UTMD Working Paper
The University of Tokyo
Market Design Center

UTMD-094

[Appendix]

Robust Exchange

Yuichiro Kamada
University of California, Berkeley

Yosuke Yasuda
The University of Osaka

September 17, 2025

For Online Publication

Online Appendix

Robust Exchange

Yuichiro Kamada[†] Yosuke Yasuda[‡]

September 17, 2025

This Online Appendix consists of four sections. Appendix C provides a formal proof of Theorem 1. Appendix D presents a modification of the k-greedy algorithm and examines the relationship of its outcome to the notion of k-robust core that we introduce there. Appendix E provides an additional discussion on the k-greedy mechanism. In Appendix F, we consider various alternative specifications for the simulations in Section 4 of the main text. Appendix G discusses potential reasons why we observe the TTC mechanism being used in school choice contexts.

C Proof of Theorem 1

Since the core idea of the proof is explained via the 3-agent example in the main text after the statement of Theorem 1, we did not provide the formal extension of that argument. For completeness, we provide the formal argument here.

Proof of Theorem 1. Consider a subset of agents, denoted by $S \subseteq I$, which consists of $k+1$ agents: $S = \{1, 2, \ldots, k+1\}$. Such a subset can be taken because $k < |I|$. Suppose for a contradiction that there exists a k-robust mechanism that is k-unanimous and strategy-proof. Let ψ be such a mechanism. Suppose that the agents' preferences are as follows.

$$\begin{aligned} &\succ_1 : 2, k+1, 1; \qquad \succ_k : k+1, 1, k; \qquad \succ_{k+1} : 1, 2, k+1; \\ &\succ_l : l+1, l \qquad \text{for all } l \neq 1, k, k+1. \end{aligned}$$

[†]Haas School of Business, University of California Berkeley, 2220 Piedmont Avenue, Berkeley, CA 94720-1900, USA, and University of Tokyo, Faculty of Economics, 7-3-1 Hongo, Bunkyoku, Tokyo, 113-0033, Japan. E-mail: y.cam.24@gmail.com

[‡]Osaka University, Department of Economics, 1-7 Machikaneyama, Toyonaka, Osaka 560-0043 Japan. E-mail: yosuke.yasuda@gmail.com

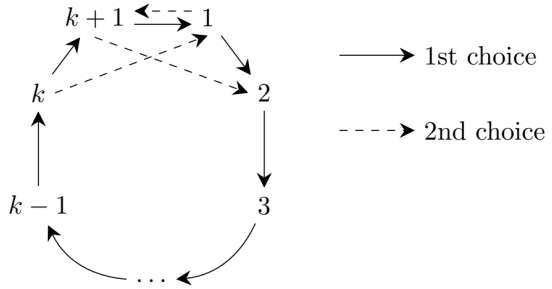


Figure 8: The counterexample for the general case in the proof of Theorem 1.

Figure 8 provides a graphical representation of these preferences.

We start with two observations. First, Lemma 1 implies that ψ must always return an individually rational exchange. Second, if a given agent i has two acceptable objects besides object i, then agent i must receive an object. This is because, otherwise, she could misreport to say that her true second choice was her first choice, thereby creating a unanimous trading cycle of size no greater than k. This second observation implies that agents k, $k+1$, and 1 must receive an object. Since the only agent besides k who regards object k as acceptable is $k-1$, agent $k-1$ receives an object as well. In the same way, all agents from 2 to $k-2$ receive an object. Overall, all agents from 1 to $k+1$ must receive an object. But there is no k-robust exchange in which every agent receives an object. This contradicts the assumption that ψ is k-robust. □

D Modified k-Greedy Algorithm and k-Robust Core

Round 0 of the k-greedy algorithm ensures that the resulting exchange is k-unanimous. One could imagine a modification of this algorithm where Round 0 is expanded so that, after eliminating the unanimous trading cycles with size k or fewer, top trading cycles with size k or fewer are iteratively implemented until no such cycles exist. Formally, the *modified k-greedy algorithm* is the same algorithm as the k-greedy algorithm except that it replaces Round 0 with the following:

Round 0: Define μ^0 as follows:
Round 0-1: Implement every unanimous trading cycle with size k or fewer. We define an exchange $\mu^{0,1}$ as follows. Let $I^{0,1}$ be the set of agents who are involved in an

implemented cycle. If agent j is in $I^{0,1}$, then $\mu_j^{0,1}$ is set to be agent j's first-choice object according to $\succ_j$. Otherwise, $\mu_j^{0,1} = j$. Define $S^{0,1} := I \setminus I^{0,1}$. If $S^{0,1} = I$, let $S^0 = S^{0,1}$ and go to Round 1. Otherwise, go to Round 0-2.

For any $n \geq 2$ such that the algorithm was determined to proceed to Round 0-n in Round 0-$(n-1)$, we define Round 0-n as follows.

Round 0-n: Implement every cycle with size k or fewer such that each agent in $S^{0,n-1}$ involved in a cycle receives the first-choice object among $S^{0,n-1}$. We define an exchange $\mu^{0,n}$ as follows. Let $I^{0,n}$ be the set of agents who are involved in an implemented cycle. If agent j is in $I^{0,n}$, then $\mu_j^{0,n}$ is set to be agent j's first-choice object according to $\succ_j$ among $S^{0,n-1}$. If agent j is in $S^{0,n-1} \setminus I^{0,n}$, $\mu_j^{0,n} = j$. If agent j is not in $S^{0,n-1}$, set $\mu_j^{0,n} = \mu_j^{0,n-1}$. Define $S^{0,n} := S^{0,n-1} \setminus I^{0,n}$. If $S^{0,n} = S^{0,n-1}$, let $S^0 = S^{0,n}$ and go to Round 1. Otherwise, go to Round 0-$(n+1)$.

Define the **modified k-greedy mechanism** to be the one that outputs the outcome of the modified k-greedy algorithm for any input. A result analogous to Theorem 3 holds:

Proposition 4. *The modified k-greedy mechanism is a k-robust mechanism, and it is individually rational, k-unanimous, and k-efficient.*

The proof is omitted as it is analogous to the one for Theorem 3.

Note that, if $k \geq |I|$, the modified k-greedy algorithm ends in Round 0, and it is equivalent to the TTC algorithm. Since the outcome of the TTC algorithm is in the core, one may hope to obtain variants of the properties of core, where we account for the restriction on the cycle size. Formally, we define the following.

Definition 4. A subset of agents $I' \subseteq I$ **blocks** an exchange μ via an exchange μ' if the following hold.

1. $\mu'_i \succeq \mu_i$ for all $i \in I'$.
2. $\mu'_j \succ \mu_j$ for some $j \in I'$.
3. $\mu'_i \in I'$ for all $i \in I'$.

Definition 5. The **k-robust core** C^k is the set of exchanges such that $\mu \in C^k$ if and only if μ is k-robust and there does not exist $I' \subseteq I$ that blocks μ such that $|I'| \leq k$.

Notice that an equivalent definition can be obtained if we do not require $|I'| \leq k$ but instead require that the resulting exchange μ' is k-robust. This is because if a coalition I' of size greater than k induces μ' that is k-robust, then we can take a subset of I' with size no greater than k that would work as a stand-alone coalition and block the original μ.

We note that the k-robust core may be empty, showing that the conjecture that the modified k-greedy mechanism induces an exchange in the k-robust core is false. One example is the preferences depicted in Figure 4 in Section 2.2 of the main text. In that example, any k-robust exchange must entail at least one unassigned agent. Let this agent be i and the agent who regards object i as the first choice be i'. Letting $I' = \{i, i'\}$ and μ' be such that $(\mu'_i, \mu'_{i'}, \mu'_j) = (i', i, j)$ where $j \in \{1, 2, 3\} \setminus \{i, i'\}$, we have that I' blocks the original exchange via μ'.

In fact, the emptiness of the k-robust core is general, in the sense the next result formalizes.

Proposition 5. *Fix (I, k) such that $1 < k < |I|$. There exists $\succ$ such that the k-robust core is empty.*

Given this result, one may hope that (i) the modified k-greedy mechanism produces an exchange in the k-robust core as long as the latter is nonempty, and also (ii) for any exchange in the k-robust core, there is a modified k-greedy mechanism with an appropriate ordering from Round 1 onward that induces the given exchange. Neither of these claims turns out to be true. Indeed, the k-robust core and the set of exchanges induced by the modified k-robust mechanisms where we vary the ordering may be disjoint even if the k-robust core is not empty, showing that both claims are false. The following example makes this point.

Example 3 (Modified k-greedy mechanism and k-robust core)**.** Consider the following preferences:

$$\succ_1 : 2, 3, 4, 1;$$
$$\succ_2 : 3, 4, 1, 2;$$
$$\succ_3 : 4, 1, 2, 3;$$
$$\succ_4 : 1, 2, 3, 4.$$

Figure 9 provides a graphical representation of these preferences. Suppose that $k = 2$.

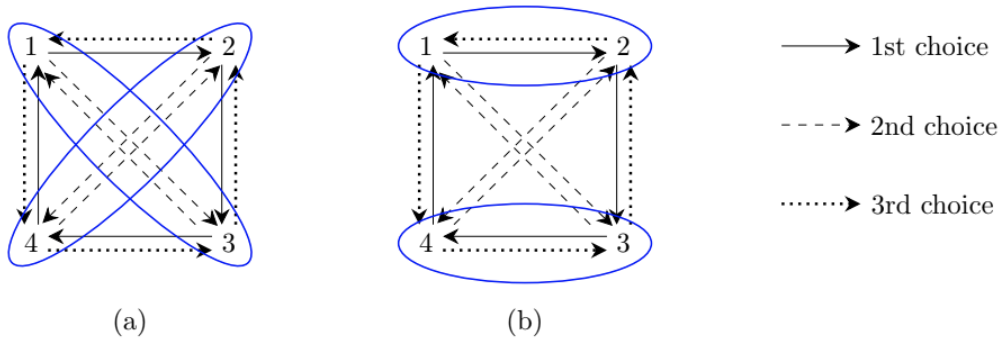


Figure 9: Example 3. Panel (a) shows the unique exchange in the k-robust core, while panel (b) shows the exchange induced by the modified k-greedy mechanism with the ordering such that agent 1 is the first.

One can check by inspection that the k-robust core is $\{\mu\}$ where $(\mu_1, \mu_2, \mu_3, \mu_4) = (3, 4, 1, 2)$. However, μ cannot be induced by any modified k-greedy mechanism. To see this, notice that Round 0 of any modified k-greedy algorithm assigns no object to anyone. Second, let i be the agent who gets the highest priority in a given modified k-greedy algorithm (i.e., $\sigma(i) = 1$.). Let agent i's first-choice object be j. Since agent j deems object i as acceptable, we must have $\mu_i = j$. Since $j = i + 1$, with the convention that $5 = 1$, we have that the outcome of the given modified k-greedy mechanism is not μ. □

This example makes another interesting point. In the standard environment, efficiency implies that at least one agent receives her first-choice object. A proof of this result goes as follows: if there is no such agent under a given exchange, then one can form a top trading cycle to obtain a Pareto-dominating exchange, which means that the original exchange was not efficient.

In our environment, such a result would not hold. Specifically, in Example 3, μ is k-efficient, but no agent receives their first-choice object. The proof that worked in the standard environment would not work here because the top trading cycle over μ would result in an exchange that involves a 4-cycle, which is not k-robust.

We note that serial dictatorship (defined to account for the constraint of k-robustness) does not induce μ, either.[1]

[1]Serial dictatorship consists of a number of steps. In the 0-th step, it identifies the set of k-robust exchanges, denoted E^0. In the l-th step, it identifies the set of all exchanges that are the best for agent l in E^{l-1}, and we denote the set by E^l. This construction implies that $E^{|I|}$ is a singleton, and the algorithm outputs its element.

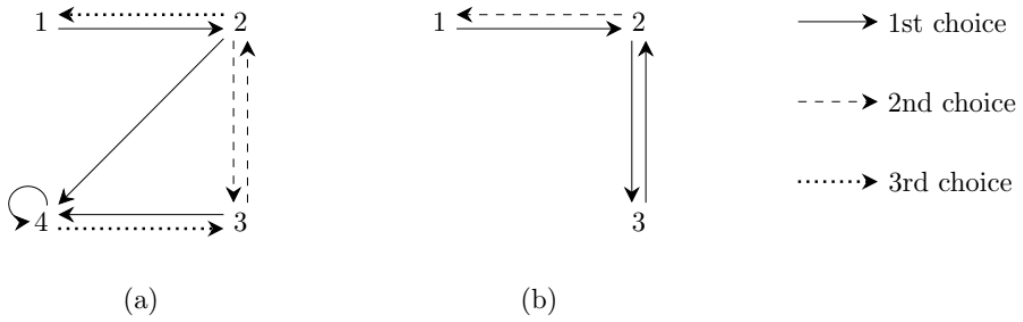


Figure 10: Example 4. After removing the unanimous trading cycle in panel (a) (just eliminating agent 4), we obtain the updated preferences as in panel (b). The k-greedy algorithm would implement the cycle $(1, 2)$ in its Round 1, while the modified k-greedy algorithm would implement the cycle $(2, 3)$ in the second step of its Round 0.

We now consider the relationship between the k-greedy mechanism and the modified k-greedy mechanism. First of all, the two mechanisms can induce different exchanges.

Example 4 (k-greedy mechanism and modified k-greedy mechanism)**.** Suppose that $I = \{1, 2, 3, 4\}$ and $k = 2$. Consider the following preferences.

$$\succ_1 : 2, 1;$$
$$\succ_2 : 4, 3, 1, 2;$$
$$\succ_3 : 4, 2, 3;$$
$$\succ_4 : 4.$$

Figure 10 provides a graphical representation of these preferences. Note that there is a single unanimous trading cycle, which is a 1-cycle consisting of agent 4. After removing agent 4, agents 2 and 3 would find themselves as respective first choices, but they are in fact second choices originally. For this reason, the two mechanisms induce different exchanges. More specifically, the k-greedy mechanism induces μ with $(\mu_1, \mu_2, \mu_3, \mu_4) = (2, 1, 3, 4)$ and the modified k-greedy mechanism induces μ' with $(\mu'_1, \mu'_2, \mu'_3, \mu'_4) = (1, 3, 2, 4)$. □

Note that, in the above example, the exchange induced by the k-greedy mechanism μ is not in the k-robust core because $(\mu', \{2, 3\})$ blocks it, while the one induced by

the modified k-greedy mechanism is in the k-robust core. This argument can be generalized to imply the following.

Proposition 6. *If a k-greedy mechanism induces an exchange μ in the k-robust core, the modified k-greedy mechanism with the same ordering induces μ as well.*

This result is a generalization of the finding in Example 4 in the sense that its contrapositive is equivalent to the following statement: if the two mechanisms induce different exchanges, then the exchange induced by the k-greedy mechanism is not in the k-robust core—the situation taking place in the example. The result suggests that, although the modified k-greedy mechanism fails to always induce an exchange in the k-robust core, it produces such an exchange "more likely."

Appendix D.3 provides two more examples to illustrate the properties of k-robust core. One example shows that the following claim is false: If all exchanges in the (unrestricted) core are not k-robust, then the k-robust core is empty. The other example shows that, although all the examples so far feature singleton k-robust cores, such a property is not general; the k-robust core may consist of multiple exchanges.

D.1 Proof of Proposition 5

Proof. Consider the following preferences for $i \in \{1, \ldots, k+1\}$ where we we let $k+2 = 1$ and $k+3 = 2$.[2]

$$\succ_i : i+1, i+2, i.$$

Moreover, for all $i \in \{k+2, \ldots, |I|\}$, let

$$\succ_i : i.$$

We show that any k-robust μ is blocked by some I' via some μ'.

Suppose first that μ has an l-cycle with $1 < l < k$. Let I'' be the set of agents involved in this cycle. Then, there must exist $i \in I''$ who is receiving her second-choice object, i.e., object $i+2$. This implies that agent $i+1$ does not receive his first-choice object. Hence, letting μ' and I' be such that

1. $I' = I'' \cup \{i+1\}$;

[2]We use this convention of $k+2 = 1$ and $k+3 = 2$ in this proof.

2. $\mu'_j = \mu_j$ for all $j \in I'' \setminus \{i\}$;

3. $\mu'_i = i + 1$;

4. $\mu'_{i+1} = i + 2$;

5. $\mu'_j = j$ for $j \notin I'$,

it follows that I' blocks μ via μ'.

Next, suppose that, for every l-cycle that μ has, $l = 1$ holds. Letting

1. $I' = \{1, \ldots, k\}$;

2. $\mu'_i = i + 1$ for all $i \in \{1, \ldots, k-1\}$;

3. $\mu'_k = 1$;

4. $\mu'_j = j$ for all $j \notin I'$,

it follows that I' blocks μ via μ'.

Finally, suppose that μ has a k-cycle, where the set of involved agents, which we denote by I'', satisfies $I'' \subseteq \{1, \ldots, k+1\}$. Let i be the unique agent such that $I'' \cup \{i\} = \{1, \ldots, k+1\}$. Without loss of generality, let $i = k$. Then, letting

1. $I' = \{1, \ldots, k\}$;

2. $\mu'_i = i + 1$ for all $i \in \{1, \ldots, k-1\}$;

3. $\mu'_k = 1$;

4. $\mu'_j = j$ for all $j \notin I'$,

it follows that I' blocks μ via μ'.

Overall, we have shown that any k-robust μ is blocked by some I' via some μ'. □

D.2 Proof of Proposition 6

Proof. Fix an ordering. Suppose that the exchange μ induced by the k-greedy algorithm and exchange μ' induced by the modified k-greedy algorithm are different from each other. Let C be the set of cycles that are formed in Round 0 of the modified k-greedy algorithm but not in the entire course of k-greedy algorithm. We first prove the following:

Lemma 2. *The set C is nonempty.*

Proof. Suppose, toward a contradiction, that C is empty. Consider the following two sets of agents:

$$D_\mu := \{i \in I | i \text{ is in the same cycle on } \mu \text{ as some } j \text{ such that } \mu_j \neq \mu'_j\}.$$

$$D_{\mu'} := \{i \in I | i \text{ is in the same cycle on } \mu' \text{ as some } j \text{ such that } \mu_j \neq \mu'_j\}.$$

Notice that, in particular, if $\mu_i \neq \mu'_i$, then we have $i \in D_\mu$ and $i \in D_{\mu'}$.

Note that we must have $D_\mu = D_{\mu'}$. This is because, if $i \notin D_\mu$, then every j in the same cycle on μ as i satisfies $\mu_j = \mu'_j$ by the definition of D_μ. But this implies $i \notin D_{\mu'}$. The same logic shows that $i \notin D_{\mu'}$ implies $i \notin D_\mu$, showing that $D_\mu = D_{\mu'}$.

Note also that and D_μ and $D_{\mu'}$ are nonempty because μ and μ' are different from each other.

Let $i^* = \min D_\mu$. Since C is empty, Rounds before Round i^* in either algorithm do not produce any cycle that does not appear in the other algorithm, and Round i^* is the first round at which different cycles appear in the two algorithms. Let the cycle produced in Round i^* of the k-greedy algorithm be $(i^\mu_1, i^\mu_2, \ldots, i^\mu_{L^\mu})$ and that of the modified k-greedy algorithm be $(i^{\mu'}_1, i^{\mu'}_2, \ldots, i^{\mu'}_{L^{\mu'}})$, where $i^\mu_1 = i^\mu_1 = i^*$. Since these two cycles are different, there must exist l^* such that $i^\mu_{l'} = i^{\mu'}_{l'}$ for all $l' < l^*$ and $i^\mu_{l^*} \neq i^{\mu'}_{l^*}$.

Let $\tilde{S}$ be the set of agents whose assignment has not been determined before Round i^* of the k-greedy algorithm, and let $\tilde{S}'$ be the set of agents whose assignment has not been determined before Round i^* of the modified k-greedy algorithm. Note that Round i^* of the k-greedy algorithm runs the (i^*, k)-greedy algorithm on $\tilde{S}$, and Round i^* of the modified k-greedy algorithm runs the (i^*, k)-greedy algorithm on $\tilde{S}'$. Notice that $\tilde{S}' \subseteq \tilde{S}$ and $\tilde{S} \setminus \tilde{S}' \subseteq C'$ hold, where C' is the set of agents involved in the cycles that are formed in Round 0 of the modified k-greedy algorithm but not in Round 0 of the k-greedy algorithm. Let $I' := \{i^\mu_1, i^\mu_2, \ldots, i^\mu_{l^*-1}\} = \{i^{\mu'}_1, i^{\mu'}_2, \ldots, i^{\mu'}_{l^*-1}\}$.

By the definition of the (i^*, k)-greedy algorithm on $\tilde{S}$, $i^\mu_{l^*}$ must be the (uniquely, because preferences of agent $i^\mu_{l^*-1}$ are strict) best object for agent $i^\mu_{l^*-1}$ among all objects in $\tilde{S} \setminus I'$ with the constraint that there is a cycle with size k or fewer such that (i) agent i^μ_l receives object i^μ_{l+1} for all $l \in \{1, \ldots, l^* - 2\}$, (ii) all agents in the cycle belongs to $\tilde{S}$, (iii) every agent in the cycle receives an acceptable object, and (iv) agent $i^\mu_{l^*-1}$ receives the object. Similarly, the definition of (i^*, k)-greedy algorithm

on $\tilde{S}'$ implies that $i^{\mu'}_{l^*}$ must be the (uniquely, because preferences of agent $i^{\mu'}_{l^*-1}$ are strict) best object for agent $i^{\mu'}_{l^*-1}$ among all objects in $\tilde{S}' \setminus I'$ with the constraint that there is a cycle with size k or fewer such that (i') agent $i^{\mu'}_{l}$ receives object $i^{\mu'}_{l+1}$ for all $l \in \{1, \ldots, l^* - 2\}$, (ii') all agents in the cycle belongs to $\tilde{S}'$, (iii') every agent in the cycle receives an acceptable object, and (iv') agent $i^{\mu'}_{l^*-1}$ receives the object.

Since C is empty, no agent in the cycle that i^* belongs to in the k-greedy algorithm are in C'. Hence, all agents in the cycle that i^* belongs to in the k-greedy algorithm are in $\tilde{S}'$ because $\tilde{S} \setminus \tilde{S}' \subseteq C'$. This implies that the constraints (ii) and (ii') are identical to each other. Moreover, constraints (i) and (i'), as well as (iv) and (iv'), are identical to each other, respectively, because $I' := \{i^{\mu}_1, i^{\mu}_2, \ldots, i^{\mu}_{l^*-1}\} = \{i^{\mu'}_1, i^{\mu'}_2, \ldots, i^{\mu'}_{l^*-1}\}$. Finally, constraints (iii) and (iii') are identical to each other. These facts imply that we must have $i^{\mu}_{l^*} = i^{\mu'}_{l^*}$. But this contradicts our earlier conclusion that $i^{\mu}_{l^*} \neq i^{\mu'}_{l^*}$. This completes the proof. □

Now, let n be the smallest integer such that some cycle in C is formed in Round 0-n of the modified k-greedy algorithm, and take an arbitrary cycle in C that is formed in Round 0-n. Let I' be the set of the agents involved in the chosen cycle. Note that $|I'| \leq k$ must hold by the definition of modified k-greedy algorithm.

Note that, under μ, each agent in I' receives an object in $S^{0,n-1}$ if $n > 1$ and in I if $n = 1$. Note also that, under μ', each agent in I' receives her first-choice object in $S^{0,n-1}$ if $n > 1$ and in I if $n = 1$. By the definition of C, there must be at least one agent in I' who receives different objects between μ and μ'. These observations imply that $\mu'_i \succeq_i \mu_i$ for every $i \in I'$ and $\mu'_j \succ_j \mu_j$ for some $j \in I'$. Moreover, $\mu'_i \in I'$ holds for all $i \in I'$. Hence, I' blocks μ via μ' and thus, μ is not in the k-robust core. Therefore, we have shown the contrapositive to the statement of the proposition. The proof is complete. □

D.3 Additional Examples

This section provides two examples to illustrate the properties of k-robust core. The first example shows that the following claim is false: If no exchanges in the (unrestricted) core are k-robust, then the k-robust core is empty.

Example 5 (Core and k-robust core)**.** Suppose that $I = \{1, 2, 3\}$ and $k = 2$. Consider

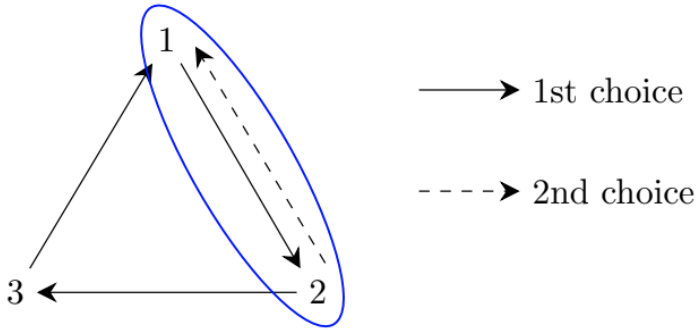


Figure 11: Example 5. Although the unique exchange in the core, in which every agent receives their first-choice object, is not k-robust, the k-robust core is nonempty (circled in the picture).

the following preferences.

$$\succ_1 : 2, 1;$$
$$\succ_2 : 3, 1, 2;$$
$$\succ_3 : 1, 3.$$

Figure 11 provides a graphical representation of these preferences. Note that the (unrestricted) core consists of a single exchange μ such that $(\mu_1, \mu_2, \mu_3) = (2, 3, 1)$ and this is not k-robust. However, k-robust core is nonempty, and it consists of a single exchange μ' such that $(\mu'_1, \mu'_2, \mu'_3) = (2, 1, 3)$. □

The second example shows that, although all the examples in Appendix D feature singleton k-robust cores, it is not general to have a singleton k-robust core.

Example 6 (Non-singleton k-robust core)**.** Suppose that $I = \{1, 2, 3, 4\}$ and $k = 2$. Consider the following preferences.

$$\succ_1 : 2, 4, 1;$$
$$\succ_2 : 3, 1, 2;$$
$$\succ_3 : 4, 2, 3;$$
$$\succ_4 : 1, 3, 4.$$

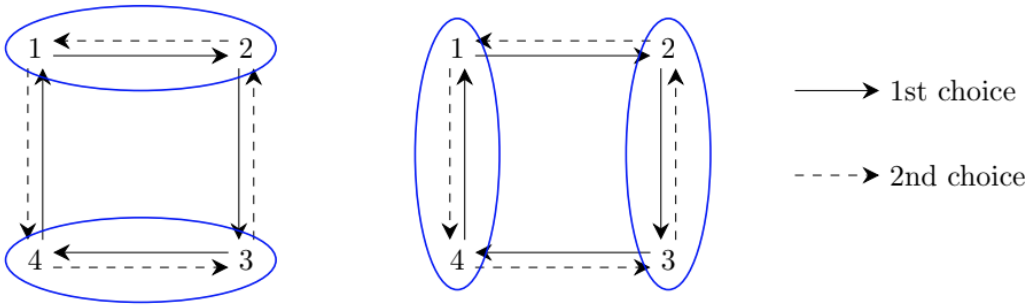


Figure 12: Example 6. The exchanges in the two panels are both in the k-robust core.

Figure 12 provides a graphical representation of these preferences. By inspection, one can show that the k-robust core consists of two exchanges, $C^k = \{\mu, \mu'\}$, where $(\mu_1, \mu_2, \mu_3, \mu_4) = (2, 1, 4, 3)$ and $(\mu'_1, \mu'_2, \mu'_3, \mu'_4) = (4, 3, 2, 1)$. □

E Additional Discussion on k-Greedy Mechanism

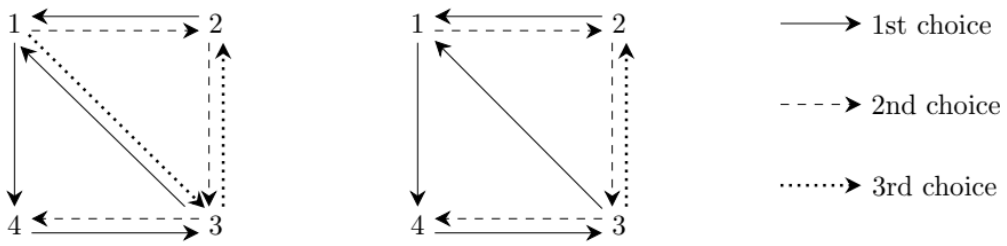
One may also consider a version of part 1 of Proposition 2 in which strict preferences are replaced with weak preferences while at least one strict preference relation is required:

Claim 3. *Fix any I, k, and $\succ$. There is no $\succ'_i$ such that for any ordering σ, we have $\psi_i(\succ'_i, \succ_{-i}) \succeq_i \psi_i(\succ)$ with at least one strict preference relation where ψ is the k-greedy mechanism with σ.*

This claim turns out to be incorrect. The next example shows this point.

Example 7 (Counterexample to Claim 3)**.** Consider an economy with four agents, $I = \{1, 2, 3, 4\}$ and set $k = 2$. Consider the following preferences:

$$\succ_1 : 4, 2, 3, 1;$$
$$\succ_2 : 1, 3, 2;$$
$$\succ_3 : 1, 4, 2, 3;$$
$$\succ_4 : 3, 4.$$



(a) Exchange under $\succ$. (b) Exchange under $(\succ_1', \succ_{-1})$.

Figure 13: Example 7. Given the preference profile as in panel (a), for any ordering σ, agent 1 would be weakly better off under the k-greedy mechanism with σ if she misreports her preferences as in panel (b). The improvement is strict if σ orders agent 3 the first.

Also, consider the following misreporting by agent 1:

$$\succ_1' : 4, 2, 1.$$

See Figure 13 for a graphical representation.

Note first that, under both $\succ$ and $(\succ_1', \succ_{-1})$, there is no pair of agents such that they find each other's object to be their first choice. This means that, whether the input is $\succ$ or $(\succ_1', \succ_{-1})$, no cycle is formed in Round 0 of any 2-greedy algorithm.

Consider Round 1. If the input is $\succ$, agent 1 is assigned object 3 if $\sigma(3) = 1$ and she is assigned object 2 otherwise. If, on the other hand, the input is $(\succ_1', \succ_{-1})$, then agent 1 is assigned object 2 for any σ. Since $2 \succ_1 3$, this shows that Claim 3 is incorrect. □

F Modifications of the Simulation Model

This section presents various alternative specifications for the simulations in Section 4 of the main text. In Appendix F.1, we vary the distribution of preferences. In any of the specifications we consider, we find the robustness of the effect that the k-greedy mechanisms perform relatively well even with small p, and we also find that some of the modifications amplify the effect. Then in Appendix F.2, we consider settings where a certain rewiring process takes place after agents drop out. Again, we find the robustness of our main insight. Each graph in this section records the average of

the relevant values for 100 runs where the preferences are drawn independently across different runs.

F.1 Changes in the Utility Distribution

In all of the following specifications, the variables ϵ_j and ξ_{ij} are independently distributed and follow $N(0,1)$. The numbers $\alpha, \beta, \gamma, \delta \in [0,1]$ and $L \in \mathbb{N}$ are parameters. Agent i's utility from receiving object j is determined as follows:

1. *Popular and unpopular objects.* In this modification, the preferences of different agents are correlated, and some objects are intrinsically more popular than others. Specifically, we set $u_{ij} = \alpha\epsilon_j + (1-\alpha)\xi_{ij}$. Here, ϵ_j depends only on j (not i), and thus measures the popularity of object j. The parameter α denotes the strength of the preference correlation. The case with $\alpha = 1$ corresponds to perfect correlation (every agent has the same preferences) and the one with $\alpha = 0$ corresponds to independent preferences, which is our base scenario. The results are shown in Figure 14 for the case of $N = 200$, $p = 0.05$, and $\alpha = 0, 0.2, 0.6$. We find that the performances of the k-greedy mechanisms relative to the performance of the TTC mechanism become better when α is smaller. The reason for this pattern is that preferences are more correlated when α is larger and hence the cycles tend to be smaller. This weakens the advantage of k-greedy mechanisms because the restrictions on the cycle sizes become more likely to be irrelevant.[3] We note that we ran simulations for other parameter combinations, and the basic results are robust. Analogous remarks apply to other modifications below where we only present graphs for $N = 200$, $p = 0.05$, and a few select parameter values.

2. *Two-sided market.* In this specification, the market is divided into two sides, and agents have a tendency to prefer the objects on the other side. Specifically, we set $u_{ij} = \beta\mathbb{I}_{i+j=\text{odd}} + (1-\beta)\xi_{ij}$. Here, we consider the scenario where odd-indexed agents/objects are on one side of the market and even-indexed agents/objects are on the other side. The parameter β denotes the strength of the tendency to prefer the objects on the other side. When $\beta = 1$, the preferences depend only on which side the given object belongs to; the case

[3] In particular, when $\alpha = 1$, every agent receives their own object, so the size of every cycle is 1.

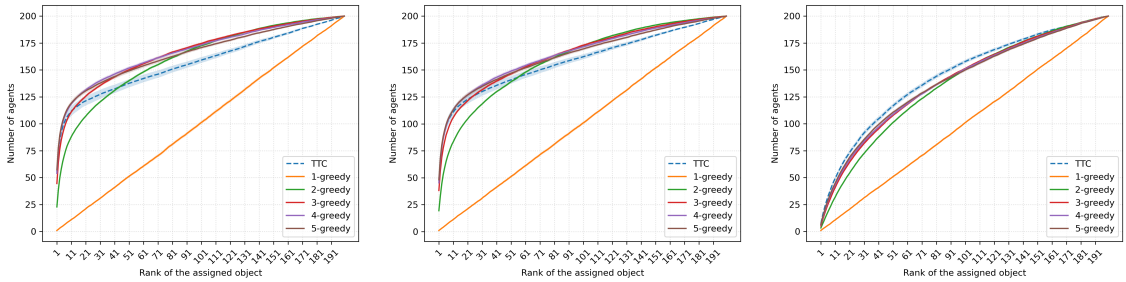


Figure 14: The performance of each mechanism under Modification 1. The strength of preference correlation (α) is 0, 0.2, and 0.6 in the left, middle, and right panels, respectively.

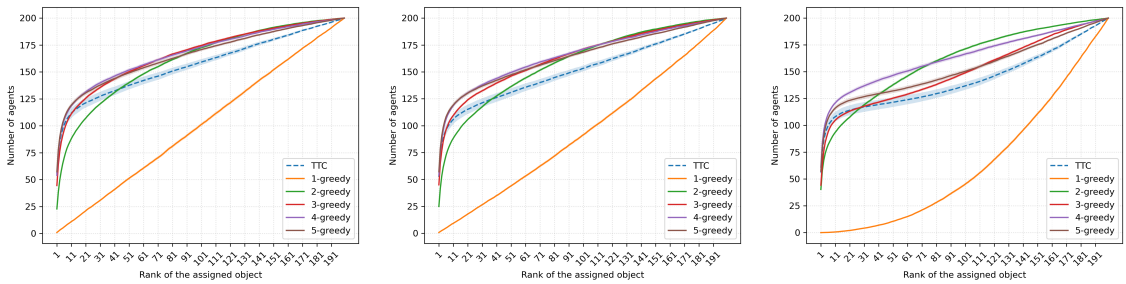


Figure 15: The performance of each mechanism under Modification 2. The strength of the tendency to prefer the objects on the other side (β) is 0, 0.2, and 0.6 in the left, middle, and right panels, respectively.

with $\beta = 0$ corresponds to our base scenario where the sides do not matter. The results are shown in Figure 15. We find that the k-greedy mechanisms perform mostly well relative to the TTC mechanism for any β, while k-greedy mechanisms with even k perform relatively better than those with odd k when β is large. The reason for this pattern is that there are many cycles with size k under the k-greedy mechanism, and when β is large, the agents in the cycle are likely to alternate the sides and, thus, the k-th agent chosen during the algorithm when forming the cycle receives a relatively desirable object when k is even and a relatively undesirable object when k is odd.

3. *Hate for the own object.* This modification corresponds to the case when agents are eager to exchange goods. We represent such preferences by the disutility of receiving their own good. Specifically, we set $u_{ij} = \gamma \mathbb{I}_{j \neq i} + (1 - \gamma)\xi_{ij}$. The parameter γ denotes the strength of the tendency to prefer the objects that other agents have. When $\gamma = 1$, the preferences depend only on whether there

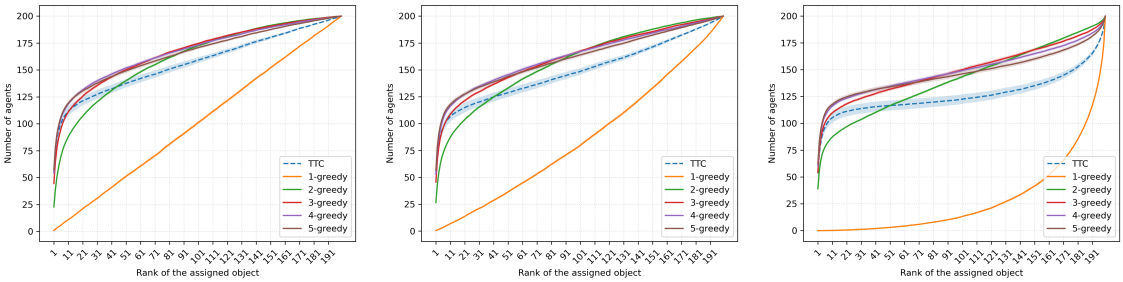


Figure 16: The performance of each mechanism under Modification 3. The strength of the tendency to prefer the objects that others have (γ) is 0, 0.2, and 0.6 in the left, middle, and right panels, respectively.

was an exchange; the case with $\gamma = 0$ corresponds to our base scenario where there is no disutility of receiving the own good. The results are shown in Figure 16. We find that the performances of the k-greedy mechanisms relative to the performance of the TTC mechanism become better when γ is larger. The reason for this pattern is that the TTC mechanism entails a greater chance for an agent to receive her own object than the k-greedy mechanisms do when there are many agents, and such an event becomes more undesirable as γ becomes larger.

4. *Love for the own object.* This modification corresponds to the case when agents are reluctant to exchange goods. We represent such preferences by the additional utility of receiving their own good. Specifically, we set $u_{ij} = \delta \mathbb{I}_{j=i} + (1-\delta)\xi_{ij}$. The parameter δ denotes the strength of the tendency to prefer the objects that they themselves are endowed with. When $\delta = 1$, the preferences depend only on whether there was an exchange; the case with $\delta = 0$ corresponds to our base scenario where there is no additional utility of receiving the own good. The results are shown in Figure 17. We find that the performances of the k-greedy mechanisms relative to the performance of the TTC mechanism become better when δ is smaller. The reason for this pattern is the opposite of the one for Modification 3: The TTC mechanism entails a greater chance for an agent to receive her own object than the k-greedy mechanisms do when there are many agents, and such an event becomes more desirable as δ becomes larger.

5. *Short preference list with informed agents.* In this modification, we assume

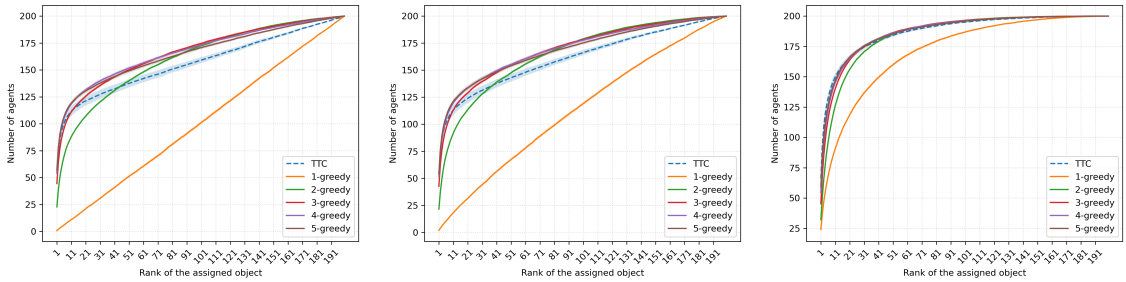


Figure 17: The performance of each mechanism under Modification 4. The strength of the tendency to prefer the objects that they themselves have (δ) is 0, 0.2, and 0.6 in the left, middle, and right panels, respectively.

that each agent submits a preference list that only includes the top L choices among all the objects, where we set the utility to be $u_{ij} = \xi_{ij}$. We call this case the case with "informed agents" because we implicitly assume that each agent knows which objects are her top L choices among all the N objects. The results are shown in Figure 18. We find that the k-greedy mechanisms perform mostly well relative to the TTC mechanism for any L except when both L and k are small, while the performances of the k-greedy mechanisms are nonmonotone in L. Specifically, the performances become better as L increases when L is small, while they become worse when L is large. The reason for this pattern is that increasing L has two effects: First, it increases the chance that a cycle can be formed given any set of agents of size no more than k due to the greater acceptability. This enhances the welfare of the agents. Meanwhile, the choice of cycles in the k-greedy algorithm implies that a cycle may assign relatively undesirable objects, especially to the agents who appear at a later stage in the formation of a given cycle. This is because increasing L adds more undesirable objects to the list of preferences as agents are informed. This effect deteriorates the welfare.

6. *Short preference list with uninformed agents.* In this modification, we assume that each agent submits a preference list that only includes L randomly chosen objects, where we set the utility to be $u_{ij} = \xi_{ij}$. This is called the case with "uninformed agents" because we implicitly assume that each agent does not necessarily know her true top L objects but rather samples random L objects out of all the N objects. The results are shown in Figure 19. We find that the performances of the k-greedy mechanisms relative to the performance of the

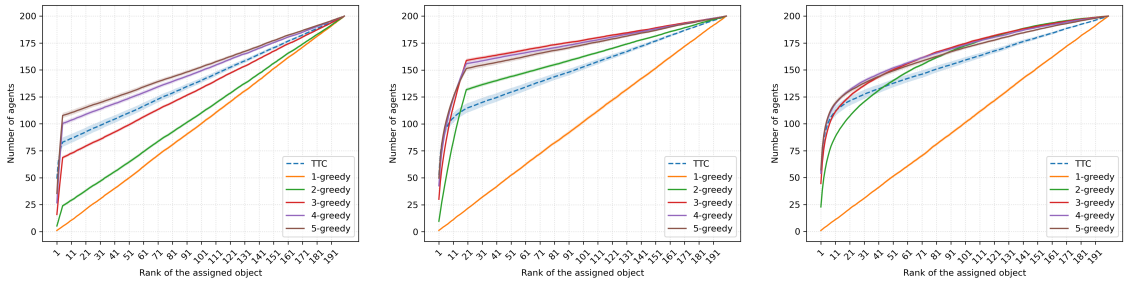


Figure 18: The performance of each mechanism under Modification 5. The length of the submitted preference list (L) is 5, 20, and unbounded in the left, middle, and right panels, respectively.

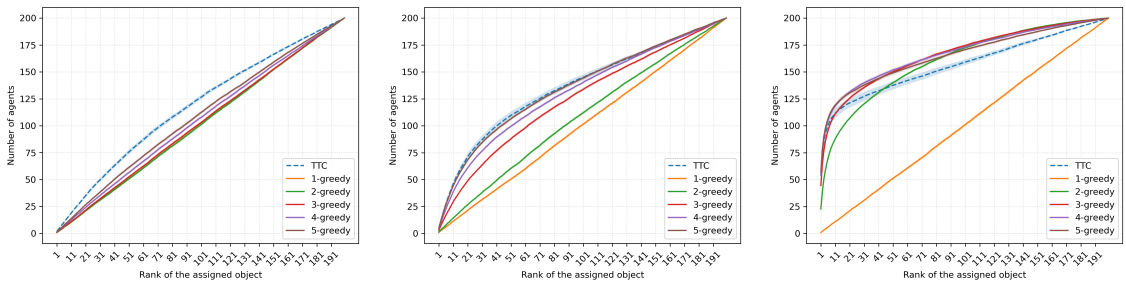


Figure 19: The performance of each mechanism under Modification 6. The length of the submitted preference list (L) is 5, 20, and unbounded in the left, middle, and right panels, respectively.

TTC mechanism become better when L is greater. The reason for this pattern can be understood by considering the two effects of increasing L in Modification 5: The first effect in which a cycle becomes more likely to be formed due to the greater acceptability still remains. However, the second effect in which agents become worse off due to the addition of the worse objects in the preference list no longer holds. This is because, as L increases, not only worse objects but also better objects can be added to the list of preferences as agents are uninformed. Overall, increasing L only enhances the welfare.

Our result that the k-greedy mechanisms perform relatively well even with small p is robust to these modifications, and we find that some of these modifications amplify the effect.

We have also examined mixtures of some of the above specifications, but do not list all the details of the results from such specifications.[4] We found that the results

[4]E.g., a mixture of Modifications 1 and 2, where $u_{ij} = \alpha\epsilon_j + \beta\mathbb{I}_{i+j=\text{odd}} + (1-\alpha-\beta)\xi_{ij}$.

are robust to considering such mixtures as well.

F.2 Rewiring among Affected Agents

In this section, we consider specifications in which, after agents drop out, the agents who are "affected" by a dropout agent take part in a certain rewiring process. Specifically, consider a family of mechanisms $(\Gamma^{N'})_{N' \subseteq N}$. We first run Γ^N. Then, each agent independently drops out with probability p. This determines the set of dropped agents, the set of affected agents, and the set of agents who are neither dropped nor affected. We consider centralized and decentralized rewriting processes. For the centralized ones, we examine the following two settings.

1. **Rewiring Process 1:** The mechanism $\Gamma^{\bar{N}}$ is run where $\bar{N}$ is the set of the affected agents. Each affected agent receives the object under the second run of Γ. Each dropped agent receives her own object. Each agent who is neither a dropped agent nor an affected agent receives the object assigned under the initial run of Γ.

2. **Rewiring Process 2:** First, we implement Rewiring Process 1. Then, each agent in $\bar{N}$ drops out with probability p. Any agents who are in the same cycle as a newly dropped agent receive their own object. Other agents receive the same object as in the outcome of Rewiring Process 1.

We view Rewiring Process 1 as an extreme scenario because, in reality, some of the affected agents may not participate in the rewiring process and even if they do, they may drop out after the process is run. Nonetheless, it helps us obtain a loose upper bound on the performance of mechanisms. We find that, although the TTC mechanism outperforms k-greedy mechanisms in Rewiring Process 1, our main insights from Section 4.2 carry over to Rewiring Process 2. The results are illustrated in Figure 20 for the case with $N = 1000$ and $p = 0.05$.[5] The utilities u_{ij} are drawn according to the normal distribution $N(0, 1)$, independently across agents and objects for 100 simulation runs. Behind these results is Theorem 5. When there are many agents, almost 100% of non-dropped agents are affected, and thus

[5]When $N = 200$, there is no clear dominance relationship between the TTC mechanism and the k-greedy mechanisms, but it is still true that the performance of the TTC mechanism becomes worse with dropouts and the k-greedy mechanisms are a viable alternative.

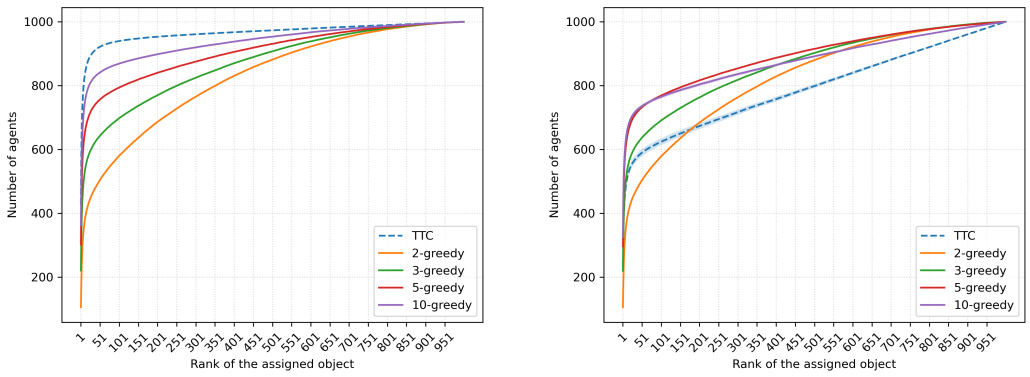


Figure 20: The performance of each mechanism under Rewiring Process 1 (left panel) and under Rewiring Process 2 (right panel).

the TTC mechanism under Rewiring Process 1 would essentially be rerunning the TTC mechanism among the non-dropped agents. This is why the TTC mechanism under Rewiring Process 1 performs well. The performance of the TTC mechanism becomes worse with the additional dropouts under Rewiring Process 2 for the same reason as the one with which the TTC mechanism is disadvantaged under our main simulations. The additional insight here is that running the k-greedy mechanisms among the affected agents (which would not be "almost 100% of non-dropped agents" due to Theorem 5) would still work better than the TTC mechanism under Rewiring Process 2.

For the decentralized processes, we consider the following two settings.

1. **Rewiring Process 3:** Randomly match agents in $\bar{N}$ to make as many pairs as possible, so that there will be $\frac{|\bar{N}|}{2}$ pairs if $|\bar{N}|$ is even, and $\frac{|\bar{N}|-1}{2}$ pairs and one remaining agent if $|\bar{N}|$ is odd (each configuration of matching occurs with equal probability). If two agents i and j in a pair regard each other's object as acceptable, then agent i receives object j and agent j receives object i. Otherwise, each of these two agents receives her own object. The remaining one agent in the case of $|\bar{N}|$ being odd receives her own object. Each dropped agent receives her own object. Each agent who is neither a dropped agent nor an affected agent receives the object assigned under the initial run of Γ.

2. **Rewiring Process 4:** First, we implement Rewiring Process 3. Then, follow the steps described below.

    - Step 1: Take the set $\bar{N}^1 \subseteq \bar{N}$ of agents who are currently receiving their

own objects. Finalize the assignment for agents in $I \setminus \bar{N}^1$. Run the same procedure as Rewiring Process 3, with replacing $\bar{N}$ with $\bar{N}^1$, to determine the assignment for agents in $\bar{N}^1$. Go to Step 2.

- Step l ($l = 2, 3, \dots$): Take the set $\bar{N}^l \subseteq \bar{N}^{l-1}$ of agents who are currently receiving their own objects. Finalize the assignment for agents in $\bar{N}^{l-1} \setminus \bar{N}^l$. If every agent in $\bar{N}^l$ regards all objects in $\bar{N}^l$ as unacceptable, let every agent in $\bar{N}^l$ receive their own object, finalize such an assignment (which would finalize the assignment for all agents in I) and finish following the steps. Otherwise, Run the same procedure as Rewiring Process 3, with replacing $\bar{N}$ with $\bar{N}^l$, to determine the assignment for agents in $\bar{N}^l$. Go to Step $l + 1$.

Note that this ends in a finite number of steps. Intuitively, Rewiring Process 3 lets the affected agents have one chance to receive someone else's object. Those agents are randomly matched with another agent, and if both agents agree, they can exchange their objects. We view this process as a realistic scenario when there is no centralized process for the affected agents. Depending on the situation, the affected agents may be able to spend a longer time searching for an object. Rewiring Process 4 is an extreme case in which agents continue searching until there is no possibility of receiving someone else's object. We view this as an unrealistic scenario because such an extensive search would likely require some type of centralization, and moreover, we assume no dropouts during the search process. We consider this case despite these problems because analyzing it would provide an upper bound on the performance of a decentralized rewiring process. We find that the k-greedy mechanisms for small values of k outperform the TTC mechanism under both Rewiring Processes 3 and 4, while the difference is more significant under Rewiring Process 3. The results are illustrated in Figure 21 for the case with $N = 1000$ and $p = 0.05$ (we observe a similar pattern for $N = 200$ as well). The utilities u_{ij} are drawn according to the normal distribution $N(0, 1)$, independently across agents and objects for 100 simulation runs. These results show that the effect of dropouts is so significant under the TTC mechanism that a decentralized system (as defined here) would not sufficiently mitigate the problem, even under an unrealistic scenario of Rewiring Process 4.

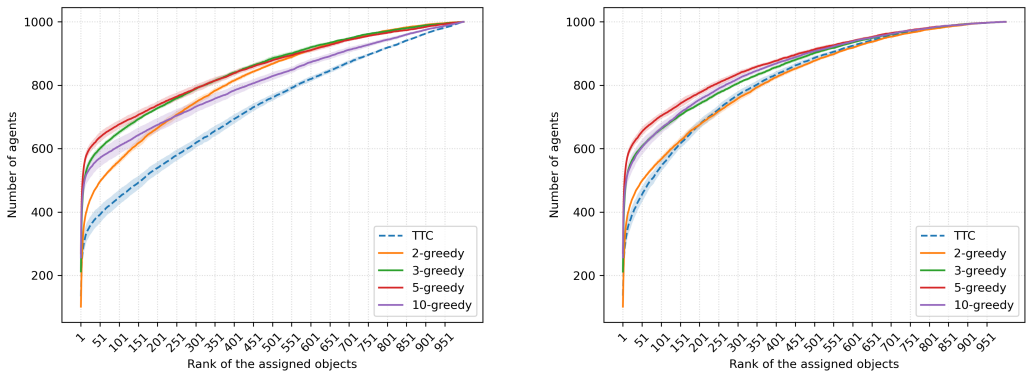


Figure 21: The performance of each mechanism under Rewiring Process 3 (left panel) and under Rewiring Process 4 (right panel).

G Discussion on School Choice

The simulation results so far demonstrate that the TTC mechanism performs poorly. One could argue that this is a possible explanation for why the TTC mechanism is not widely used in practice despite its well-known desirable properties in the standard environment without dropouts, such as efficiency, being in the core and strategy-proofness. However, this mechanism was used in a few school choice contexts such as New York City and New Orleans (cf. **??**). Why is this the case? More specifically, what is a feature distinct in the school choice problem that makes the TTC mechanism usable in practice?

We view that the key difference is in how "soft" the capacity constraint is. In the house exchange problem, it would be practically infeasible to let multiple agents consume one object. For this reason, once an agent in a given cycle drops out, all agents in the cycle have to become unassigned. The standard school-choice situation is different in that it would not be fatal for schools to accommodate slightly more students than their capacity because they typically have many available seats. Hence, even if one student drops out of a cycle, one could imagine a system that lets the other agents in the cycle receive the school seat specified in the cycle and only changes the assignment of the dropped student to the school she was initially endowed with, such as the "walk-zone school" (or to her outside option). One concern, then, is that such a system may result in too much excess of the number of students compared to the capacity when many dropout students come back to the same school. By simulations, we show that such a concern may not be too much of an issue.

Specifically, we consider the following model. There are N agents and K schools. Each student is endowed with a school, and each school has N/K students who are endowed with that school and has a capacity of N/K. Each student has preferences over schools and each school has priority over the students who are endowed with that school. First, the TTC mechanism is run.[6] After that, each student drops out of the currently assigned school with probability p and is assigned the endowed school.[7] If a student is assigned its endowed school and drops out, then the student is reassigned that school.

In the simulations, we generated the students' preferences and the schools' priorities uniformly randomly and independently. The result of the simulations for the case with $N/K = 100$ is presented in Figure 22. One can see that, although the number of students who come back to the endowed school (represented by orange bars) can be large, the resulting excess (i.e., the number of come-backs minus the number of students who leave, represented by blue bars) is typically much smaller.[8] The reason is that, although a given school may have to accept the dropped-out students who return to the school, there may also exist students who are assigned to the school under the TTC mechanism but drop out and leave the school. These two effects mostly cancel each other out, resulting in only a small excess beyond the capacity.

[6]We consider the standard modification of the TTC mechanism adapted to the school choice context: In each step of the TTC algorithm, each student points to the most desirable school that still has vacant seats. Each school has priority over the N/K students who are endowed with that school, and points to the highest student according to such priority among those who still remain in the market.

[7]Another possible specification would be that some dropped-out students will not be assigned any school (e.g., they go to a school outside of the market, such as a private school), but such an assumption would only strengthen our claim that the excesses are liklely to be small, as the excess becomes even less likely to emerge.

[8]In the graphs, the blue bar at the zero excess is extremely high. This is because it accounts for all the schools for which the resulting number of assignments is no greater than their capacity.

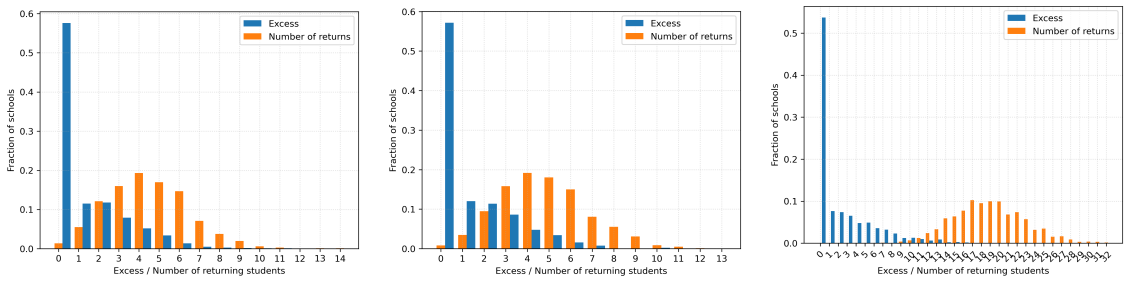


Figure 22: The distribution of the number of returns and excess at each school in the school choice problem. Each school has a capacity of 100. *Left*: 10 schools and $p = 0.05$; *Middle*: 20 schools and $p = 0.05$; *Right*: 20 schools and $p = 0.2$.