One-sided matching with neighbor and critical descent permission

Group: Yufei Liu 2022533037; Yida Zhao 2023233225; Pengyu Ji 2023233142

ABSTRACT

Mechanism design on social networks has recently become a hot research direction. Compared to the traditional settings, the new challenge is that we need to incentivize the agents of the game to invite their neighbors on the social network to join the game, with a proper guarantee of their no worse allocation. In one-sided matching, existing works have tried to add constraints on the Top Trading Cycle (TTC) to obtain the incentive under the new setting, but it only works in highly restricted scenarios. In this paper, we propose **Swap With Neighbour Plus** (**SWNP**), which not only achieves incentives in all networks but also gives a larger selection space for each agent with the help of the critical diffusion tree induced from the social network. Through simulation experiments, we confirmed that our mechanism consistently outperforms other baselines on both common and specific social network graphs.

1 INTRODUCTION

A new trend in mechanism design involves motivating agents to bring in new participants via their social connections. Unlike traditional static settings, these mechanisms leverage agents' connections to grow the market [8]. By encouraging current agents to invite their neighbors, the market expands. Typically, a larger market leads to better outcomes in most games. Specifically, in one-sided matching, an increase in participants often leads to more satisfying matches.

However, the motivation to invite more agents may not always be in the best interest of the inviters. For instance, if an agent invites others but finds that their preferred exchanges are disrupted by these new participants, they may lose the incentive to extend further invitations. This occurs because the involvement of additional agents can interfere with the original agent's optimal exchanges, reducing their overall benefit.

To remove one agent's hesitation to invite all her neighbors, we should guarantee that the inviters' match does not get worse after inviting their neighbors. The well-known Top Trading Cycle (TTC) mechanism, which gives a unique truthful, stable, and optimal solution under traditional one-sided matching scenarios, fails to keep Incentive Compatibility (IC) under social network setting [1]. To achieve the incentive, many mechanisms such as Swap With Neighbour (SWN), Swap with Children (SWC), and Leave and Share (LS) add different kinds of constraints on TTC [1, 9], including restrictions on the structure of social networks or each agent's selection space.

In this paper, we propose **Swap With Neighbour Plus** (**SWNP**), a new mechanism that uses LS and SWN as a base. We induce the critical diffusion tree from the social network. In each round, agents are allowed to swap with their neighbors of the original graph. Apart from that, each agent is allowed to **get** item from their descents of the diffusion critical tree. After the exchange, we share the neighbors of the matched agents just like LS. In the next round, we regenerate a diffusion critical tree of the new graph after sharing. It releases the restrictions on agents' selection space while keeping Incentive Compatibility (IC), Individual Rationality (IR) and Stable-CC. Wild-range simulations have been conducted to verify the improvement brought by SWNP. In brief, our contributions involve three folds:

- We propose **Swap With Neighbour Plus** (**SWNP**), a mechanism based on LS and SWN while releasing selection space with the help of the critical diffusion tree.
- SWNP is proven to achieve both Incentive Compatibility (IC), Individual Rationality (IR), and Stable-CC.
- To see the matching improvement under SWNP, we conduct simulations to compare SWNP with SWN and LS. SWNP indeed outperforms the other two baselines in all settings under consideration.

2 RELATED WORKS

Mechanism design in social networks is a prominent research area, where the mechanisms take into account the social connections and interactions of agents. A widely used approach to leverage these connections is to attract new participants, which has led to significant advancements in areas such as auctions and cooperative games [2, 3, 7]. For one-sided matching, which is a more specific field, there have been several existing mechanisms based on social networks such as SWC, SWN, LS that can be introduced.

We first provide the definition for the Top Trading Cycle (TTC) [4, 5], which serves as the foundation for the subsequent mechanisms over social networks.

Definition 2.1 (Top Trading Cycle). The Top Trading Cycle (TTC) method creates a directed graph where each agent points to the agent holding their most preferred available item in the matching market. At least one cycle will form. For each cycle, assign the item to the agent pointing to it, and then eliminate the cycle. This process is repeated until no agents remain.

TTC cannot ensure Incentive Compatibility (IC) in the social network setting. Multiple kinds of restrictions have been added to TTC to achieve the incentive and preserve the selection space for each agent as much as possible. One attempt is to restrict the structure of the social networks to trees and allow each agent to swap with her neighbors and descendants. It is called Swap With Children (SWC) [1].

Definition 2.2 (Swap With Children). Swap With Children (SWC) creates a directed graph where each agent points to the agent holding their most preferred available item among herself, her neighbors, and her descendants in the matching market. At least one cycle will form. For each cycle, assign the item to the agent

pointing to it, and then eliminate the cycle. This process is repeated until no agents remain.

Instead of posing restrictions on the structure of social networks, another trivial extension of TTC is Swap With Neighbors (SWN), which only allows agents to swap items with their neighbors. Leave and Share (LS) [6] uses SWN as a base and adds a natural sharing process to enlarge agents' selection space, trying to avoid conflicts with the core goal of expanding the market and providing a better allocation. We provide definitions for SWN and LS as follows.

Definition 2.3 (Swap With Neighbour). Swap With Neighbour (SWN) creates a directed graph where each agent points to the agent holding their most preferred available item among herself and her neighbors in the matching market. At least one cycle will form. For each cycle, assign the item to the agent pointing to it, and then eliminate the cycle. This process is repeated until no agents remain.

Definition 2.4 (Leave and Share). Leave and Share (LS) only allows agents to exchange with their neighbors. When a group of agents are matched and allocated corresponding items, they are eliminated (leave the market) and their remaining neighbors will be regarded as each other's neighbors (shared). We also provide the specific process of LS is as follows:

(1) A random order of agents is given in advance.
(2) We push the agent with the highest order which remains in the market into the stack. The agent in the stack top can point to the owner of her most preferred available item in neighbors. Especially, the agent can point to the agent in the stack bottom. The agent she points to will be pushed into the stack unless she is already in. Once there is a swap loop formed between a part of the agents in the stack, these agents can exchange their items and be popped from the stack, also leaving the market.
(3) Once the stack is empty, we eliminate those agents who have left the market. Their neighbors will be connected pairwisely.
(4) Repeat the above procedure until there is no agent left.

3 MODEL

In this part, We will specify our setting and some relevant definitions similar to [6].

We consider a one-sided matching problem in a social network denoted by an undirected graph $G = (N \cup Sp, E)$, which contains n agents $N = \{1, \dots, n\}$ and a virtual super node Sp. Each agent $i \in N$ is endowed with an indivisible item h_i and $H = \{h_1, \dots, h_n\}$ is the set of all agents' items. We define agent i as j's neighbor if there is an edge $e \in E$ between agent i and j, and let $r_i \subseteq N$ be i's neighbor set.

Note that different from the traditional setting, here we assume that only a subset $N_0 \subseteq N$ of the agents are initially in the game and they are independent of each other, i.e., no edge exists between these agents. Then these agents will invite others to join the game and the invited agents will also perform the invitation, i.e., adding an undirected edge between the inviting and invited agent. To better illustrate our mechanism, we add a virtual node S_p, which connects with each initial agent, i.e., there exists an edge $\langle S_p, i\rangle \in E$ for each $i \in N_0$. An agent i is included in the matching if and only if there exists a path from S_p to i. The graph G remains connected even if the virtual super node does not exist.

Each agent $i \in N$ has a strict preference $\succ_i$ over H. $h \succ_i h'$ means i prefers h to h' and we use $\succeq_i$ to represent the weak preference. Denote agent i's private type as $\theta_i = (\succ_i, r_i)$ and $\theta = (\theta_1, \cdots, \theta_n)$ as the type profile of all agents, where r_i is the neighbour set of agent i. Let θ_{-i} be the type profile of all agents except for agent i, then θ can be written as (θ_i, θ_{-i}). Let Θ be the type profile space of all agents. Similarly, we have $\Theta = (\Theta_i, \Theta_{-i})$.

In a matching mechanism, each agent is required to report her type (reporting neighbor set is treated as inviting neighbors in practice). We denote agent i's reported type as $\theta'_i = (\succ'_i, r'_i)$, where $\succ'_i$ is the reported preference and $r'_i \subseteq r_i$ is the reported neighbor set. Let $\theta' = (\theta'_1, \cdots, \theta'_n)$ be the reported type profile of all agents.

Definition 3.1. A one-sided matching mechanism is defined by an allocation policy $\pi = (\pi_i)_{i\in N}$, where $\pi_i : \Theta \to H$ satisfies for all $\theta \in \Theta$, for all i, $\pi_i(\theta) \in H$, and $\pi_i(\theta) \neq \pi_j(\theta)$ for all $i \neq j$.

For a given report profile θ', we generate a directed graph $G(\theta') = (N(\theta'), E(\theta'))$, where edge $\langle i, j\rangle \in E(\theta')$ if and only if $j \in r'_i$. Under θ', we say agent i is qualified if and only if there is a path from any agent in N_0 to i in $G(\theta')$. That is, i can be properly invited by the invitation chain from agent set N_0. Let $Q(\theta')$ be the set of all qualified agents under θ'. Then the matching mechanism can only use $Q(\theta')$.

Definition 3.2. A diffusion one-sided matching mechanism in social networks is a one-sided matching mechanism, $\pi = (\pi_i)_{i\in N}$, such that for all reported type profile θ', it satisfies:

(1) for all unqualified agents $i \notin Q(\theta')$, $\pi_i(\theta') = h_i$.
(2) for all qualified agents $i \in Q(\theta')$, $\pi_i(\theta')$ is independent of the reports of all unqualified agents.

The reported type of each agent could affect the qualification of other agents, i.e., misreporting the neighbor set may make some agents become unqualified.

Next, we define two desirable properties for diffusion one-sided matching mechanisms: individual rationality and incentive compatibility.

Individual rationality requires that for each agent, reporting her type truthfully guarantees that she gets an item no worse than her own.

Definition 3.3. Given a report profile θ', and the induced graph $G(\theta')$, we can generate a diffusion critical tree $T(\theta')$ from $G(\theta')$ as the following(only uncluding all the qualified agents).

- $T(\theta')$ is a rooted tree, where the virtual super node S_p is the root and there is an edge (S_p, i) for all $i \in N_0$.
- For all agents $i, j \in N$, there is an edge (i, j) if and only if (1) i is a cut-point to disconnect j from S_p, (2) there is no cut-point to disconnect j from i. Intuitively, agent i is the closest agent to j whose leaving will block j's participation.

An example of a critical diffusion tree is shown in Figure 1.

Definition 3.4 (Individual Rationality (IR)). A diffusion one-sided matching mechanism π is individually rational if for all $i \in N$, all $\theta_i \in \Theta_i$, and all $\theta'_{-i} \in \Theta_{-i}$, we have $\pi_i(\theta_i, \theta'_{-i}) \succeq_i h_i$.

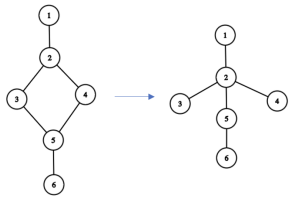

Figure 1: A simple example of the graph and its corresponding critical diffusion tree. Node 1 is the virtual super node S_p.

For incentive compatibility, it means reporting type truthfully is a dominant strategy for each agent. No agent is incentivized to misreport on either her neighbor set or her preference.

Definition 3.5 (Incentive Compatibility (IC)). A diffusion one-sided matching mechanism π is incentive compatible if for all $i \in N$, all $\theta'_{-i} \in \Theta_{-i}$ and all $\theta_i, \theta'_i \in \Theta_i$, we have $\pi_i(\theta_i, \theta'_{-i}) \succeq_i \pi_i(\theta'_i, \theta'_{-i})$.

Then to evaluate the performance of a matching mechanism in social networks, we directly use the same definition with regard to stability in [6].

Definition 3.6 (Blocking Coalition under Complete Components). Given an allocation $\pi(\theta)$, we say a set of agents $S \subseteq N$ (with item set $H_S \subseteq H$) is a **blocking coalition under complete components** for $\pi(\theta)$ if S forms a complete component in $G(\theta)$ and there exists an allocation $z(\theta)$ such that for all $i \in S, z_i(\theta) \in H_S$ and $z_i(\theta) \succeq_i \pi_i(\theta)$ with at least one $j \in S$ such that $z_j(\theta) \succ_j \pi_j(\theta)$.

Definition 3.7 (Stability under Complete Components (Stable-CC)). We say a mechanism π is **stable under complete components** if for all type profiles θ, there is no blocking coalition for $\pi(\theta)$.

In this report, we design a matching mechanism that satisfies IC, IR, and Stable-CC.

4 METHOD

We propose a new mechanism called **Swap With Neighbour Plus (SWNP)**, which uses LS as a base and adds extra exchange options to enlarge agents' selection space, trying to provide a better allocation. Firstly, we generate the diffusion critical tree of the original graph. Then in each round, agents are allowed to swap with their neighbors of the original graph. Different from naive SWN, we also allow the agent to **get** item from their descent of the diffusion critical tree. After the exchange, we share the neighbors of the matched agents in this round by connecting their neighbors to each other, thus their neighbors can have new neighbors in the next round. In the next round, we start by generating a new diffusion critical tree of the new graph after sharing. We give specific definitions and algorithms as follows.

Definition 4.1. We denote D'_i as the descent of i in the diffusion critical tree in $T(\theta')$.

Definition 4.2. We add a directed edge $\langle i, j \rangle$ in graph G if $j \in D'_i$ and $j \notin r'_i$, thus forming a new graph $G'(\theta')$.

Definition 4.3. Given a set $A \subseteq N$, we say $f_i(A) = j \in A$ is i's favorite agent in A if for any agent $k \in A, h_j \succeq'_i h_k$.

Definition 4.4. An ordering of agents is a one-to-one function $\mathcal{P} : \mathbb{N}^+ \to N$, where agent $\mathcal{P}(i)$ is the i^{th} agent in the ordering. Agents in $\mathcal{P}$ are sorted in ascending order by the length of the shortest path from agent set N_0 to them. Especially, for any agent $i \in N_0$, its shortest path length is 0. When multiple agents have the same length of the shortest path, we use a random tie-breaking.

The detailed process of our algorithm is as follows:

SWNP:

(1) Initialize $N_{out} = \emptyset$ and an empty stack S. Define the top and bottom of S as S_{top} and S_{bottom} respectively, and let $R_i = r'_i \cup D'_i \cup \{i\}$.
(2) While $N_{out} \neq N$:
  (a) Find the minimum t such that $\mathcal{P}(t) \notin N_{out}$. Push $\mathcal{P}(t)$ into S.
  (b) While S is not empty:
    (i) While $f_{S_{top}}(R_{S_{top}}) \notin S$, push $f_{S_{top}}(R_{S_{top}})$ into S.
    (ii) Pop off all agents from S_{top} to $f_{S_{top}}(R_{S_{top}})$, who already formed a trading cycle C following their favorite agents. Allocate each agent $i \in C$ the item $h_{f_i(R_i)}$. Add C to N^t_{out}.
    (iii) Update the neighbor set of C's remaining neighbors by removing C, i.e., for all $j \in \bigcup_{i \in C} r'_i \setminus N^t_{out}$, set $r'_j = r'_j \setminus C$. Then also update D'_i. Let $D'_i = D'_i \setminus C$ for all remaining agents.
  (c) Add N^t_{out} to N_{out}. Let all remaining neighbors of N^t_{out} connect with each other, i.e., they become neighbors of each other. That is, let $X = \bigcup_{i \in N^t_{out}} r'_i \setminus N^t_{out}$ and for all $j \in X$, set $r'_j = r'_j \cup X$. The diffusion critical tree is also regenerated on the new graph and thus D'_i are recomputed.

Note that SWNP differs from the original LS mainly in two aspects:

- Any exchange occurs only if all the agents included in the exchange form a directed cycle in the graph $G'(\theta')$. (While the original LS allows agents to directly get the item of S_{bottom})
- We allow all the agents to directly get the item of their critical descent.

5 PROPERTIES OF SWNP

In this section, we prove that SWNP is IR, IC, and Stable-CC.

Theorem 5.1. *For any ordering P, SWNP is IR.*

Proof. In SWNP, agent i leaves only when she gets an item h_j. She can always choose to keep her initial item as her favorite one, then she will be allocated h_i. Therefore, SWNP is IR. □

Theorem 5.2. *For any ordering P, SWNP is IC.*

First, we prove a lemma that no trading cycles will share the same agent, i.e., two different trading cycles consist of exactly distinct agents.

Lemma 5.3. *For a trading cycle $C_p = \{p_1, p_2, ..., p_k\}$ and another cycle $C_q = \{q_1, q_2, ..., q_l\}$, $C_p \neq C_q$, then $p_i \neq q_j$ for $\forall i \in \{1, 2, ..., k\}, \forall j \in \{1, 2, ..., l\}$.*

Suppose there exists $p_i = q_j$. Then in a trading cycle, she will get her favorite item among her neighbors, here labeled as p_{i_1} and q_{j_1}. We have $p_{i_1} = q_{j_1}$. To repeat this process, we have all the agents in C_p are exactly the agents in C_q, that is, $C_p = C_q$, which contradicts the assumption that $C_p \neq C_q$. □

Proof. Then we prove misreporting neither $\succ_i$ nor r_i can improve her allocation.

Misreport on $\succ$: For agent i, suppose her reported preference is $\succ_i'$ and is allocated $h_{j'}$ and the truthful type and corresponding allocation is $\succ_i$ and h_j. Before it is pushed onto the stack, all trading cycles are irrelevant to $\succ_i'$. So we consider the situation when it is pushed onto the stack.

Suppose $h_{j'} \succ_i h_j$. If i reported truthfully, i would choose j' before j. However, i was finally allocated h_j, which means j' forms a cycle $C_{j'}$ without i. If i reported $\succ_i'$, it gets $h_{j'}$, which indicates that there exists a trading cycle including j' and i. This is proved to be false by Lemma 5.3. Therefore, the assumption $h_{j'} \succ_i h_j$ does not exist.

Misreport on r: For each agent i, suppose her report neighbour set is r_i' and her truthful neighbour set is r_i. The corresponding allocation is $h_{j'}$ and h_j each. When i misreports, she could only affect her neighbors who are further away from S_p (and may also reduce the critical edges from her critical ancestor to her children in $G'(\theta')$). We consider two different situations.

First, i misreports r_i' and results in some $k \in r_i$ unqualified to get a better allocation, which means i is a critical father of k in $T(\theta')$. From Lemma 5.3, considering reporting neighbor set of i, the only way to break a trading cycle that does not contain i is to make some agents in the cycle unqualified (as i cannot control the edges in the cycle). Also, if there is no trading cycle C_k that contains a better allocation for i, i is definitely not incentivized to make k unqualified. Therefore, when i tries to make some k unqualified, i and k must be competing for some item $h_{j'}$ and k can form a trading cycle with j'. There are two possible competitions between i and k in the matching:

- k competes with i for a $h_{j'}$, where j' is no further from S_p than k (closer or of the same distance to the virtual root). As k is a critical child of i, the path between j' and k must contain i. Thus i has strictly higher priority than k when matching with $h_{j'}$, i.e., only if i is allocated with a better item than $h_{j'}$ and j' remains in the stack, k can match with $h_{j'}$. Therefore, i has no incentive to disconnect k.
- k competes with i for a $h_{j'}$, where j' is further from S_p than i. If k could swap with j', there exists a trading cycle containing k and j'. If the cycle contains i, then there does not exist the competition as i can get her favorite item at the same time with k.(if they compete with one item, the cycle won't ever form) If the cycle does not contain i, then we can prove that when k is disconnected from the graph, j' is also disconnected. If not, there exists a path from k to j' as they can form a trading cycle without passing i. i's misreporting cannot break this path. So if j' remains qualified, k is also qualified. Therefore, when k is disconnected, j' is also disconnected. i will not benefit from such disconnection if she aims at $h_{j'}$.

Considering both situations, i will not get a better allocation if she makes k unqualified.

Second, i misreports r_i' and does not make any agents unqualified. This will only reduce the option of i itself, while not affecting any edges in $G'(\theta')$. i will have no incentives to do so under any circumstance.

In conclusion, i has no incentive to misreport her neighbor set. □

Theorem 5.4. *For any ordering P, SWNP is Stable-CC.*

Proof. For every $S \subseteq N$ and their item set H_S. Let the allocation given by SWNP be $\pi(\theta)$. If there is a blocking coalition S, where $S \subseteq N$ is the node set of a complete component in $G(\theta)$, we have $\forall i \in S, S \subseteq r_i$. A blocking coalition S suggests there is a $p(\theta)$ such that for all $i \in S$, $p_i(\theta) \in H_S$, $p_i(\theta) \succeq_i \pi_i(\theta)$ with at least one $j \in S$ we have $p_j(\theta) \succ_j \pi_j(\theta)$. Therefore, for all $j \in S$, the blocking coalition guarantees the owner of $p_j(\theta)$ and j are in one trading cycle, that is, if a trading cycle contains any agent in the coalition, all the agents in the trading cycle are in the coalition. Based on SWNP, $p_j(\theta) \succeq \pi_j(\theta)$ means the agent with the initial item as $p_j(\theta)$ will be pushed into the stack before that with $\pi_j(\theta)$. Thus, the trading cycle which contains the agent with the initial item $p_j(\theta)$ and j can trade by the cycle (i.e.,$\forall i \in S, p_i(\theta) = \pi_i(\theta)$). This contradicts the assumption of $j \in S, p_j(\theta) \succ_j \pi_j(\theta)$. Hence, SWNP is Stable-CC. □

6 EXPERIMENTS

In this section, we compare our mechanism with SWN and LS. We define a cardinal index D to measure the performance and run experiments in various random graphs to show the performance of our mechanism. The lower bound is given by SWN since agents should be able to swap with their neighbors.

We define $\succ_i (j)$ as the j^{th} favorite item of i. Assuming that h_i is $\succ_i (j)$ and $\pi_i(\theta)$ is $\succ_i (k)$, where $j \geq k$ for IR property, we define the ascension of i as $d_i = j - k$. The average ascension of agents is defined as $D = \frac{\sum_{i \in N} d_i}{n}$. We use D to measure the average improvement of agents' satisfaction in a one-sided matching mechanism.

In our experiment, the number of agents is 200 and we assume that only agent 1 is in the market at the beginning without a specific explanation. Agents' preferences and initial items are generated randomly from all permutations of the items. And each setting we will repeat the experiment 100 times and take the average.

We first evaluate the three algorithms in a random graph. To generate random networks, we define the probability of an edge between any two nodes as p. A higher p leads to a denser connected graph. Especially, when $p = 1$, the graph is complete.

We observe that our mechanism performs worse than LS but better than SWN. This is because the randomly generated graphs have too many edges, resulting in a very small number of cut vertices. The lack of cut vertices limits agents' potential for descents in the diffusion critical tree, thereby restricting their exchange range.

Considering the lack of social network patterns in the random graphs, we adopted two special graphs, a tree and another special graph that we defined. Those graphs are also common in reality,

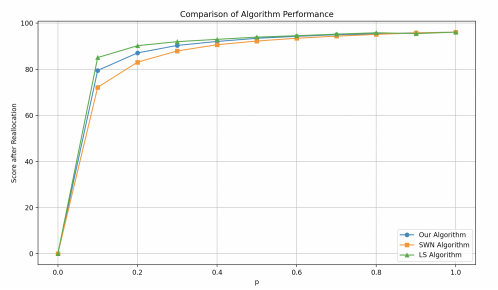

Figure 2: How D changes accordingly for each p. p is from 0 to 1 and the minimum scale for p is 0.1.

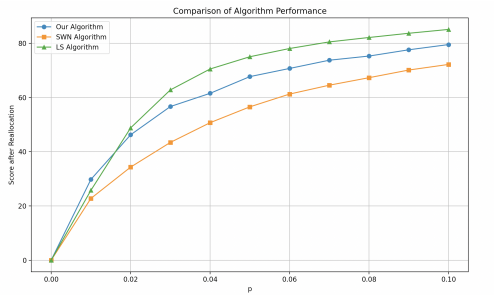

Figure 3: How D changes accordingly for each p. p is from 0 to 0.1 and the minimum scale for p is 0.01.

and our mechanism performs much better compared to that in the random graphs.

Now we consider the tree structure. To generate a tree, we uniformly select an integer f_i from $[1, i]$ as the father of node i. We execute the process for each node i.

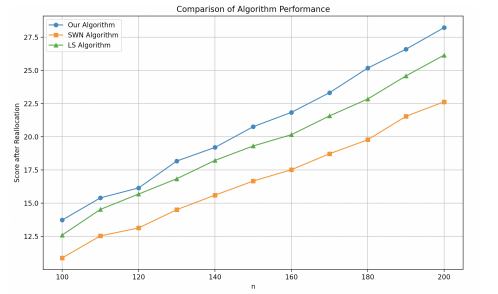

Figure 4: How D changes accordingly for each n. n is from 100 to 200 and the minimum scale for n is 10.

We observe that our algorithm outperforms both LS and SWN for every value of n. This is because, in the tree structure, every agent is a cut vertex, meaning every agent can obtain items from their descendants. This feature significantly increases the exchange range for each agent, leading to better allocation.

Next, we introduce the graph that we defined. Suppose there are n agents and we assume the size of a block is t. We divide n agents into several blocks, with each block having a size of t. Notice that if t can not divide n evenly, there will be a block with a size of $n\%t$. We let p be the probability of an edge between any two nodes **within the same block**. Then, we use a **bridge** to connect two blocks.

This type of graph is also common in reality because each block can represent a specific social circle, with few individuals belonging to multiple social circles simultaneously. These critical individuals are important in real-world scenarios, and our mechanism effectively leverages their presence to achieve better performance.

We set $p = 0.2$ and vary the size of a block.

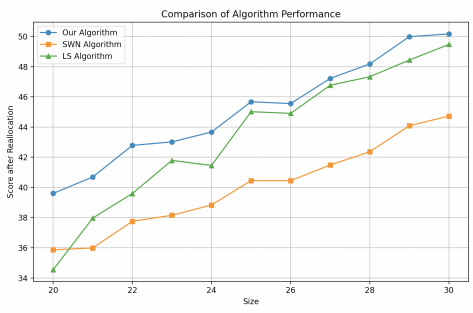

Figure 5: How D changes accordingly for each size. size is from 20 to 30 and the minimum scale for size is 1.

In these graphs, our algorithm also outperforms LS and SWN, maintaining the best performance for every block size. This is because the graph we construct contains many cut vertices, which our mechanism utilizes more effectively than LS and SWN, thus achieving better performance.

Overall, the experiments demonstrate that our mechanism performs well in structured graphs like trees and custom-defined graphs with many cut vertices, leveraging the presence of cut vertices to achieve superior allocation outcomes. However, in random graphs with high edge density, the performance of our mechanism is slightly limited, however, still better than SWN.

7 CONCLUSION

In this paper, we proposed a new one-sided matching mechanism called Swap With Neighbour Plus (SWNP) that satisfies IC, IR, and Stable-CC. This mechanism works in all kinds of network settings and significantly surpasses other mechanism baselines. Possible future work includes finding practical scenarios where this mechanism is suitable and looking for other extensions that further enlarge agents' selection space while retaining necessary properties.

REFERENCES

[1] Takehiro Kawasaki, Ryoji Wada, Taiki Todo, and Makoto Yokoo. 2021. Mechanism Design for Housing Markets over Social Networks. In AAMAS '21: 20th International Conference on Autonomous Agents and Multiagent Systems, Virtual Event, United Kingdom, May 3-7, 2021. 692–700.

[2] Bin Li, Dong Hao, Hui Gao, and Dengji Zhao. 2022. Diffusion auction design. Artificial Intelligence 303 (2022), 103631.

[3] Bin Li, Dong Hao, Dengji Zhao, and Tao Zhou. 2017. Mechanism Design in Social Networks. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA. 586–592.

[4] Jinpeng Ma. 1994. Strategy-proofness and the strict core in a market with indivisibilities. International Journal of Game Theory 23, 1 (1994), 75–83.

[5] Lloyd Shapley and Herbert Scarf. 1974. On cores and indivisibility. Journal of mathematical economics 1, 1 (1974), 23–37.
[6] Tianyi Yang, Yuxiang Zhai, Dengji Zhao, Xinwei Song, and Miao Li. 2023. Truthful and Stable One-sided Matching on Networks. (2023). arXiv:cs.GT/2201.05787
[7] Yao Zhang and Dengji Zhao. 2022. Incentives to Invite Others to Form Larger Coalitions. In Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems. 1509–1517.
[8] Dengji Zhao. 2021. Mechanism Design Powered by Social Interactions. In AAMAS '21: 20th International Conference on Autonomous Agents and Multiagent Systems, Virtual Event, United Kingdom, May 3-7, 2021. 63–67.
[9] Yue Zheng, Tianyi Yang, Wen Zhang, and Dengji Zhao. 2020. Barter Exchange via Friends' Friends. arXiv preprint arXiv:2010.04933 (2020).