Essays in Matching Theory

Author: Jinyong Jeong

ESSAYS IN MATCHING THEORY

Jinyong Jeong

A dissertation
submitted to the Faculty of
the department of Economics
in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

Boston College
Morrissey College of Arts and Sciences
Graduate School

April 2018

ESSAYS IN MATCHING THEORY

Jinyong Jeong

Advisors: Prof. Utku Ünver, Prof. Tayfun Sönmez, Prof. Bumin Yenmez

My doctoral research focuses on the matching theory and its market design application. Specifically, I work on matching with property rights, where property rights not only mean the ownership, but also refer to the ability to determine how the good is used. In the matching with property rights model, an agent who owns a resource can claim how her resource is offered, depending on what she gets from the system. For example, in a housing exchange for vacation, an agent who gets a house with a car will offer her house also with a car. However, if she is assigned only a house without a car, she might refuse to offer a car. This restriction can be thought as a matching with externality, as someone's consuming my resource in certain way affects my utility. With property rights present, it is not clear how we can achieve a desirable outcome while satisfying the rights. I am currently pursuing two main lines of research in this topic that constitute the two chapters dissertation.

In *Matching with Property Rights: an Application to a Parking Space Assignment Problem*, I introduce parking in urban areas as a matching problem. First, I model the street-parking market as a strategic game and show that the set of Nash equilibrium outcomes is equivalent to the set of stable allocations. However, it is not reasonable to expect drivers to reach a Nash equilibrium in the decentralized system due to lack of information and coordination failure. Therefore, I suggest a centralized mechanism that would enable a parking authority to assign available spaces to drivers in a stable way. The model incorporates resident parking spaces, such that visitors could access vacant resident spaces. To use the resident parking spaces, the system needs to protect exclusive property rights over their parking spaces. I show that, however, there is no mechanism that is stable and protects residents' rights. To resolve this issue, I introduce a new concept, a claim contract, and suggest a mechanism that protects property rights, is strategy proof for the drivers, and approximates a stable matching. Besides its market-design focus, this paper handles both priority-based and property right-based assignment, which considered separately in the matching theory literature.

In *Housing Market with Contracts*, I study matching with property rights problem in the

housing market framework. To introduce property rights in housing market, I assume the house can be offered in two contractual terms. Property rights requires that when an agent gets a house in a certain term, her house should also be offered as the same term. Moreover, when every agent owns a house, property rights reduces to an equal-term matching. After defining efficiency and core in equal-term domain, I show that, in a housing market with contracts problem, core may be empty. However, there always exists an efficient, individually rational, and equal-term matching in every housing market with contracts problem. Then I present a mechanism that always produces an efficient, individually rational, and equal-term matching. This is the first attempt to model a matching with contract in a exchange economy.

To Yuhwa and Jinyoo

ACKNOWLEDGEMENTS

This dissertation has been a long journey, which I cannot accomplish alone. Along the way, many people have supported me, and my work would have been absolutely impossible without them.

I am grateful to my advisors for their guidance, advice, and for being a truthful evaluator of me and my work. I thank Utku Ünver, who encouraged me at times when I was in lack of confidence. Being his student, from my first year of the program, having him as my advisor, is my honor and it made me a better researcher and economist. I thank Tayfun Sönmez, who helped me in focusing on crucial part of the research, which was scattered all around. I learned a lot from him in his class and seminars, who has a genuine interest in learning new things. I thank Bumin Yenmez, who have helped and talked me a lot, even shortly after he joined the department. Thanks to him, I had a chance to introduce my work to the Pittsmart team, which ended up my new position. I owe my advisors all my accomplishment.

I thank Marvin Kraus, who has guided me to learn urban economics, and introduced me various conferences including ITEA in Barcelona. Thanks to him, I had great chances to present my work and get to know the urban and transportation economics field. I thank Hideo Konishi, who gave me a chance to have a great opportunity to teach, which also helped me to overcome financial difficulties. His consideration, kindness, and support will remain me all the time.

My deepest thanks go to my wife, Yuhwa Hong. She has been my pillar for all my life. Her love, patient, sacrifice, and support has shaped me. Without her, I could not come to this point. I owe her everything.

Writing this dissertation has had a big impact on me, which has been a period of learning myself as well. I thank Jesus, who has been always inside me, even when I was only looking at myself. God's endless love changed my life, and will change me forever.

TABLE OF CONTENTS

CHAPTER 1

MATCHING WITH PROPERTY RIGHTS: AN APPLICATION TO A PARKING SPACE ASSIGNMENT PROBLEM

In a modern city, such as Boston, finding a parking space may be challenging. A driver often must cruise to find a parking space, sometimes wasting more time than the time spent in driving[1]. This cruising behavior not only wastes the driver's own time and energy, but also contributes to traffic congestion and air pollution. The demand for parking is derived from the demand for driving, which itself is derived from the demand for various activities. Considering this interconnection between parking and other economic activities, parking problems have implications on traffic regulation, parking fee, urban transportation policy, and city design, among many other policy concerns.

One way to mitigate these problems is to assign available parking spaces in a centralized system. The main objective of this paper is to identify how to allocate parking spaces to drivers, while minimizing negative side effects of cruising-for-parking behavior. Inefficiencies in the decentralized parking problem are a result of wasted parking spaces (both the visitor and resident spaces) and mismatch of spaces (two drivers who prefer each other's assignment). I suggest a centralized system to assign available parking with a matching and mechanism design approach to mitigate parking problems.

In this paper, I apply matching theory to address the parking problem often studied in urban economics. I first analyze the decentralized parking problem as a cruising game, where drivers attempt to find a parking space while competing with each other for spaces. In this game, a driver who is closer (normalized by speed, road condition, etc.) to a space can arrive at that space earlier than others. Therefore, the driver has an incentive to drive

[1] According to Shoup (2005), an average, 30% of vehicles cruise for a parking spot, spending an average of eight minutes.

to a closer space rather than their preferred space. This observation shows the similarity of this game to the well-known Boston mechanism of Ergin and Sönmez (2006) in the school choice context, where parents have incentive to rank nearby schools above their preferred choice. This similarity leads to the observation that the set of Nash equilibrium for the cruising game is equivalent to the set of stable matchings. In this game, the "distance" of a driver to a parking space is similar to the "priority" of a student at a school. Therefore, stability appears as a natural solution concept for the equilibria of the decentralized street-parking game. Moreover, other features, such as prices, inherent parking permits, etc, may lead to more general priority structures. Thus, I model the parking problem as a matching model, and build a mechanism that always produces a stable allocation.

This work has two main contributions, practical and theoretical. Practically, it studies downtown parking problems in a matching and market design approach. Theoretically, this paper introduces a new matching model for both a priority-based and property-right-based allocation problems.

I study a special case of matching, where some of the agents have property rights while the rest of the allocation is done using priority.[2] When property rights are respected, no mechanism can always produce a stable allocation. To resolve this incompatibility between stability and property rights, I introduce a new concept, a **claim contract**. A claim contract is an auxiliary contract and does not involve any physical space, but it serves as a protection for property rights. With a claim contract, I construct the agents' choice functions. Due to the nature of the claim contract, the choice functions do not satisfy any of the previously known conditions that guarantee the existence of a stable allocation. However, I show it is still possible to produce a "practically" stable allocation, which approximates stability in an acceptable manner.

After I analyze the model, I suggest practical implementation of this model. First, I include price into the contract, so that the drivers can bid up for the spaces. Second, I suggest different priority structures that a parking authority can take depending on its objective. Third, I suggest a simple way of collecting preferences from the drivers, named as GDP

[2]Property rights refer to the ownership of specific property by individuals and the ability to determine how such property is used. In this paper, the latter part of the definition is the focus.

system. Finally, I discuss how the process would work in real-time (i.e., dynamically).

Ayala et al. (2012) analyze a parking space assignment game and devised an algorithm for the Nash equilibrium outcome. However, they implicitly assume the drivers' preferences are solely determined by their proximity to the parking spaces. While this assumption holds well in certain situations, it would not hold if drivers expect to park closer to their final destinations. In this paper, drivers' preferences are fully general, so one driver may prefer a space closer to her destination, while another prefers a space closer to his current location.

The seminal papers van Ommeren et al. (2011) and van Ommeren et al. (2015) study the welfare implications of resident parking permits. Considering the presence of a welfare loss with resident parking permits, utilizing resident spaces when the residents are absent can be a partial solution. Therefore, I incorporate resident parking spaces into the model, such that the temporally vacant resident spaces can be used more efficiently.

Geng and Cassandras (2012) investigate a smart parking system from an engineering standpoint. A smart parking system provides parking information to drivers. It is better than providing no information to drivers, but assigning a driver a space that is exclusively reserved for that particular driver is preferred.

Matching theory has been widely studied, starting with Gale and Shapley (1962), and has since been applied to real-life problems including matching for labor markets, house allocation, school choice, and organ exchanges. Hatfield and Milgrom (2005) extend the matching model to a matching-with-contracts model, and Hatfield and Kojima (2010) show that the bilateral substitutes condition is sufficient for the existence of a stable allocation in a matching-with-contracts framework. Hatfield and Kominers (2016) introduce substitutably completable preferences, which guarantee the existence of stable outcomes while remaining independent of bilateral substitutes' preferences. One of the examples of substitutably completable preferences, task-based preferences, is closely related to this paper, as each parking space has specific preferences depending on the purpose of the space, either for residents or visitors. Yenmez (2017) defines path-independence and path-independent modifications. While being independent of the substitutes condition, he shows the condition that also guarantees an existence of stable allocations.

This paper is organized as follows. Section 2 defines the parking problem as a game and shows that the Nash equilibrium of the game is stable in a matching framework. Section 3 includes residents and their spaces in the system. Section 4 introduces a new idea of dealing with residents' property rights and constructs choice functions. Section 5 introduces a mechanism that protects residents' rights and produces a practically stable allocation. Section 6 suggests a practical application of the model and shows positive results by restricting underlying preferences or priorities. Section 7 concludes. In the Appendix, I touches a dynamic nature of the mechanism and a many-to-one extension of the model.

1 PARKING MODEL AND THE CRUISING GAME

1.1 PARKING MODEL

In a parking space assignment problem, there are a number of available parking spaces and a number of drivers who demand a parking space. Drivers have strict preferences over parking spaces, and parking spaces are assigned to the first-arriving driver, which we can interpret as proximity priority.

A formal description of the parking space assignment problem is:

- a set of n drivers $I = \{i_1, \cdots, i_n\}$,
- a set of m parking spaces $S = \{s_1, \cdots, s_m\}$,
- drivers' strict preferences $\succ = (\succ_{i_1}, \cdots, \succ_{i_n})$, and
- a distance function $d : IXS \to \mathbb{R}$, which denotes the distance between a driver and a space.

A driver i, who is closer to a space s in comparison to another driver i', can drive to the space s faster than i'. In this sense, distances can be interpreted as priorities of the drivers for each space.

- Priority order P_s for a space s is given by the function d, so that the closer driver to space s has a higher priority at s, i.e.,

$$iP_si' \quad \text{if and only if} \quad d(i,s) < d(i',s).$$

The outcome of the problem is called a matching, which is a function that assigns the spaces to the drivers. Formally, a matching $\mu : I \to S$ is a function from the set of drivers to the set of spaces, and $\mu(i)$ is the space that matched to driver i. $\mu^{-1}(s)$ will denote the driver assigned to space s.

The parking space assignment problem can also be viewed as a two-sided matching market(Gale and Shapley, 1962). To do that, we treat spaces as agents by interpreting the priorities of the spaces as strict preferences so that spaces prefer closer drivers, i.e., $iP_s i'$ means space s prefers driver i to driver i'.

In the two-sided matching context, a pair (i, s) blocks a matching μ if

$$s \succ_i \mu(i) \quad \text{and} \quad iP_s\mu(s).$$

The matching μ is stable if there is no blocking pair. The set of stable matchings is non-empty for any two-sided matching market, and there exists a single stable matching that every driver weakly prefers to any other stable matching (Hatfield and Milgrom, 2005). The subsequent sections of this paper refers to this matching as the driver-optimal stable matching, or driver-optimal stable allocation.

1.2 CRUISING GAME

In a decentralized system, drivers face a game situation, namely a cruising game. In the cruising game, the drivers are the players, each of whom demands a parking space. The strategy set is S, the set of spaces. Each driver chooses a parking space by searching and park there if it is empty when they arrive. Let σ_i denote the strategy of driver i, and $\sigma_I = \{\sigma_1, \cdots, \sigma_n\}$ be a strategy profile of all the drivers.

Definition 1 (Nash Equilibrium)**.** *Let $A(\sigma; I, S) : I \to S$ be a assignment function from a strategy profile to the set of spaces, and $A_i(\sigma)$ be the space assigned to driver i when the drivers' strategy is σ. A strategy profile $\sigma^* = \{\sigma_1^*, \cdots, \sigma_n^*\}$ is a Nash equilibrium of the cruising game if, for all i and σ_i,*

$$A_i(\sigma^*) \succeq_i A_i(\sigma_i, \sigma_{-i}^*),$$

where σ^*_{-i} denotes the strategy where all drivers except i follow the equilibrium strategy. In other words, a strategy profile is a Nash equilibrium if there is no driver who can improve his matching by changing his strategy alone.

1.3 STABILITY OF EQUILIBRIUM

The cruising game is similar to the well-known Boston mechanism in the following sense: drivers apply for the spaces and the highest priority (closest) driver is allocated the space. However, the driver's priorities are lost if they do not apply for the space. In other words, the drivers compete for a space among the drivers who ranked the space first. By applying the result of Ergin and Sönmez (2006), we see that the Nash equilibrium outcome of the cruising game is stable.

Theorem 1. *The set of Nash equilibrium outcomes of the cruising game is equal to the set of stable matchings of the parking space assignment problem.*

Proof. 1. If μ is a Nash equilibrium outcome, then it is stable.

Let $\sigma^* = (\sigma^*_1, \cdots, \sigma^*_n)$ be a Nash equilibrium strategy profile and μ be the resulting matching such that $A_i(\sigma^*) = \mu(i)$. Assume μ is not stable. Then there is a driver-space pair (i, s) such that driver i prefers space s to his assignment $\mu(i)$, and either space s remains unmatched or i is closer to s than the driver $i' = \mu^{-1}(s)$. If i changes his strategy to $\sigma_i = s$, then under the strategy profile $\sigma = (\sigma^*_{-i}, \sigma_i)$, driver i will be assigned s. Therefore, μ is not a Nash equilibrium outcome, contradicting the assumption.

2. If μ is stable, then it is a Nash equilibrium outcome.

Let μ be a stable matching. In a cruising game, if each driver goes to the space that they are assigned under μ, i.e., if the strategy profile is $\sigma = (\mu(i_1), \cdots, \mu(i_n))$, then the cruising game ends at the first step and the resulting matching is μ.

σ is a Nash equilibrium because no driver can profitably change his strategy from σ, hence μ is a Nash equilibrium outcome. If a driver i prefers another space s to his matching $\mu(i)$, the driver who is matched to s has higher priority than i by stability.

□

Theorem 1 shows that stability is a natural equilibrium concept in the cruising game, and it justifies introducing a centralized mechanism to this market. In the decentralized cruising game, it is challenging for drivers to reach a Nash equilibrium. First, drivers are not aware of all the available parking spaces, resulting in a waste of some of the spaces. In addition, they are not aware of the locations and preferences of other drivers, so coordination failure could result in allocations that are not in equilibrium. Therefore, by introducing a centralized system which assigns available spaces to drivers in a stable manner, a better-than-market, if not the best, outcome can be obtained.

Moreover, among all possible equilibrium outcomes, there exists one that is best for the drivers, the driver-optimal stable allocation. This allocation can be found by the following deferred acceptance algorithm.(Gale and Shapley, 1962)

Example 1. *DPDA: Drivers Proposing Deferred Acceptance.*

Step 1: *Each driver i proposes to her first choice (among all acceptable choices).*
Each space s tentatively holds the closest proposal, if any, and reject the others.

⋮

Step k: *Any driver who was rejected at step k-1 proposes to the best acceptable space which she hasn't yet made an offer.*
Each space holds the closest proposal among all the offers including it was holding, and rejects the others.
If no rejections occur, finalize the mechanism and match the "holding" offers.

DPDA results in a driver-optimal stable matching, that is, all drivers prefer it at least as well as any other stable matching. Therefore, it is best for the drivers under the stability requirement. DPDA is strategy-proof for drivers, so it is the dominant strategy for each driver to submit his/her true preferences.

DPDA is worst for parking spaces, implying the total distance traveled is maximized among all stable matchings. Since we want to minimize the negative side effect of cruising, it would be better if we could minimize distance traveled, using a space-proposing version

of deferred acceptance. Note that, DPDA would be better than a decentralized system, however, since the drivers will not be cruising for parking spaces.

2 RESIDENTS AND THEIR SPACES

In this section, by extending the parking model to a matching-with-contract model, I include resident parking spaces and different contractual terms.

We could build a model with exclusive visitor parking spaces, treating resident parking as a distinct market from the public parking market. However, by including residents' parking spaces in the system, we can allocate parking spaces more efficiently, because resident parking spaces are not always occupied.

Therefore, the set of drivers, I, now includes residents, $r \in I_R$, and visitors, $v \in I_V$, where $I_R \cup I_V = I$. Each resident r has her own space, called $s^r \in S$, and each resident has a highest priority at her own space.

When a resident intends to park at another location through the centralized system, we can assign her own space to another driver until she returns. As the resident parking holder should always be able to reclaim the resident space, it is important to identify how long each driver is willing to park.

I model this feature as a contractual term, and for simplicity, there are two duration terms, long-term and short-term.[3] The set of contractual terms is denoted by $T = \{t^+, t^-\}$.

With the aforementioned elements, there is a finite set of contracts $\mathbb{X} = I \times S \times T$. A contract $x = (i, s, t)$ specifies an agent $i = i(x)$, a space $s = s(x)$, and the term of the contract $t = t(x)$.

Given a set of contracts $X \subset \mathbb{X}$, let X_a be the set of contracts associated with agent $a \in I \cup S$. For example, $X_i = \{x \in X \mid i(x) = i\}$ is the subset of contracts in X that the driver i is associated with and $X_s = \{x \in X \mid s(x) = s\}$ is the subset of contracts in X that the space s is associated with.

Each driver has a unit demand, and has a strict preference over contracts in X_a and the

[3] Short term parking is not only theoretically interesting but also captures the real situation in a typical downtown. There are some short-term (usually less than 30 minutes) parking spaces, intended for high turnover in crowded districts.

null contract $\emptyset$. Let $\succ_i$ denote the strict preference relationship of the driver $i \in I$ over the set of contracts. A resident's option to be unmatched (occupying her own space) will be denoted by a contract (r, s^r, t^+).[4]

The preference relationship can be general, especially regarding the contractual terms, so that one can prefer short-term parking to long-term parking at the same space. In later sections, I will show how we can achieve some desirable property by restricting these preference relations.

Each space has a basic priority ordering π over the contracts in X_s. The priority ordering can be given by the distances as in the basic model, but it can be more general, reflecting the parking authority's objective. In this section, π is treated as exogenously given, and, in later sections, I will propose suggested priority structures for different objectives.

In this matching-with-contract model for parking, I define stability over the set of contracts. The stability concept is same as before, and it ensures individual rationality and a no-blocking-pair condition.

Definition 2. *Let $\mu(a)$ be a contract that an agent a is matched to in matching μ. A matching μ is **stable** if,*

i) for all i, $\mu(i) \succ_i \emptyset$

ii) there does not exist an individual-space pair (i, s), and a contract $x = (i, s, \cdot)$, where $x \succ_i \mu(i)$ and $x \succ_s \mu(s)$.

Next, I impose the following condition: a resident should be able to park at her own space whenever she returns. To implement this condition, I define protecting residents' property rights in the following way.

Definition 3. *Let $t(\mu(a))$ denote the term of the contract that agent a is matched to in matching μ. A matching μ **protects residents' rights** (or, simply, **protects residents**) if, for any resident r and her space s^r,*

$$t(\mu(r)) = t^- \implies t(\mu(s^r)) \neq t^+.$$

[4] It does not matter which contractual term is used to describe the resident's demand for her own space.

In words, the condition requires that, when a resident is assigned a short-term parking, her space should only be assigned to short-term parking as well (or remain unmatched). Note that if a resident r is matched to a long-term contract, her space s^r can be assigned to either a long-term or a short-term contract.

Protecting residents is a desirable property, and even a necessary one, to incentivize residents to participate in the centralized parking system. If the system doesn't protect their rights, residents will not participate, or participate as visitors, so their spaces will be wasted while the competition among drivers increases. I therefore require the system to satisfy this condition, and try to find a matching with other desirable properties.

2.1 IMPOSSIBILITY RESULT

To assign parking spaces to drivers, we need a centralized process, which is called a mechanism. A mechanism is a systematic procedure that selects a matching for a given preference profile. Given a mechanism ϕ, let $\phi(\succ)$ be the matching that mechanism ϕ selects when agents' (reported) preference profile is $\succ$. A mechanism ϕ is stable if $\phi(\succ)$ is stable for every preference profile $\succ$. A mechanism ϕ *protects residents' rights* if $\phi(\succ)$ protects residents' rights for every preference profile $\succ$.

The first observation in the parking problem with residents says that protecting residents is not compatible with stability.

Proposition 1. *There is no mechanism that is both stable and protects residents' rights.*

Proof. Suppose there are two individuals, a resident r and a visitor i. The resident has a space s^r, and there is a vacant space s^v. Let $x_0 = (r, s^r, t^+)$, $x_1 = (r, s^v, t^-)$, $y_0 = (i, s^r, t^+)$, and $y_1 = (i, s^r, t^-)$. The individuals and the spaces have the following preferences:

$$r : \{x_1\} \succ_r \{x_0\} \succ_r \emptyset$$
$$i : \{y_0\} \succ_i \{y_1\} \succ_i \emptyset$$
$$s^r : \{x_0\} \succ^{s^r} \{y_0\} \succ_{s^r} \{y_1\} \succ_{s^r} \emptyset$$
$$s^v : \{x_1\} \succ_{s^v} \{y_1\} \succ_{s^v} \emptyset$$

The only stable matching in this economy is $\{x_1, y_0\}$, which can be obtained by either a drivers-proposing or a spaces-proposing deferred acceptance algorithm. It assigns $\{x_1\}$ to r and $\{y_0\}$ to i. However, this does not protect r's property right because s^r is assigned a t^+ contract when r is assigned a t^- contract. To protect the resident's right, i should be assigned a contract $\{y_1\}$, but it is blocked by the (i, s^r) pair with the contract $\{y_0\}$.

□

We want to find a matching that has desirable properties, conditional on protecting residents' rights. As the above example shows, however, it is not possible to have a mechanism that always produces a stable matching that protects residents. The main reason for this incompatibility is because a resident's preference does not reflect her property right, nor does the space's priority structure reflect it. In the next section, I introduce a new idea, a claim contract, to reflect residents' rights. With the claim contract, I define choice functions of the individuals and the spaces, which are derived from the preference profiles. Once a mechanism chooses a matching under those choice functions, it will be evaluated in the preference domain.

3 CHOICE FUNCTION DESIGN

3.1 CHOICE FUNCTION AND CLAIM CONTRACT

Agent a's choice function, denoted by $Ch_a(X)$, is a systematic procedure that selects a set of contracts from a choice set X. For example, preference profile $\succ$ itself can be converted to a choice function by letting each choice function of agent a select the highest ranked contract under a's preference, i.e., $Ch_a(X) = \max_{\succ_a}[\{x \in X_a\} \cup \emptyset]$.

Before I construct choice functions of the agents in the parking problem, I introduce a new concept, a **claim contract**, which reflects residents' rights over spaces. When a resident may receive a short-term contract, this resident **claims** her right so that her space cannot accept a long-term contract. Formally, a claim contract $c_r = (r, s^r, \cdot) \in \mathbb{X}$, when it is in the space s^r's choice set, complements a t^- contract while ruling out t^+ contracts.[5]

[5]The claim contract should be distinguished from the contract (r, s^r, t^+), which indicates that resident r

Here is an example of resident space s^r's choice function with a claim contract:

Example 2. *Let x be a contract with the term t^+, and let y be a contract with the term t^-. The choice function of a resident space s^r is given by the following.*

$$Ch_{s^r}(\{x\}) = \{x\}$$
$$Ch_{s^r}(\{y\}) = \{y\}$$
$$Ch_{s^r}(\{c_r\}) = \{c_r\}$$
$$Ch_{s^r}(\{x, y\}) = \{x\}$$
$$Ch_{s^r}(\{x, c_r\}) = \{c_r\}$$
$$Ch_{s^r}(\{c_r, y\}) = \{c_r, y\}$$
$$Ch_{s^r}(\{x, y, c_r\}) = \{c_r, y\}$$

In this example, contract x has a higher priority in space s^r. However, when the claim contract c_r is in space s^r's choice set, the choice rule of s^r gives higher priority (or exclusive right) to contract y. Therefore, c_r complements t^- contracts at space s^r.

3.2 CHOICE FUNCTIONS

Now I formally define choice functions with the claim contract for all types of agents.

The choice functions of visitors $v \in I_V$ and public spaces $s^v \in S^V$ are straightforward. Each of the choice functions chooses the best contract in its preference lists and it can only choose one contract.

Definition 4. *For the given set of contract X, the choice function of the visitor v and of the public space s^v are defined as:*

$$Ch_v(X) = \max_{\succ_v}[\{x \mid x \in X_v\} \cup \emptyset]$$

is matched to her own space.

$$Ch_{s^v}(X) = \max_{\succ_{s^v}}[\{x \mid x \in X_{s^v}\}].$$

The choice function of residents $r \in I_R$ is similar, except it can possibly choose up to two contracts, one regular contract combined with a claim contract c_r. Inclusion of the claim, however, does not affect choice over other contracts. Rather, this helps us to define the stability.

Definition 5. *Given a set of contracts X and a base preference ordering from X, a resident r's choice $Ch_r(X)$ is obtained as follows:*

Phase 0: Remove all the contracts for another individual i, add them to the rejected set $R_r(X)$, and proceed with phase 1.

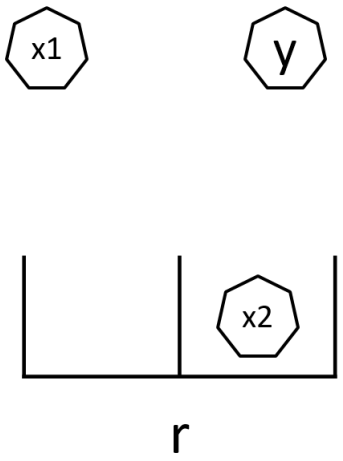
Phase 1: If there is no claim contract (for resident r) in X, then choose the most preferred contract in X and terminate the procedure. Otherwise, proceed with phase 2.

Phase 2: When there is a claim contract $c_r = (r, s^r, \cdot)$ in X, choose the claim and the most preferred contract in X together. If there is no contract other than the claim, choose only the claim contract. Terminate the procedure.

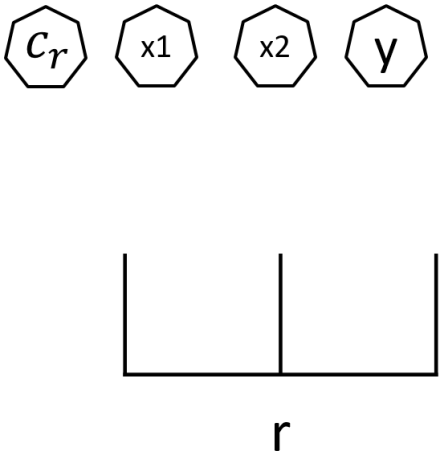
Choice of the claim contract in resident's choice function is due to a technical reason, to ensure the stability well defined. It does not change the preference of the resident, but only includes the claim contract whenever it is available in the choice set. The following is an illustration of a resident's choice function.

r

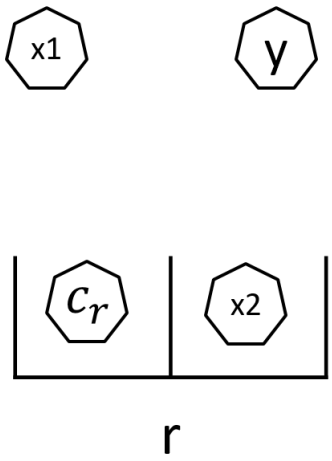
In resident r's choice set, there are two t^+ contracts, $x1$ and $x2$, and one t^- contract, y, in resident r's choice set.



The choice function chooses the top ranked contract among the available ones, which, in this example, is $x2$.



If there is a claim contract,



Then the claim contract is chosen along with the one that was chosen before.

The choice function of a resident space s^r is not trivial, since the choice rule depends on the presence of the claim, and there is a complementarity. When there is a claim, it gives higher priority to the contract with a t^- term. Formally, the choice function of a residential space is the following:

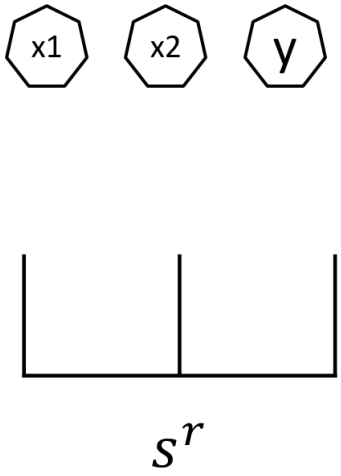
Definition 6. *Given a set of contracts X and a base priority ordering, a resident space s^r's choice $Ch_{s^r}(X)$ is obtained as follows:*

Phase 0: Remove all the contracts for another space s', add them to the rejected set $R_{s^r}(X)$ and proceed with phase 1.

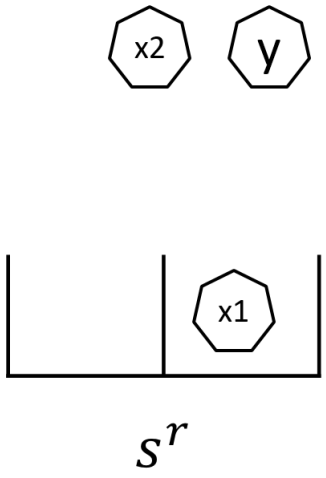
Phase 1: If there is no claim contract in X (i.e., regarding the space s^r), then choose the top priority contract and terminate the procedure. Otherwise, proceed with phase 2.

Phase 2: When there is a claim contract $c_r = (r, s^r, \cdot)$, choose the claim and the top priority contract among the contracts with the term t^-. If there is no contract with the term t^-, choose only the claim contract. Terminate the procedure.

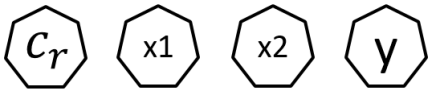
An illustration of the choice function of a resident space s^r is as follows:

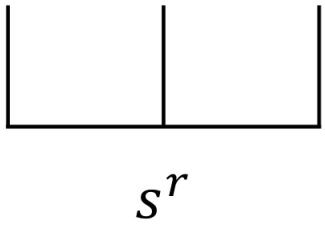


This is a choice function of a resident space, s^r, with two long-term contracts and one short-term contract.

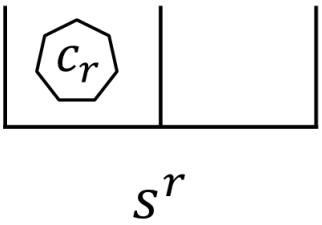


When there is no claim contract, it chooses the highest priority contract, which is $x1$ in this example.

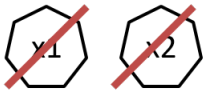


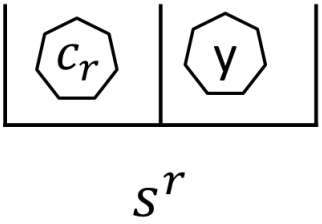


When there is a claim contract in the choice set,



The claim contract is chosen first, and the two long-term contracts ($x1$ and $x2$) are rejected because of the claim contract.





As a result, the t^- type contract is chosen after the claim contract.

The choice function of the resident space is constructed to protect the resident's right. The claim contract substitutes out the long-term contract while complementing the short-term contract. A suggested centralized system will use these claim contracts along the allocation mechanism and make sure to protect residents' rights. Note that the choice function of a resident space does not satisfy the substitute condition, as the following example illustrates.[6]

Example 3. *Let $x = (i, s^r, t^+), y = (i', s^r, t^-)$, and $c_r = (r, s^r, \cdot)$, and suppose x has higher priority than y in space s^r without property rights. Then the space's choice function will choose the subset that maximizes the following preference:*

$$\{c_r, y\} \succ_{s_r} \{c_r\} \succ_r \{x\} \succ_{s_r} \{y\} \succ_{s_r} \emptyset$$

Especially,

$$Ch_{s^r}(\{x, y\}) = \{x\}$$

$$Ch_{s^r}(\{x, y, c_r\}) = \{y, c_r\},$$

violating the substitute condition.

When the substitute condition is violated, there may not exist a stable outcome (Hatfield and Milgrom, 2005). Therefore, the parking model with residents also can fail to produce a stable matching. In the following sections, I define stability with respect to the choice functions defined above, which reflects a stability notion while protecting residents.

3.3 Stability with Choice Functions

When choice functions are determined, stability can be redefined with respect to them.

Definition 7. *A set of contracts X is stable (w.r.t. choice function Ch_a) if,*

i) for all $a \in I \cup S$, $Ch_a(X) = X_a$ and

[6] It does not satisfy other known conditions in the literature either; see the Appendix.

ii) there does not exist a set of contracts Y such that $Y \cap X = \emptyset$, for every a, $Y_a \subseteq Ch_a(Y \cup X)$.

I intentionally use a set of contracts X to define the stability w.r.t. a choice function, to distinguish it from the stability w.r.t. preference, where a matching μ was used.

In the previous section, we observed incompatibility of stability and protecting residents when we use drivers' preferences and spaces' priorities. To overcome this issue, I will suggest a mechanism which works with choice functions, not with preferences and priorities.

4 CENTRALIZED MECHANISM

4.1 BENCHMARK

One way to protect residents' rights is to restrict resident spaces only to t^- contracts. This can be done by running a cumulative-offer algorithm after putting claim contracts in each corresponding resident space's choice set at the beginning of the algorithm. Then, by construction, the resulting allocation will protect residents, as well as being stable w.r.t. the choice functions.

Definition 8 (Benchmark algorithm)**.** [7]

- Step 0: Put each claim contract into its corresponding resident space's choice set. For every s^r, let $A_{s^r}(0) = \{c_r\}$. For other (visitors') spaces s, let $A_s(0) = \emptyset$.
- Step 1: One (randomly chosen) driver offers her most preferred contract $x_1 = (i(1), s(1), t)$. Let $A_{s(1)}(1) = A_{s(1)}(0) \cup \{x_1\}$ and $A_s(1) = A_s(0)$ for all $s \neq s(1)$. The space that is offered the contract, $s(1)$, holds the contract if $x_1 \in Ch_{s(1)}(A_{s(1)}(1))$ and rejects it otherwise.

In general

- Step k: One of the agents without a contract held by any space offers her most preferred contract among the ones that have not been previously rejected, $x_k =$

[7] This and the next algorithms are based on Hatfield and Kojima (2010) and Sönmez (2013).

$(i(k), s(k), t)$. Let $A_{s(k)}(k) = A_{s(k)}(k-1) \cup \{x_k\}$ and $A_s(k) = A_s(k-1)$ for all $s \neq s(k)$. Space $s(k)$ holds the contract if $x_k \in Ch_{s(k)}(A_{s(k)}(k))$ and rejects it otherwise.

The algorithm terminates if every individual has a contract held by a space or no remaining acceptable contract. By construction of the algorithm, the Benchmark protects residents' rights, and it produces a stable outcome. Moreover, the mechanism is strategy proof for the drivers.

Theorem 2. *Benchmark is stable w.r.t. the choice functions, protects residents' rights, and is strategy proof for the drivers.*

Proof. The proof comes from the fact that, once the claim contract is in the choice set of the resident space, contracts are substitutes for all the agents.

□

Many of the resident spaces may be wasted, if the demand for short-term parking is small. Therefore, this benchmark mechanism, even though it is technically stable, is worse than the one that is suggested in the next sub-section.

Definition 9. *Suppose X is a set of contract and a resident r is assigned a space other than s^r, i.e., $X_r \neq (r, s^r, \cdot)$. Under X, the space s^r is **wasted** if there exists an agent i, either another resident or a visitor, such that $y \succ_i X_i$ where $y = \{i, s^r, \cdot\}$.*

In words, a resident space is wasted when it is not matched to any driver and there is some driver who prefers a contract at the resident space to the contract that she is assigned. When a resident is assigned a t^- contract and a visitor wants a t^+ contract in that resident space, it is inevitable that the space will be wasted to protect property rights. However, if a resident space is wasted even when the resident is assigned a t^+ contract, we could fix it by modifying the rule of the mechanism.

4.2 CUMULATIVE OFFER WITH CLAIM CONTRACT

I suggest a centralized system for a parking-space-assignment problem with contracts. The following cumulative-offer algorithm is a generalized version of the drivers proposing de-

ferred acceptance algorithm.

Definition 10 (Cumulative offer with claim contract (CC))**.**

- Step 1: One (randomly chosen) driver offers her most preferred contract $x_1 = (i(1), s(1), t)$. Let $A_{s(1)}(1) = \{x_1\}$ and $A_s(1) = \emptyset$ for all $s \neq s(1)$. The space that is offered the contract, $s(1)$, holds the contract if $x_1 \in Ch_{s(1)}(A_{s(1)}(1))$ and rejects it otherwise. If $i(1)$ is a resident and $t = t^s$, go to step 1′. Otherwise, proceed with step 2.

- Step 1′ : Let $i(1) = r_1$. After r_1 offers a contract x_1, she also sends her claim contract to her resident space, i.e., $c_r = (r_1, s^{r_1}, \cdot)$ is sent to the space s^{r_1}. Then, space s^{r_1} holds the claim contract, letting $A_{s^{r_1}(1)} = \{c_r\}$. Proceed with step 2.

In general

- Step k: One of the drivers without a contract held by any space offers her most preferred contract among the ones that have not been previously rejected, $x_k = (i(k), s(k), p, t)$. Let $A_{s(k)}(k) = A_{s(k)}(k-1) \cup \{x_k\}$ and $A_s(k) = A_s(k-1)$ for all $s \neq s(k)$. Space $s(k)$ holds the contract if $x_k \in Ch_{s(k)}(A_{s(k)}(k))$ and rejects it otherwise. If $i(k)$ is a resident and $t = t^s$, go to step k′. Otherwise, proceed with step k+1.

- Step k′ : Let $i(k) = r_k$. After r_k offers a contract x_k, she also sends her claim contract to her resident space, i.e., $c_{r_k} = (r_k, s^{r_k}, \cdot)$ is sent to space s^{r_k}. Let $A_{s^{r_k}}(k) = A_{s^{r_k}}(k-1) \cup \{c_{r_k}\}$. The space s^{r_k} rerun its choice function.[8] If there is a driver who has two contracts held by spaces, she keeps the preferred one and remove the other from $\cup_s A_s(k)$.[9] Proceed with step k+1.

This algorithm terminates when every driver is matched to a contract or every unmatched driver has no remaining acceptable contracts. Since there is a finite number of contracts, the algorithm terminates in some finite number of steps, K. All contracts held at step K are finalized, resulting in allocation $\cup_{s \in S} C_s(A_K)$. I define two properties of the set of contracts that will be used to describe the final assignment.

[8] As a result, the claim contract will be chosen, and the contract held by s^{r_k} may be rejected in this step, if it is a long-term contract.

[9] The removed one is not “rejected" yet, so it can be used for another offer later.

Definition 11 (Practical stability)**.** *A set of contracts X is **practically stable** (w.r.t. Ch_a) if,*

i) for all $a \in I \cup S$, $Ch_a(X) = X_a$ and

ii) if there exists a set of blocking contracts Y such that $Y \cap X = \emptyset$, $x \in Y$ implies $x = c_r$ for some r.

If a set of contracts X is practically stable, then the only possible blocking contracts are claim contracts, which cannot affect the resident's physical assignment, and which will not appear in the cumulative offer process.

Definition 12. *A mechanism is **non-wasteful** if it always produces a allocation without any wasted space. A mechanism is **wasteful** if it is not non-wasteful. A mechanism ϕ is **less wasteful** than a mechanism ψ if, whenever ϕ produces a wasteful allocation with a given problem, ψ also produces a wasteful allocation. A mechanism ϕ is **strictly less wasteful** than ψ if ϕ is less wasteful than ψ but not vice versa.*

Theorem 3. *cumulative offer with claim contract protects residents' rights, is practically stable w.r.t. the choice function, is strategy proof, and is strictly less wasteful than the Benchmark.*

Proof. It is trivial to see that the CC mechanism is less wasteful than the Benchmark mechanism. Whenever there is a wasted space in CC, there is a (r, s^r) pair such that r gets a short-term contract and s^r gets no contract, and there is at least one driver i who demanded s^r as a long-term contract. Since no long-term contract is acceptable at s^r under the Benchmark mechanism, any wasted space under CC is also wasted under the Benchmark.

Also, by construction of the mechanism, CC protects residents' rights. Once a resident r offers a short-term contract, a claim contract stays in s^r's choice set until the end of the algorithm, preventing s^r from being assigned a long-term contract. Indeed, CC protects residents too much to allow wasted spaces.

To see that CC is practically stable, let X be the final assignment of CC and suppose there is a contract $y \notin X$ such that $y \neq c_r$ for some r, and $Ch_i(X \cup \{y\}) = \{y\}$ and $Ch_s(X \cup \{y\}) = \{y\}$ for some individual i and space s. If s is a visitor's space, then i would

have offered contract y before $Ch_i(X_i)$, and it would have been in space s's accumulated set $A_s(K)$ at the end of the procedure. Then, by the definition of the visitor's choice function, $Ch_s(A_K) = X_s$ is the most preferred contract for the space among the contracts in A_K. However, $y \notin X$ and $Ch_s(X \cup \{y\}) = \{y\}$ implies that, y is most preferred contract in $X \cup \{y\}$, a contradiction.

Now suppose s is a resident r's space. If $c_r \notin A_s(K)$, then as the same argument, y should be the most preferred contract in X, leading to a contradiction. In words, if there is no claim contract in s's accumulated set, y should have been chosen by the space s. Therefore, for the contract y not to be chosen from the space s, there should be a claim contract in $A_s(K)$. Since short-term contracts are always acceptable in s^r, if there were a short-term contract demand at s^r, and if it was most preferred one, it should have been accepted. This observation implies y is not a short-term contract. Then, by the definition of the resident space's choice function, c_r is always chosen by the space. A claim contract is always chosen, so $c_r \in X_s$, implying no t^+ contract is acceptable in r. This contradicts the fact that $Ch_s(X \cup \{y\}) = \{y\}$, as y is a long-term contract. Therefore, there is no possible blocking contract other than claim contracts, leading to a practical stability.

Lastly, the CC mechanism is strategy-proof the drivers. The choice function of each visitor space satisfies the substitute condition, so we only need to consider manipulation of a driver whose long-term contract is rejected by a resident space.[10] Suppose a driver i prefers a contract $x = (i, s^r, t^+)$ to her assignment under true preference. In the algorithm, she offers x to a resident space s^r. Assume $c_r \in A_{s^r}(K)$. The contract $x = (i, r^s, t^+)$ is rejected because of the claim contract, implying the resident r has offered a short-term contract in the mechanism process. The only possible way for i to get contract x is to have r not offer the short-term contract. To do that, r should get a long-term contract before she offers a short-term. Therefore, suppose further that r offered a long-term contract $z = (r, s, t^+)$, but it was rejected by i's contract, $y = (i, s, t^+)$.[11] Note that i prefers y to x since it was offered earlier.

To see if the driver i can manipulate her preference to get y instead of x, suppose she

[10] If a short-term contract is rejected, then it is rejected by another short-term contract, so that there is no way to manipulate.

[11] The term does not need to be a t^+.

reports y is not acceptable for her. Then, when r offers z to space s, it is not yet rejected. However, the fact that y is also rejected in the true preferences implies there is another contract that is more preferred than y in space s, which in turn will be preferred to z. Therefore, z will be rejected again, so that the resident will offer a short-term contract. As a result, r will send c_r to r^s, which will reject x. Therefore, the driver i's manipulation does not give him a better contract. □

The cumulative offer with claim contract mechanism is not first best since it is still wasteful. However, it protects residents' rights so that it does not prevent residents from participating in the centralized system. Also, it utilizes resident spaces, which will be wasted anyway.[12]

4.3 PRACTICAL STABILITY IN PREFERENCE DOMAIN

Claim contract, choice functions, and the concept of practical stability were arbitrarily made up to ensure protecting property rights, and therefore it is not clear what practical stability means in preference domain. Clearly, practically stable outcome is not stable, so there can be a set of blocking contracts in resident spaces. However, it totally rules out wastes from the visiting spaces. Also, if the resident is matched to a short-term parking, blocking contract will violate property rights. So these wastes are inevitable costs for protecting residents' rights. One particular type of wastes that we want to avoid, however, still remains: when a resident (who at least once demanded a short-term parking) is matched to a long-term parking, there could be a wasted space that does not violate property rights.

This observation gives a positive result by restricting drivers' (especially residents') preferences. Stability and protecting residents are not compatible, so any resident-protecting mechanism protects residents at the cost of wasted spaces. This waste occurs when the resident demands a short-term contract during the algorithm but gets a long-term contract in the final allocation. In some circumstances where we can model the residents' preferences as lexicographic, such that they all prefer long-term contracts to short-term contracts, we can achieve the following positive result.

[12] It is not known yet whether CC is the least wasteful mechanism among all strategy-proof and resident-protecting mechanisms.

Theorem 4. *Suppose every resident prefers every long-term contract to any short-term contract, i.e., for all r and contracts x and y, $t(x) = t^+$ and $t(y) = t^- \Rightarrow x \succ_r y$. Then, the cumulative offer with claim contract algorithm is non-wasteful.*

Proof. Let the final assignment be the set of contracts X and suppose a resident space s^r is wasted under X. That means that resident r is assigned a long-term contract, but s^r is not assigned any contract. Let i be a driver who prefers a contract $y = (i, s^r, t^+)$ to her assignment X_i.

If $c_r \notin X_{s^r}$, then the contract y forms a blocking contract, violating the practical stability of X. But X satisfies practical stability by theorem 3.

If $c_r \in X_{s^r}$, then resident r has offered a short-term contract at some point in the algorithm. Since she prefers all long-term contracts to short-term contracts, all of her long-term contracts should have been rejected at that point. Therefore, r should have offered only short-term contracts from then on. This violates the assumption that r is assigned a long-term contract.

□

5 Market Design: A Practical Implementation Proposal

This section investigates how the parking model can be implemented in practice. I first include price in the contract, so that drivers can bid up the prices. Then I consider how to prioritize the drivers with distances and prices. Lastly, I suggest a simple way of collecting preferences information from the drivers.

5.1 Price

The first question that might arise from the previous sections is the role of prices in the parking space assignment process. I implicitly modeled a fixed price, mainly because I wanted to focus on matching with property rights, but also because it may not be politically acceptable to dramatically increase parking prices.[13] Even if it is not possible to change

[13] Shoup (2005) points out that public parking spaces are generally treated as free goods and that the low cost of parking is one of the main causes of urban traffic problems.

prices, however, it is worth modeling the problem with prices to see if the market works better than without the prices.

When we include prices in the model, they will be treated as one of the components of contracts. Let $P = (p_0, p_1, \cdots, p_{max})$ be the finite set of discrete prices. Each $p \in P$ corresponds the price per given time duration, for example, 30 minutes. p_0 is the price a driver can expect to pay to park in an area with unsaturated parking spaces. We can set p_0 to zero or to the current price.

Now a contract has a price as an additional component, i.e., $x = (i, s, p, t) \in \chi = I \times S \times P \times T$ and $p(x) = p$. Preferences of the drivers and priorities (choice functions) of spaces have been defined over the set of contracts, so including price in the contract will work well with those structures. The only assumption imposed on drivers' preferences is that they prefer a cheaper contract to a more expensive one, other things being equal; for all i and s, $(i, s, p, t) \succ_i (i, s, p', t)$ if $p < p'$ with the same contractual term t. Note that, it is possible that a driver prefers a expensive long-term contract to a cheap short-term contract. With well-defined preferences and priorities over the contracts, we can apply the cumulative offer with claim contract mechanism discussed in the earlier section.

For the resident spaces, I still require the protection of residents' property rights, even with high prices. For any resident space s^r, any long-term contract will be rejected when there is a claim contract c_r in s^r's choice set no matter the price of the contract . Therefore, we will use the same choice functions as in the previous section. Then the following result is obvious because the choice functions remain the same for contracts with prices.

Definition 13. *In the parking space assignment problem, price **interferes** with claim contract c_r, if, for some $p \in P$ and a set of contracts X, there exists a long-term contract $x = (i, s^r, p, t^+)$ such that $x \in Ch_{s^r}(\{x\} \cup \{c_r\} \cup X)$.*

In words, price interferes with a claim contract if a resident space accepts a long-term contract even when there is a claim contract in the choice set.

Corollary 1. *If price does not interfere with any claim contract, then cumulative offer with claim contract with set of contracts χ protects residents' rights, is practically stable w.r.t. the choice function, is strategy proof, and is less wasteful than the Benchmark.*

Since resident street parking is generally a city-assigned right, we do not allow residents to “sell" their spaces and collect money in return. Instead, the parking fee is collected by the central parking authority and the system protects the property rights of the residents automatically.

When price is included in the contract, it is not clear what is a best way to give priority based on distances and prices. Moreover, preferences are in general complex and not easily expressible in full. The next two sub-sections propose a method to design priorities for urban planners and a language to easily express preferences for drivers.

5.2 DESIGNING PRIORITIES

The second question is how to prioritize drivers in the system, especially with prices. Distance priority can be used without prices because we can reduce the negative externality of cruising by minimizing the total distance traveled. With prices, however, it depends on the parking authority's objective. If the parking authority aims for maximum profit, then it might not be a bad idea to give priorities based upon prices.

PRICE-ONLY PRIORITY

Prices can be treated as priorities to maximize revenue from the parking allocation. Formally, for every space s, the price-only priority ordering π_s^p satisfies $x \ \pi_s^p \ y$ if $p(x) > p(y)$. If price solely determines priority and distance is merely used to break ties, the mechanism corresponds to an auction system. It resembles a second-price auction, but because of discreteness, the winner may pay one incremental price higher than the second bidder's price.

If we model the parking problem with only public parking spaces and visitors, then deferred acceptance can be used to find the proposing-side-optimal stable allocation. In that case, the system can maximize revenue within a stability constraint.

Proposition 2. *Suppose there are only public parking spaces and visiting drivers. The spaces proposing deferred acceptance algorithm with price-only priority π^p maximizes revenue among all stable allocations.*

Proof. It is enough to note that the spaces' preferences are based on the prices, and the spaces proposing version of deferred acceptance yields an optimal stable matching for the spaces. □

DISTANCE-ONLY PRIORITY

When prices are fixed, or it's not feasible to increase prices, we may think of giving a closer driver higher priority. This priority reflects the nature of the problem, as priority is given by distances in the decentralized cruising game. Even when prices can be adjusted, the parking authority may want to keep the distance-only priority if their objective is to minimize the total distance traveled by the drivers.

Again, with only public spaces and visitors, we can achieve an optimal stable allocation, in this case to minimize total distance traveled.

Proposition 3. *Suppose there are only public parking spaces and visiting drivers. The spaces proposing deferred acceptance algorithm with distance-only priority π^d minimizes total distance traveled among all stable allocations.*

Both the π^p-based and the π^d-based spaces proposing deferred acceptance is not strategy proof for drivers. If manipulation on the drivers' end is not a concern, we can use these versions according to the objective of the parking authority. If manipulation creates a significant problem in the algorithm, we could use the drivers-proposing versions of DA instead.

MIXED-PRIORITY

I suggest a mixed-priority structure where the parking authority can decide how much to weigh price and distance based on the relative importance of the two. This priority can be used if both price and distance traveled are important to the parking authority. The distance function $d(i, s)$ gives the distance between the driver i and the space s. Let k_s be the rate of substitution between price and distance at space s. For example, if $k_s = 500m/\$$, a \$1 higher price is equivalent to the 500m less distance traveled, so assigning the space to a car 100m apart at \$1 is equally preferred to assigning the space to a car 600m apart at \$2.

With each space's rate of substitution k_s, we can define the base priority π_s^m as follows. For contracts $x = (i, s, p, t)$ and $y = (i', s, p', t)$,

$$x \; \pi_s^m \; y \quad \text{iff} \quad p - \frac{d_{is}}{k_s} > p - \frac{d_{is}}{k_s}.$$

We can think of the $\frac{d_{is}}{k_s}$ term as the *distance-price* of driver i at space s. The rates of substitution could be the same across all spaces, or they could have different ones depending on the practical parameters such as average traffic, demand of parking, supply of spaces, etc.

These various priority structures are possible suggestions of practical application of the model, and there is yet no qualitative analysis on the consequences of each priority.

5.3 DESIGNING PREFERENCES EXPRESSION LANGUAGE

Another component we need to consider in our application is how drivers can express preferences. I first suggest a simple way of collecting drivers' preferences.

On the drivers' side, it is not safe for them to submit their full list of preferences while driving. On top of safety concerns, they may not be fully aware of available parking spaces nearby. Therefore, we need to ask minimal information from the drivers to construct preferences for them. I suggest collecting the following GDP information as one way of submitting their preferences.

Definition 14. *GDP information system*

G oal: final destination of the driver

D istance: additional distance the driver is willing to walk for unit price decrease

P rice: maximum price that the driver is willing to pay to park at the destination

The *distance* asks the driver's marginal rate of substitution between walking more and paying more. Assuming the marginal rate of substitution is constant, we can construct a full list of preferences for a given driver.

Definition 15. *Given the GDP information of a driver i, who submits (G_i, D_i, P_i) to the system, the preference of i is constructed as the following:*

$$(i, s, p) \succ_i (i, s', p')$$

$$\Leftrightarrow$$

$$p + \frac{d(s, G_i)}{D_i} < p' + \frac{d(s', G_i)}{D_i}$$

where $d(s, G)$ is the walking distance between space s and destination G.

Since the system has the information for all of the available spaces, which the drivers may not be aware of, it can construct the preference profile for each driver given his or her GDP information. With the GDP information system, a driver prefers a space with a lower price, as measured by the direct parking price and the indirect walking cost.

By no means do I argue this is the way that the system should be implemented. Instead, I present GDP as an example to show that preferences can be constructed with relatively limited information. With technology nowadays, GDP or any other form of information can be accumulated to suggest a better method of preference submission, such as preset preferences, a non-constant rate of substitution, etc.

5.4 DYNAMIC MODEL

We considered a static model, assuming drivers and available parking spaces are fixed. However, the cruising game itself is a dynamic game, where drivers enter and exit, and the available spaces also change. Therefore, when we introduce a centralized system in this market, we need to consider its dynamic properties as well. One natural question is whether the drivers submit their preferences to the system when they start searching for parking spaces or if they wait on the street for the next round. This is important from a practical point of view, as the wait is costly for the drivers and the system. If they find it profitable to wait, then the system will fail to reduce the negative side effect of cruising behavior.

The dynamic model of the parking space assignment problem consists of multiple pe-

riods. The static cumulative algorithm will still be used to match the contracts at each period, and I will focus on dynamic consequences of the repeated algorithm. A number of new drivers and a number of new parking spaces arrive at each period $\tau = \{1, \cdots, \mathcal{T}\}$. Then the set of drivers, the set of spaces, the preferences, and the priority order are updated accordingly.

A **dynamic parking space assignment problem with contracts** at each period τ is a list $(I^\tau, S, P, T, \succ_{I^\tau}, \pi^\tau)$ with:

1. a set of drivers at each period I^τ,
2. a set of parking spaces $S = \{s_1, \cdots, s_m\}$,
3. a set of discrete prices $P = \{p_0, \cdots, p_{max}\}$,
4. a set of terms $T = \{t^-, t^+\}$,
5. a list of drivers' strict preferences over contracts $\succ_{I^\tau}$ and
6. a list of base priority rankings $\pi^\tau = (\pi^\tau_{s_1}, \cdots \pi^\tau_{s_m})$.

The centralized system assigns available parking spaces to drivers who submit their preferences at each period using the cumulative offer with claim contract, and updates the available spaces for the next period $\tau + 1$. A driver who demands a parking space can choose any period to submit his preferences. However, we don't want the driver wait too long before participating in the system. To capture the waiting decision of the drivers, I define **postpone** in the following way.

Definition 16. *A driver **postpones** participating in the system if he submits his preferences after period τ, when there are acceptable parking spaces for him at period τ.*

As I require drivers to submit their preferences when they start searching for spaces, I define the following property.

Definition 17. *A parking space assignment mechanism is **delay proof** if it is a dominant strategy for each driver not to postpone participation.*

We don't want drivers to postpone their participation in the system because they will occupy space on the road while they are waiting, increasing negative side-effect of cruising, not because they could get a better matching by postponing. In the following subsections, I present a dynamic extension of the static model.

REPEATING MECHANISM

One method of extending the static mechanism to a dynamic version is to simply repeat the static cumulative algorithm every period. The system will finalize the assignment in each period, so that if a driver who is assigned a contract wants to participate again in the next period, then they should surrender their contracts and pay the price for it.

Formally, the set of drivers matched at period τ, $C_I^\tau(X)$, leaves the system, and a set of drivers comes into the system at period $\tau + 1$, denoted by $\{i_1^{\tau+1}, i_2^{\tau+1} \cdots\}$. A driver who was matched at period τ but decided to enter the system again will be treated as a new driver at period $\tau + 1$. Also, the priority rule does not change across periods. Therefore, the dynamic problem is a repeat of the static one with new drivers and new spaces.

Remark 1. *If the system uses distance-based priority, then the repeating mechanism is not delay proof.*

If drivers accept the contract that is matched to them, they may lose an opportunity to be matched to a better one in future rounds. This concern may encourage postponing, which will increase cruising behavior, particularly if priorities are given by distance only.

PRIORITY-UPDATING DYNAMIC MECHANISM

In a sophisticated dynamic mechanism, the system will run the static cumulative algorithm every period, with updated priority orderings based on the assignments in the previous period. Specifically, a contract x that is assigned at a space $s(x)$ in round τ will get highest priority at space $s(x)$ in period $\tau + 1$. If driver $i(x)$ decides to participate again in period $\tau + 1$, therefore, he will be guaranteed a contract at least as good as contract x.

Formally, let $A_I^\tau \subset C_I^\tau(X)$ be the set of drivers who are matched at period τ but participate in the system again in the next period. The priority-updating dynamic problem

replaces the basic dynamic model as follows.

1′. a set of drivers at each period $I^\tau = A_I^{\tau-1} \cup \{i_1^\tau, i_2^\tau, \cdots\}$,

5′. a list of drivers' strict preferences over contracts $\succ_{I^\tau} = \{\succ_1^\tau, \succ_2^\tau, \cdots\}$,

6′. a list of updated priority rankings $\pi^\tau = (\pi_{s_1}, \cdots \pi_{s_m})$.

The updated priority ranking gives highest priority to the driver who was matched in the previous period. The priority-updating dynamic mechanism runs the CC algorithm every period with the updated information. Thus, it is a weakly dominant strategy for drivers to participate immediately, as they cannot be assigned a worse outcome as period proceeds.

Remark 2. *The priority-updating dynamic mechanism for a parking space assignment problem is delay proof.*

6 CONCLUSION

In this paper, I introduced a matching model for a downtown parking space assignment problem. I tackled the problem in three ways: first, in an unregulated world as a decentralized game; second, motivated by the equilibria of the decentralized game, as a centralized stable matching problem; third, by introducing resident street parking as a source open to visitors, I introduced a novel model of matching with priorities and property rights. This new model incorporates choice functions that do not satisfy any previously known conditions that guarantee the existence of a stable allocation. Despite this, a choice function stable matching always exists. I call this concept practical stability with respect to underlying priorities and preferences.

Finally, I proposed a practical implementation method combining not only priorities and property rights but also prices into the system. This implementation also considered methods to design priorities, express preferences, and run in real time.

This work is the very first step to solve downtown parking problems, and it is a partial equilibrium analysis. The price system will affect the drivers' modal choice of traveling, and ease of parking will affect their demand for driving. Cruising behavior has environmental and traffic externalities. To fully study and solve parking-related problems, we should

consider all these consequences in a more general model. This paper can be a building block for that line of future work. In addition to that, one interesting and important practical question, which is not addressed in this paper, is how to incentivize residents to offer their spaces when they demand parking from the centralized system.

With driverless cars are coming into our life, parking will become easier in the near future, and it will be reasonable to introduce a centralized parking assignment system into the market. My new approach to the parking problem can assign spaces in a better, and residents-protecting way.

APPENDIX

1.A PROPERTIES OF RESIDENT SPACE'S CHOICE FUNCTION

Chambers and Yenmez (2013) study another concept of the choice rules, path-independence, and show that a stable matching exists when all agents' choice rules are path-independent. Furthermore, Yenmez (2017) extends this finding to show the existence of a stable matching when one party has path-independent choice rules and the other party has path-independent modifications for all agents. In the parking problem with residents, however, the choice function of the resident space does have path-independent modifications.

Lemma 1. *The choice rules of the resident space (s^r) are not path-independent, and they do not have path-independent modifications.*

Proof. Suppose there is one resident space s^r, a resident r, and two visitors, v_1 and v_2. There is one t^+ contract $x_1 = (v_1, s^r, t^+)$ for v_1, one t^- contract $y_2 = (v_2, s^r, t^-)$ for v_2, and the claim contract $c_r = (r, s^r, t)$ for the resident r. For the drivers (resident and visitors), the choice rules are as follows: $C_r(\{c_r\}) = \{c_r\}$, $C_{v_1}(\{x_1\}) = \{x_1\}$, and $C_{v_2}(\{y_2\}) = \{y_2\}$. For any agent, the choice from a set of the contracts involving the agent is an empty set if it is not specified.

As for the space, according to the choice rule $Ch_{s^r}(X)$, the following contracts will be chosen by s^r.

$$Ch_{s^r}(\{x_1\}) = \{x_1\}$$

$$Ch_{s^r}(\{y_2\}) = \{y_2\}$$

$$Ch_{s^r}(\{c_r\}) = \{c_r\}$$

$$Ch_{s^r}(\{x_1, y_2\}) = \{x_1\}$$

$$Ch_{s^r}(\{x_1, c_r\}) = \{c_r\}$$

$$Ch_{s^r}(\{c_r, y_2\}) = \{c_r, y_2\}$$

$$Ch_{s^r}(\{x_1, y_2, c_r\}) = \{c_r, y_2\}.$$[14]

[14] We can construct the choice as the subset maximizing the space's preference given by $\{c_r, y_2\} \succ \{c_r\} \succ \{x_1\} \succ \{y_2\} \succ \emptyset$.

This choice rule is not path-independent, since $Ch_{s^r}(\{x_1, y_2, c_r\}) = \{c_r, y_2\} \neq \{c_r\} = Ch_{s^r}(c_r \cup Ch_{s^r}(\{x_1, y_2\}))$. However, there is no modification other than the trivial one, Ch_{s^r} itself, because for any subset $X' \subseteq \{x_1, y_2, c_r\}$, any possible choice $C'_{s^r}(X')$ is individually rational. Since Ch_{s^r} is not path-independent, therefore, there is no path-independent modification of Ch_{s^r}.

□

1.B REPEATED CUMULATIVE OFFER ALGORITHM

Definition 18 (Repeated cumulative offer (duration-match) algorithm:)**.** *The following steps describe the repeated cumulative offer algorithm.*

Let δ be the set of all claim contracts.

Step 0: Run the cumulative offer algorithm under choice functions Ch^{π}, but without claim contracts. That is, every agent is restricted to offering contracts in $X \setminus \delta$. Let $A_1 = \cup_s A_s(K)$ be the set of all contracts that are offered in the cumulative offer algorithm.

Step 1: After the cumulative offer algorithm terminates, any resident (r) with a short-term contract held by a space applies his claim contract to his own block. Call this set of claim contracts δ_1. Each block with any claim contract offered in this step reevaluates its choice function with the claim. Let $A'_1 = A_1 \cup \delta_1$ be the set of contracts that are offered before or at this step.

Step 2: After step 1, there could be agents whose multiple contracts are held by some spaces. Let $J \subset I$ be the set of such agents. Randomly pick one agent, $j_1 \in J$, and choose the best contract among the ones that are held, and remove the other contracts from the space's accumulated set $A_s(k)$, updating it to $A'_s(k) = A_s(k) \setminus z$. Remove j from the set J. The space s where j's contract was removed reevaluates it's choice function with updated accumulated set, and chooses $C_s(A'_s(k))$. If, by reevaluating, a new agent has multiple contracts held by spaces, include that agent in J. Repeat the process until there is no agent in J.

Step 3: Resume the cumulative offer algorithm with A'_1 held by the spaces. At this step, every agent is restricted to offer contracts in $X \setminus (A'_1 \cup \delta)$, and the set of contracts that every space will be given is restricted to the subsets of $(X \setminus \delta) \cup \delta_1$. In words, an agent can only offer a contract that was not offered before among non-claim contracts. As a result, a space will choose from a set that does not contain any new claim contract.

Theorem 5. *The repeated cumulative offer algorithm produces a practically stable allocation for the parking space assignment problem.*

Theorem 6. *The repeated cumulative offer algorithm protects residents' rights, is less wasteful than the cumulative offer with claim contract, but not strategy proof.*

1.C MANY-TO-ONE MODEL

In this section, I extend the model as a many-to-one matching, and I use blocks (with multiple spaces) as resources to be allocated. Each block has several parking spaces, possibly with different contract terms. The set of blocks is denoted by $B = (b_1, \cdots, b_m)$, and each block b has capacity q_b.

Figure 1.1: A block with different term spaces.

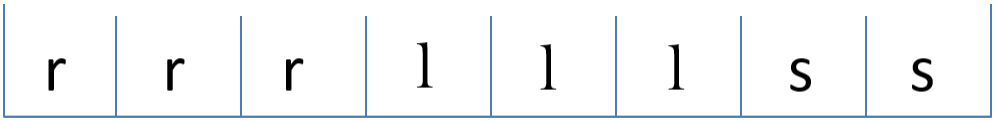


Figure 1.1 is an example of a typical block. It may have resident spaces, short-term only-spaces, and long-term spaces. Each block b has its own capacity q_b and has a basic priority ordering over contracts in X_s.

With blocks that may have multiple resident spaces, protecting residents' rights is defined as the following:

Definition 19. *A set of contracts $X \subseteq \mathbb{X}$ protects residents' rights if, for any block b and the set of residents $r(b) = \{r | s^r \in b\}$,*

$$|s \in T(X_b)| \geq |s \in T(X_{r(b)})|.$$

In words, a set of contracts protects residents' rights if the number of short-term contracts in a block b is greater than or equal to the number of short-term contracts that the residents in b are assigned. Then, when the residents return after their short-term parking, there will be enough space for them to park on their own block.

1.C.1 BLOCK'S CHOICE FUNCTION

A choice function of a block b, under the priority ordering π, is a systematic procedure to select a number of contracts given a set of demanding contracts.

In a block b, q_b^r and q_b^s spaces are reserved for residents and short-term parking, respectively. Residents have priority at these q_b^r spaces, and the number of resident parking spaces is equal to the number of resident permit holders. Only short-term parking contracts are acceptable at q_b^s spaces. The remaining spaces, $q_b^l = q_b - q_b^r - q_b^s$, are for long-term visitors' parking.

When we construct a block's choice function, we need to make sure that any resident who is assigned a space on another block can take back his own resident space when he returns. One way to achieve this property is to assign, as we did in one-to-one matching, only short-term contracts to resident spaces. The following choice function for a block b is constructed in this way.

Definition 20 (Block's choice function)**.** *Given a set of contracts X' and a base priority ordering π, a block b's choice $C_b^\pi(X')$ is obtained as follows:*

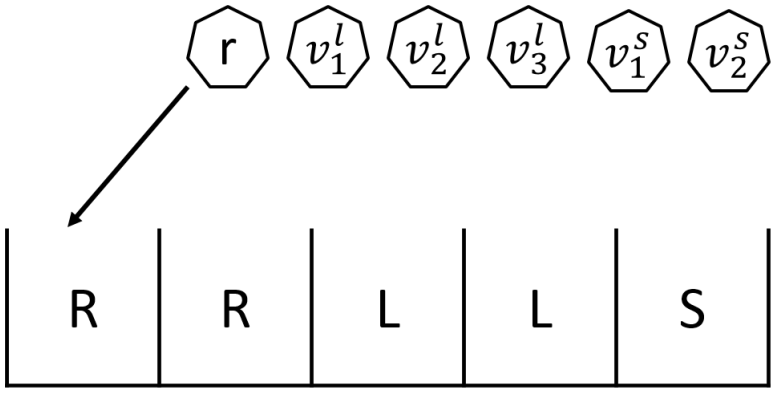
Phase 0: Remove all the contracts for another block b', add them to the rejected set $R_b^\pi(X')$, and proceed with phase 1.

Phase 1: For the first q_b^r potential elements of $C_b^\pi(X')$, choose any resident contract one at a time. If all contracts are considered in this phase, terminate the procedure; if q_b^r elements are chosen, set $q_b^{s'} = q_b^s$ and proceed with phase 2; if less than q_b^r elements are chosen, let $q_b^{s'} = q_b^s + (q_b^r - |(\cdot, \cdot, \cdot, r) \in C_b^\pi(X')|)$ and proceed with phase 2.

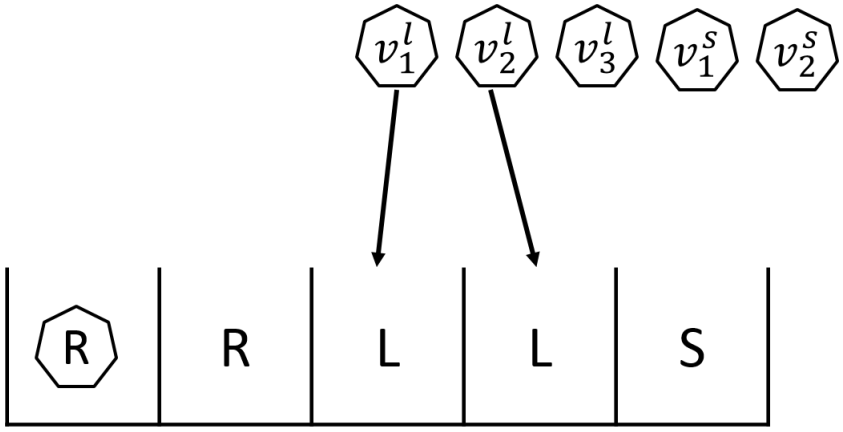
Phase 2: For the next q_b^l potential elements, choose the highest-ranked long-term contracts one at a time according to ranking π. Remove all surviving long-term contracts, adding them to $R_b^\pi(X')$. Also remove all short-term contracts involving any driver in $C_b^\pi(X')$. If there is no surviving contract, then terminate the procedure. Otherwise, let $q_b^{s''} = q_b^{s'} + (q_b^l - |(\cdot, \cdot, \cdot, l) \in C_b^\pi(X')|)$, and proceed with phase 3.

Phase 3: For the last $q_b^{s''}$ potential elements of $C_b^\pi(X')$, choose only the short-term contracts with highest π ranking one at a time, adding them to $C_b^\pi(X')$. Remove all remaining contracts and terminate the procedure. Note that it is possible that some of the $q_b^{s''}$ spaces remain unassigned.

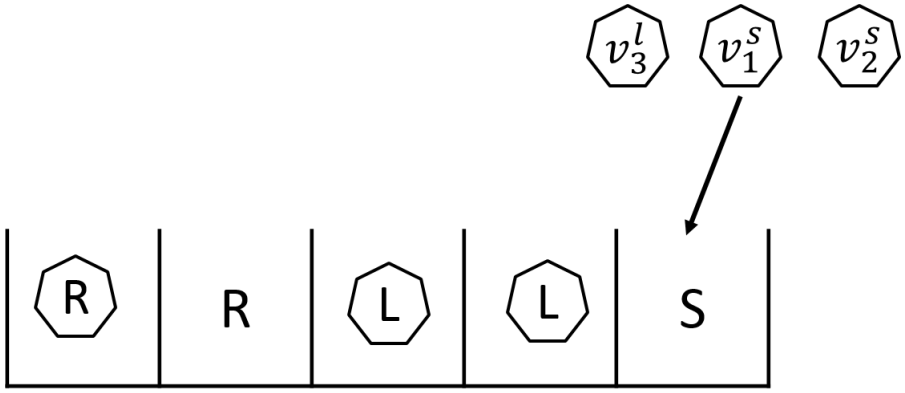
Example 4. *Here is an example of how the block's choice works.*



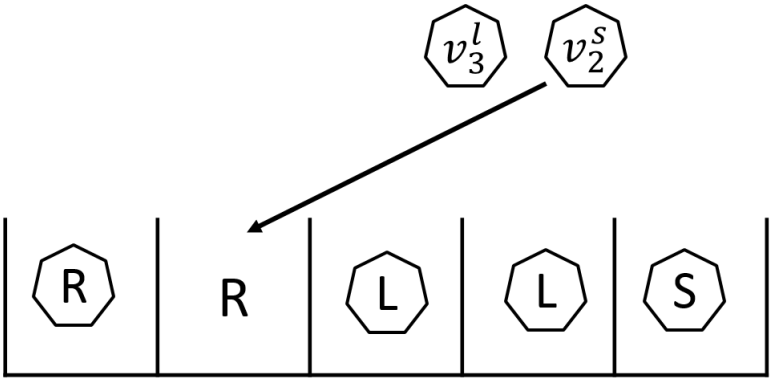
A block has all 3 types of spaces, R for residents, L for long-term visitors, and S for short-term visitors. There are six contracts in the block's choice set, one resident contract, three long-term visitor contracts, and two short-term visitor contracts. The block first chooses the resident contract, and assigns it to a R space.



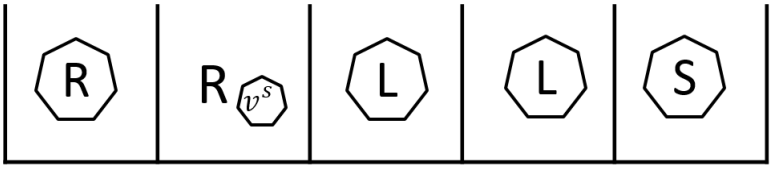
Next, long-term visitor contracts are chosen one by one up to the capacity of long-term spaces, according to the priority ordering.



Lastly, short-term visitor contracts are chosen.



Any vacant resident spaces are assigned to short-term visitor contracts according to the priority ordering.



The figure above shows the final result of the block's choice function. Note that, a short-term visitor contract is occupying one vacant resident space.

In this choice function, any vacant resident parking space is assigned only to short-term visitors. This will protect residents' rights. However, there will be an efficiency loss if a resident is allocated a long-term visitor parking and their space is not assigned a contract. I close this section by stating the choice function for the blocks.

Definition 21 (Choice function with claim contract)**.** *Given a set of contracts X' and a base priority ordering π, a block b's duration-match choice $D_b^\pi(X')$ is obtained as follows:*

Phase 0: Remove all the contracts for another block b', add them to the rejected set $R_b^\pi(X')$, and proceed with phase 1. Contracts that survive phase 0 involve only block b.

Phase 1.1: For the first q_b^r potential elements of $D_b^\pi(X')$, choose any resident contract one at a time. Remove any claim contract involving residents in $D_b^\pi(X')$. If all contracts are considered in this phase, terminate the procedure; otherwise, proceed with phase 1.2.

Phase 1.2: If all q_b^r spaces are filled, let $q_b^{l'} = q_b^l$ and proceed with phase 2; if there are remaining resident spaces, choose any claim contract one at a time. If resident spaces still

remain, add them to long-term spaces, i.e., $q_b^{l'} = q_b^l + (q_b^r - |(\cdot,\cdot,\cdot r^l) \in D_b^\pi(X')| - |(\cdot,\cdot,\cdot,r^s) \in D_b^\pi(X'))|$*, and proceed with phase 2.*

Phase 2: For the next $q_b^{l'}$ *potential elements, choose highest-ranked long-term visitor contracts one at a time according to ranking* π*. Remove all short-term visitor contracts involving any driver in* $D_b^\pi(X')$*, as well as any long-term visitor contracts. If there is no surviving contract, then terminate the procedure; otherwise, let* $q_b^{s'} = q_b^s + (q_b^l - |(\cdot,\cdot,\cdot,l) \in D_b^\pi(X')|) + |(\cdot,\cdot,\cdot,r^s) \in D_b^\pi(X')|$ *and proceed with phase 3.*

Phase 3: For the next $q_b^{s'}$ *potential elements of* $D_b^\pi(X')$*, choose only visitor-short contracts with highest* π *ranking one at a time, adding them to* $D_b^\pi(X')$*. Remove all remaining contracts and terminate the procedure.*

CHAPTER 2
HOUSING MARKET WITH CONTRACTS

In this chapter, I study matching with property rights problem in the housing market framework. Housing market problem is first introduced by Shapley and Scarf (1974), which presented a Top Trading Cycle algorithm that always finds a core matching. Since then, it is shown that the outcome of TTC is the unique core matching for each housing market (Roth and Postlewaite, 1977), the core mechanism is strategy-proof (Roth, 1982), and the core mechanism is the only mechanism that is Pareto efficient, individually rational, and strategy-proof (Ma, 1994). Abdulkadiroŭlu and Sönmez (2013) gives well organized review in this line of studies.

To introduce property rights in housing market, I assume the house can be offered in two contractual terms. Property rights requires that when an agent gets a house in a certain term, her house should also be offered as the same term. Moreover, when every agent owns a house, property rights reduces to an equal-term matching. After defining efficiency and core in equal-term domain, I show that, in a housing market with contracts problem, core may be empty. However, there always exists an efficient, individually rational, and equal-term matching in every housing market with contracts problem. Then I present a mechanism that always produces an efficient, individually rational, and equal-term matching.

1 MODEL

As an extension of the **housing market** (Shapley and Scarf, 1974), a **housing problem with contract** is a tuple $(I, H, X, \mu \succ)$. There are finite set of n agents I and finite set of n houses H, where agent i owns house h_i. A contract $x = (i, h, t) \in X$ specifies an

agent $i(x)$, a house $h(x)$, and the contractual term $t(x)$. A house can be offered in two contractual terms, $t \in \{R, L\}$. Agent i's option of remaining unmatched (or matched to her own house) is denoted by $\bar{i} = (i, h_i, R)$, and any agent can be matched to her own house only with R-term. Each i has a strict preference over contracts, $\succ_i$, and and a weak preference is denoted by $\succeq_i$.

A matching $\mu : I \to X$ is a function that matches a contract to an agent, i.e., $\mu(i)$ is a contract that an agent i is matched to. I define feasibility and individual rationality of the matching.

Definition 22. *A matching μ is **feasible** if,*

- $|\mu(i)| = 1$ *for all* $i \in I$,
- $i \neq j$ *implies* $h(\mu(i)) \neq h(\mu(j))$.

Definition 23. *A matching μ is **individually rational** if each agent is matched to a contract that is weakly preferred to remaining unmatched, i.e., for every i,*

$$\mu(i) \succeq_i \bar{i}$$

In the housing market problem, Pareto efficiency is an important solution concept along with the individual rationality.

Definition 24. *A matching μ is **Pareto efficient** if there is no matching ν such that $\nu(i) \succeq_i \mu(i)$ for all $i \in I$ and $\nu(i) \succ_i \mu(i)$ for some $i \in I$.*

Property rights can be interpreted as a restriction in this model. If i gets a L type contract, her house h_i can only be assigned to a L type contract.

Definition 25. *A matching μ **protects property rights** if, for any agent i,*

$$t(\mu(i)) = L \Rightarrow t(\{x | h(x) = h_i\}) = L$$

Definition 26. *A matching μ is **equal-term** if each agent is matched to a contract that has the same term that her house is matched to, i.e., for every i,*

$$t(\mu(i)) = t(\{x|h(x) = h_i\})$$

In general, **equal-term** implies **protecting property rights**, and the two are equivalent if every agent has her own house. Therefore, from now on, I use equal-term as a requirement to protect property rights. The problem becomes to find an efficient, individually rational matching under equal-term domain.

2 INCOMPATIBILITY BETWEEN PARETO EFFICIENT AND EQUAL-TERM

With equal-term as additional constraint, we may still want to acheive Pareto efficient outcome. However, equal-term is not compatible with Pareto efficiency in general. Pareto efficient matching may be not equal-term as the following example shows.

Example 5.

i_1	i_2	i_3
$(2, L)$	$(3, R)$	$(1, L)$
$(3, L)$	$(1, L)$	$(2, R)$
$\bar{1}$	$\bar{2}$	$\bar{3}$

In this example, i_1's most preferred contract is h_2 as L-term, i_2's most preferred one is h_3 as R-term, and i_3's most preferred one is h_1 as L-term. If they each receive their top choices, the matching is Pareto efficient, but clearly it is not equal-term, so it cannot respect property rights.

If equal-term property is required, we cannot expect a Pareto efficeint outcome in general. In order to incorporate this restriction, I define efficiency with equal-term constraint. , namely **equal-term efficient**. Also, I also define core under the same constraint.

Definition 27. *A matching μ is **equal-term-efficient**, or **EE**, if there is no equal-term matching ν such that $\nu(i) \succeq_i \mu(i)$ for all $i \in I$ and $\nu(i) \succ_i \mu(i)$ for some $i \in I$.*

Definition 28. *An equal-term matching μ is **in the equal-term core**, or, simply **in the core**, if there is no coalition $T \subseteq I$ and a equal-term matching ν such that,*

- $h(\nu(i)) \in \{h_j\}_{j \in T}$ *for all* $i \in T$,
- $\nu(i) \succeq_i \mu(i)$ *for all* $i \in T$,
- $\nu(i) \succ_i \mu(i)$ *for some* $i \in T$.

Equal-term efficiency is less restrictive than Pareto efficiency as it can be dominated by a matching that is not equal-term. Note that, equal-term efficienct matching does not need to be equal-term itself, and therefore Pareto efficienct matching is always equal-term efficient as well. Equal-term efficiency, along with equal-term, will be a solution concept in this housing market with contracts model.

3 PROPERTIES OF EQUAL-TERM MATCHING

This section discusses properties of equal-term matching in the housing market problem with contracts. The first observation is negative. Unlike the original hosing market problem, core may be empty in this model.

Remark 3. *In a house exchange problem with contract, core may be empty. The following example illustrates this.*

There are three individuals, and their preferences are given by the table, where $\bar{i}$ *indicates* i*'s option of matched to his own house.*

i_1	i_2	i_3
$(2, L)$	$(3, R)$	$(1, L)$
$(3, L)$	$(1, L)$	$(2, R)$
$\bar{1}$	$\bar{2}$	$\bar{3}$

There are three efficient matching in this problem. $(i_1 - 2L, i_2 - 1L, i_3 - \bar{3})$; $(i_1 - \bar{1}, i_2 - 3R, i_3 - 2R)$; $(i_1 - 3L, i_2 - \bar{2}, i_3 - 1L)$. *The first one is blocked by* (i_2, i_3) *coalition, second one by* (i_3, i_1)*, and the third one by* (i_1, i_2).

One can see that there is a conflict of interest between R-term contracts and L-term contracts, which results in empty core, or multiple core matchings. If all agents prefer one type to the other, then there is no conflict, so that we can recover the uniqueness of the core matching.

Definition 29. *An agent is R-type (L-type) if all of her acceptable R-term (L-term) contract is preferred to any acceptable L-term (R-term) contract.*

If the agents are all the same type so that there is no conflict of interests, then the market becomes almost similar to traditional housing market.

Proposition 4. *If all agents are R-type or L-type, then the core always exists and is unique.*

Proof. See appendix. □

We can find the core matching by executing Top Trading Cycle. This result does not hold if there is at least one agent who is not the same type.

Proposition 5. *If all but one agent is different type, then core may not exist.*

In general, therefore, one cannot assume that the core is non-empty. Even when core is empty, however, there always exists an equal-term efficient matching in housing market with contracts. Therefore, our goal is to find an efficient matching under equal-term constraint.

Theorem 7. *There exists an EE, IR, and equal-term matching in every house exchange problem.*

Proof. Take all EE and IR matchings. Since the initial endowment matching is EE and IR, there is at least one such a matching. There should be at least one that is not dominated by any other matchings in the set. That matching is the EE, IR, and equal-term matching. □

Even though the core may be empty, the set of EE, IR, and equal-term matching is always non-empty. Therefore, the natural question is how to find an EE and IR outcome under equal-term constraint. Next section will address this issue.

4 EFFICIENT MATCHING MECHANISMS

In this chapter, I show a mechanism that produces a EE, IR, and equal-term matching. In order to do that, I first formally define the mechanism and its properties.

Definition 30. *Let M be the set of all matchings. A mechanism is a function $\phi :\succ \to M$, which selects a matching for a given preference profile.*

Definition 31. *A mechanism is strategy proof (SP) if for any $\succ$ and $\succ'$, and for any $i \in I$,*

$$\phi(\succ) \succ_i \phi(\succ'_i, \succ_{-i}).$$

A mechanism ϕ is equal-term efficient if it always selects an equal-term matching, individually rational if it always selects an individually rationoal matching, and equal-term if it always selects an equal-term matching. The next proposition shows incompatibility of SP with other properties.

Before introducing a mechanism, I present impossibility result. Housing market with contracts model, fails to have SP property if we were keeping EE, IR, and equal-term.

Proposition 6. *There is no mechanism that is EE, IR, equal-term, and SP.*[1]

Proof. Suppose there are two agents with the following preferences.

i_1	i_2
$(2, L)$	$(1, R)$
$(2, R)$	$(1, L)$
$\bar{1}$	$\bar{2}$

There are two efficient and equal-term matchings; $(i_1 - 2L, i_2 - 1L); (i_1 - 2R, i_2 - 1R)$. If the mechanism chooses the first one, then i_2 can manipulate by stating $(1, L)$ is not acceptable for her. If the mechanism chooses the second one, then i_1 can manipulate by stating $(2, R)$ is not acceptable for her. □

The following is obvious observation from the definition of core.

Corollary 2. *There is no strategy-proof mechanism that produces a matching in the core (whenever core exists).*

[1] Note that, normal TTC is a PE and SP mechanism (but not equal-term), and keeping each agent's endowment is an equal-term and SP mechanism (but not efficient).

One may wonder how far we can reach if we were to keep SP. The following theorem says that there is not much that we can improve upon the initial endowment in strategy proof way.

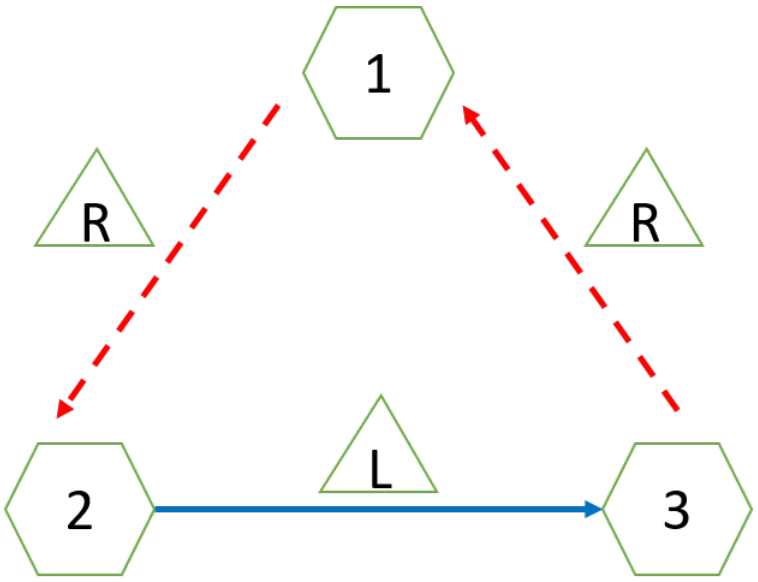
Definition 32. *A mechanism is* ***Pareto improving from endowment*** *if it chooses a matching that Pareto dominates initial endowment matching whenever it's available.*

Theorem 8. *If a mechanism is SP, IR, and equal-term, then it cannot be Pareto improving from endowment.*

Proof. See appendix. □

When there are two possible cycles, R-cycle and L-cycle, the mechanism should choose between the two at some point. When there is an agent who dislikes the mechanism's choice, he can manipulate by stating that the chosen contractual term is not acceptable. This incompatibility is inevitable when we require equal-term. Therefore, we drop SP and try to find a matching that is EE, IR, and equal-term.

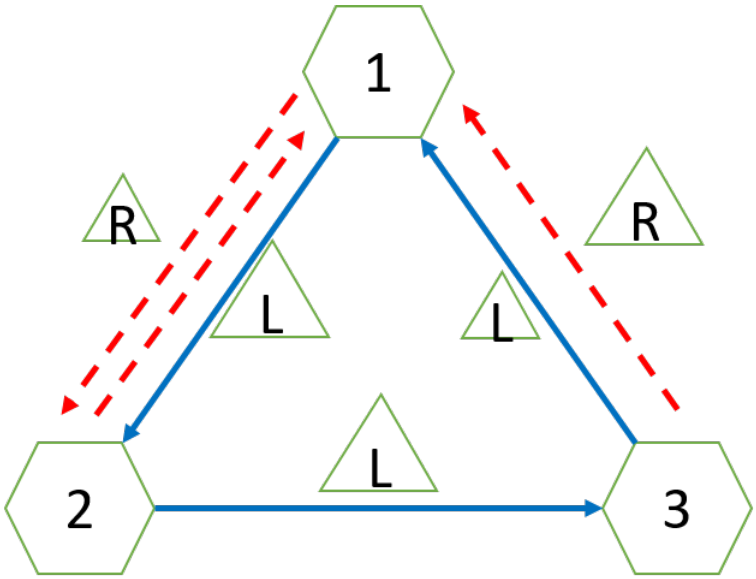
The first candidate for the housing market with contracts would be Top Trading Cycle, as it is the core mechanism in housing market problem. However, TTC does not work in this setup, and especially it fails to produce an equal-term matching as the following figure illustrates.



Top trading cycle does not work.

In the above graph, each agents points to her most preferred contract and forms a cycle. However, when i_1's top choice is h_2 as R-term, i_2's top choice is h_3 as L-term, and i_3's top choice is h_1 as R-term, we cannot clear the cycle since it will violate equal-term property.

As a way of producing an efficient equal-term matching, I propose a *dual-TTC* in which the algorithm tries to find equal-term matching both in R-terms and L-terms.



Dual-TTC may work.

4.1 ONE-TYPE FAVORED TTC

Unlike the original housing market, there could be overlapping cycles with the same set of agents in dual-TTC, and it is important how to choose one among them. At this point, without specifying the cycle choice rule yet, I define the new algorithm as follows.

Definition 33. *A **loop** is an ordered list of agents $(i_1, i_2, \cdots, i_k)$ such that agent i_1 demands agent i_2's house, agent i_2 demands agent i_3's house, $\cdots$, agent i_k demands agent i_1's house. An **equal-term-loop** is a loop that each agent demands a house of the next agent, all with the same term. A **top rank contained** equal-term-loop is an equal-term-loop that at least one agent is demanding her most preferred available contract.*

One can see that the top rank contained equal-term-loop is crucial for EE and equal-term matching. Now I introduce an extreme case of dual-TTC, which favors only R-type agents.

Definition 34. *R-type favored Top Trading Cycle algorithm works as follows.*

- *Each agent i points to her most preferred R contract. This will yield one of the following three cases.*

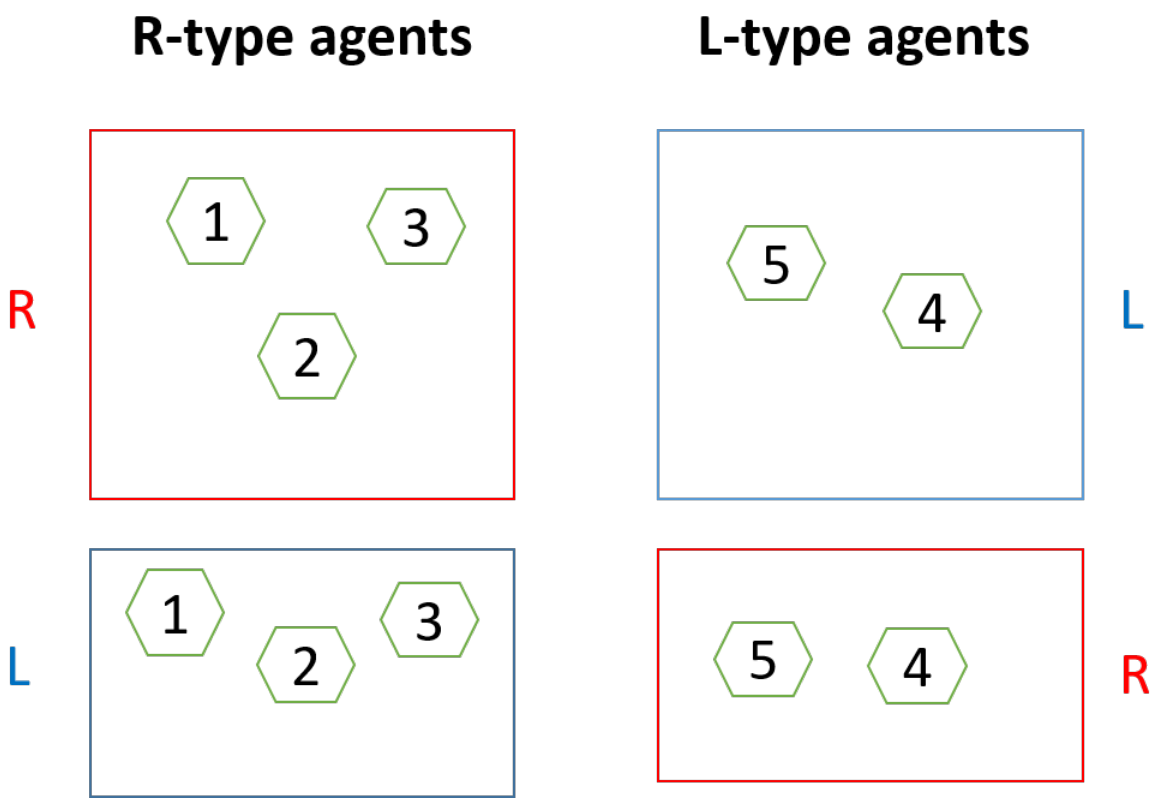
  case1 : It forms a loop with at least one R-type agent: Clear the loop by assigning each agent's demanded contract and remove all the agents in the loop and all the contracts that are involved these agents. Then go back to the initial step.

  case2 : No cycle is formed: There are some agents who do not have any acceptable R-term contract. Remove all R-term contracts involving these agents and go back to the initial step.

  case3 : It forms a cycle which only involves L-type agents: There is at least one L-type agent whose house is demanded by at least one R-type agent. Call this L-type agent i_1. Let i_2 be the agent whose house is demanded by i_1, $\cdots$, i_k be the agent who closes the loop. Now, remove i_k's most preferred R-term contract and go back to initial step.

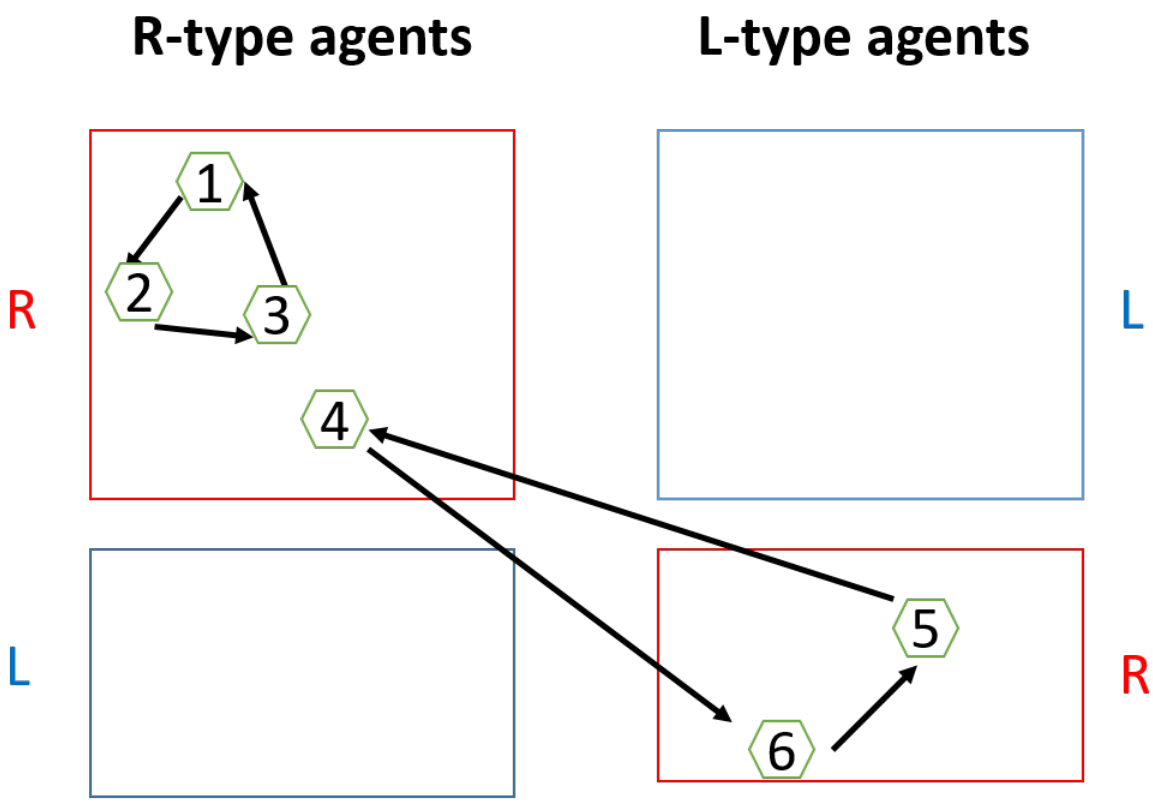
- *When there is no more R-term contract, then run the TTC with L-term contracts among remaining agents.*

Example 6. *R-type favored TTC*

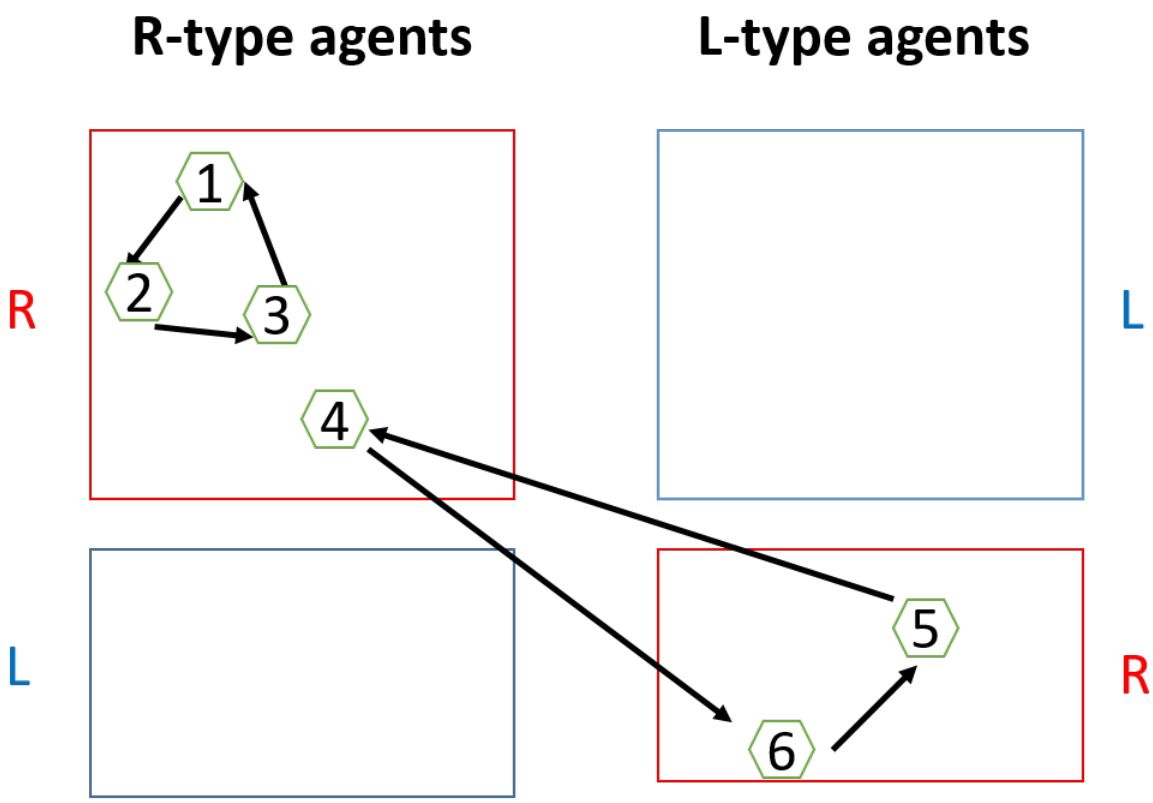


There are R-type agents and L-type agents. Each agent appears twice.

[Case 1]

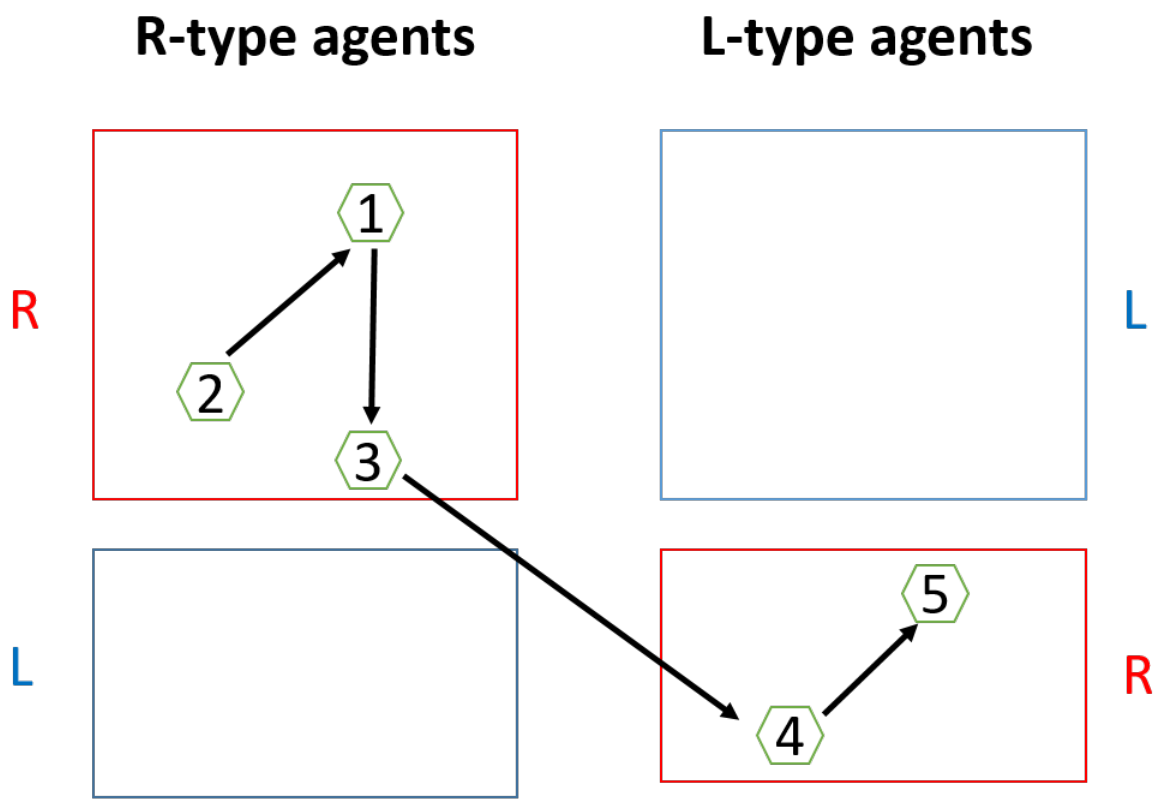


Cycles with at least one R-type agent.

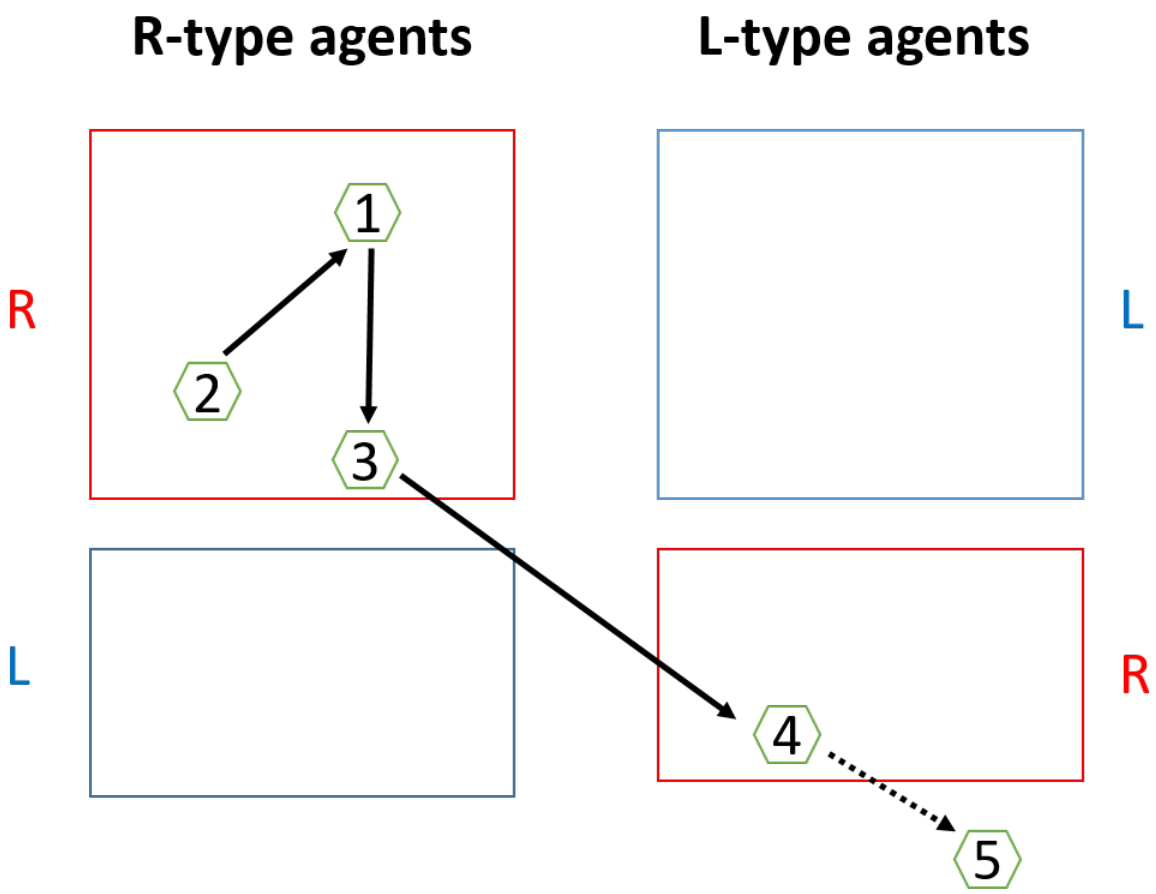


Each agent in the cycle gets what he demands.

[Case 2]

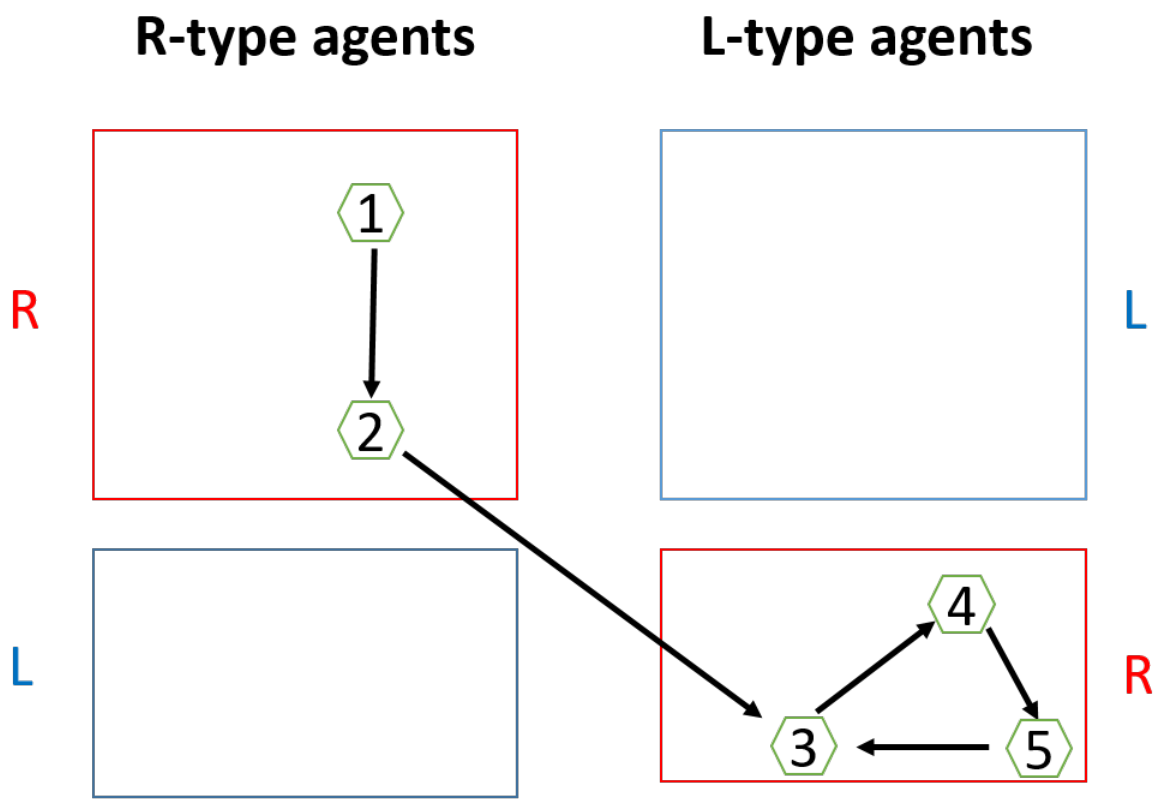


No cycle has formed. There is at least one agent who does not have any acceptable R contract. In this case, it is the agent 5.

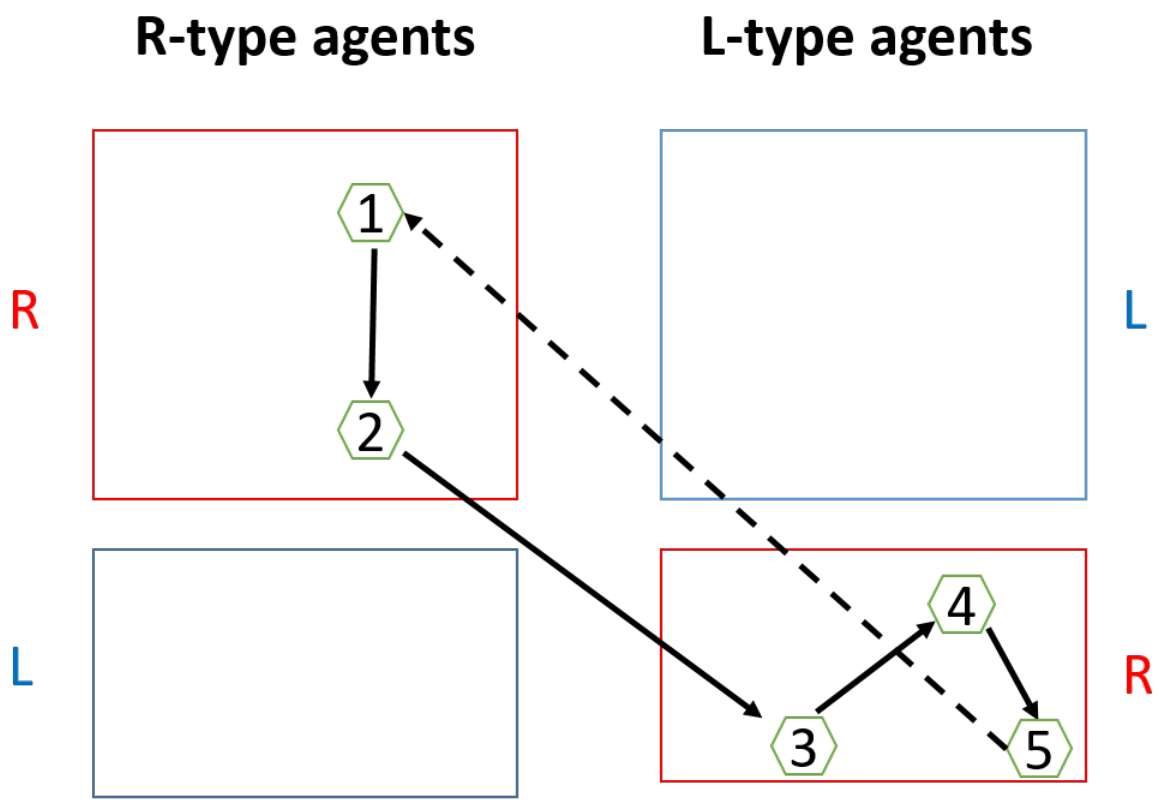


Remove all R-term contracts that involve agent 5.

[Case 3]



Cycle with only L-type agents.



Agent 5 points to her second preferred R-term contract.

Theorem 9. *Suppose all agents are either L-type or R-type. Then one-type favored TTC produces an efficient, individually rational, and equal-term matching. Moreover, the R-type favored (L-type favored) TTC is strategy proof for R-type (L-type) agents.*

Proof.

- Individual rationality: every agent is getting one of the contract they pointed, including the option of remaining unmatched. Therefore, no one will be matched to a contract that is less preferred than their own house.

- Equal-term: Since the algorithm clears equal-term cycles only, the final matching is obviously equal-term.

- Efficiency: Let matching μ be the outcome of the algorithm and let ν be a matching that Pareto dominates μ. There is at least one agent who strictly prefers the contract under ν to the contract under μ. Call this agent i. Since i prefers $\nu(i)$ to $\mu(i)$, he must have pointed to $\nu(i)$ before $\mu(i)$. Let j be the agent who pointed a contract other than

$\nu(j)$. If $\mu(j) = \nu(j)$, then j would also pointed to $\nu(j)$. If everyone in the loop has pointed ν, it would have been cleared. Therefore, there is an agent who pointed other than ν and breaks the loop, when i pointed to $\nu(i)$. Since j less prefer $\nu(j)$ to what he pointed, he will be worse off under ν, contradicting that ν Paroto dominates μ.

- Strategy proof for R-type: when a cycle is cleared in the process, all R-type agent gets the most preferred contract among all acceptable R-term contracts for each of them. Therefore, no R-type agent has an incentive to get cleared in the later rounds than his own. Moreover, none of them can get a contract that is cleared earlier, as it will violates Pareoto efficiency of the final outcome. Hence, any R-type agent's weakly dominant strategy is to report his preference truthfully.

□

Remark 4. *One-type favored TTC outcome may not be in the core (even when the core is non-empty).*

5 CONCLUSION

In this paper, I study a matching with property rights in housing market model. When every agent has own house, protecting property rights reduces to equal-term matching. I defined core and efficiency under equal-term domain. Even though core may be empty in this environment, I showed there still exits an equal-term efficient and individual matching. Then I proposed algorithm that produces equal-term efficient, individually rational, and equal-term matching. However, due to the conflict of the different contractual terms, no strategy proof mechanism can produce an efficient, individually rational, and equal-term matching.

This work is an attempt of matching with contract model in Top Trading Cycle approach. Even though the results are limited, it can be applied to many real world problems including pet-sitting exchange, Sabbatical housing exchange, etc.

APPENDIX

2.A PROOFS

2.A.1 PROOF OF PROPOSITION4

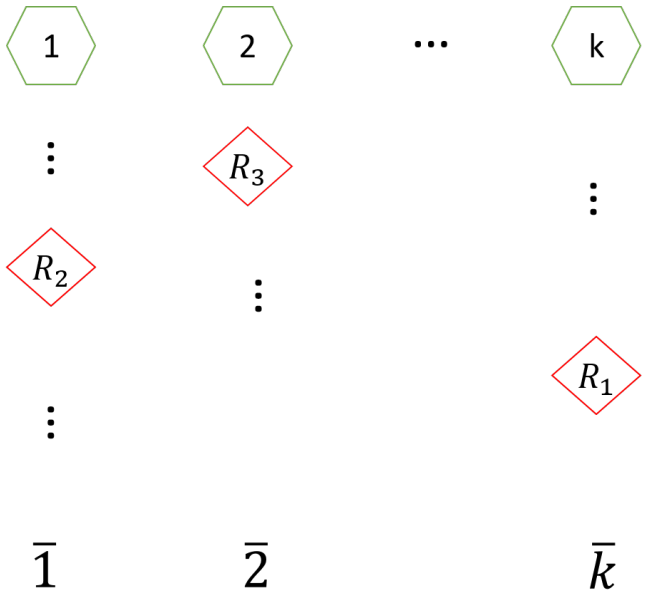
Proof. Core can be obtained by running TTC twice, for R-term contracts first, followed by L-term contracts. Let μ be the outcome of TTC and consider R-term coalition with matching ν. Since everyone shares the same preference over R-term and L-term, the agents in the coalition should have pointed $\nu(i)$, earlier than pointing to $\mu(i)$. However, the fact that they get $\mu(i)$ implies that at least one $\nu(i)$ cannot form a clearing cycle, so that the owner of $\nu(i)$ gets a better outcome than what he gets in the ν. Therefore, under ν, at least one is worse off, contradicting that ν is a blocking matching of the coalition. Same argument applies to L-term coalition.

□

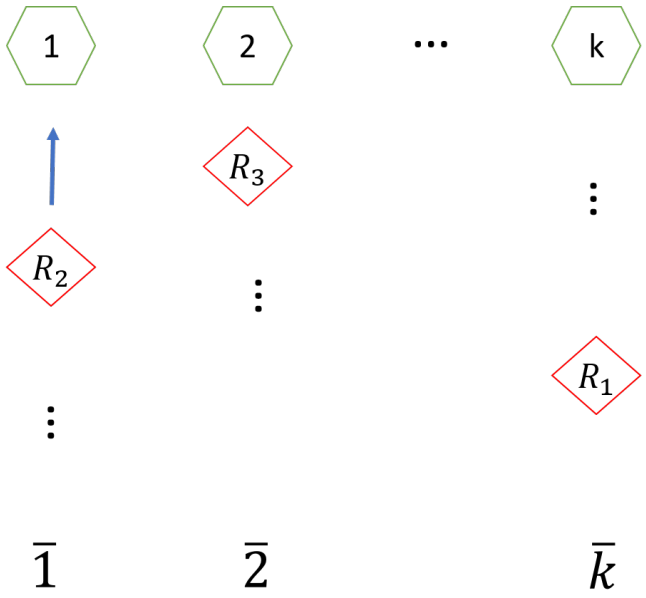
2.A.2 PROOF OF THEOREM8

Proof. Suppose not. An EE, IR, and equal-term mechanism ϕ chooses a matching that Pareto dominates endowment. Let a group of agents, $\{i_1, \cdots, i_k\}$, be the ones who each gets the next agent's house as R-term contract (without loss of generality). Let R_2 be the contract that agent i_1 is matched to, and R_3 the contract that i_2 is matched to, and so on, and lastly R_1 be the contract that agent i_k is mathced to.

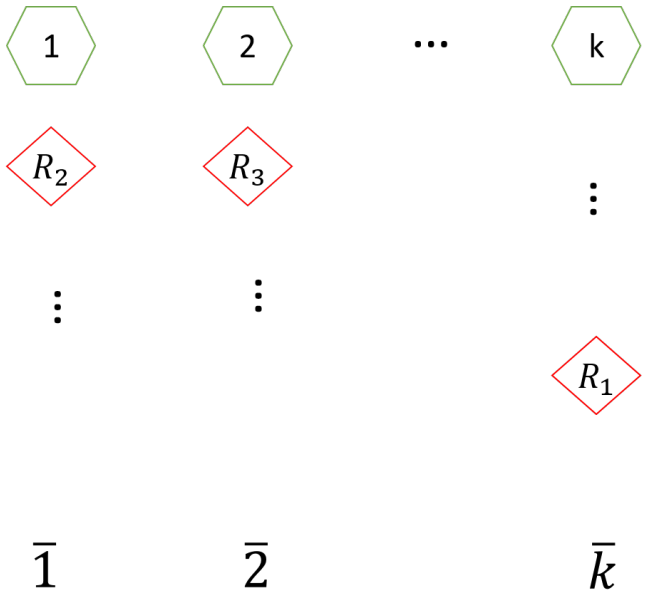
Call the initial preference profile the profile 1. Starting from the first agent, i_1, change her preference by moving R_2 as her top choice. By PE, this cannot change i_1's matching under ϕ.



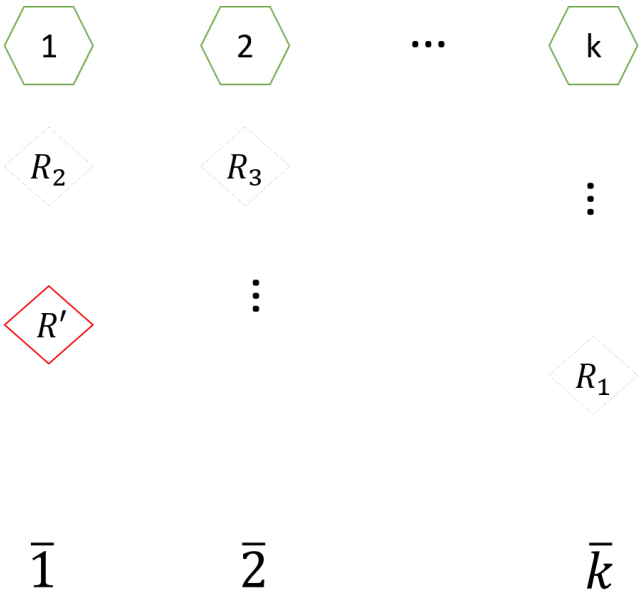
Profile 1



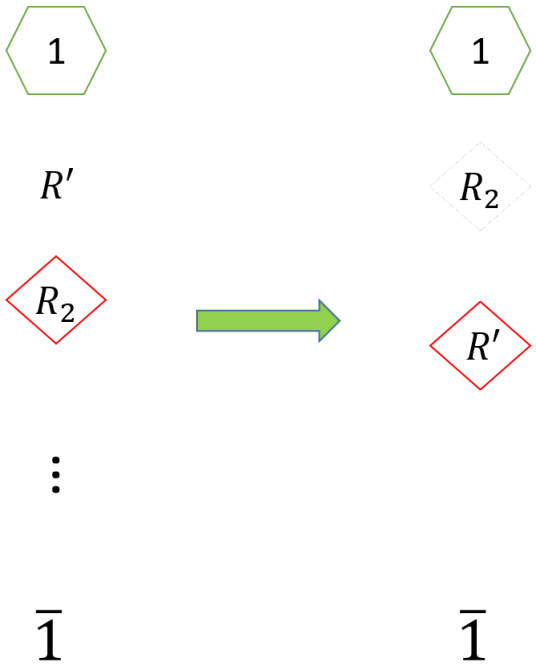
Move the final matching to the top one by one.



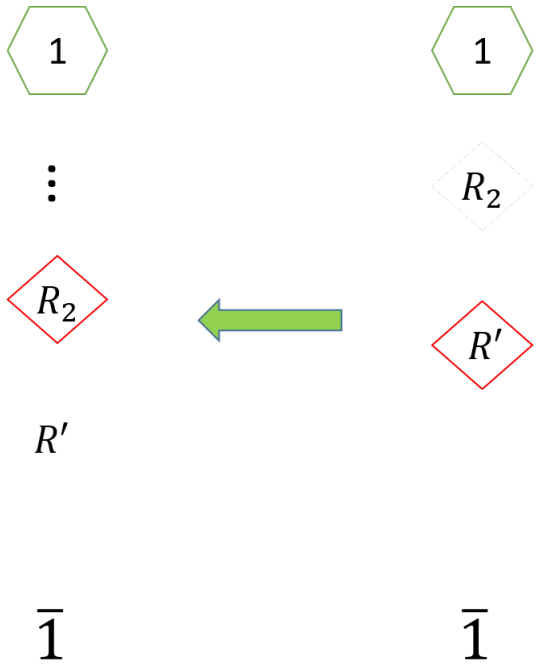
Final matching should be the same for i_1.



If R' is chosen, it violates SP.



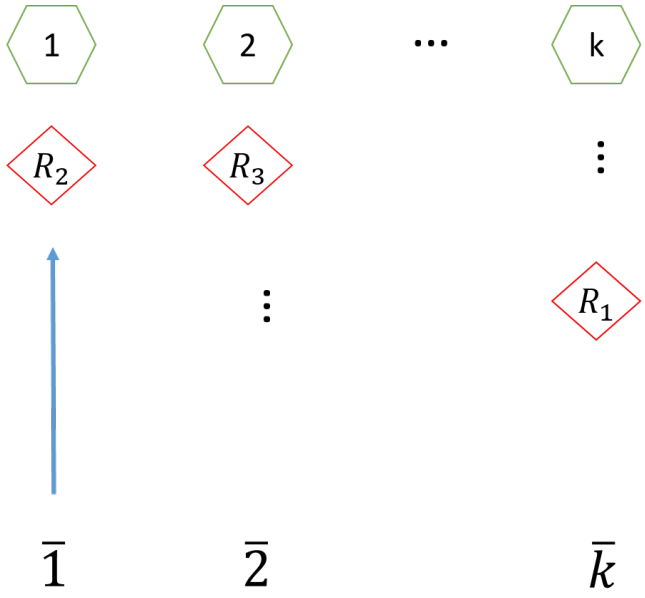
If $R' \succ_i R_2$ in profile 1, then i_1 can manipulate when the profile 1 is his true preference.



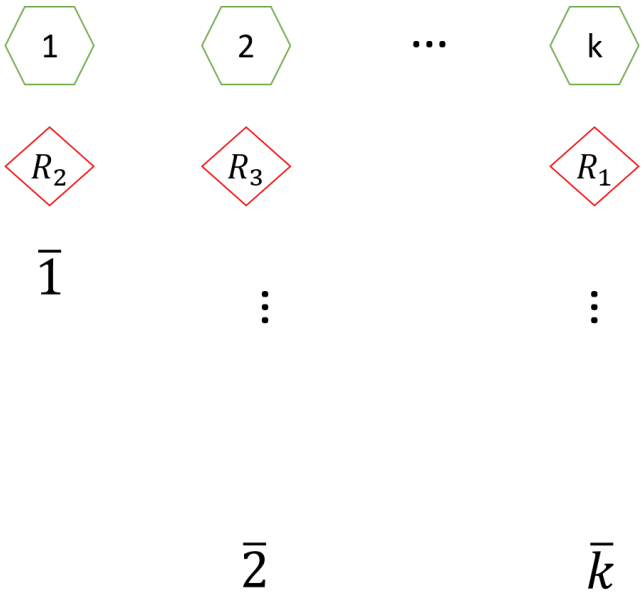
If $R_2 \succ_i R'$ in profile 1, then i_1 can manipulate in opposite direction.

After that, move i_1's endowment $\bar{1}$ to the second of her preference. By IR, the mechanism should assign either R_2 or $\bar{1}$ to i_1. By SP, i_1 keeps her original matching R_2. In the

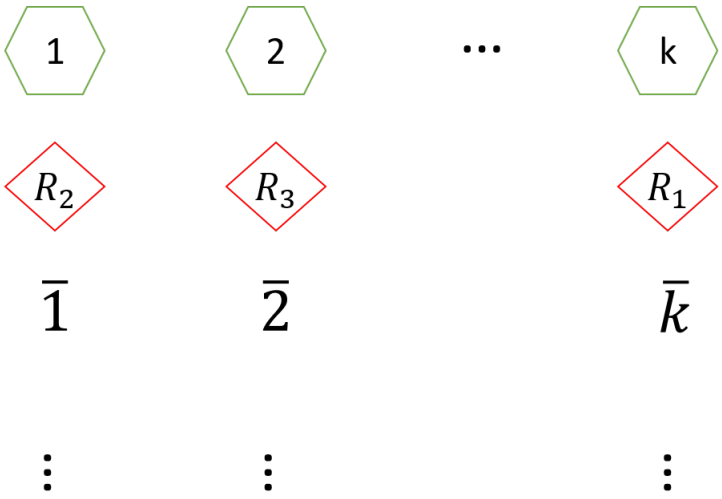
same way, move each agent's matching under ϕ and endowment one by one, resulting in profile 2.



Move endowment of i_1 to her second choice.

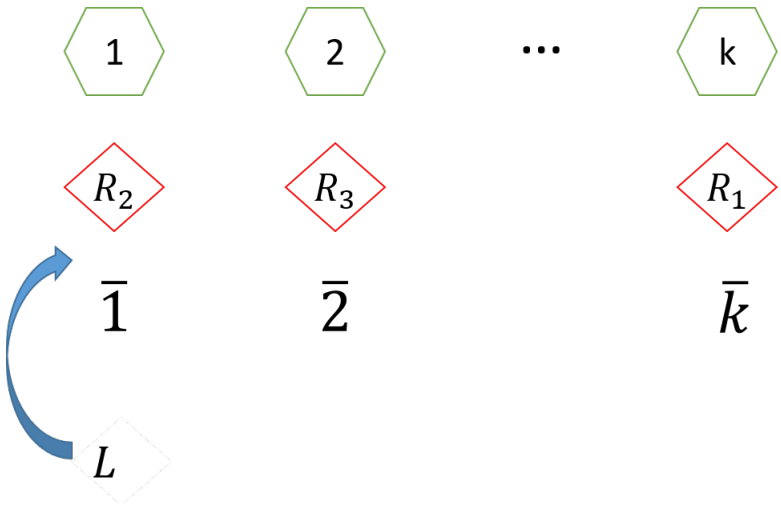


Final matching remains the same by SP and IR.

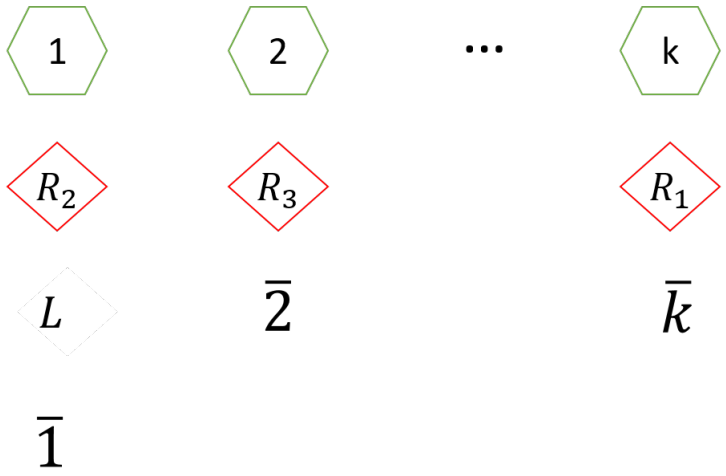


Profile 2

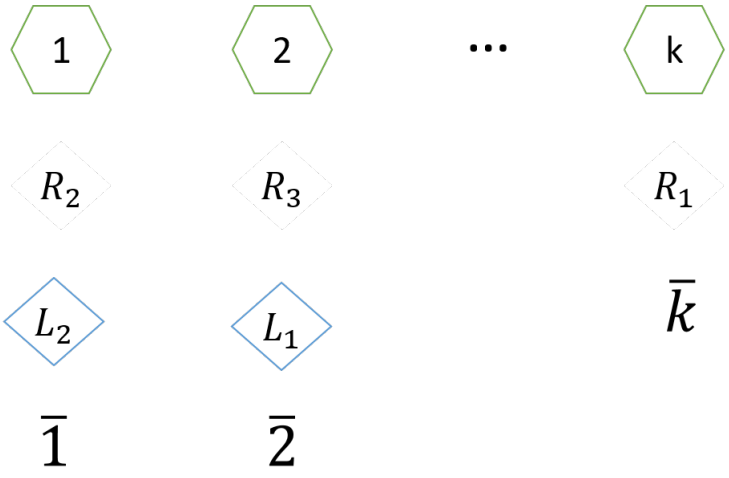
At this stage, move all others' (i_j with $j > k$) endowment to their top choices. Now the mechanism will assign endowments for those agents. This will not change the agents up to k, as the initial cycle is the unique Pareto improving matching from endowment in this economy. Now, one by one, move the top L-term contract to the second of each agent, resulting in profile 3.



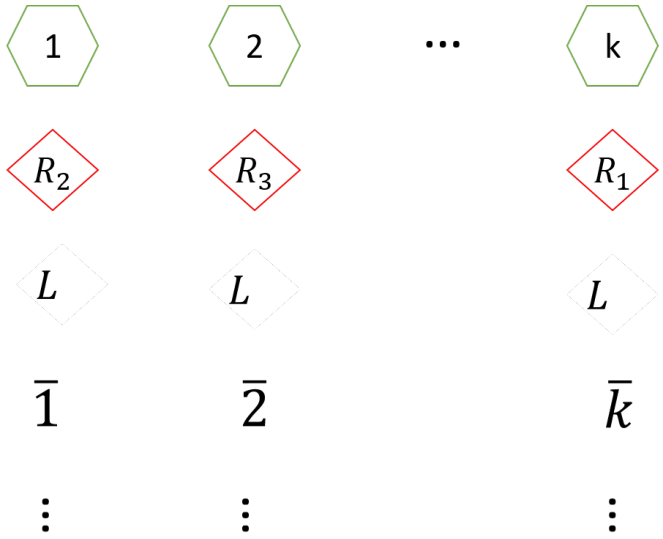
Move top L-term contract one by one.



Final matching does not change as assigning L to i_1 will, by equal-term, assign another agent a L contract, which violates IR.

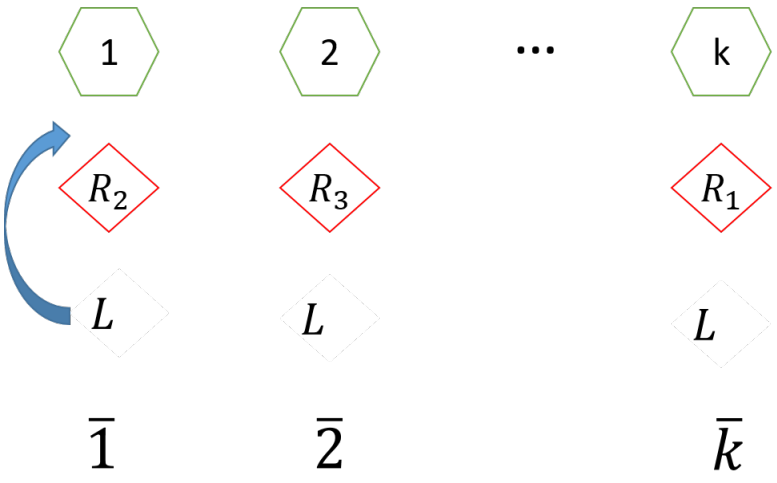


Move i_2's L does not change the final matching. For example, this matching will violate SP.

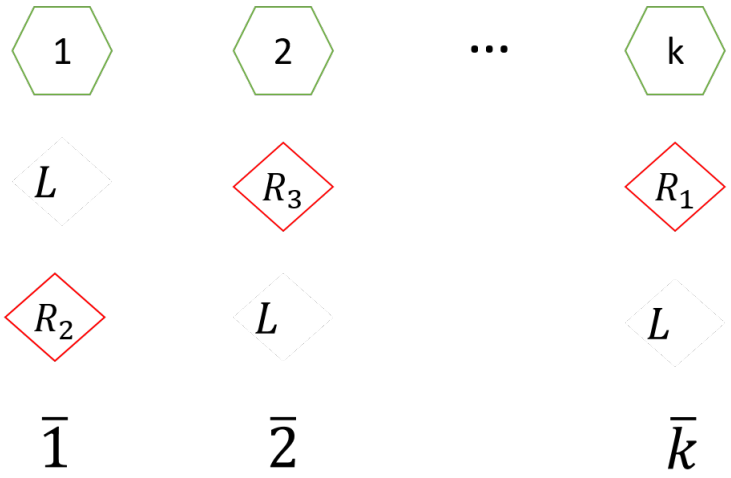


Profile 3

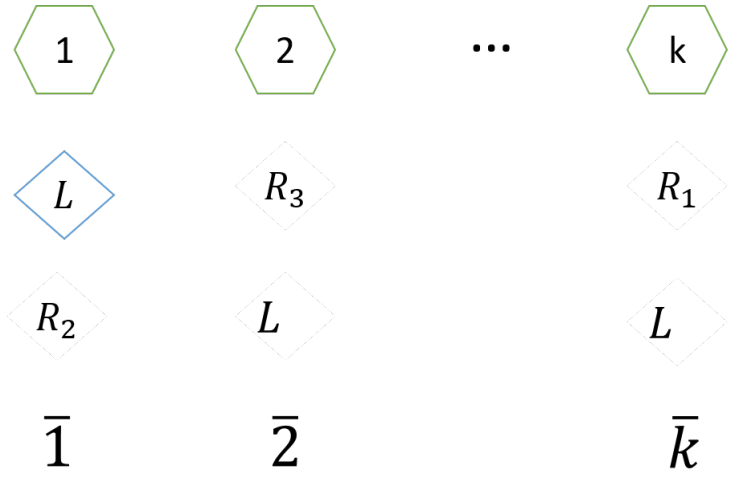
In profile 3, the mechanism still chooses the R-term cycle. Lastly, move i_1's second ranked contract (L-term contract) to the top, to see contradiction. If the mechanism chooses the L-term contract for i_1, then i_1 succesfully manipulated, contradicting SP. If R_2 is remain to be chosen, then i_1 can report only her top contract is accepable, resulting in L-term cycle matching by IR.



Move L to the top.



Violates SP by i_1.

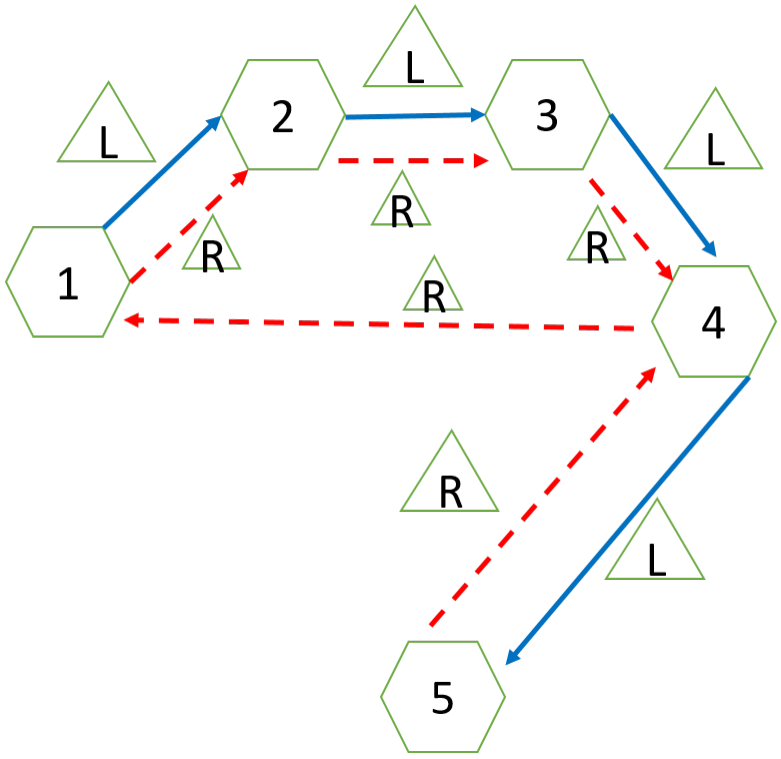


Violates SP by the i_1.

□

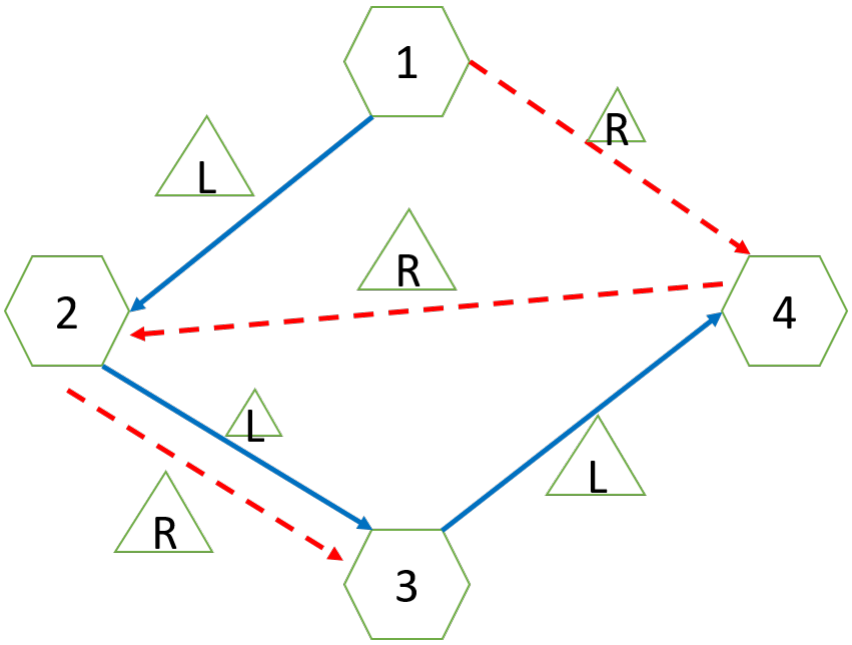
2.A.3 PROOF OF THEOREM10

Lemma 2. *If every agent has at least one of each R-type and L-type acceptable contract, then there is at least one equal-term-loop. Furthermore, at least one equal-term-loop is top rank contained.*



No top rank contained equal-term-loop. i_5 has only R-type contract in this case. If i_5 had an acceptable L-type contract, there should have been a top rank contained equal-term-loop.

Lemma 3. *If there is no equal-duration-loop, then there exist at least one agent whose acceptable contracts are all L-type, and at least one agent whose acceptable contracts are all R-type.*



No equal-duration-loop has formed. i_3 has only L-type acceptable contract, and i_4 has only R-type acceptable contract.

2.B DUAL TOP TRADING CYCLER

In this chapter I generalize the one-type favored TTC. To do that, we need a priority order of agents, or choice rule for equal-term loops. If there is a pre-determined priority, then the mechanism can simply choose the one that contains highest priority agent. Alternatively, it may choose the loop that contains maximum number of top-ranked contracts. The following algorithm takes priority order of the agents or the choice rule among efficient cycles as given.

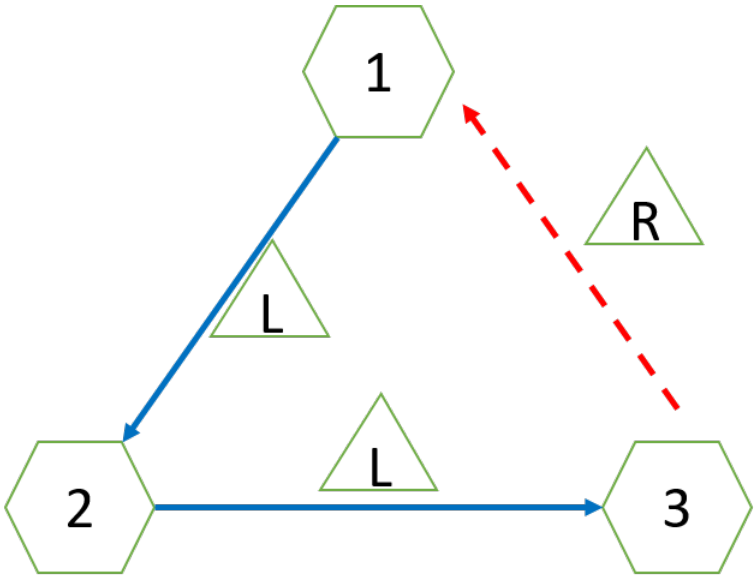
step 1 : Each agent points to her most preferred contract among all L-type acceptable ones, and to her most preferred contract among all R-type acceptable ones. If any agent has only one type of acceptable contracts, then she only points to one contract. Proceed with step 2.

step 2 : If there is any **top rank contained equal-term-loop**, clear one loop according to the choice rule. Assign each agent's demanded contract and remove all the agents in the loop and all the contracts that are involved these agents. Go back to step 1. Otherwise, proceed with step 3.

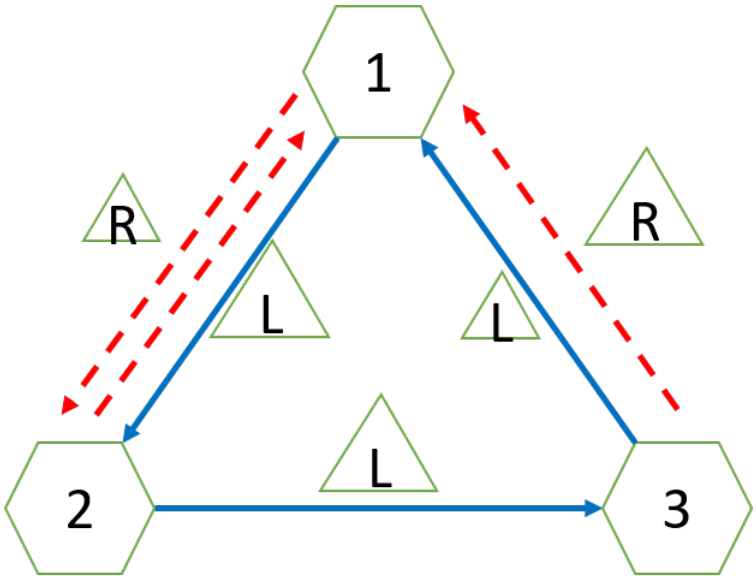
step 3 : If there is no **top rank contained equal-term-loop**, but an **equal-term-loop**, then temporally remove the agent who is not in the loop and has only a single type available contracts. Then, the agent who was pointing the removed one, points to her next available contract. Continue until a top rank contained equal-term-loop forms. If no such a loop has formed until everyone is removed, then recover all removed agents in this phase and clear the equal-term-loop.

Example 7.

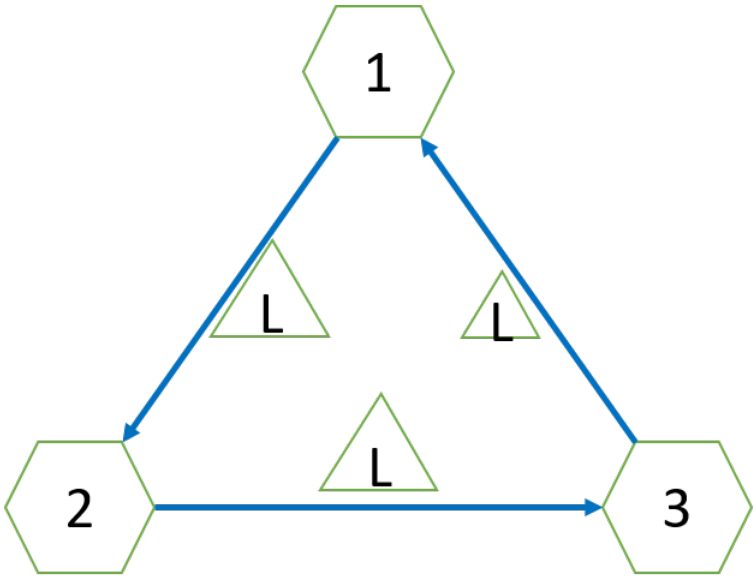
i_1	i_2	i_3
$(2, L)$	$(3, L)$	$(1, R)$
$(2, R)$	$(1, R)$	$(1, L)$
$\bar{1}$	$\bar{2}$	$\bar{3}$



Each points to the top preferred contract.



Also points to the top preferred contract of the other term.



Clear the loop that contains top-ranked contract.

Theorem 10. *The dual TTC produces an efficient, individually rational, and equal-term matching.*

2.C TOP TRADING CYCLE WITH COUNTER OFFER

This chapter introduces a new approach to housing market with contracts problem. This allows agents to *counter offer* when they are demanded by a less preferred contractual term. For example, if someone demands my house as R-term, but if my top choice is his L-term, I counter offer him with L-term contract. Since it was not known yet whether this algorithm produces an efficient outcome, I present the algorithm without giving any result yet. The key idea in this algorithm is to check if there exists a equal-term-loop within a loop whenever a loop is formed. This can be thought as an extension of YRMH-IGYT algorithm. (Sönmez and Ünver, 2010)

step 0 : Set counter for each agent to zero.

step 1-1 : For ordering π restricted to available agents, increase the counter number of the first agent by 1, and the first agent demands his top available contract.[2] Proceed to step 1-2.

⋮

step 1-k : Insert the agent whose space was demanded in the previous step to the top of the ordering and increase the counter of this agent by 1, until someone has counter number 2. If someone has a counter number 2, that means a loop has formed. Proceed to step 2-1.

step 2-1 : Within the current loop, name the agent whose space is recently demanded i_1, and the one whose space i_1 demanded i_2, $\cdots$, and lastly, the one who demands i_1 becomes i_k, so that each agent demands the space of the next agent in the loop. Proceed to step 2-1.

step 2-2 : If the loop is an **equal-term-loop**, then assign each agent's demanded contract and remove all the agents in the loop and all the contracts that are involved these agents. Then go back to step 0. Otherwise, proceed to step 2-3.

[2]The counter counts how many times an agent demanded in this phase.

step 2-3 : Let $t(i_n)$ be the contract term that i_n demanded. If counter number of i_2 is 1, then i_1 demands her top contract again. Then increase i_2's counter number by 1, and go back to step 2-1. If it is greater than or equal to 2, go to step 2-4.

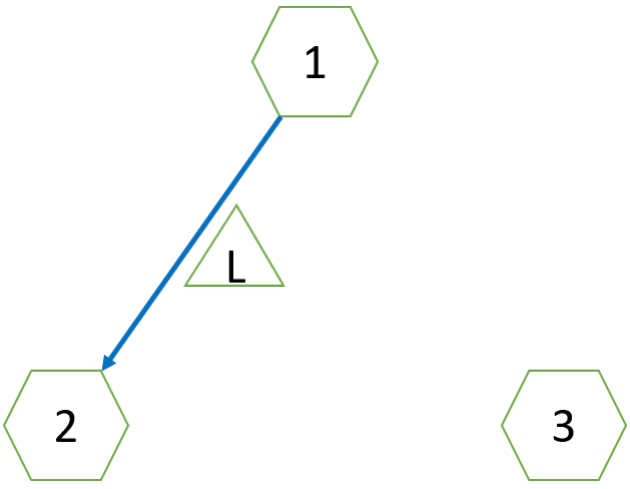
step 2-4 : If $t(i_k) = t(i_1)$, i.e., the contract term that i_1 demands is the same as the contract term that i_1's place is demanded by i_k, then i_1 demands her top choice again. Then go back to step 2-2. Otherwise, go to step 2-5.

step 2-5 : If $t(i_k) \neq t(i_1)$, then i_1 demands her second top contract.

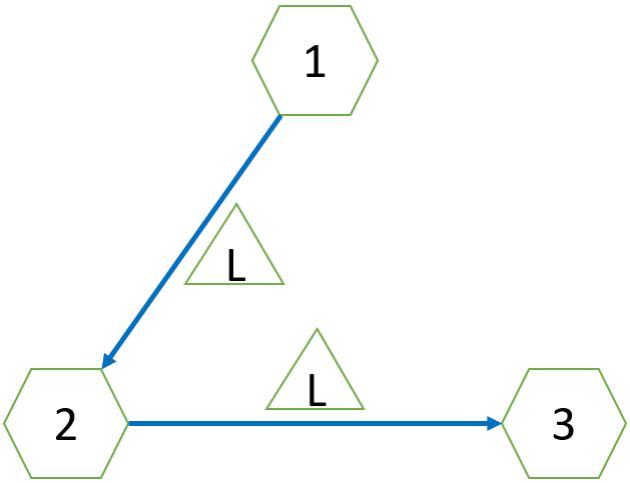
(case 1) If i_1's second top choice is involved one of the agents in the current loop, it starts a new loop. It could be i_2, forming the same loop, or i_k, forming a two-way loop. Go to step 2-1.

(case 2) If i_1's second top choice doesn't belong to anyone in the current loop, it begins a new chain. Go to step 1-2.

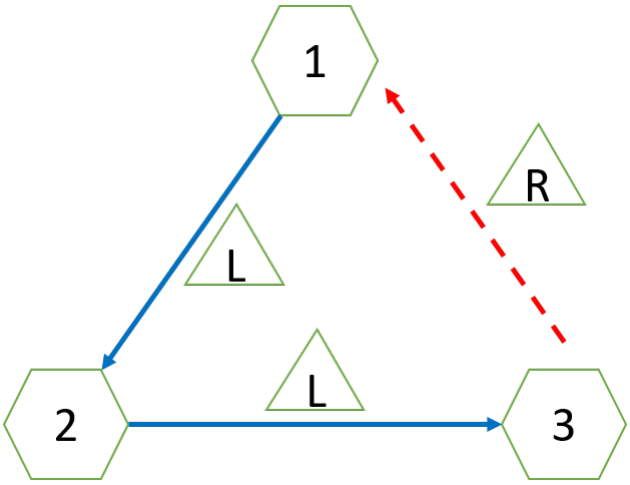
i_1	i_2	i_3
$(2, L)$	$(3, L)$	$(1, R)$
$(2, R)$	$(1, R)$	$(1, L)$
$\bar{1}$	$\bar{2}$	$\bar{3}$



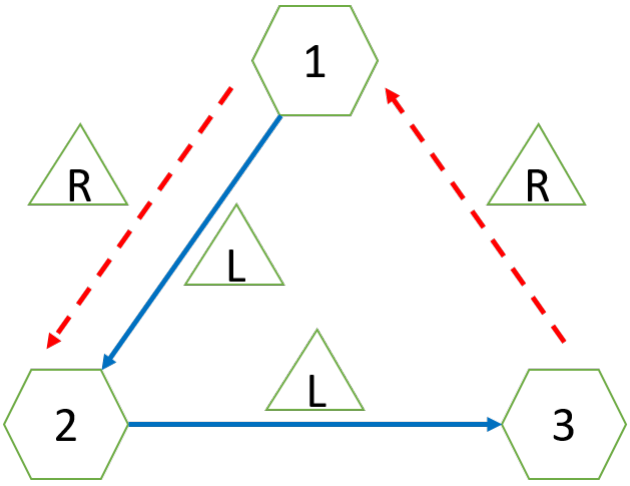
i_1 demands his top contract, $(2, L)$.



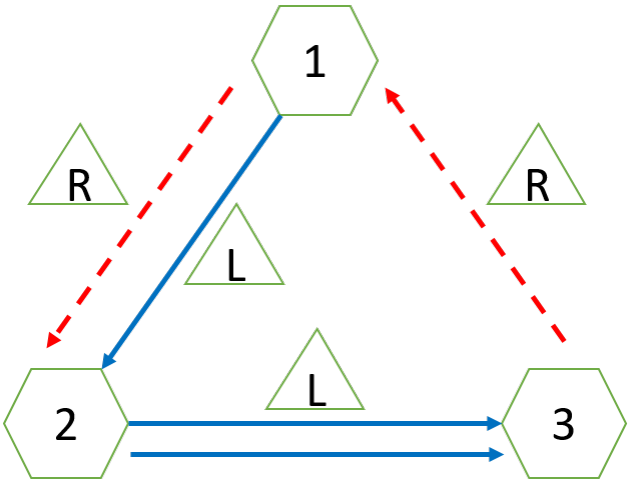
i_2 demands his top contract $(3, L)$.



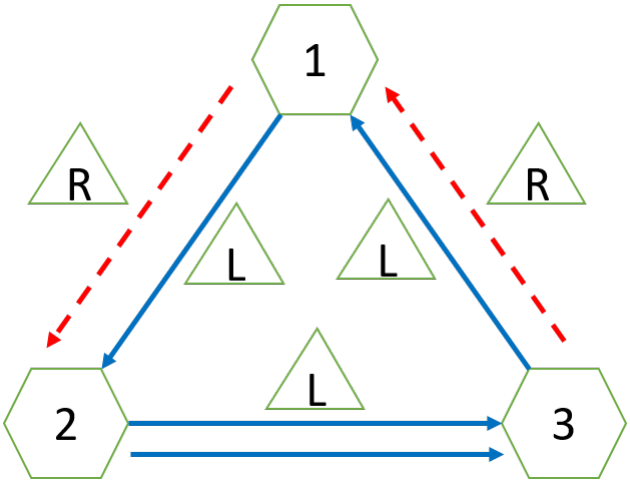
i_3 demands his top contract, $(1, R)$, forming a **loop**.



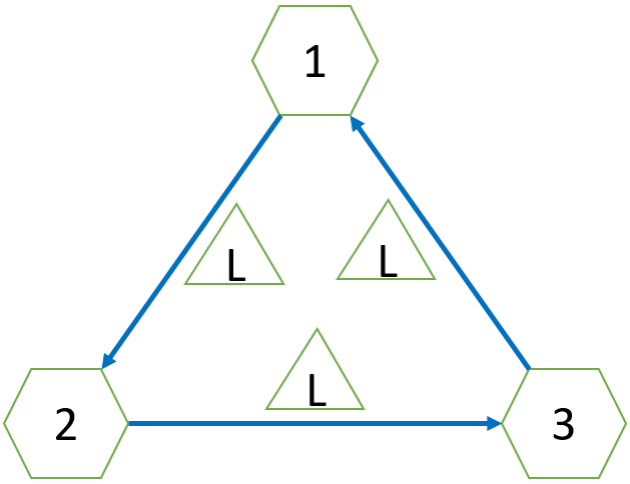
Since i_1's second choice involves the next agent in the **loop**, i_1 demands his second contract, $(2, R)$.



i_2's second choice does not involve the next agent in the **loop**, so he demands his first choice again.



i_3's second choice involve the next agent in the **loop**, so he demands his second choice, $(1, R)$, forming an **equal-term loop.**



Each of the agents gets a L-type contract from the next agent in the **equal-term loop.**

Bibliography

Abdulkadiroŭlu, A. and Sönmez, T. (2013). Matching markets: Theory and practice. *Advances in Economics and Econometrics*, Vol.1:3–47.

Ayala, D. et al. (2012). Stability of marriage and vehicular parking. *MATCH-UP 2012: the Second International Workshop on Matching Under Preferences*.

Chambers, C. and Yenmez, B. (2013). Choice and matching.

Ergin, H. and Sönmez, T. (2006). Games of school choice under the boston mechanism. *Journal of Public Economics*, 90:215–237.

Gale, D. and Shapley, L. (1962). College admissions and the stability of marriage. *American Mathematical Monthly*, 69, 1:9–15.

Geng, Y. and Cassandras, C. G. (2012). A new smart parking system infrastructure and implementation. *Procedia - Social and Behavioral Sciences*, 54:1278–1287.

Hatfield, J. W. and Kojima, F. (2010). Substitutes and stability for matching with contracts. *Journal of Economic Theory*, 145:1704–1723.

Hatfield, J. W. and Kominers, S. D. (2016). Hidden substitutes.

Hatfield, J. W. and Milgrom, P. R. (2005). Matching with contracts. *American Economic Review*, 95(4):913–935.

Ma, J. (1994). Strategy-proofness and the strict core in a market with indivisibilities. *International Journal of Game Theory*, 23:75–83.

Roth, A. E. (1982). Incentive compatibility in a market with indivisible goods. *Economic Letters*, 9:127–132.

Roth, A. E. and Postlewaite, A. (1977). Weak versus strong domination in a market with indivisible goods. *Journal of Mathematical Economics*, Vol.4:131–137.

Shapley, L. and Scarf, H. (1974). On cores and indivisibility. *Journal of Mathematical Economics*, Vol.1(issue 1):23–37.

Shoup, D. C. (2005). *The High Cost of Free Parking*. Planners Press, American Planning Association, Chicago.

Sönmez, T. (2013). Bidding for army career specialties: Improving the rotc branching mechanism. *Journal of Political Economy*, 121(1):186–219.

Sönmez, T. and Ünver, M. U. (2010). House allocation with existing tenants: A characterization. *Games and Economic Behavior*, 69:425–445.

van Ommeren, J. et al. (2011). The real price of parking policy. *Journal of Urban Economics*, 70:25–31.

van Ommeren, J. et al. (2015). Car ownership and residential parking subsidies: Evidence from amsterdam. *Tinbergen Institute Discussion Paper*, 2015-116/VIII.

Yenmez, B. (2017). College admissions. *GSIA working paper*, No. 2014 - E24.