Refugee Resettlement: an Impossibility Result

by

M. Emin Birkan

A Dissertation Submitted to the
Graduate School of Social Sciences and Humanities
in Partial Fulfillment of the Requirements for
the Degree of

Master of Arts

in

Economics

KOÇ ÜNİVERSİTESİ

February 1, 2024

Refugee Resettlement: an Impossibility Result

Koç University

Graduate School of Social Sciences and Humanities

This is to certify that I have examined this copy of a master's thesis by

M. Emin Birkan

and have found that it is complete and satisfactory in all respects,

and that any and all revisions required by the final

examining committee have been made.

Committee Members:

_____

Asst. Prof. Özgür Yılmaz

_____

Assoc. Prof. Emre Ekinci

_____

Assoc. Prof. Murat Yılmaz

Date: _____

[To my parents]

ABSTRACT

Refugee Resettlement: an Impossibility Result
M. Emin Birkan
Master of Arts in Economics
February 1, 2024

Many countries hosting refugees has no particular method of settling them. In most of these countries there are cities with a large concentration of refugees. We propose two Pareto efficient and strategyproof mechanisms to resettle refugees among cities. These mechanisms always yield results where the number of refugees does not increase in over concentrated cities. Further, we compare the success of these mechanisms on reducing the concentration of refugees on particular cities. None of these mechanisms always performs better than the other one, their success depends on the problem we consider. This observation is persistent even under a restrictive assumption. Furthermore, we show that there is no Pareto efficient and strategyproof mechanism that always performs weakly better than both of the mechanisms we offer. For a mechanism, minimizing the concentration of refugees on some cities is not compatible with being Pareto efficient and strategyproof.

ÖZETÇE

Mülteci Yerleştirme Mekanizmalarının Bir Yetersizliği Üzerine
M. Emin Birkan
Ekonomi, Yüksek Lisans
1 Şubat 2024

Pek çok ülke halihazırda ağırladığı mültecilerin ikamet ettikleri şehirlerde değişikliklerin nasıl yapılacağını düzenleyen bir method belirlememiştir. Mülteci ağırlayan önemli sayıda ülkede mülteci yoğunluğu bazı şehirlerde yoğunlaşmış durumda. Biz bu makalede mültecilerin ikamet ettikleri şehirlerin değiştirilmesi için takip edilebilecek iki tane Pareto optimal ve strategyproof mekanizma öneriyoruz. Bu mekanizmaların her ikisi de mültecilerin yoğunlaştığı şehirlerdeki mülteci sayısının artmasına izin vermiyor. Bu makalede ayrıca, önerdiğimiz mekanizmaların şehirlerdeki yoğunluğun azaltılması açısından performanslarını değerlendiriyoruz. Hangi mekanizmanın diğerinden daha iyi sonuç vereceği üzerinde çalışılan probleme göre değişiyor. Bu özellik mültecilerin tercihleri hakkında yaptığımız kısıtlayıcı bir varsayım altında dahi değişmiyor. Son olarak hem strategyproof hem Pareto optimal olan, ve önerdiğimiz iki mekanizmadan da hiçbir problem için daha kötü bir sonuç vermeyen bir mekanizmanın var olmadığını gösteriyoruz.

ACKNOWLEDGMENTS

I would like to thank my thesis advisor, Prof. Özgür Yılmaz, for his guidance throughout my first research attempt. I am also thankful to Ömer Faruk Şahin for his contribution to the final result of my thesis.

I would also like to thank Prof. Emre Ekinci and Prof. Murat Demirci for their support in my PhD application process.

Lastly, I would like to express my gratitude to my parents, Zekeriya Birkan and Nokta Birkan, and my wife, Emine İnci Birkan, for their invaluable support throughout my master's degree.

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

By the first half of 2021, the number of refugees in the world has reached 20.8 million. Some countries such as Colombia, Turkey and Germany host more than one million refugees. In some of the countries hosting refugees, the lack of central planning leads to over concentration of refugees to certain cities. This concentration both worsens the living conditions of refugees and creates problems for natives.

Turkey is an example for these countries. There are currently 3.6 million registered Syrian refugees in Turkey. In some major cities in the south-eastern and southern regions of country, the percentage of Syrian refugees to the total population is above 20%, whereas in the twenty least-concentrated cities this percentage is below 0.5%. The local congestion leads to poverty and insufficient infrastructure in over concentrated cities. The inadequacy of refugees' access to social services is another negative aspect of over concentration. For example, the education enrollment rate among school-aged children is only 27% for Syrian refugees in Turkey.On the other hand, in the cities of low concentration, the refugees' access to social services is far from being a problem.

Another concern is the integration of refugees to the labor force. When there is a massive migration, an extensive infrastructure for social services such as education, health and housing is needed. Further, introducing large number of refugees into the workforce may effect unemployment and wages of natives. In the example of Turkey, Syrian refugees have great difficulties in meeting even their basic needs, since market mechanisms and market selection have failed, leading to both

regional and sectoral congestion. Thus, central planning is urgent. In this paper, we propose several mechanisms for resettlement of refugees in cities. Our main motivation is to find a mechanism that improves population distribution imbalances.

Chapter 2

MODEL

Let $\mathcal{C} = \{c_1, c_2, .., c_n\}$ be a finite set of cities and for each city c $\in \mathcal{C}$, I_c the set of agents residing in c. Let $\mathcal{I} = \bigcup_{c\in\mathcal{C}} I_c$. Each agent $i \in \mathcal{I}$ has (strict) preferences over $\mathcal{C}$. Let $R = (R_i)_{i\in\mathcal{I}}$ denote a preference profile. Each city c_l has a quota q_l. Let $q = (q_l)_{l=1}^{l=n}$. A **refugee allocation problem** is a list $\langle \mathcal{C}, (I_c)_{c\in\mathcal{C}}, q, R\rangle$. Since $\mathcal{C}, (I_c)_{c\in C}$ and q are all fixed, we can identify a problem by just referring to the preference profile of agents, R.

Given a problem R, a **matching** is a mapping $\mu : \mathcal{I} \to \mathcal{C}$. Note that a problem itself identifies a matching. We refer this initial matching as the **status-quo** and denote it by μ_0. Let $\mathcal{M}$ be the set of all matchings and let $\mathcal{R}$ be the set of all preference profiles. A **mechanism** $\rho : \mathcal{R} \to \mathcal{M}$ is a step by step procedure that assigns a matching for each problem. A city c is **overpopulated** if $|I_c| > q_c$, **underpopulated** if $|I_c| < q_c$, and **balanced** if $|I_c| = q_c$. The set of agents in city c_l after Step k is denoted by $I^k_{c_l}$. At Step $k+1$, a city $c_l \in \mathcal{C}$ is **overpopulated at Step k+1** if $|I^k_{c_l}| > q_c$. A city $c_l \in \mathcal{C}$ is **underpopulated at Step k+1** if $|I^k_{c_l}| < q_c$. A city $c_l \in \mathcal{C}$ is **balanced at Step k+1** if $|I^k_{c_l}| = q_c$. A city c_l is an **overpopulated city with room at Step k+1** if c_l is an *overpopulated* city so that $|I^k_{c_l}| < |I_c|$. A city c_l is an **balanced city with room at Step k+1** if c_l is a *balanced* city so that $|I^k_{c_l}| < |I_c|$. In general, there is no ambiguity about the step we consider, so we generally refer an *overpopulated, underpopulated, and balanced city at a particular step* just as *currently overpopula currently underpopulated*, and *currently balanced*.

Chapter 3

CHAIN MECHANISMS

When each city points to a unique agent and any agent points to her most preferred city, a set of cycles and chains occurs. The existing mechanisms for similar problems rely on executions of those cycles and chains. In the refugee allocation problem, only the execution of a chain decreases the number of refugees in an overpopulated city and increases the number of refugees in an underpopulated city. But, execution of a cycle does not change the population of cities. While mechanisms such as Top Trading Cycles and Chains mechanism[1] for similar problems precede execution of cycles, we propose a mechanism that prioritizes execution of chains over execution of cycles in order to improve population distribution imbalances across cities.

A chain-prioritized mechanism:

Let f be a strict priority ordering of cities and for each city $c_l \in \mathcal{C}$, let g_l be a strict priority ordering of agents in I_{c_l}.

At the beginning of the mechanism, we recursively define the set A_0 as follows:

- $c \in A_0^1$ if either for each $i \in I_c$, for each $c' \neq c$, $c\mathcal{P}_i c'$ and c is either *overpopulated* or *balanced*, or for each $i \in I_c$, for each $c' \neq c$, $c\mathcal{P}_i c'$, and for all $i \in \mathcal{I}$ agent i prefers her own city to c, and c is *underpopulated*.
- For each $k \geq 2$, $c \in A_0^k$ if c is not *underpopulated* and for each i $\in I_c$, $c'\mathcal{P}_i c$, implies $c' \in A_0^l$ for some $l < k$.
- $A_0 = \cup_{k=1}^{\infty} A_0^k$

[1] Roth, A., Sonmez, T., and Unver, U.(2004)."Kidney Exchange", Quarterly Journal of Economics

Only the cities that are not in A_0 are included in the algorithm. Each city points to highest priority agent that currently resides in that city. We call an agent pointed by a city as an *active agent* and each active agent points to her most preferred available city. At the end of Step n, we define the set A_n as above.

- $c \in A_n^1$ if either for each $i \in I_c$, for each $c' \neq c$, $c\mathcal{P}_i c'$ and c is either *currently overpopulated* or *currently balanced*, or for each $i \in I_c$, for each $c' \neq c$, $c\mathcal{P}_i c'$, and for each $i \in \mathcal{I}$ agent i prefers her own city to c, and c is *currently underpopulated*.
- For each $k \geq 2$, $c \in A_n^k$ if c is not *currently underpopulated* and for each i $\in I_c^n$, $c'\mathcal{P}_i c$, implies $c' \in A_n^l$ for some $l < k$.
- $A_n = \cup_{k=1}^{\infty} A_n^k$

A **cycle** $(c_1', i_1', c_2', i_2', ..., c_m', i_m')$ is an ordered list of cities and agents such that city c_1' points to agent i_1', agent i_1' points to c_2',$\cdots$, city c_m' points to agent i_m' and agent i_m' points to city c_1'. A **chain** $(c_1', i_1', c_2', i_2', ..., c_{m-1}', i_{m-1}', c_m')$, is an ordered list of cities and agents such that the city c_1' points to agent i_1', agent i_1' points to c_2',$\cdots$, agent i_{m-1}' points to city c_m' where c_m' is either an *underpopulated city*, or an *overpopulated city with room*, or a *balanced city with room*. Given a chain $(c_1', i_1', c_2', i_2', ..., c_{m-1}', i_{m-1}', c_m')$, c_1' is the *tail* or *last city* of the chain, and c_m' is the *head* or *first city* of the chain. The **execution** of a cycle or a chain is moving all the agents in that cycle or chain to the city they point. If chain $(c_1', i_1', c_2', i_2', ..., c_{m-1}', i_{m-1}', c_m')$ is executed, then we say that agent i_1' is the agent who moves *last* at the execution and agent i_{m-1}' is the agent who moves *first* at the execution.

Chain Selection Rules

1. **Minimal Chain Selection Rule:** Given the relevant priority order at a given step, choose the highest priority city among all cities that are tail of some chain and execute the minimal (shortest) chain ending with that city.
2. **Maximal Chain Selection Rule:** Given the relevant priority order at a given step, choose the highest priority city among all cities that are tail of some chain and execute the maximal (longest) chain ending with that city.

Generalized Top Trading Chains and Cycles Mechanism (gTTCh)

Step 0: Choose and fix a chain selection rule.

Step 1 Define f_1 as follows: For any two cities c and c', if both cities are *overpopulated* cf_1c' if and only if cfc'; if both cities are *underpopulated* or *balanced* again cf_1c' if and only if cfc'; if c is *overpopulated* and c' is *underpopulated* or *balanced*, cf_1c'; and if c is *underpopulated* or *balanced* and c' is *overpopulated*, $c'f_1c$. f_1 is the priority ordering among cities at Step 1.

- If there are cycles that only contains *overpopulated* or *balanced* cities, then select these cycles and execute them. Remove the cities in A_1 and agents currently assigned to them. Update the set of active agents.
  - If all cities are removed, the algorithm terminates.
  - Otherwise, proceed to Step 2
- If there is no cycle of *overpopulated* or *balanced* cities;
  - Execute the appropriate chain using the chain selection rule. Remove the cities in A_1 and agents assigned currently assigned to them. Update the set of active agents.
    - If all cities are removed, the algorithm terminates.
    - Otherwise, proceed to step 2

Step k

Define f_k as follows: For any two cities c and c', if both cities are *currently overpopulated* at Step k, cf_kc' if and only if cfc'; if both cities are *currently underpopulated* or *currently balanced*, again cf_kc' if and only if cfc'; if c is *currently overpopulated* and c' is *currently underpopulated* or *currently balanced*, cf_kc'; and if c is *currently underpopulate* or *currently balanced* and c' is *currently overpopulated*, $c'f_kc$. f_k is the priority ordering among cities at Step k.

- If there are cycles that only contains *currently overpopulated* or *currently balanced* cities, then select these cycles and execute them. Remove the cities in A_k

and agents currently assigned to them. Update the set of active agents.

- If all cities are removed, the algorithm terminates.
- Otherwise, proceed to Step $k+1$

- If there is no cycle that only contains *currently overpopulated* or *currently balanced* cities;
  - Execute the appropriate chain using the chain selection rule. Remove the cities in A_k and agents currently assigned to them. Update the set of active agents.
    - If all cities are removed, the algorithm terminates.
    - Otherwise, proceed to step $k+1$

Lemma 1. *Let (R_i, R_{-i}) and (R'_i, R_{-i}) be two problems and fix the priority orderings among agents and cities. If agent i moves only in one of the problems, set Step k as the step that agent i moves in that problem. If agent i moves in both problems, let m and n be the steps that agent i moves to another city in the algorithm for (R_i, R_{-i}) and (R'_i, R_{-i}), respectively, and set $k = \min\{m, n\}$. In the gTTChC algorithm, the first $k-1$ executions, priority orderings f_1,...,f_{k+1}, and the sets A_0,..., A_{k-1} are the same for both (R_i, R_{-i}) and (R'_i, R_{-i}).*

Proof. Note that for a given problem $\langle \mathcal{C}, (I_c)_{c\in\mathcal{C}}, q, R\rangle$, the second step of the algorithm is equivalent to the first step of the algorithm for the (reduced) problem $\langle \mathcal{C} \setminus A_1, (I_{c_{l_1}})_{c_l \in \mathcal{C}\setminus A_1}, q, R'\rangle$ where R' is the preference profile of the agents in $\bigcup_{c_l \in \mathcal{C}\setminus A_1}(c_{l_1})$ over $\mathcal{C} \setminus A_1$ and assuming the priority ordering among cities, f, and priority ordering among agents within a city are same in these two problems. Thus, it is enough to prove the result only for the first step of two arbitrary problems (R_i, R_{-i}) and (R'_i, R_{-i}) so that $k > 1$ (for case $k = 1$ the result trivially holds.). Let $k > 1$. Suppose the execution(s) in the first step of algorithm are different for these two problems. There are two cases:

1. Cycle(s) that contains only *overpopulated* or *balanced* cities are executed in

the first step of the $gTTCh$ for (R'_i, R_{-i}).

- A chain is executed in the first step of the $gTTCh$ for (R_i, R_{-i}): This implies there is no cycle that contains only *overpopulated* or *balanced* cities at the beginning of the algorithm. If any executed cycle in the first step of the $gTTCh$ for (R'_i, R_{-i}) doesn't contain agent i, in the $gTTCh$ for (R_i, R_{-i}), any agent/city in that cycle points to the same city/agent and this cycle would also be formed in the first step of the $gTTCh$ for (R_i, R_{-i}). But, as we explained, this is impossible. Thus, the cycle executed in the first step of the $gTTCh$ for (R'_i, R_{-i}) contains agent i. But this contradicts with $k > 1$.

- Cycle(s) that contains only *overpopulated* or *balanced* cities are executed in the first step of the $gTTCh$ for (R_i, R_{-i}): There is a cycle that contains only *overpopulated* or *balanced* cities so that it is formed in the first step of the $gTTCh$ for one of the problems, but not in the other one. But, such a cycle contains agent i since otherwise any agent/city in that cycle would point to same city/agent in both of the problems and same cycle would be formed in both problems. This means that agent i moves in the first step of the $gTTCh$ for one of the problems, which is impossible as $k > 1$.

2. A chain is executed in the first step of the $gTTCh$ for (R'_i, R_{-i}):

   - A chain is executed in first step of the $gTTCh$ for (R_i, R_{-i}): This implies that the two executed chains are different in two problems. If any of these chains contains agent i then we obtain a contradiction with $k > 1$. Suppose this is not the case and consider the chain that is executed in the first step of the $gTTCh$ for (R'_i, R_{-i}). This chain is also exists in the same step of the mechanism for other problem since any agent/city in this chain points to the same city/agent in the first step of the $gTTCh$ for other problem. Similarly, the chain executed in the first step of the $gTTCh$ for (R_i, R_{-i}) also exists in the first step of the

$gTTCh$ for (R'_i, R_{-i}). But, since the executed chains are different, this leads to a contradiction as the chain selection rule would consistently select the same chain among these two chains when both of them are present.

- Cycle(s) $gTTCh$ cities are executed in the first step of the $gTTCh$ for (R_i, R_{-i}): Any cycle $gTTCh$ such that it is formed in the first step of the $gTTCh$ for (R_i, R_{-i}) and doesn't contain agent i would also be formed in the first step of the $gTTCh$ for (R'_i, R_{-i}). Thus, we deduce that a cycle that contains agent i is executed in the first step of (R_i, R_{-i}), which contradicts with $k > 1$.

Thus, the execution(s) in the first step of the gTTChC are the same for these two problems. Thus, f_2 and the set A_1 are also same in both problems. ■

Theorem 1. *The gTTCh with the minimal chain selection rule is strategyproof.*

Proof. Suppose that the $gTTCh$ using the minimal chain selection rule is not strategyproof. Then, there exists an agent, say agent i, such that if she submits her preferences as R'_i instead of her true preferences R_i whereas the other refugees preferences, R_{-i}, remain the same she settles in a better city. First of all, observe that agent i has to move to another city in the $gTTCh$ for (R'_i, R_{-i}). By the following argument, this observation implies that agent i moves also in the $gTTCh$ for (R'_i, R_{-i}). By Lemma 1, in both problems, all the available cities and the priority orderings are the same at the step that agent i moves in the $gTTCh$ for (R'_i, R_{-i}), say Step k. If agent i doesn't move to any city in the $gTTCh$ for (R_i, R_{-i}), then at Step k, she is pointing to another city since there is at least one available city and agent i prefers this city to her city. If, for (R'_i, R_{-i}), a cycle that contains only *currently overpopulated* or *currently balanced* cities is executed at the k'th step of the $gTTCh$, then at Step k of the $gTTCh$ for (R_i, R_{-i}), there exists a sequence of *overpopulated* or *balanced* cities and agents, which points to each other, starting with the city that agent i is placed in the $gTTCh$ for (R'_i, R_{-i}), and

ending at agent i. This sequence exists in the following steps of algorithm until agent i is placed. Thus, agent i should be placed to the same or to a better city as a result of the $gTTCh$ also in problem (R_i, R_{-i}), which is a contradiction. If, on the other hand, a chain is executed at the k'th step of the $gTTCh$ for (R'_i, R_{-i}), then the part of this chain up to agent i also exists in the k'th step of the $gTTCh$ for (R_i, R_{-i}), and also agent i points to some city. Thus, agent i is either contained in a cycle that contains only *currently overpopulated* or *currently balanced* cities or in a chain at Step k of the $gTTChC$ for (R_i, R_{-i}). If agent i contained in a cycle that contains only *currently overpopulated* or *currently balanced* cities, then she moves in Step k. If agent i is contained in a chain, then there is no cycle that contains only *currently overpopulated* or *currently balanced* cities at Step k of the $gTTCh$ for (R_i, R_{-i}), since otherwise that cycle would also be formed in the same step of the $gTTChC$ for (R'_i, R_{-i}). Furthermore, the minimal chain that contains agent i and all the chains that doesn't contain agent i are the same at Step k of the $gTTCh$ for both problems. This implies a contradiction as the chain selection rule identifies the same chain on both problems, and so agent i moves in both problems. Thus, we can consider only the cases that agent i moves in both problems.

Let m and n be the steps that agent i is placed in (R_i, R_{-i}) and (R'_i, R_{-i}) respectively and set $k = min\{m, n\}$. By Lemma 1, $k = n < m$, since otherwise, in (R_i, R_{-i}), after the first $k-1$ steps, agent i moves to her best available city, which contradicts with our assumption. For (R_i, R_{-i}), there are three cases:

Agent i is contained in a cycle that contains only *currently overpopulated* or *currently balanced* cities after first $k-1$ steps of the $gTTCh$ for (R_i, R_{-i}): After the first $k-1$ steps, agent i points to her most preferred city in (R_i, R_{-i}), and the cycle that she is contained is executed at Step k of the $gTTCH$ for (R_i, R_{-i}). This contradicts with $k < m$.

Agent i is related with (but not contained in) a cycle that contains only

***currently overpopulated* or *currently balanced* cities after first $k-1$ steps of the *gTTCh* for (R_i, R_{-i}):** By Lemma 1, the cycle that agent i is related exists in both problems after the first $k-1$ steps of the *gTTCh*. Thus, agent i should be contained in a cycle that contains only *currently overpopulated* or *currently balanced* cities after Step $k-1$ of the *gTTCh* for (R'_i, R_{-i}). This is only possible if in (R_i, R_{-i}), after first $k-1$ steps of the *gTTChC*, there exists a sequence of *currently overpopulated* or *currently balanced* cities and agents that, each city in that sequence points to the active agent of the next city, it starts with the city that agent i is placed in the *gTTCh* for (R'_i, R_{-i}) and ends with agent i. Again by Lemma 1, this sequence also exists in (R_i, R_{-i}) after the first $k-1$ executions. Note that any active agent will remain active in the cities of this sequence and thus, this sequence will persist in (R_i, R_{-i}) unless agent i is placed in the *gTTCh*. Thus, in the following steps of the *gTTCh* for (R_i, R_{-i}), agent i can always form a cycle that contains only *currently overpopulated* or *currently balanced* cities cities by pointing to the first city in this sequence. But, since agent i is placed to another city in the *gTTCh* for (R_i, R_{-i}), agent i prefers this city to the city that agent i is placed in the *gTTCh* for problem (R'_i, R_{-i}) with respect to her true preferences. This is a contradiction.

Agent i is contained in a chain after first $k-1$ steps of the *gTTCh* for (R_i, R_{-i}): There are two subcases:

- There exists a cycle that contains only *currently overpopulated* or *currently balanced* cities which doesn't contain agent i after the first $k-1$ steps of the *gTTCh* for (R_i, R_{-i}): By Lemma 1, this cycle also exists after the first $k-1$ steps of the *gTTCh* for (R'_i, R_{-i}). But, since agent i is placed at k'th step of the *gTTCh* for (R'_i, R_{-i}), this requires the existence of a different cycle that contains only *currently overpopulated* or *currently balanced* cities which contains the city of agent i in the k'th step of the *gTTCh* for (R'_i, R_{-i}). As in the previous case, this implies a sequence which contains only *currently overpopulated* or *currently balanced* cities so that will persist until agent i settled in the *gTTCh* for (R_i, R_{-i}). Thus, the outcome of the *gTTChC* for (R_i, R_{-i}) is at

least as good as the outcome of the $gTTCh$ for (R_i', R_{-i}). This is a contradiction.

- After the first $k-1$ steps in the $gTTCh$ for (R_i, R_{-i}), there is another chain that doesn't contain agent i and is executed at Step k: By Lemma 1, this chain is also exists in Step k of the $gTTCh$ for (R_i', R_{-i}). If this chain is executed at the k'th step of the $gTTCh$ for (R_i', R_{-i}) we get a contradiction with $k = n$. If not, then either a cycle that contains only *currently overpopulated* or *currently balanced* cities which contains agent i or another chain that contains agent i is executed at Step k of the $gTTCh$ for (R_i', R_{-i}). The former leads to a contradiction as in the previous cases. If the latter is true, then denote the cities that are the tails of executed chains at Step k of the $gTTCh$ for (R_i, R_{-i}) and (R_i', R_{-i}) by c_{l_1} and c_{l_2}, respectively. Note that both chains exist at Step k of the $gTTCh$ for (R_i', R_{-i}) and since the chain starting with c_{l_2} is executed, either we have $c_{l_2} f_k c_{l_1}$ or these cities are the same. Indeed, these cities cannot be the same since in this case, the fact that minimal chain starting with c_{l_2} contains agent i would imply that the part of this chain until the city of agent i contains no *underpopulated* city or *overpopulated (or balanced) city with room*. But, Lemma 1 implies that this part of the chain also exists at the k'th step of the $gTTCh$ for (R_i, R_{-i}), which implies that any chain starting with c_{l_1} in that problem contains agent i. This is impossible since the minimal chain starting c_{l_1} at k'th step of the $gTTCh$ for (R_i, R_{-i}) doesn't contain agent i. Thus, we conclude $c_{l_2} f_k c_{l_1}$.
  The part of the chain that is executed at the k'th step of the $gTTCh$ for (R_i', R_{-i}) up to agent i also exists at the k'th step of the $gTTCh$ for (R_i, R_{-i}), and agent i points to some city. But then, either there exists a chain starting with c_{l_2} or agent i is contained in or related with a cycle that contains only *currently overpopulated* or *currently balanced* cities at Step k of the $gTTCh$ for (R_i, R_{-i}). Both imply to a contradiction since in this step of the $gTTCh$ for (R_i, R_{-i}), the chain starting with c_{l_1} is executed.

Thus, the $gTTCh$ with the minimal chain selection rule is strategyproof. ■

Chapter 4

CYCLE MECHANISMS

Generalized Top Trading Cycles and Chains Mechanism (gTTC)

Step 0: Choose and fix a chain selection rule.

Step 1 Define f_1 as follows: For any two cities c and c', if both cities are *overpopulated* cf_1c' if and only if cfc'; if both cities are *underpopulated* or *balanced* again cf_1c' if and only if cfc'; if c is *overpopulated* and c' is *underpopulated* or *balanced*, cf_1c'; and if c is *underpopulated* or *balanced* and c' is *overpopulated*, $c'f_1c$. f_1 is the priority ordering among cities at Step 1.

- If there are any cycles, execute them. Remove the cities in A_1 and agents currently assigned to them. Update the set of active agents.
  - If all cities are removed, the algorithm terminates.
  - Otherwise, proceed to Step 2
- If there is no cycle;
  - Execute the appropriate chain using the chain selection rule. Remove the cities in A_1 and agents currently assigned to them. Update the set of active agents.
    - If all cities are removed, the algorithm terminates.
    - Otherwise, proceed to step 2

Step k

Define f_k as follows: For any two cities c and c', if both cities are *currently overpopulated* at Step k, cf_kc' if and only if cfc'; if both cities are *currently underpopulated* or *currently balanced*, again cf_kc' if and only if cfc'; if c is *currently overpopulated*

and c' is *currently underpopulated* or *currently balanced*, cf_kc'; and if c is *currently underpopulate* or *currently balanced* and c' is *currently overpopulated*, $c'f_kc$. f_k is the priority ordering among cities at Step k.

- If there are any cycles, execute them. Remove the cities in A_k and agents currently assigned to them. Update the set of active agents.
  - If all cities are removed, the algorithm terminates.
  - Otherwise, proceed to Step $k+1$
- If there is no cycle;
  - Execute the appropriate chain using the chain selection rule. Remove the cities in A_k and agents currently assigned to them. Update the set of active agents.
    - If all cities are removed, the algorithm terminates.
    - Otherwise, proceed to Step $k+1$

Chapter 5

AN IMPOSSIBILITY RESULT

In this chapter, first we compare the performance of the two mechanisms on reducing the concentration of refugees in different problems. We see that which mechanism performs better depends on the particular problem. Second, we see that this result doesn't change even under a restrictive assumption on agents' preferences. Lastly, we show that there is no strategyproof and Pareto Optimal mechanism that always perform weakly better than the two mechanisms we defined. We begin the chapter by defining some necessary concepts.

A matching μ **improves distribution** if $\sum_{c\in\mathcal{C}} \max\{\min\{|I_c|-q_c, |I_c|-|\mu(c)|\}, 0\} \geq 0$. We say that μ is **weakly better** than μ' if $\sum_{c\in\mathcal{C}} \max\{\min\{|I_c| - q_c, |I_c| - |\mu(c)|\}, 0\} \geq \sum_{c\in\mathcal{C}} \max\{\min\{|I_c|-q_c, |I_c|-|\mu'(c)|\}, 0\}$. We call the sum $\sum_{c\in\mathcal{C}} \max\{\min\{|I_c|-q_c, |I_c| - |\mu(c)|\}, 0\}$ as the **improvement number** of μ. Lastly, mechanism ρ is **weakly better** than mechanism v if for each problem the resulting matching of the mechanism ρ is weakly better than the resulting matching of the mechanism v for every problem.

Example 1.

$\mathcal{C} = \{c_1, c_2, c_3, c_4, c_5, c_6\}$. There are two agents in each city, and for each $j \in \{1, 2, 3, 4, 5, 6\}$, $I_{c_j} = \{i_j^1, i_j^2\}$. For each $j \in \{1, 2, 3, 5, 6\}$ $q_j = 1$, and $q_4 = 3$. For each c_j, the i_j^1 has higher priority than i_j^2. The priority ordering among cities is $c_1 f c_2 f c_3 f c_5 f c_6 f c_4$. The preferences of agents are as follows:

$i_1^1 : c_2 \succ c_1 \succ c_3 \succ c_4 \succ c_5 \succ c_6$

$i_1^2 : c_1 \succ c_2 \succ c_3 \succ c_4 \succ c_5 \succ c_6$

$i_2^1 : c_3 \succ c_2 \succ c_1 \succ c_4 \succ c_5 \succ c_6$

$i_2^2 : c_2 \succ c_3 \succ c_1 \succ c_4 \succ c_5 \succ c_6$

$i_3^1 : c_4 \succ c_3 \succ c_1 \succ c_2 \succ c_5 \succ c_6$

$i_3^2 : c_3 \succ c_4 \succ c_1 \succ c_2 \succ c_5 \succ c_6$

$i_4^1 : c_5 \succ c_4 \succ c_1 \succ c_2 \succ c_3 \succ c_6$

$i_4^2 : c_4 \succ c_5 \succ c_1 \succ c_2 \succ c_3 \succ c_6$

$i_5^1 : c_6 \succ c_5 \succ c_1 \succ c_2 \succ c_3 \succ c_4$

$i_5^2 : c_5 \succ c_6 \succ c_1 \succ c_2 \succ c_3 \succ c_4$

$i_6^1 : c_3 \succ c_6 \succ c_1 \succ c_2 \succ c_4 \succ c_5$

$i_6^2 : c_6 \succ c_3 \succ c_1 \succ c_2 \succ c_4 \succ c_5$

Generalized Top Trading Cycles and Chains Mechanism (gTTC) on Example 1:

Step 0 $A_0 = \emptyset$ and use the minimal chain selection rule.

Step 1 The only cycle is $(c_3, i_3^1, c_4, i_4^1, c_5, i_5^1, c_6, i_6^1)$. This cycle is executed at Step 1. Then, $A_1 = \{c_1, c_2, c_3, c_4, c_5, c_6\}$.

All cities are removed and the algorithm terminates.

Generalized Top Trading Chains and Cycles Mechanism (gTTCh) on Example 1:

Step 0 $A_0 = \emptyset$ and use the minimal chain selection rule.

Step 1 There is a chain starting with the highest priority city: $(c_1, i_1^1, c_2, i_2^1, c_3, i_3^1, c_4)$. It is also the minimal chain starting with c_1. This chain is executed at Step 1. Then, $A_1 = \{c_1, c_2, c_3, c_4, c_5, c_6\}$.

All cities are removed and the algorithm terminates.

In terms of reducing population distribution imbalances, the **gTTCh** performs better than the **gTTC** in Example 1: The populations of the cities remained unchanged at the resulting matching of the **gTTC** whereas the population of c_1 has reduced to its allowed level and population of c_4 has increased by one at the resulting matching of the **gTTCh**. We next present an example in which the **gTTC** performs better.

Example 2.

$\mathcal{C} = (c_1, c_2, c_3, c_4, c_5, c_6)$. There are two agents in each city, and for each $j \in \{1, 2, 3, 4, 5, 6\}$ $I_{c_j} = \{i_j^1, i_j^2\}$. For each $j \in \{2, 3, 5, 6\}$ $q_j = 1$, $q_1 = 3$ and $q_4 = 3$.

For each city c_j i_j^1 has higher priority than i_j^2. The priority ordering among cities is $c_2 f c_3 f c_5 f c_6 f c_4 f c_1$. The preferences of agents are as follows:

$i_1^1 : c_2 \succ c_1 \succ c_3 \succ c_4 \succ c_5 \succ c_6$

$i_1^2 : c_1 \succ c_2 \succ c_3 \succ c_4 \succ c_5 \succ c_6$

$i_2^1 : c_3 \succ c_2 \succ c_1 \succ c_4 \succ c_5 \succ c_6$

$i_2^2 : c_2 \succ c_3 \succ c_1 \succ c_4 \succ c_5 \succ c_6$

$i_3^1 : c_4 \succ c_3 \succ c_1 \succ c_2 \succ c_5 \succ c_6$

$i_3^2 : c_3 \succ c_4 \succ c_1 \succ c_2 \succ c_5 \succ c_6$

$i_4^1 : c_5 \succ c_4 \succ c_1 \succ c_2 \succ c_3 \succ c_6$

$i_4^2 : c_4 \succ c_5 \succ c_1 \succ c_2 \succ c_3 \succ c_6$

$i_5^1 : c_6 \succ c_5 \succ c_1 \succ c_2 \succ c_3 \succ c_4$

$i_5^2 : c_4 \succ c_5 \succ c_6 \succ c_1 \succ c_3 \succ c_2$

$i_6^1 : c_3 \succ c_6 \succ c_1 \succ c_2 \succ c_4 \succ c_5$

$i_6^2 : c_6 \succ c_3 \succ c_1 \succ c_2 \succ c_4 \succ c_5$

Generalized Top Trading Cycles and Chains Mechanism (gTTC) on Example 2:

Step 0 $A_0 = \emptyset$ and use the minimal chain selection rule.

Step 1 The only cycle is $(c_3, i_3^1, c_4, i_4^1, c_5, i_5^1, c_6, i_6^1)$. This cycle is executed at Step 1. Then, $A_1 = \{c_1, c_2, c_3, c_6\}$.

Step 2 There are no cycles and the only chain is (c_5, i_5^2, c_4). This chain is executed in Step 2. Then $A_2 = \{c_1, c_2, c_3, c_4, c_5, c_6\}$

All cities are removed and the algorithm terminates.

Generalized Top Trading Chains and Cycles Mechanism (gTTCh) on Example 2:

Step 0 $A_0 = \emptyset$ and use the minimal chain selection rule.

Step 1 There is a chain starting with the highest priority city, $(c_2, i_2^1, c_3, i_3^1, c_4)$. It is also the minimal chain starting with c_2. This chain is executed at Step 1. Then, $A_1 = \{c_3, c_4, c_5, c_6\}$, and c_4 becomes an overpopulated city and c_2 becomes an overpopulated city with room.

Step 2 The only chain is (c_1, i_1^1, c_2). This chain is executed at Step 2. Then, $A_2 = \{c_1, c_2, c_3, c_4, c_5, c_6\}$.

All cities are removed and the algorithm terminates.

In terms of reducing population distribution imbalances the **gTTC** performs better in this example: Population at the overpopulated cities has remained unchanged at the resulting matching of the **gTTCh**, only the population of c_1 has decreased by one and the population of c_4 has increased by one. On the other hand, at the resulting matching of the **gTTC** mechanism, the population of the overpopulated city c_5 has reduced to its allowed level and population of underpopulated city c_4 has increased by one.

These examples demonstrate that the success of the mechanism on improving population distribution imbalances depends not only the prioritized executions but also the preferences of the lower priority agents in the cities and preferences of agents who not move at the step of execution. At a given step, improving the population distribution imbalances is only possible with the execution of a chain. But, executing a chain instead of a cycle might also be harmful. Even if the execution of a chain improves the distribution at first, this effect might be eliniminated in the following steps as this execution by allowing for an agent to move from an underpopulated city to an overpopulated city. Moreover, by increasing the population of an underpopulated city, the execution of this chain instead of a cycle may prevent the execution of another chain as in Example 2. Under the following assumption, this is not a concern since the execution of a chain cannot prevent execution of another chain by increasing the population of an underpopulated city.

Assumption 1. *Let $\langle \mathcal{C}, (I_c)_{c\in\mathcal{C}}, q, R\rangle$ be a problem, for any underpopulated city c, the number of agents who prefer city c to their own city, is less than $q_c - |I_c|$.*

Although the aforementioned concern is eliminated, the **gTTC** mechanism is still can still perform better than the **gTTCh** mechanism in some problems. Example 3 below illustrates such a case. Furthermore, there is no strategyproof, individually rational and Pareto efficient mechanism that is weakly better than both **gTTC** and **gTTCh**. We conclude our discussion by proving this result in Theorem 2.

Example 3.

$\mathcal{C} = \{c_1, c_2, c_3, c_4\}$. There is only one agent in each city, and for every city c_j we denote the agent in that city as i_j. $q_1 = q_2 = 1$, and $q_3 = q_4 = 3$. The priority ordering among cities is $c_1 f c_2 f c_3 f c_4$. The preferences of agents are as follows:

$i_1 : c_2 \succ c_3 \succ c_1 \succ c_4$

$i_2 : c_4 \succ c_2 \succ c_1 \succ c_3$

$i_3 : c_3 \succ c_1 \succ c_2 \succ c_4$

$i_4 : c_2 \succ c_1 \succ c_4 \succ c_3$

Generalized Top Trading Cycles and Chains Mechanism (gTTC) on Example 3:

Step 0 $A_0 = \emptyset$ and use the minimal chain selection rule.

Step 1 The only cycle is (c_2, i_2, c_4, i_4). This cycle is executed at Step 1. Then, $A_1 = \{c_2, c_4\}$.

Step 2 The agent who wants to move is i_1, The chain (c_1, i_1, c_3) s executed. Then, $A_2 = \{c_1, c_2, c_3, c_4\}$

All cities are removed and the algorithm terminates.

Generalized Top Trading Chains and Cycles Mechanism (gTTCh) on Example 3:

Step 0 $A_0 = \emptyset$ and use the minimal chain selection rule.

Step 1 The minimal chain starting with the highest priority city is $(c_1, i_1, c_2, i_2, c_4$. This chain is executed. Then, $A_1 = \{c_2, c_3\}$, and c_1 becomes an underpopulated city with room.

Step 2 The only agent who wants to move is i_4. Thus, the chain (c_4, i_4, c_1) is executed. Then, $A_2 = \{c_1, c_2, c_3, c_4\}$.

All cities are removed and the algorithm terminates.

In Example 3, for each underpopulated city c, there is only one agent who prefers city c to their own city. Furthermore, in both of underpopulated cities, there are room for two more agents. Therefore, Example 3 satisfies the Assumption 1. However, in the Cycle-mechanism agent i_1 moves to c_3 and the improvement number is 1, whereas in the Chain-mechanism the improvement number is 0.**gTTC** performs better than **gTTCh**.

Now we will present the proof for the impossibility of existence of an individually rational, Pareto efficient and strategyproof mechanism that is weakly better than both **gTTC** and **gTTCh**. This impossibility re is also true for problems satisfying Assumption 1

Theorem 2. *There is no individually rational, Pareto efficient and strategyproof mechanism that is weakly better than both* ***gTTC*** *and* ***gTTCh****, even under Assumption 1.*

Proof. Assume that there exists an individually rational, Pareto efficient and strategyproof mechanism ϕ such that $\phi \succeq$ **gTTC** and $\phi \succeq$ **gTTCh**. Let $\mathcal{C} = \{c_1, c_2, c_3, c_4\}$ with quota levels $q_1 = q_2 = 1$, and $q_3 = q_4 = 3$. We will consider three different problems that differ only by preferences of the agents. In all of the problems, there is only one agent in each city. We denote the agent in city c_j in problem k as i_j^k. The three problems can therefore be characterized by the preferences of agents, R_1, R_2, and R_3, which we define as follows:

R_1	R_2	R_3
$i_1^1 : c_2 \succ c_3 \succ c_1 \succ c_4$	$i_1^2 : c_2 \succ c_1 \succ c_3 \succ c_4$	$i_1^3 : c_2 \succ c_1 \succ c_3 \succ c_4$
$i_2^1 : c_4 \succ c_2 \succ c_1 \succ c_3$	$i_2^2 : c_4 \succ c_2 \succ c_1 \succ c_3$	$i_2^3 : c_4 \succ c_2 \succ c_1 \succ c_3$
$i_3^1 : c_3 \succ c_1 \succ c_2 \succ c_4$	$i_3^2 : c_3 \succ c_1 \succ c_2 \succ c_4$	$i_3^3 : c_3 \succ c_1 \succ c_2 \succ c_4$
$i_4^1 : c_2 \succ c_1 \succ c_4 \succ c_3$	$i_4^2 : c_2 \succ c_1 \succ c_4 \succ c_3$	$i_4^3 : c_2 \succ c_4 \succ c_1 \succ c_3$

Note that in all of these problems, Assumption 1 is satisfied. Now, let's apply **gTTC** and **gTTCh** on these problems:

Generalized Top Trading Cycles and Chains Mechanism (gTTC) on R_1 :

Step 0 $A_0 = \emptyset$ and use the minimal chain selection rule.

Step 1 The only cycle is (c_2, i_2^1, c_4, i_4^1). This cycle is executed at Step 1. Then, $A_1 = \{c_2, c_4\}$.

Step 2 The agent who wants to move is i_1^1, The chain (c_1, i_1^1, c_3) s executed. Then,

$A_2 = \{c_1, c_2, c_3, c_4\}$

All cities are removed and the algorithm terminates.

Generalized Top Trading Chains and Cycles Mechanism (gTTCh) on R_1 :

Step 0 $A_0 = \emptyset$ and use the minimal chain selection rule.

Step 1 The minimal chain starting with the highest priority city is $(c_1, i_1^1, c_2, i_2^1, c_4$. This chain is executed. Then, $A_1 = \{c_2, c_3\}$, and c_1 becomes an underpopulated city with room.

Step 2 The only agent who wants to move is i_4^1. Thus, the chain (c_4, i_4^1, c_1) is executed. Then, $A_2 = \{c_1, c_2, c_3, c_4\}$.

All cities are removed and the algorithm terminates.

Note that in R_1, **gTTC** has an improvement number of 1, whereas **gTTCh** has an improvement number of 0.

Generalized Top Trading Cycles and Chains Mechanism (gTTC) on R_2 :

Step 0 $A_0 = c_3$ and use the minimal chain selection rule.

Step 1 The only cycle is (c_2, i_2^2, c_4, i_4^2). This cycle is executed at Step 1. Then, $A_1 = \{c_1, c_2, c_3, c_4\}$. All cities are removed and the algorithm terminates.

Generalized Top Trading Chains and Cycles Mechanism (gTTCh) on R_2 :

Step 0 $A_0 = \{c_3\}$ and use the minimal chain selection rule.

Step 1 The minimal chain starting with the highest priority city is $(c_1, i_1^2, c_2, i_2^2, c_4$. This chain is executed. Then, $A_1 = \{c_2, c_3\}$, and c_1 becomes an underpopulated city with room.

Step 2 The only agent who wants to move is i_4^2. Thus, the chain (c_4, i_4^2, c_1) is executed. Then, $A_2 = \{c_1, c_2, c_3, c_4\}$.

All cities are removed and the algorithm terminates.

Generalized Top Trading Cycles and Chains Mechanism (gTTC) on R_3 :

Step 0 $A_0 = \{c_3\}$ and use the minimal chain selection rule.

Step 1 The only cycle is (c_2, i_2^3, c_4, i_4^3). This cycle is executed at Step 1. Then, $A_1 = \{c_1, c_2, c_3, c_4\}$.

All cities are removed and the algorithm terminates.

Generalized Top Trading Chains and Cycles Mechanism (gTTCh) on R_3 :

Step 0 $A_0 = \{c_3\}$ and use the minimal chain selection rule.

Step 1 The minimal chain starting with the highest priority city is $(c_1, i_1^3, c_2, i_2^3, c_4)$. This chain is executed. Then, $A_1 = \{c_1, c_2, c_3, c_4\}$

All cities are removed and the algorithm terminates.

Thus, in R_3, the improvement number of **gTTCh** is 1.

We know that in R_1, the improvement number of **gTTC** is 1. Furthermore, if ϕ places i_1^1 to city c_2 in R_1, ϕ would have an improvement number that is less than 1, as i_4^1 would move to c_1 due to Pareto efficiency, and overpopulated cities would remain overpopulated. Thus, ϕ places i_1^1 to city c_3.

In R_2, i_1^2 has to stay in city c_1 if ϕ is implemented: Due to individual rationality, the only other option is i_1^2 moves to c_2. This is not possible as it would imply that in R_1, i_1^1 can misrepresent her preferences to move c_2, which contradicts with ϕ being strategyproof. Thus, if ϕ is implemented, only the agents i_2^2 and i_4^2 moves in R_2: Each of them has to be placed to other one's city, as required by Pareto efficiency.

In R_3, i_4^3 has to move to c_2 if ϕ is implemented, since otherwise i_4^3 can misrepresent her preferences to move c_2 as i_4^2 in R_2. Given that i_4^3 moves to c_2 in R_3, i_2^3 moves to c_4 as required by Pareto efficiency. There is no other possibility of move, as only agent who wants to move is i_1^3, but c_2 is overpopulated. Therefore, the improvement number of ϕ in R_3 is 0. However, we know that the improvement number of **gTTCh** is 1 in R_3. Which contradicts with $\phi \succeq$ **gTTCh**.

■

Chapter 6

CONCLUSION

In this paper, we defined two strategyproof and Pareto efficient mechanisms. Although in any step an improvement in the concentration of agents in some cities is only possible via execution of a chain, the mechanism that prioritizes execution of chains doesn't necessarily performs better than the mechanism that prioritizes execution of cycles. This result is persistent even under the Assumption 1 that we assumed on agents' preferences.

Furthermore, with the last theorem, we prove that there is no strategyproof and Pareto efficient mechanism that performs weakly better than any of the two mechanisms we defined. This result implies that among all strategyproof and Pareto efficient mechanisms, there is no particular mechanism that performs better than all other such mechanisms for every problem. It is still an interesting question whether a mechanism can perform better than another mechanism in most of the problems. Our result doesn't rule out such a possibility.

BIBLIOGRAPHY

[1] Culberton, S., and Constant, L.(2015). "Education of Syrian Refugee Children: Managing the crisis in Turkey, Lebanon, and Jordan". *Santa Monica, CA:Rand Corporation*, https://www.rand.org/pubs/research_reports/RR859.html

[2] United Nations High Commissioner for Refugees (2020) "2020 Global Report". https://www.unhcr.org/flagship-reports/globalreport/

[3] United Nations High Commissioner for Refugees (2021)."The 2021 Mid-year Trends Report" https://www.unhcr.org/statistics/unhcrstats/618ae4694/mid-year-trends-2021.html

[4] Roth, Alvin E., Tayfun Sönmez and M. Utku ünver.(2004). "Kidney exchange". *Quarterly Journal of Economics 119(2): 457-488.* https://www.jstor.org/stable/25098691