributes with the other members in this channel.

- Token Channel (TC): Processes in TC managing all users' account and balance. If a service-exchange is accomplished, evoking TC to confirm whether the corresponding service status on SC is completed, then approving sending the token to SPs from associated SRs.



doi:10.6342/NTU201902682

The proposed system is designed based on Fabric's multiple-channel architecture for increasing the scalability of the system. Because Time Banks are community-based institutions, the circulatory nature of time credits is also limited. Therefore, multiple Time Banks are allowed to join in the proposed system. All Time Banks in the network share the same TC, but each of them has its own SC. Notice that in Fig. 3.3, each SC manages its community's services, and the services cannot match with each other if they belong to different SCs. The conditions for supporting cross channel service-exchanges will be addressed later.

## 3.3 System Assumption

This thesis will simulate the whole system processes employed by valid members with the aid of the proposed matching mechanism. Some system related assumptions must be given for making our discussions more focused. Firstly, each service inquiry has a fixed available time when the members post it onto the blockchain. As mentioned in Section 3.1, the response to each service inquiry is not immediate, so SPs and SRs can wait for a while after a service inquiry is posted. Setting available time periods also helps SPs and SRs know the service inquiry has been successful matched or not.

Second, different from traditional server-client based system where users need to call the matching function on local-side after the service is posted and cannot continuous ex-

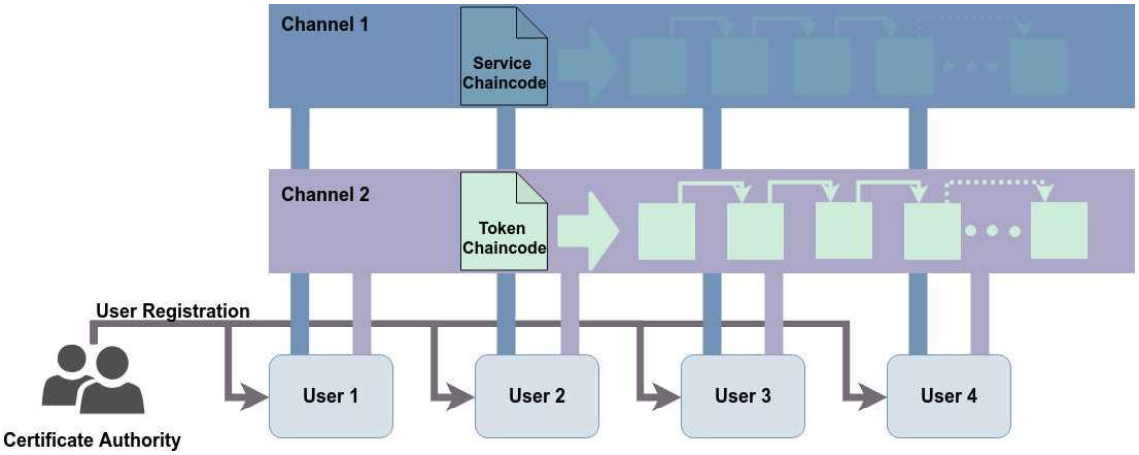



Figure 3.2: Hyperledger Fabric's multi-channel network in the time bank system



doi:10.6342/NTU201902682

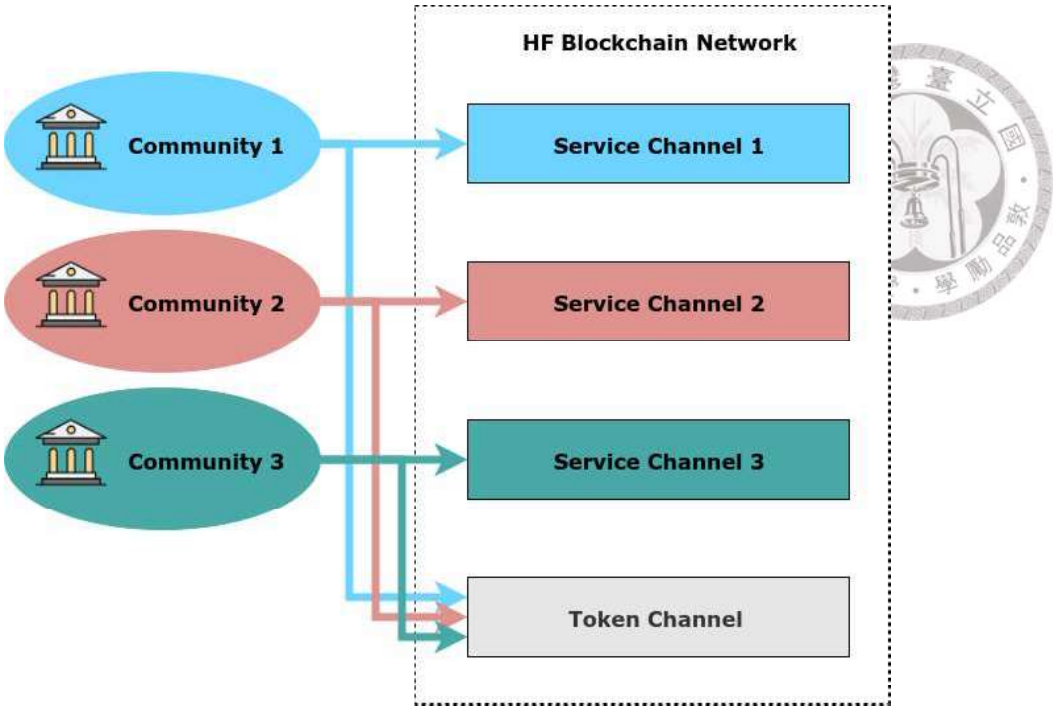



Figure 3.3: The scalable time bank network structure.

ecute the match function. In the proposed system, there is no global planner to handle the match process, so during the available time periods, each inquiry tries to evoke the match function through the service chaincodes autonomously until a match is found.

Third, since all SPs and SPs find their matches in Fabric network, so the Fabric network can be viewed as a market of the proposed Time Bank. The arrivals of SPs and/or SPs to the market is stochastic in nature, which complicate the corresponding analysis a lot. For the simplicity of analyzing, dynamic market matching methodologies are considered and a market where balanced supplies and demands are assumed. In the rest of our discussions, a market fulfills the above assumption is called a two-sided market. For a two-sided market, there are some works focused on the incentive strategy design for enhancing selling [3], where a balance between supply and demand is achieved through an advanced price and/or reward strategy. In this thesis, we discuss the matching strategy based on the pre-described "waiting for match", or equivalently, the "thickening market" strategy. Our strategy will be detailed in Chapter 5.



doi:10.6342/NTU201902682

# Chapter 4

# Functional Modules of the Proposed System

This section details each module of the system and addresses the overall transaction processes, including user registration, posting service inquiries, matching processes, and service exchange processes.

## 4.1 User Registration Module

Fabric is a permissioned blockchain, so all on-chain members must complete their user registration process before evoking and/or querying the on-chain information. The interaction between an applicant and CA in the user registration process is shown in Fig. 4.1. First, the applicant sends a registration request to CA through his or her client App, and the request contains the real identity of the applicant. Second, after the registration, CA sends and stores the public and the private keys and the identity certificate to and on the user's wallet. Third, CA calls token chaincodes to create a user's account, where the user's time token is stored.

Every time a user invokes the chaincode, Fabric blockchain checks the validity of the identity certificate. Cryptography techniques allow users to present their certificates to others for proving their identities, so long as the other parties trust the certificate issuer. Through reading and checking the certificate, one can make sure that the information about



doi:10.6342/NTU201902682

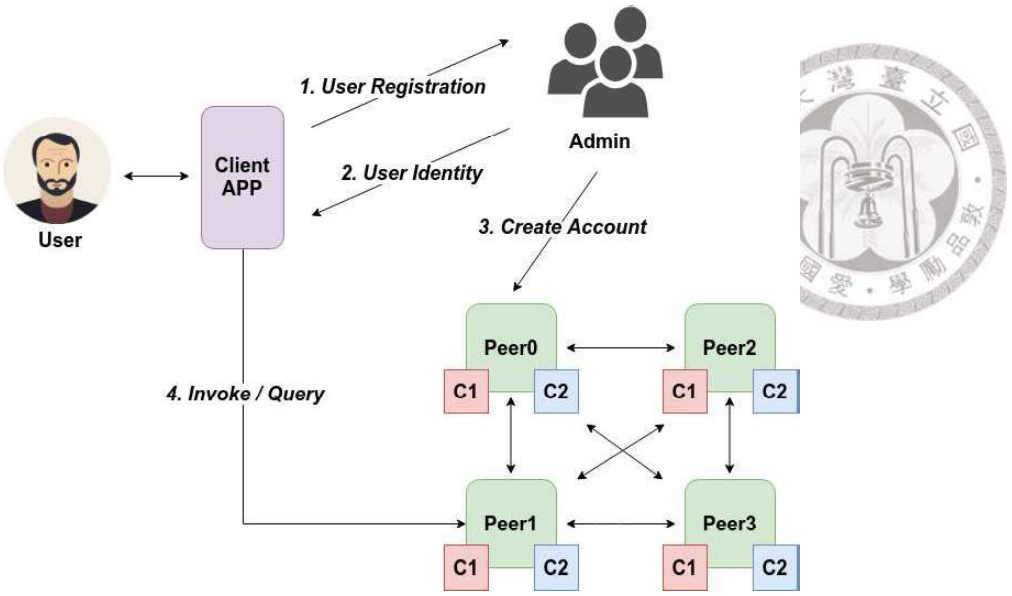



Figure 4.1: The interaction with the CA in the user registration process

the user has not been tampered.

## 4.2 Service Inquiry Posting Module

An Enrolled member, with issued key pair and wallet, can start to post service inquiries on the Fabric blockchain. Using the issued key pair, a valid member accesses the peer, uploads his or her service inquiry, and evokes a service posting function, called the service proposal, in form of Service Chaincodes. Then, the service proposal is posted on SC. For maximizing the liquidity and circulation of Time Coins, each posted service has a fixed available time slot for waiting for the match up with other qualified services, which fulfilled the conditions defined in each service inquiry.

Each service inquiry has a lifecycle on the blockchain, and Fig.4.2 shows the state machine for the service inquiry. The state of service inquiries can be divided into 5 stages: $NewPosted$, $Candidate$, $Designation$, $Confirmation$, and $Expiration$, which exactly depict the life cycle of a service. Although a service will be terminated sooner or later, the related data will be recorded on the blockchain, forever. The state information will help the system to query for services that can be matched with high probability. Furthermore, the state helps the system solve the potential double spending problem. The detail explanation



doi:10.6342/NTU201902682

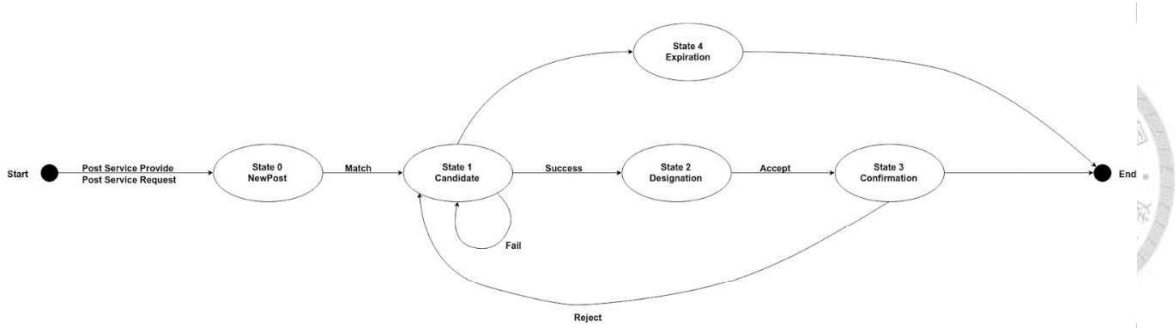



Figure 4.2: State machine of the service

about the double spending problem will be mentioned in Section 4.4.

## 4.3 Service Inquiry Evoking Module

In this work, each matching process is represented by a set of chaincode, which instantiated on the blockchain by the two-side users(SPs and SRs) in the market. As shown in Table 4.1, the enrolled members published their services inquiries in SC, in which the types of services, denoted as $serviceType$, are divided into the following two kinds: request and provide.

Each newly posted service inquiry, as above-mentioned, has a fixed available time. As mentioned in Section 3.3, each service inquiry attempts to evoke the match function in service chaincodes at a fixed period until it is matched. Therefore, $NOAI$ (Number of available invokes) records the number of times each service inquiry can invoke the match function. Whenever a match function executes for a service inquiry, the $NOAI$ in the inquiry will be decreased. $serviceClass$ represents the service substances in the Time Bank system.

Fig.4.3 shows some service substances taken from TimeBanks USA, and their supply-versus-demand ratios. It is worth mentioning that the number of offers may not be realistic. Because when someone wants to join TimeBanks USA system, he or she needs to clearly tell the system what services he or she the can provide. When someone needs help, the staffs of TimeBanks USA will, use the phone, to ask members who can provide the service. Therefore, the number of members who can actually provide services must be less than the



doi:10.6342/NTU201902682

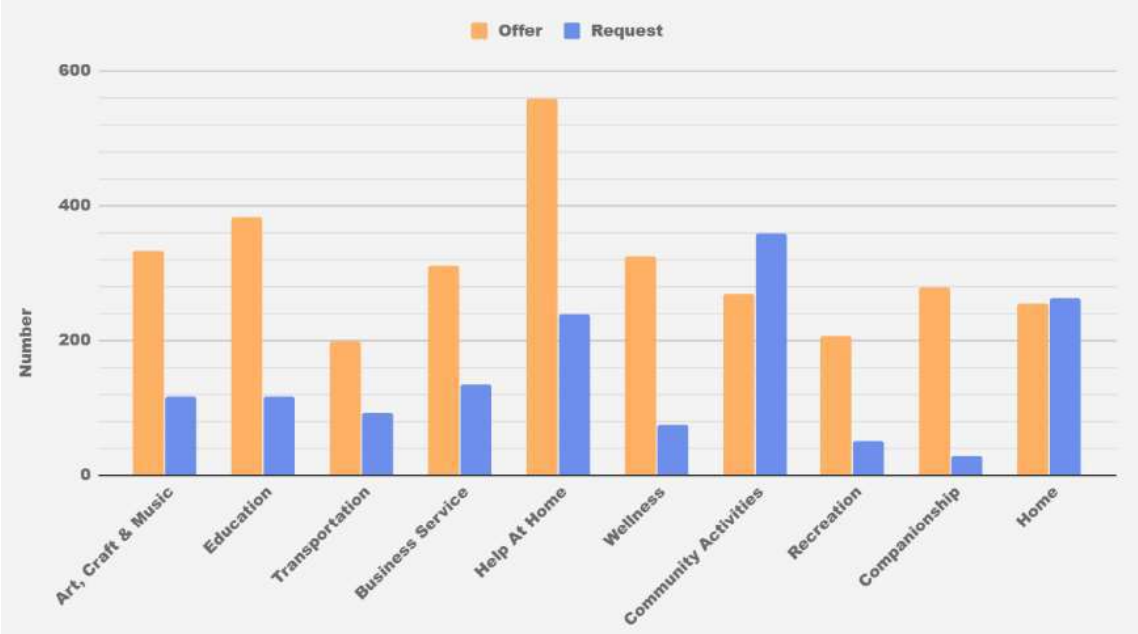



Figure 4.3: Relative Service Ratios of Supply-versus-Demand Measured in TimeBanks USA

number of offers in the figure. For simplicity, we select the three most demanded services as the service substances in the proposed system, including $HelpatHome$, $CommunityActivities$, and $Home$. Services in $HelpatHome$ includes parenting, cooking, hairdressing, housework, pet care, respite care, and more. Services in $CommunityActivities$ include dance teaching, talent teaching, and more. Services in $Home$ include garden and yard work or other services for house repairs.

As mentioned in Section 3.2, the state of service inquiries can be divided into 5 stages, and each sate is recorded as an element if the State Machine of the service. The service

| Elements in the Service Chaincode | Explanation |
|---|---|
| $serviceType$ | The service can be provided or requested. |
| $posterName$ | The user who posted the service on the blockchain. |
| $serviceClass$ | Substances of the service. |
| $serviceTime$ | The time interval for supplying or demanding services. |
| $postTime$ | A timestamp of the publishing the service. |
| $state$ | Each service has five status in its life cycle. |
| $serviceOwner$ | Other services matched up with the target service. |
| $NOAI$ | Number of available invokes. |
| $v$ | A voting parameter depends on the situation of matching. |
| $\alpha$ | A ”0 to 1” tuning parameter for reducing redundant waiting time. |

Table 4.1: The parameters in the service proposal.



doi:10.6342/NTU201902682

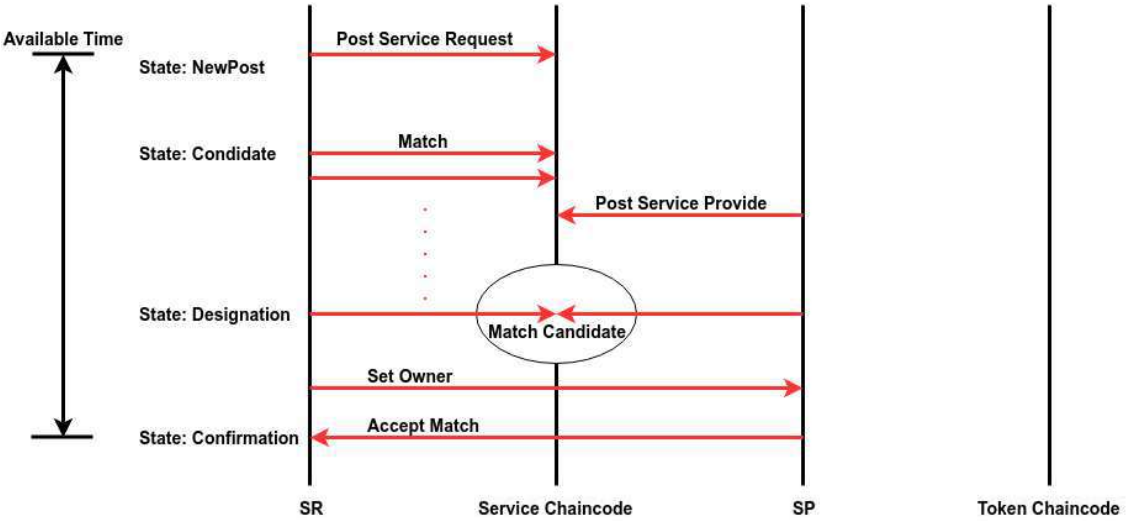



Figure 4.4: Transaction Flow of the Service Matching Phase.

chaincode will classify these service inquiries in the blockchain based on the service $state$. In our system, each service inquiry becomes the matching candidate of another service not only if they should have the same $serviceClass$ but also need to be within the same available time interval. And $serviceTime$ records the time interval in the service inquiry. Only the demand service inquiries' $serviceTime$ are included in the supply service inquiries' SER is eligible for becoming a candidate. In addition,the two system parameters $v$, $\alpha$ are designed to determine when each service will be matched, the detailed explanation will be presented in the chapter 5.

## 4.4 Matching Process

### 4.4.1 Match Flow Chart

Figure 4.4 shows the detailed transaction flow of the service matching process. First, If an SR posts his or her own service inquiry onto the blockchain. If the posting process is successful, the state of the service inquiry will be modified to $NewPost$. Other service inquiries within the same interval will no longer be posted. Second, as long as that service inquiry is posted, the Client App will automatically invoke the match function at a fixed frequency. And the state of the service inquiry will be searchable by other services when the state is denoted as $Candidate$. Third, if the SP posts a service inquiry which is com-



doi:10.6342/NTU201902682

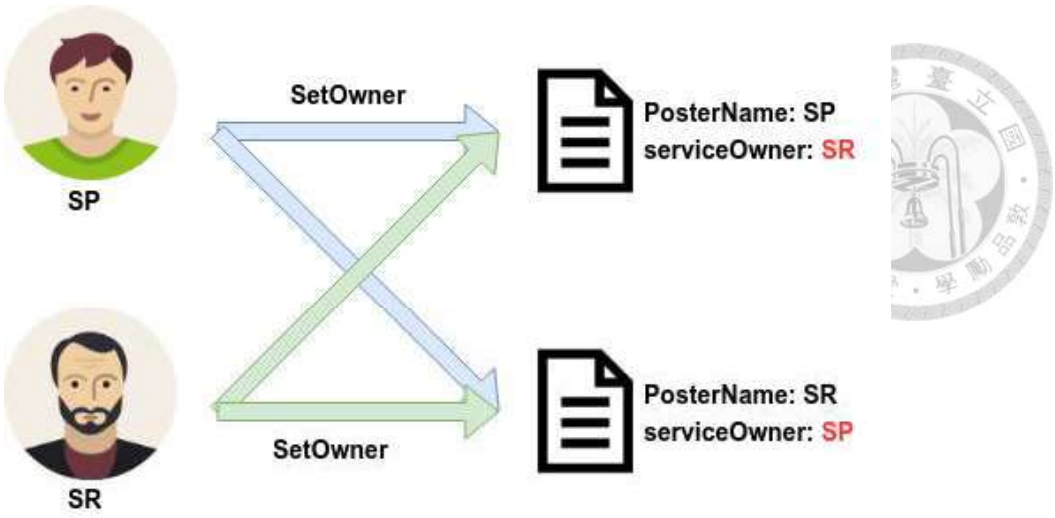



Figure 4.5: How to set the $serviceOwner$ between an SP and an SR.

patible with that of the SR. After the service proposal is posted, the data will be uploaded to the blockchain, and the target SR will find the candidate SP's service inquiry in next time the matching function is invoked. Fourth, after the service inquiries of the SP and the SR fulfill a matching strategy, the SR sets the SP's $serviceOwner$ to himself or herself and vice versa, as shown in 4.5. Finally, the SP can choose to accept or not accept this service match. If not, both parties will return to their $candidate$ state and continue to look for other compatible service inquiries.

### 4.4.2 The Double Spending Problem

If a service inquiries matches with another one, they set each other as the service owner but this may lead to a problem. For example, there are three service inquiries in the blockchain; two request services: R1, R2, and one provide service: P1. Now assume R1 and R2 match up with P1 at the same time. Then, R1 and R2 set the same P1 as their service owners, this situation is similar to the well-known double spending problem in Bitcoin blockchain. Fabric uses MVCC(Multiversion concurrency control) to eliminate the risk of double spending. When updating a state, a new version of an existing state will be marked to overwrite the old one. Any transaction executed between this time will be swept, and shows up an MVCC error about the information of the fail transactions.

In a nutshell, when invoking the set owner function, Fabric builds a read-write set. The



doi:10.6342/NTU201902682

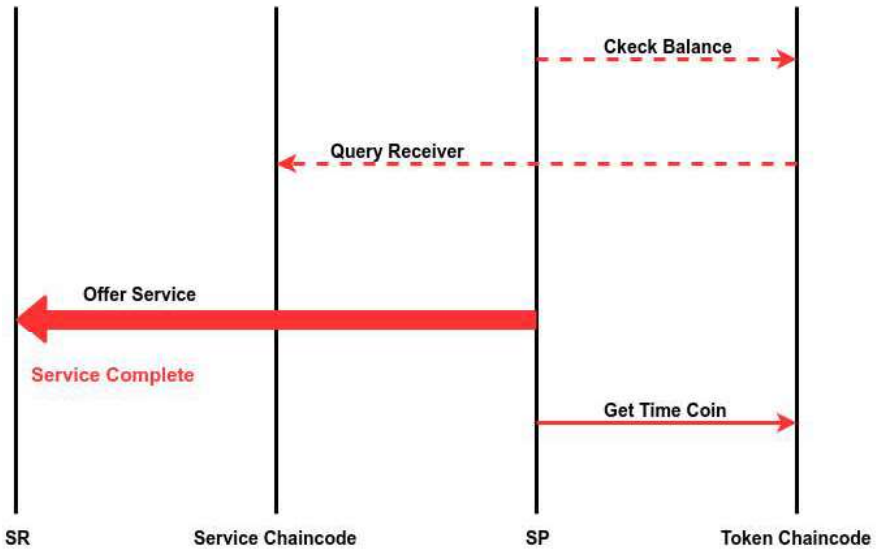



Figure 4.6: Transaction Flow of the Service Exchanging Phase.

ordering service would sequence the proposed transactions in a block. If R1's transaction occurs before R2's, All peers validates R1's transaction but invalidates R2's transaction since the balance version in R2's read set no longer matches up with the new balance version in the current state.

## 4.5 Service Exchange Process

In this chapter, the representative service has been matched, and the actual service and time credit exchange are executed ongoing. Figure 4.6 shows the exchange process between service offering and time credit spending. First, the SP first queries TC for checking the balance of the SR, and TC examines whether the states of the SP and the SR are changed to matched, then TC checks whether the balance of the SP is sufficient to pay the service charge. Second, SP actually provide the service to the SR, and after the SP completed the service, he or she gets the SR' s time credit from TC.



doi:10.6342/NTU201902682

# Chapter 5

# Dynamic Matching in Two-sided Markets

In this Chapter, the timing and the network structure issues of a dynamic matching system are introduced first. Then, our suggested solutions to deal with those challenges are presented. For simplicity, the dynamic matching in two-sided markets is chosen as our discussion focus.

## 5.1 The Model

The thesis model the tasks of dynamic service matching on blockchain as a continuous-time stochastic process in a market with two-side binding participants. In which, each node denotes a valid service inquiry posted on the blockchain. In other words, two linked nodes stand for a candidate matched pair and each link represents that there is a potential matching relationship between the two ends of the link. The two nodes with potential matching relationships are called neighbors we use $N(n_i)$ to denote the number of neighbors of node $n_i$. For example, in Fig. 5.1, node $n_1$ links to nodes $n_2$, $n_3$, $n_4$, and $n_5$, so $N(n_1) = 4$.

Once a user posted his or her service inquiries on a node, we say that node is entering the market, and as pre-described, users in the market can be divided into SPs and SRs. For simplicity, a user in the market is denoted by the symbol $x$, where $x \in \{SP_i, SR_j\}, i, j =$



doi:10.6342/NTU201902682

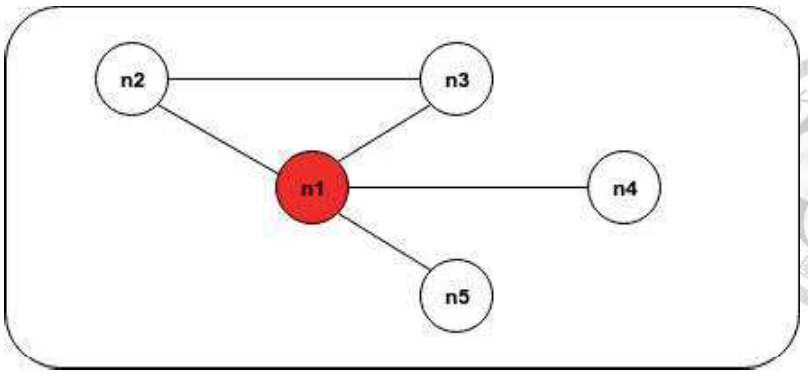



Figure 5.1: The Neighbor of the $n_1$, and $N(n_1) = 4$.

$1, 2, 3.....$ Without loss of generality, we assume users arrive at the market with average time interval $T_n$ according to a Poisson process, where $n$ denotes the time interval cycle. That is the probability of an $SP$ arriving to the market is$Pr[SP]$, and that of an $SR$ is $Pr[SR]$. Moreover, we assume that the amounts of supply and demand in the market are equal. Hence, $Pr[SP] = Pr[SR] = 0.5$. Assume all the matched services will leave the market immediately, and each $SP$ and $SR$ has a fixed average available time $T_a$. Therefore, each user has examined $n$ services when he or she leaves the market without being matched, where $n = T_a/T_n$. If a node has not been matched during its available time, as pre-described, it will leave the market immediately. The work assumes a normal matching cycle time is $T_x$, if a service is unable to match with others in $T_x$, it will wait until the next cycle to match again. So, each service can invoke match processes up to $m$ times in its lifecycle, where $m = T_a/T_x$. Furthermore, let $p$ denotes the probability of two random service inquiries that are compatible, where $p$ is related to the services' type, class, and available time.

## 5.2 Timing and Structure

In the stochastic arrival market, we have two directions to optimize the dynamic matching problem, that is, through timing and node connecting structures. Each node tends to find a match when the node itself becomes critical, the market size becomes thicker, and the number of objects that can be paired increases. Since the market gets thicker, each node has more choice to match or to be matched, this means the connection structure of nodes



doi:10.6342/NTU201902682

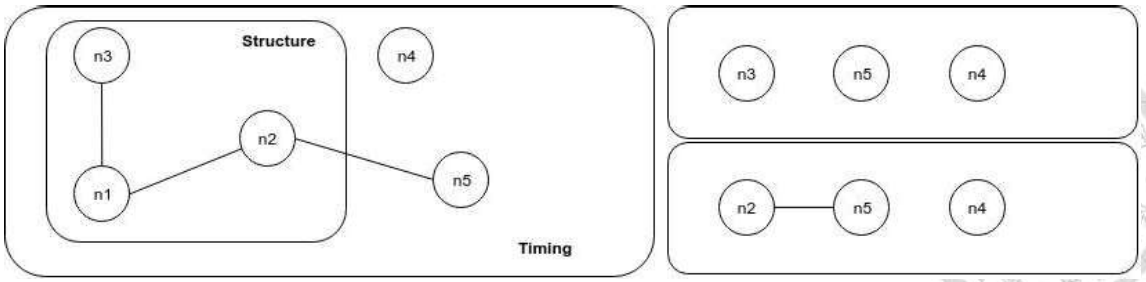



Figure 5.2: Different Market States Caused by Different Node $n_1$'s Matching Choice.

matters.

As shown in Figure 5.2, the node $n_i$ where $i = 1, 2, ..., 5$ arrives at the market in order. If each node matches immediately upon arrival, nodes $n_1$ will find a match whenever node $n_2$ arrives at the market, then no node can be matched with node $n_3$ when it arrives at the market. On one hand, if each node prefers to wait for a while, nodes $_n1$ will have the other matching choices. This is an example of the so-called timing issue. On the other hand, at this moment, node $n_1$ could choice nodes $n_2$ or $n_3$ but not both of them this is clearly a structural issue.

However, choosing the node which benefits the whole market will complicate the system a lot. For example, if node $n_1$ is aware of that its match with node $n_3$ will make nodes $n_2$ and $n_5$ get matched, which leads to a better system state than its choice to match with node $n_2$ directly. However, to reach such a better system state, one must globally compute all the possible linking situations of the market. Clearly, it will take $O(n^2)$ complexity, if the market size is $n$. Another fact is, in the dynamic matching market, if any node arrives at the market or leaves the market, all the linking possibility must be recalculated. Unfortunately, our system is blockchain based, there is no centralized planner and/or executor to handle the above-mentioned global computation. Therefore, as shown in Fig. 5.3, this thesis suggests postpending the actual match somewhat to thicken the market size, which in turn may increase the overall system matching rate.

In addition, timing and structure issues related to each other. If each node waits for a while to find a match, the number of nodes in the market will increase, and the nodes will have more possible links among one another.

For example, if there is only one link in $n_1$, as shown in Fig. 5.2, there is no other



doi:10.6342/NTU201902682

choice that will be more conducive to market structure. On the other hand, if the market already has sufficient thickness, the choice of each node will not have much impact on the structure of the market. For example, in Fig. 5.3, if $n_1$ match anyone of its neighbors, the other nodes always can find someone to match with. Compared with Fig. 5.1, if $n_1$ selects its match with $n_2$, the choice will make nodes $n_3$ and $n_5$ unable to find matches, we say this choice causes higher loss to the market.

For measuring the density of a market, we define the density parameter as $d = np$, where $d$ is the number of expected links of a node does not match anyone yet. After each node enters the market, there is a fixed available time $Ta$. If a node continues to wait for the entire available time, the node will meet the other $n$ nodes during this time period. Then the number of expected links of the node will be $np$.

## 5.3 The Greedy and the Patient Matching Strategies

This section presents two simple static matching algorithms based on waiting time and compares the impact of the two algorithms on the market size.

### 5.3.1 Greedy Algorithm

Greedy algorithm means that any node tries to find a match as soon as possible when it arrives at the market. If no match can be found, the node will remain in the market. Fig. 5.4 shows the processing snapshots of our greedy algorithm, with 4 continuous time units, in a market. When $n_3$ arrives at the market, it immediately matches $n_1$ and leaves the

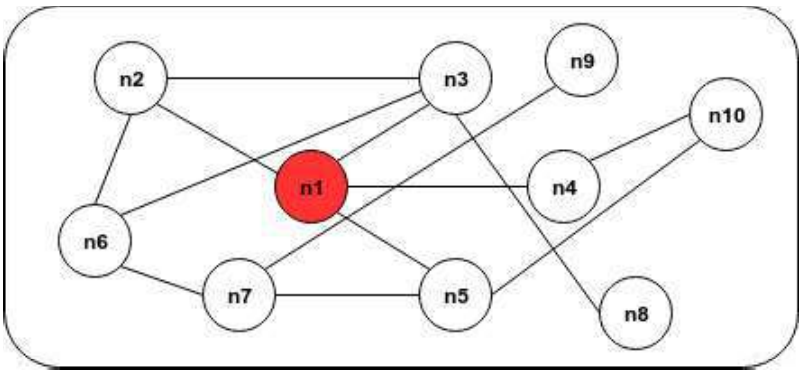



Figure 5.3: Node Connecting Structures and the Thickness of a Market.



doi:10.6342/NTU201902682

market. Therefore, any node remains in the market will have no link with others. And they can only wait for matches with nodes arriving at the market later.

When greedy algorithm is applied to the market, according to relevant researches, the market size will highly concentrate on a fixed range. This implies a balance between the arrival rate and the departure rate will be reached. Where the departure nodes include those, who leaving the market immediately after they found matches successfully and who leaving without match because of lifecycle expiration. Without loss of generality, the node arrival rate to the market can be assumed following the Poisson Process. As shown in Fig. 5.5, the change of the node number in the market can be expressed by a Markov Chain, where $x$ denotes the number of nodes in the market. First, the market arrival rate under the greedy algorithm can be written as:

$$r_{x,x+1} = n(1 - d/n)^x \tag{5.1}$$

where the probability $p$ is replaced by $d/n$. Equation 5.1 means any node gets a match when it arrives at the market while the nodes added to the market are those didn't find matches with the other nodes in the market. Second, the departure rate in the market can be expressed as:

$$r_{x,x-1} = x + n(1 - (1 - d/n)^x) \tag{5.2}$$

Equation 5.2 can be understood by the following two explanations:

- Leave without matched: Suppose we use $T_a$ as a time unit in the equation. After a $T_a$, there are $n$ nodes, in average, arriving at the market. However, as all nodes are

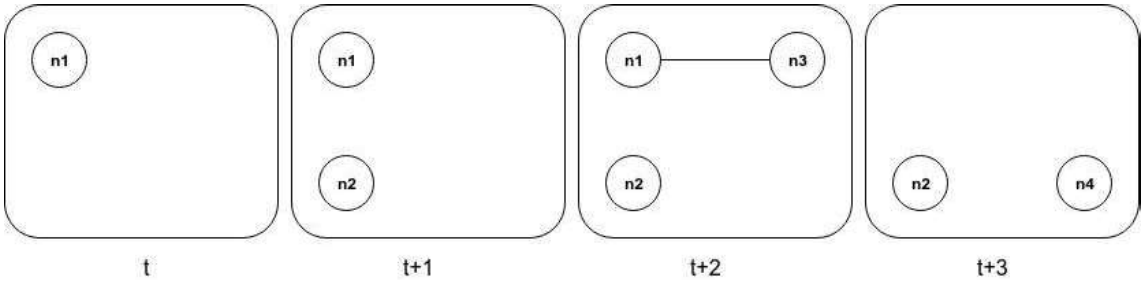



Figure 5.4: Matching Process Under the Greedy Algorithm.



doi:10.6342/NTU201902682

assumed didn't find matches, all nodes in the market will leave the market after a $T_a$ is passed. So, the number of departure nodes equals to the number of nodes $x$ in the market.

- Get match then leave: When the node arrives at the market and gets a match immediately, then the number of nodes in the market will be reduced accordingly, because the matched node will leave the market right away.

Therefore, the balance equation of the size of a stable market can be written as:

$$f(x) = n(1 - d/n)^x - (x + n(1 - (1 - d/n)^x)) \tag{5.3}$$

Equation 5.3 can be solved to find the highly concentrated number of market size, that is:

$$x* = \frac{n}{(2d + 1)} \tag{5.4}$$

The loss of the market under the greedy algorithm can be defined as the ratio of the node arrival rate to its departure counterpart as:

$$loss = \frac{1}{(2d + 1)} \tag{5.5}$$

### 5.3.2 Patient Algorithm

Patient algorithm means that any node gets a match only when it becomes critical, where critical means that the node has exhausted its available time. Fig.5.6 processing snapshots of our patient algorithm, with 8 continuous time units. In which node $n_1$ becomes critical,

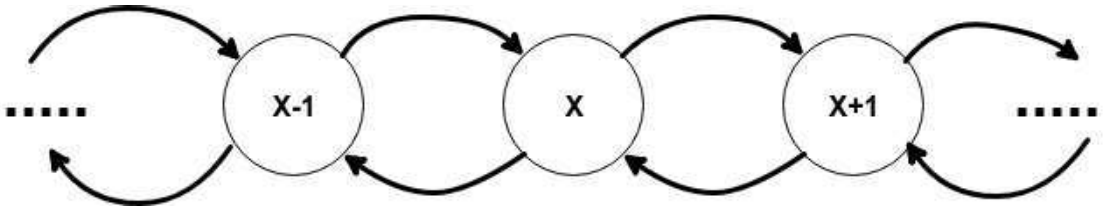



Figure 5.5: State Transition Graph of the Markov Chain under the Greedy Algorithm.



doi:10.6342/NTU201902682

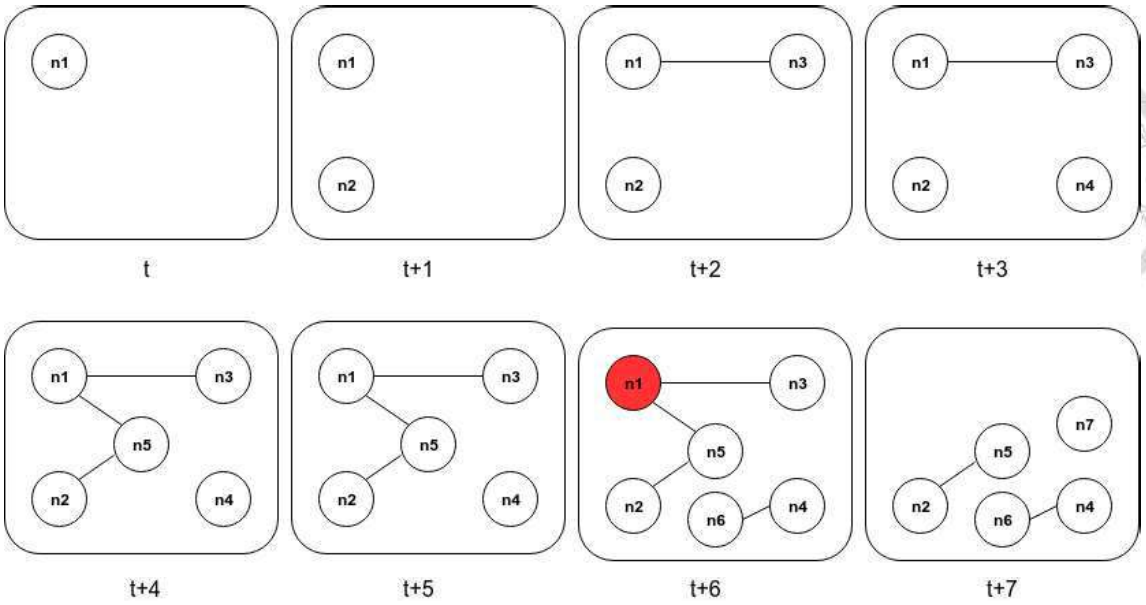



Figure 5.6: Dynamic matching market under the patient algorithm.

it randomly matches one of its neighbors. Because of the patient algorithm is applied, our system will postpone the possible matching and keep the nodes in the market, and therefore, increase the node density of the market.

Interestingly, similar to the case of greedy algorithm, when patient algorithm is adopted, the market size will also be highly concentrated within a fixed range. As shown in Fig. 5.7, the change (or the transition) of the number of nodes can also be expressed by a Markov Chain. Now, the analysis of node number transitions can be divided into the following three cases. First, the market arrival rate under the patient algorithm will be

$$r_{x,x+1} = n \tag{5.6}$$

Equation 5.6, reflects the fact that, under patient strategy, any node gets a match only when it becomes critical (i.e., its lifecycle time is nearly passed). So, any node enters the

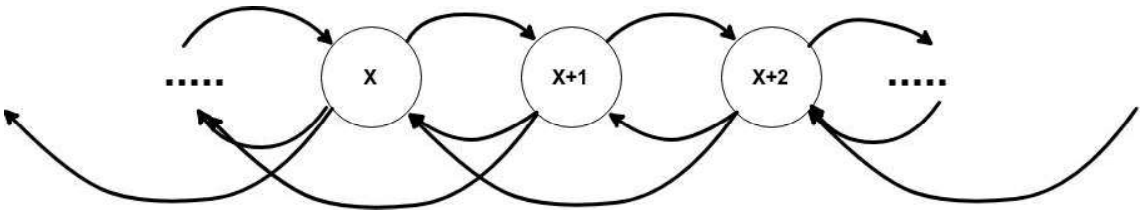



Figure 5.7: State Transition Graph of the Markov Chain under the Patient Algorithm.



doi:10.6342/NTU201902682

market will be remanded in the market. Second, the departure rate of the market can be written as:

$$r_{x,x-1} = x(1 - d/n)^{x-1} \tag{5.7}$$

Now if the number of nodes in the market changes from $x$ to $x - 1$, it will be caused only by node' s timeout. On the other hand, if a node matches with another node, there are two nodes leaving the market at the same time. So, the departure rate equals to the expected number of nodes in the market that cannot find matches and can be expressed by Equation 5.7. Third, the leave rate of the matched nodes can be written as:

$$r_{x,x-2} = x(1 - (1 - d/n)^{x-1}) \tag{5.8}$$

Equation 5.7 says nodes in the market get matched when they become critical and leave the market in pairs. So, a balance equation of the market size can be deduced as:

$$f(x) = n - (x + 1) - (x + 2)(1 - (1 - d/n)^{x+1}) \tag{5.9}$$

There is another more straightforward way to find the above-mentioned highly concentrated range. As shown in Figure 5.8, if any node matching with none of the critical nodes, then the market size should be $n$. On the other hand, if all critical nodes do find their corresponding matches, then the market size would be $n/2$. This is because when the arrival rate is $n$, the market size is $x$, all of them found their matches, the output rate should be $2x$, Now if an equilibrium state is reached, then $n = 2x$, so the $x$ will be $n/2$.

According to the concentrated node number of the market, the loss of the market under

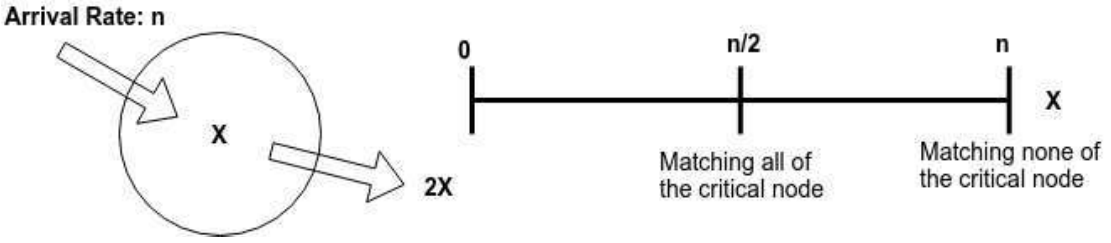



Figure 5.8: An easy way to find the highly concentrated range with the patient algorithm.



doi:10.6342/NTU201902682

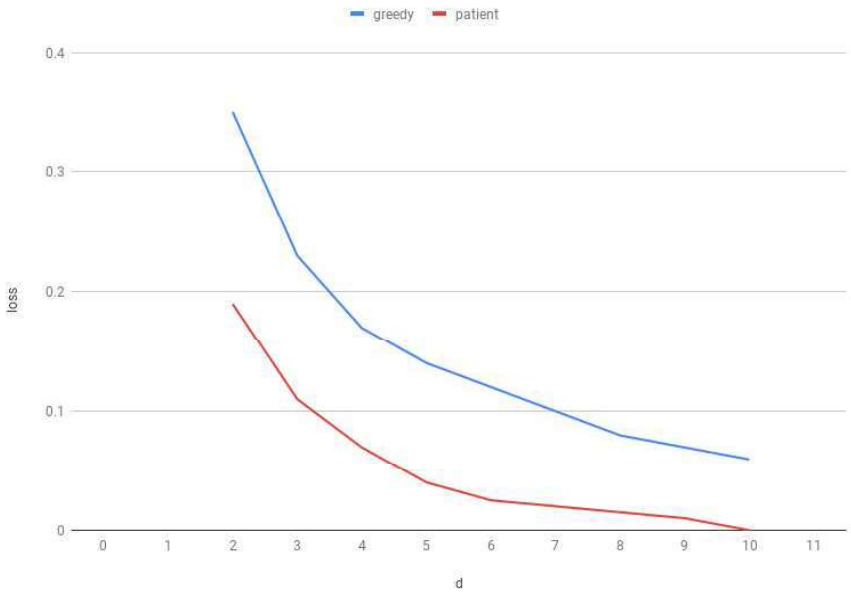



Figure 5.9: System Losses vs. the Number of Expected Links per Node (d), if the Patient and the Greedy Matching Strategies are adopted.

the patient algorithm can be computed as the ratio of the node arrival rate to the departure ratio, that is

$$loss = \frac{e^{-d/2}}{2} \tag{5.10}$$

### 5.3.3 Comparison

Fig. 5.9 shows the system loss under different market density conditions. Intuitively, greedy algorithm causes higher system loss than its patient counterpart. If the market density getting higher, the losses caused by the structural issues becomes smaller.

## 5.4 Dynamic Matching strategy

As pre-described our Time Bank is a community-based system, there are limited number of members in the market; therefore, the system performance is highly sensitive to the quantity of the market size. That is, for promoting the embracement of Time Bank, node density (or system size) of our system should be enlarged to an economical scale so as



doi:10.6342/NTU201902682

to provide attractive system performance. Therefore, in this section, a market thickness-based dynamic matching strategy is presented for bettering our system performance. The basic idea is that "increase the possible links per node does give each node more matching candidates". However, in a blockchain-based Time Bank, all of the matching strategies are realized on the chaincodes. That means the on-ledger matching strategy does not have the flexibility to meet the dynamically changed market conditions. For example, if the market is very thick, but all nodes are programed to follow the patient algorithm, it will cost a lot of time to find a match. So, what we need is a more flexible (or dynamic changing) strategy to handle different real market situations.

In some sense, a Dynamic Tuning Strategy (DTS) will also thicken the market to reduce system losses, just like the patient algorithm. But DTS does not need to postpone the match of a node until the node becomes critical. In this work, a system parameter $\alpha$ parameter that decides whether the market prefers the patient algorithm or the greedy is defined to control the system's preference about patient or greedy during the matching process, as shown in 5.10. When the market is thin, all in-market nodes prefer to take patient-oriented strategy. On the other hand, when the market is thick, all the nodes prefer not to wait too long to get a match.

Since a blockchain-based system does not have a global planner, so the matching process has to be run by all of the users. When someone posts a service inquiry to the blockchain, then the corresponding node will invoke the matching function. And this matching function should be DTS-based. Conceptually, the proposed DST mechanism is a mixture of the pre-described Greedy and Patient mechanisms, and it will be detailed in the next subsection.

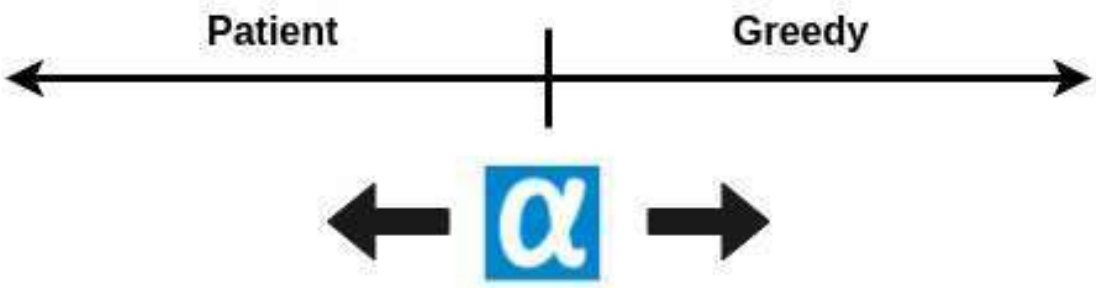



Figure 5.10: The Tuning Parameter Alpha.



doi:10.6342/NTU201902682

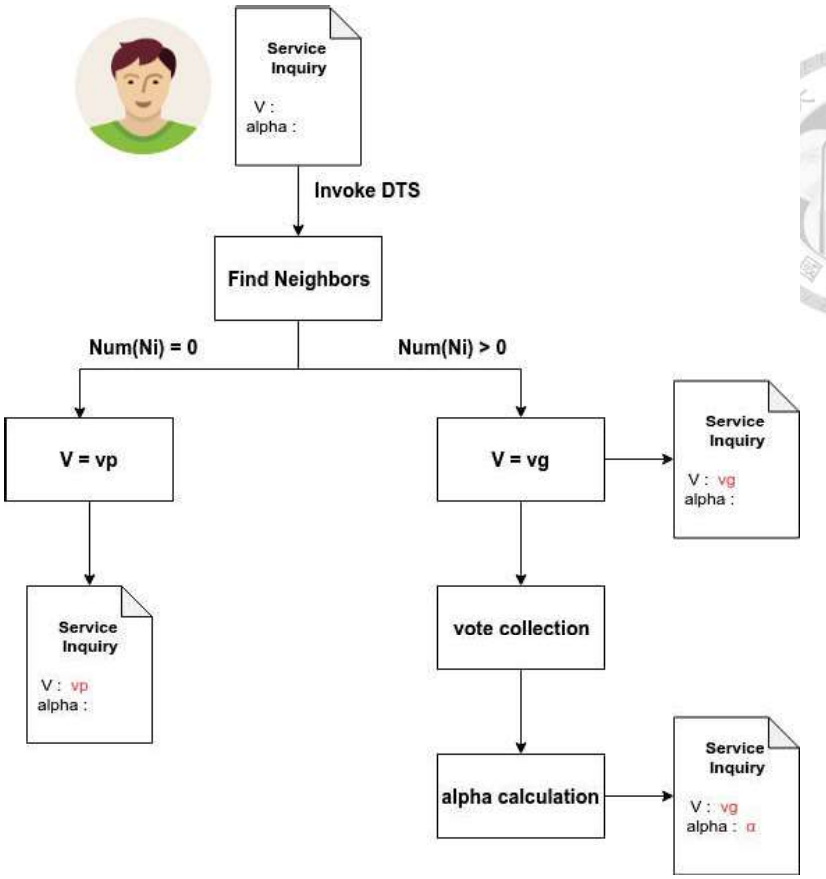



Figure 5.11: The Block Diagram of the Proposed Voting Mechanism, where $Num(N_i)$ denotes the number of neighbors of node $N_i$.

### 5.4.1 Voting Mechanism

After a user posted his or her service proposal on the blockchain, the corresponding node queries matching related data of the other inquiries on the blockchain. This thesis assumes $N_i$ is the set of neighbors to a service $S_i$, where a neighbor to the service $S_i$ means the matching condition of that node is compatible to that of the service $S_i$. Furthermore, twe also denote the set of voting state as $V$, for each posted service inquiry. Where the set of voting states, $V$, is consists of the following two elements:

- $v_p$: the state of the service inquiry tends to be patient for the current market.
- $v_g$: the state of the service inquiry tends to be greedy for the current market.

Every time the user invokes a DTS-based matching, the service inquiry can flag itself with vote state $v_p$ or $v_g$, and write the chosen state into the set $V$ in the service inquiry as shown in Fig. 5.11.



doi:10.6342/NTU201902682

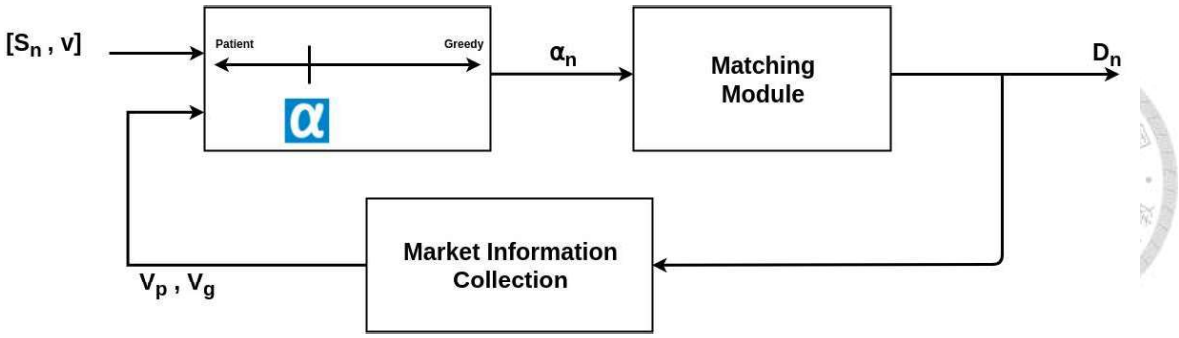



Figure 5.12: The Block Diagram of the Proposed DTS.

If $S_i$ does not have any neighbor, it means $S_i$ does not compatible with the matching conditions of other service inquiries in the current market. That $S_i$ cannot complete the match in this time, so it prefers other nodes will thicken the market so that it can find neighbors at the next time DTS-based matching is invoked. Therefore, that $S_i$ will flag its state of vote to $v_p$ when invoking DTS-based matching this time, which means that $S_i$ expecting every other node to wait for matching patiently.

If $S_i$ has neighbors, it means there are other service inquiries in the market whose matching conditions are compatible with that of $S_i$. Since $S_i$ has neighbors, it prefers completing the match as soon as possible. So the $S_i$ will flag its state of vote to $v_g$, which means that $S_i$ embracing the greedy market strategy more. Fig. 5.12 depicts the Black Diagram of the proposed DTs for matching.

After voting, if the number of neighbors in the $S_i$ is greater than zero, then $S_i$ investigates the voting states of other service inquiries in the blockchain and calculates the value of $\alpha$. In this work, $\alpha$ is defined to be the ratio of the number of $v_p$ with respect to the size of the current market, that is

$$\alpha = \frac{|V_p|}{|V_p| + |V_g|}, 0 \leq \alpha \leq 1 \tag{5.11}$$

First let's discuss two extreme cases for $\alpha$:

- If all nodes in the market do not have any neighbor, all inquiries will change their voting states to $v_p$ . At this time, $\alpha = 1$, and the effect of our DTS-based matching is equivalent to that of a patient algorithm, and all nodes in the market have a consistent choice: to wait patiently.



doi:10.6342/NTU201902682

- If all nodes in the market do have neighbors, all inquiries will change their voting states to $v_g$. At this time, $\alpha = 0$, and the effect of our DTS-based matching is equivalent to a greedy algorithm, and all nodes in the market have a consistent choice: to match as soon as possible.

Second, from Equation (5.11), the value of alpha will be dynamically tuned according to the ration of the number of $v_p$ with respect to the total votes of the market. As the voting states of the system members are changing, the strategy for matching is also changed dynamically.

### 5.4.2 Match Decision Function

The realization of the proposed DTS depends the following Match Decision Function:

$$TH = m - \lfloor m * \alpha \rfloor \tag{5.12}$$

$$D = NOAI - TH = \begin{cases} Wait & Otherwise \\ Match & D \leq 0 \end{cases} \tag{5.13}$$

Where $NOAI$ denotes the remaining number of available time periods for the service inquiry. In addition, $m$ is the maximum times which the service inquiry can invokes the matching process in its lifecycle.

For example, if $m = 10$ and the service inquiry $S_i$ still have $6$ chances left in the current pairing procedure, then $NOAI = 6$. assuming $\alpha = 0.5$, then $TH = 5$ And $D = NOAI - TH = 6 - 5 = 1$ and $S_i$ will continue to wait until the next call for the matching function.

As one matching period is passed, $S_i$ in the new matching process will have $NOAI = 5$. Assuming $\alpha = 0.4$ again, then $D = 0$ and the corresponding $S_i$ will try its best to match with one of its neighbors and leave the market. The total process of our DTS-based matching process is illustrated in Fig. 5.13

In addition, the longer a node remains on the market, the easier it is to be matched.



doi:10.6342/NTU201902682

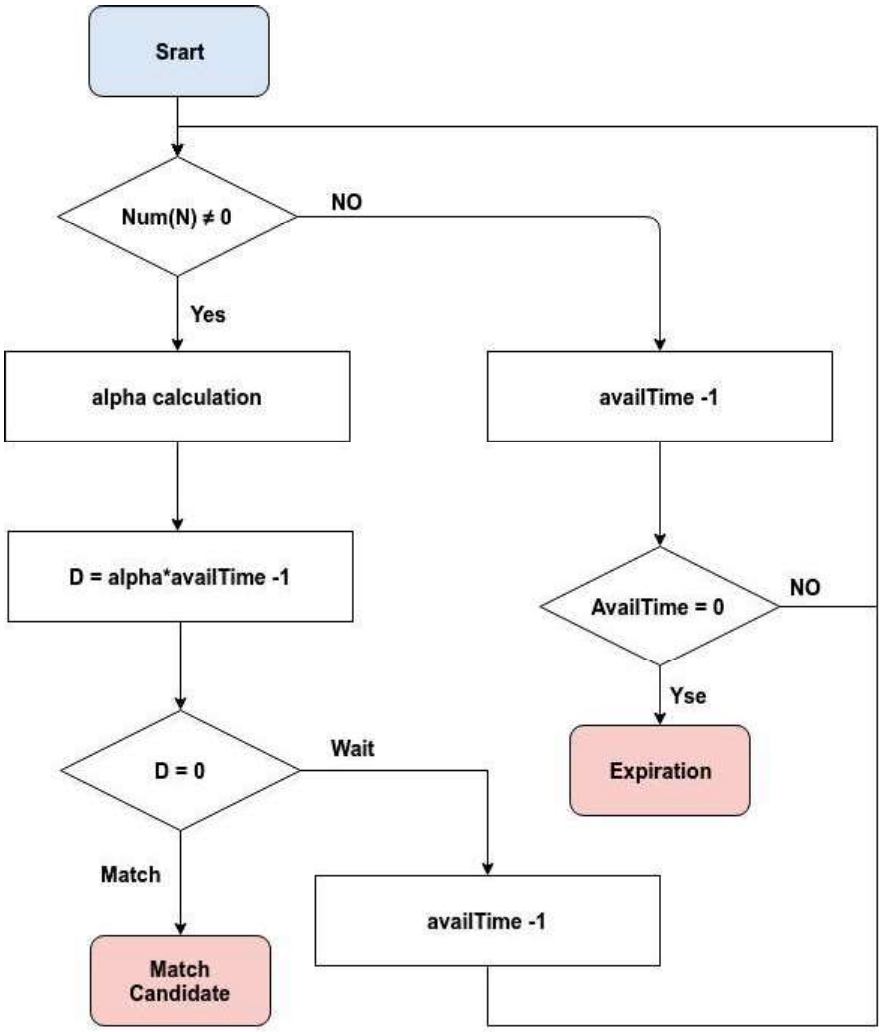



Figure 5.13: The Flow Chart of the Proposed DTS-based Matching Mechanism.

Therefore, all nodes in the market will operate according to the remaining feasible time and will have the above-mentioned characteristics.



doi:10.6342/NTU201902682

# Chapter 6

# Experiment

In this section, we simulate the whole process starting from users' stochastically arrivals to the blockchain, posting their service inquiries, then waiting for being matched. Our experimental system and simulation are conducted on a personal computer with Intel 6700K CPU, 24GB RAM, and using Ubuntu 16.04. The Fabric version is 1.4. And all the nodes are running in docker where the version is 18.09.3.

## 6.1 Service Posting

The user uses the client application to post the service inquiry to the blockchain network as shown in 6.1. If the service inquiry is successfully posted, the system will return the TxID(Transaction ID) in this transaction. When we invoke the chaincode in the

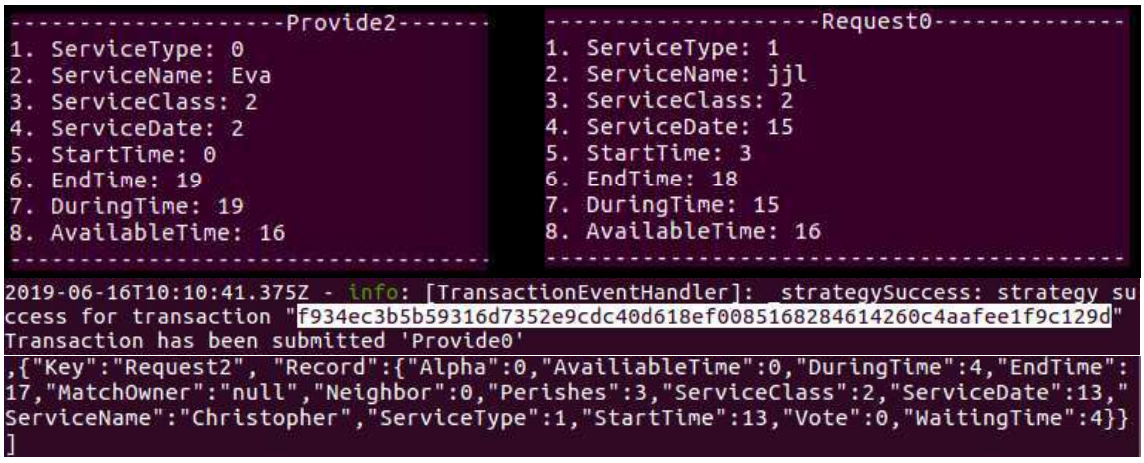



Figure 6.1: The two types of the service inquiries.



doi:10.6342/NTU201902682

```
provideNum: 13 requestNum: 4 unmatchNum: 3 matchedNum: 2 perishNum: 10 averageNum: 1 timeHour: 22 timeMin: 48 timeSec: 2 neighbor 3
provideNum: 13 requestNum: 4 unmatchNum: 3 matchedNum: 2 perishNum: 10 averageNum: 1 timeHour: 22 timeMin: 48 timeSec: 5 neighbor 3
provideNum: 13 requestNum: 4 unmatchNum: 3 matchedNum: 2 perishNum: 10 averageNum: 1 timeHour: 22 timeMin: 48 timeSec: 8 neighbor 3
provideNum: 13 requestNum: 4 unmatchNum: 1 matchedNum: 3 perishNum: 10 averageNum: 1 timeHour: 22 timeMin: 48 timeSec: 11 neighbor 5
```

Figure 6.2: The ServiceScan.

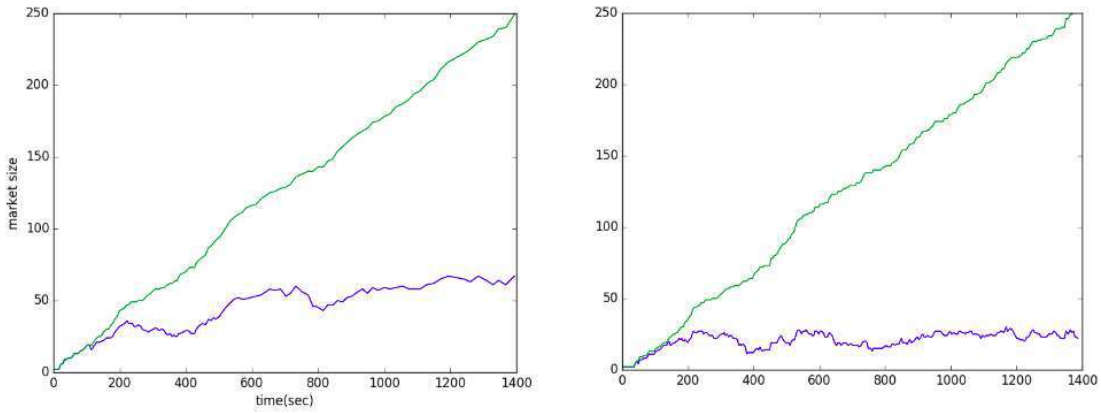



Figure 6.3: Simulated Market Sizes for Patient and Greedy Algorithms.

blockchain, we can call this action is sending a transaction to the blockchain. And each transaction can be traced according to the TxID. Also, the service inquiry can be queried on the ledger of the blockchain with the state, like Provide2 or Request0. So, if user wants to query his service inquiry, he can using the state to check the status with the service inquiry.

The experiment simulates a real time banking situation: there are lots of members using the application and posting many different service inquiries. Fig. 6.1 shows that there are 2 types of service inquiries are posted from different users, Eva and JJL. The experiment also established a ServiceScan which is a monitor to observe the total service inquiry in the blockchain network as shown in Fig. 6.2. ServiceScan collects all of the services and them according to different states.

## 6.2 Matching

Our matching experiments consist of two parts. The first is an experiment of matching under the greedy and patient algorithms. The second is to simulate finding a match in the real market scenario under the greedy, the patient, and the DTS-based matching algorithms. Where the system loss is used to measure the corresponding performance.



doi:10.6342/NTU201902682

| | Loss | Waiting Cost | Average Neighbor |
|---|---|---|---|
| Greedy | 0.328 | 0.12 | 1.184 |
| Patient | 0.264 | 0.86 | 3.864 |
| DTS | 0.283 | 0.36 | 3.12 |

Table 6.1: The Performance Comparison Among Three Different Strategies.

In our simulation:

- The average arrival time interval is $5$ second and follows a Poisson Process.
- The total number of services = $250$.
- The matching period is $40s$ seconds
- The Available time: $200$ seconds

Fig. 6.3 shows the simulation results under the two static algorithms. he market will highly concentrated on quantity ranges under both algorithms. The concentrated range under Greedy algorithm should be approximately equal to that of $[6, 15]$, and the range under the Patient algorithm should be approximately equal to that of $[20, 40]$. It is worth mentioning that because the service inquiry performs the matching process on the blockchain platform, there is a delay in querying and writing records to the ledger. The time units in our simulation is in seconds, which makes the delay time more obvious. The delay time will cause the service to stay in the market for a longer period, so the result of the simulation is slightly higher than that of the calculated one.

Table. 6.1 shows the system performances under the different three algorithms. The loss of the greedy algorithm is the largest, but the waiting time is the shortest. While the patient algorithm causes the smallest loss; however, it takes the longest waiting time to find a match. The loss obtained in the DTS-based matching algorithm is only a little worse than that of the patient algorithm, but it greatly reduces the waiting time. That is to say, if DTS-based matching algorithm is used, users can save more than half of the waiting time, and the loss can be controlled to an acceptable range. More importantly, as for the average neighbors, the greedy algorithm gives only one neighbor to one node when it finds a match.



doi:10.6342/NTU201902682

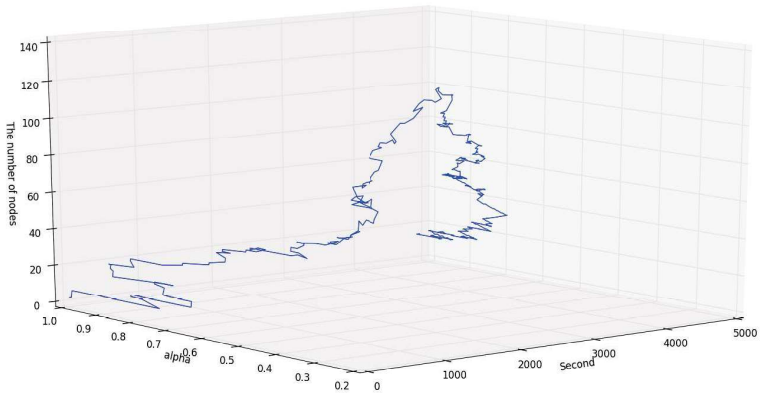



Figure 6.4: Simulated Market Sizes amd alpha values for DTS-based matching Algorithms.

## 6.3 DTS-based Matching Simulation

In Fig. 6.4, we use a DTS-based matching algorithm to simulate the matching scenario in the time banking system. Unlike previous simulations, no service will arrive at the market after a certain point in time. This simulation will show that when the service to the market changes drastically, the system's strategy will automatically tune the alpha and maximize the overall matching rate.

From the results of the simulation, we can divide it into three phases:

- When the system first started running, the market was very small, so the vote for each service caused the alpha changing greatly. On the other hand, because the market is small and the probability of matching is low, alpha will be biased towards 1, and the timing strategy of the system tends to wait.

- Until the market gets thicker, the alpha will slowly become smaller and smaller, and the alpha value changes will become smoother, representing that services in the market can get matched in shorter waiting times.

- When the arrival of the service is interrupted, the alpha will gradually increase, allowing the market to maintain the match rate in the market through the waiting



doi:10.6342/NTU201902682

mechanism.

For time-based banks in the community-based market, DTS can stabilize the market thickness and effectively provide a high matching rate. Compared to the static matching algorithm, DTS will be able to flexibly change the timing strategy based on the number of nodes in the market and the number of links for each node in the market. In addition, time banks in different communities will have different arrival rates and matching rates.DTS-matching does not need to design different matching algorithms according to different markets. As nodes arrive in the market in sequence, the algorithm will automatically tune to the appropriate timing strategy.



doi:10.6342/NTU201902682

# Chapter 7

# Conclusion and Future Work

This thesis presents a blockchain based Time Bank system with service-exchange and flexible matching mechanism. In the system, because the matching system is realized on the blockchain platform. It brings the system data transparency, high scalability, and the fairness of the matching strategy. However, build on the blockchain platform also have some challenge and compromise. Like the system can't globally compute the optimize choice in the network and need the user running the matching process locally. The thesis also proposes a useful matching mechanism on the blockchain. The dynamic tuning strategy increases each node's neighbor without cost a lot of waiting time. when the market structure changed, becoming thicker or becoming thinner, the matching strategy will dynamically adjust.

In future work, the system will integrate the user's grading mechanism into the matching mechanism. It will allow the system to select matching objects with reference to personal preferences. This integrated system will bring good user experience and feedback. Furthermore, In the design of the matching mechanism, an incentive mechanism for unbalanced supply and demand will be considered. The SP and SR can use the different matching strategy independently in the unbalanced supply and demand market. If there are lots of SR, but just a few SP in the market, the matching strategy of SP may can more greedy than SP.



doi:10.6342/NTU201902682

# References


[1] *S. Nakamoto,” Bitcoin: A Peer-to-Peer Electronic Cash System,” 31 October 2008 .[Online] https://bitcoin.org/bitcoin.pdf*

[2] *V. Buterin,” Ethereum: a next generation smart contract and decentralized application platform,” 2013.[Online] https://github.com/ethereum/wiki/wiki/White-Paper*

[3] *Hemant K. Bhargava, ”Incentives for Selling Two-Sided Markets”*

[4] *A Chakraborty, ”Fair Sharing for Sharing Economy Platforms”,August 2017*

[5] *E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. D. Caro et al.” Hyperledger fabric: A distributed operating system for permissioned blockchains,” Proceedings of the 13th ACM SIGOPS European Conference on Computer Systems, 2018.*

[6] *M. Akbarpour,”Thickness and Information in Dynamic Matching Markets”, December 2018*

[7] *Itai Ashlagi, ”On Matching and Thickness in Dynamic Markets”,20 January 2016*

[8] *Itai Ashlagi, ”Matching in Dynamic Imbalanced Markets”,9 March 2019*

[9] *Morimitsu Kurino, House allocation with overlapping agents: A dynamic mechanism design approach. Jena economic research papers, 2009.*

[10] *Francis Bloch and Nicolas Houy, Optimal assignment of durable objects to successive agents. Economic Theory, September 2012.*




doi:10.6342/NTU201902682

[11] *D. Gale and L. S. Shapley, College Admissions and the Stability of Marriage, Amer. Math. Monthly 1962.*

[12] *K. Iwama, "A Survey of the Stable Marriage Problem and Its Variants", 2008 IEEE Informatics Education and Research for Knowledge-Circulating Society*



doi:10.6342/NTU201902682