Stable Marriage in Euclidean Space

Yinghui Wen
Shandong University
Qingdao, China
yhwen@mail.sdu.edu.cn

Zhongyi Zhang
Shandong University
Qingdao, China
zhangzhongyi@mail.sdu.edu.cn

Jiong Guo
Shandong University
Qingdao, China
jguo@sdu.edu.cn

ABSTRACT

We study stable marriage problems in the d-Euclidean space. Under this setting, each agent is represented as a point in the d-dimensional space, and for each agent a, the preference of a is based on the sorting according to the Euclidean distances between a and agents from the opposite gender. Let $\delta(a, b)$ being the Euclidean distance between two points a and b. A man u prefers a woman w_1 to another woman w_2 if and only if $\delta(u, w_1) < \delta(u, w_2)$. If $\delta(u, w_1) = \delta(u, w_2)$, then u ranks w_1 and w_2 indifferently, and we say there is a tie between w_1 and w_2 in u's preference list. A lot of variants of Stable Marriage with Ties (SMT) have been shown to be NP-complete when ties occur in preference lists. In this paper, we study the most famous hard variants of SMT in d-Euclidean space, namely, Regret-SMT, Forced-SMT, and Egalitarian-SMT. We prove that with $d = 1$, Forced-SMT and Regert-SMT can be solved in polynomial-time, while with $d = 2$, all of the three problems are NP-hard. Then we show that if the preference list can be incomplete (agents are allowed to not give a full rank of the opposite gender), the three problems and another variant Max-SMTI are NP-hard even with $d = 1$. Finally, we provide an algorithm to recognize whether a given preference profile can be embedded into 1-Euclidean space.

KEYWORDS

Social Choice; Stable Matching; Computational Complexity.

ACM Reference Format:
Yinghui Wen, Zhongyi Zhang, and Jiong Guo. 2023. Stable Marriage in Euclidean Space. In *Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), London, United Kingdom, May 29 – June 2, 2023*, IFAAMAS, 9 pages.

1 INTRODUCTION

Matching problems have received a considerable amount of attention from both economics and computer science communities and have been studied for several decades, due to their rich applications in the real world, for instance, assignment of students to colleges [1], kidney patients to donors [29], refugees to host countries [5].

One of the most prominent matching models is the Stable Marriage problem, which was introduced by Gale and Shapley [19]. Given two disjoint sets U and W with each $u \in U$ providing a strictly ordered list $\succ_u$ of the members of W and vice versa, the Stable Marriage (SM) problem seeks for a matching M without *blocking pair*. Herein, a blocking pair is a pair of $u \in U$ and $w \in W$ such that u and w are not matched by M but u prefers w to $M(u)$ in $\succ_u$ and w prefers u to $M(w)$ in $\succ_w$. Conventionally, the members of U are called men and the member of W are called women. We refer to both men and women as agents. The most classic SM problem requires that each agent provides a fully, strictly ordered list of the members of the opposite gender as his preference. According to the criticism that full, strict preference orders rarely suits real-world applications [7], a lot of variants with less restrictive preference structures haven been proposed, such as incomplete and ties preferences [22, 23, 28], general preferences [17], pairwise preferences [2, 4, 27], and multi-modal preferences [10, 34], etc.

We study a "geometric" variant of the Stable Marriage problem, called d-Euclidean Stable Marriage (d-Euclid-SM). In this variant, each $a \in U \cup W$ is represented by a point in the d-dimensional Euclidean space, and the preference list of a is based on the ranking of the Euclidean distances between a and agents from the opposite gender. More precisely, given two agents b_i, b_j from the opposite gender, a prefers b_i to b_j if and only if $\delta(a, b_i) < \delta(a, b_j)$ with $\delta(a, b)$ being the Euclidean distances between two points a and b.

The introduction of d-Euclid-SM is mainly motivated by the following application scenario. Consider a dating agency, which characterize each man and woman by a vector of length d. Each entry of the vector corresponds to one of d questions, whose answer range from "strongly agree" to "strongly disagree", which could be encoded as integers. The questions should measure the attitude of the men and women towards various issues like pets, children, hobbies, etc. Then, the preference of each man (or woman) is computed based on the Euclidean distance between the vector of this man (or woman) and the vectors of all women (or men). Computing a stable matching using these preferences is essentially the d-Euclid-SM problem. In addition, there are many real-world problems that can be modeled as d-Euclid-SM, especially when the agents' ranking criteria for opposite gender is defined by distance, e.g., assigning employees to factories. d-Euclidean space can be seen as a domain restriction of the preference lists, which guarantees many nice properties. Thus, it is interesting to check whether a hard variant of Stable Marriage problem remains NP-hard in d-Euclidean space.

A lot of variants of Stable Marriage have been introduced, which seek for a stable matching satisfying some constraints. Some ask for a stable matching satisfying a score bound, such as Egalitarian [24], Regret [21], Balanced [18], and Sex-equal [25], etc. Some variants focus on finding a matching with restricted edges, such as Forced [13–15], Forbidden [13–15], and Distinguished [31]. Let π denote a constraint. We say a stable matching M is π-*stable* if M satisfies π. With π being Egalitarian/Regret/Forced, it is NP-hard to find a π-stable matching [28] when ties occur in preference lists.

In this paper, we study the computational complexity of hard variants of Stable Marriage in d-Euclidean space. More precisely, we study d-Euclidean π-Stable Marriage with Ties (d-Euclid-π-SMT) with π being a constraint. Note that, an Euclidean instance of Stable Marriage problem without ties always admits a unique

Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), A. Ricci, W. Yeoh, N. Agmon, B. An (eds.), May 29 – June 2, 2023, London, United Kingdom.

Table 1: Overview of our results. "-" stands for that it is meaningless to study Max-SMT under this setting, since there always exists a prefect stable matching when preference lists are complete.

	d=2	d=1	
	complete	complete	incomplete
Regret	NP-hard	P	NP-hard
Egalitarian	NP-hard	?	NP-hard
Forced	NP-hard	P	NP-hard
Max	-	-	NP-hard

matching and can be found in $O(n^2)$ by matching, removing the current closest man-woman pair iteratively. In fact, this trivial algorithm still holds for the more general setting, that is, d-Euclidean Stable Roommates [3]. Regret-SMT, Egalitarian-SMT, Forced-SMT and MAX-SMTI are proved to be NP-hard by Manlove et al. [28]. Here, SMTI is the abbreviation of Stable Marriage with Ties and Incomplete Preferences. Refer to Preliminaries for the definitions of Regret, Egalitarian, Forced, and Max. We study these four problems in the d-Euclidean space. We find that when all preference lists are complete, Regret-SMT, Egalitarian-SMT, and Forced-SMT are NP-hard even with $d = 2$, while Regret-SMT and Forced-SMT admit polynomial-time algorithms when $d = 1$. When preference lists are permitted to be incomplete, the four problems are NP-hard even with $d = 1$. Refer to Table 1 for an overview of our results. Due to lack of space, the proofs of the lemmas and theorems marked with (*) are moved to Appendix.

Related Work. Arkin et al. investigated d-Euclidean Stable Roommates, under the name Geometric Stable Roommates, and proved that when the preference lists are complete and without ties, the problem can be solved in polynomial time [3]. Chen and Roy studied Multi-Dimensional Stable Roommates in 2-Dimensional Euclidean Space, they proved that k-Stable Roommates is NP-hard even in 2-Euclidean space and $k = 3$ [12]. In voting system, d-Euclidean space is studied as a kind of domain restriction [8, 9, 11, 16, 26, 33]. However, the definition is a little bit different, that is, only voters have preference lists based on Euclidean distance, the candidates do not need to rank the voters.

2 PRELIMINARIES

Let $U = \{u_1, \cdots, u_n\}$ and $W = \{w_1, \cdots, w_n\}$ be two n-elements disjoint sets of agents. We call the members in U *men*, and the members in W *women*. The *preference list* of $u \in U$ is an ordered subset that ranks a subset of the members in W, denoted as $\succ_u$. If the length of $\succ_u$ is less than n, we say $\succ_u$ is *incomplete*. If there are two women w_i and w_j, who are considered equally good as partner of u, we say $\succ_u$ contains a *tie* and use $w_i \sim w_j$ to denote the relation of w_i and w_j in $\succ_u$. The preference list $\succ_w$ of $w \in W$ is defined analogously. A *matching* $M \subseteq \{(u, w) | u \in U \wedge w \in W\}$ is a set of pairwise disjoint pairs. M is *stable* if M does not contain *blocking pairs*; a blocking pair is a pair $\{u, w\} \notin M$ such that u prefers w to $M(u)$ and w prefers u to $M(w)$. We say M is a *perfect matching* if $|M| = n$. If $(u, w) \in M$, we say that w is the *partner* of u matched by M, denoted as $M(u)$, and vice versa. If u has no partner, we define $M(u) = \emptyset$. Given an agent a, $P_a(b)$ stands for the position of b in $\succ_a$, and $\delta(a, b)$ stands for the Euclidean distance between a and b, where b is an agent from the opposite gender. A *preference profile* L is the set of all preference lists. We say L contains ties if at least one preference list in L contains ties and L is incomplete if at least one preference list in L is incomplete.

Preference Lists in Euclidean Space. Each agent $a \in U \cup W$ is represented by a point in the d-dimensional Euclidean space, and the preference list of a is based on the ranking of the Euclidean distances between a and agents from the opposite gender. Let $\delta(a, b)$ be the Euclidean distance between two points a and b. If $\succ_a$ is incomplete, the agents in $\succ_a$ can be inconsistent with the agents of the opposite gender, that is, an agent b might not be a member of $\succ_a$ even if $\delta(b, a) < \delta(c, a)$ and c is in $\succ_a$.

Constraints and Problems. There are four constraints studied in this paper, namely, Regret, Egalitarian, Forced, and Max. We use Reg and Egal to denote Regret and Egalitarian. Each stable marriage problem with a constraint π has an additional input π_i, and an additional requirement π_r for the solution matching M. We define them in Table 2.

Table 2: The four constraints studied.

π	π_i	π_r
Reg	an integer t	$\max_{a \in U \cup W} P_a(M(a)) \leq t$
Egal	an integer t	$\sum_{a \in U \cup W} P_a(M(a)) \leq t$
Forced	a forced pair set F	$F \subseteq M$
Max	an integer t	$\lvert M \rvert \geq t$

Now, we formally define the problem we study in this paper. Let $\pi \in \{$ Reg, Forced, Egal, Max$\}$.

d-Euclidean π-Stable Marriage with Ties (d-Euclid-π-SMT)
Input: Two sets of agents U and W with $|U| = |W| = n$, an embedding $U \cup W \to \mathbb{R}^d$ of the agents into d-dimensional Euclidean space, and π_i.
Question: Is there a matching M satisfying π_r?

3 NP-HARDNESS IN 2-EUCLIDEAN SPACE

In this section, we prove 2-Euclid-π-SMT is NP-hard by providing a polynomial-time reduction from Planar and Cubic Exact Cover by 3 Sets problem [32], which is an NP-complete special case of the Exact Cover by 3 Sets problem [20].

Planar and Cubic Exact Cover by 3 Sets (PC-X3C)
Input: A $3n$-element set $\mathcal{X} = \{x_1, \cdots, x_{3n}\}$ and a collection $\mathcal{S} = \{S_1, \cdots, S_m\}$ with each $S_j \in \mathcal{S}$ being a 3-elements subset of $\mathcal{X}$ and each element occurring in exactly three sets, and the associated graph is planar.
Output: a subcollection $\mathcal{K} \subseteq \mathcal{S}$ such that each $x_i \in X$ occurs in exactly one member of $\mathcal{K}$.

Here, we say a graph $G = (V, E)$ is an associated graph of a PC-X3C instance $I = (\mathcal{X}, \mathcal{S})$, if (1) $V = V_{\mathcal{X}} \cup V_{\mathcal{S}}$ with $V_{\mathcal{X}} = \{v_i | x_i \in \mathcal{X}\}$ and $V_{\mathcal{S}} = \{v_j | S_j \in \mathcal{S}\}$, and (2) $E = \{e_{ij} | x_i \in S_j, x_i \in \mathcal{X}, S_j \in \mathcal{S}\}$. Then, G is a planar graph with the vertex degree being three. Thus, based on the work of Battista et al. [6], G can be embedded in the grid $\mathbb{Z}^2$ in polynomial time, such that its vertices are at the integer grid points and its edges are drawn using at most one horizontal and one vertical segment in the grid.

THEOREM 3.1. *For every $d \geq 2$, d-Euclid-π-SMT is NP-hard. Here, $\pi \in \{Reg, Egal, Forced\}$ and all preference lists are complete.*

Due to lack of space, we only prove the NP-hardness of 2-Euclid-Reg-SMT with $t = 4$. 2-Euclid-π-SMT with $\pi \in \{Egal, Forced\}$ can be proved in a similar way by slightly modifying the auxiliary agents.

3.1 The Construction

Main idea. Given an instance $I = (\mathcal{X}, \mathcal{S})$ of PC-X3C, we first embed the associated graph $G(I)$ into a 2-dimensional grid with edges drawn using line segments of length at least $L > 1000$, and the distance between two parallel line segments is at least L. In the construction process, we used 6 kinds of gadgets, each of which is composed of agents. For each set $S_j \in \mathcal{S}$, we create a set gadget, consisting of four *reception gadgets*, and guarantee that the agents in the gadget can only be matched in two ways, one standing for S_j in the solution, and the other for not. For each element $x_i \in \mathcal{X}$, we create a gadget, called *selection gadget*, whose agents can only be matched in three ways, denoted as $M_{x_i}^{S_a}, M_{x_i}^{S_b}, M_{x_i}^{S_c}$ with S_a, S_b, S_c being three sets containing x_i, each of which encodes how x_i is covered by the PC-X3C solution. We use *chain gadgets* to “link” element gadgets and set gadgets, that is, to coordinate the matching decisions of a element gadget and a set gadget, whose corresponding element x_i and set S_j satisfy $x_i \in S_j$. Thus, the element and set gadgets are placed in the positions of their corresponding elements and sets in the 2-dimensional plane, while the chain gadgets are placed along the edges of $G(I)$. To fulfill their purpose, chain gadgets need to be able to deal with the following three tasks: (1) the matching decision of an element/set gadget need to be broadcasted to its corresponding set/element gadgets; (2) a chain gadget need to be able to turn 90 degrees; (3) chain gadgets can cross each other. For each of the tasks, we create a gadget, which can be considered as a component of chain gadget, called “broadcast gadget”, “turn gadget”, and “crossover gadget”.

Chain gadget. Each chain gadget consists of the so-called “basic components”, each of which contains two pairs of agents. There are two types of basic components, which alternatively occur in a chain gadget. See Figure 1 for an illustration. Given a chain gadget Γ, let $|\Gamma|$ be the number of basic components of Γ, and $\Gamma[i]$ be the i-th component of Γ with $1 \leq i \leq |\Gamma|$. For a Type-A component, we have $\delta(u_{2i-1}^\Gamma, w_{2i}^\Gamma) = 2 + \epsilon$, and for a Type-B component, we have $\delta(w_{2i-1}^\Gamma, u_{2i}^\Gamma) = 2 + \epsilon$, where $\epsilon = \frac{1}{10000}$. Given two components $\Gamma[i]$ and $\Gamma[i+1]$ of chain gadget Γ, define $\delta(\Gamma[i], \Gamma[i+1]) = \delta(u_{2i}^\Gamma, w_{2i+1}^\Gamma)$ if $\Gamma[i]$ is Type-A, and $\delta(\Gamma[i], \Gamma[i+1]) = \delta(w_{2i}^\Gamma, u_{2i+1}^\Gamma)$ if $\Gamma[i]$ is Type-B. For each $1 \leq i \leq |\Gamma| - 1$, we set $\sqrt{2} < \delta(\Gamma[i], \Gamma[i+1]) \leq 2$. In one chain gadget, we always let $\Gamma[i]$ and $\Gamma[i+1]$ be on a straight line, except for turn gadget. A chain gadget must be set between two agents or gadgets. We say “a chain from P to Q” if $\Gamma[1]$ is around the position of P and $\Gamma[|\Gamma|]$ is around the position of Q with P and Q being agents or gadgets, denoted as $\Gamma(P, Q)$. In the following, we will use “chain” for “chain gadget”.

Selection gadget. For each $x_i \in \mathcal{X}$ with $x_i \in S_a, S_b, S_c$, we create one selection gadget. A selection gadget, denoted as Δ^{x_i}, consists of three men $u_{S_a}^{x_i}, u_{S_b}^{x_i}, u_{S_c}^{x_i}$, and three women $w_1^{x_i}, w_2^{x_i}, w_3^{x_i}$. We put $w_1^{x_i}$ and $w_2^{x_i}$ at the same position. The distance between $w_1^{x_i}$ (or $w_2^{x_i}$) and $u_{S_a}^{x_i}$ (or $u_{S_b}^{x_i}, u_{S_c}^{x_i}$) equals 1, which is less than the distance between $w_3^{x_i}$ and the three men, which is at least $\sqrt{5}$. See Figure 2a for an illustration. There are three chains around Δ^{x_i}. The distances between $u_{S_a}^{x_i}, u_{S_b}^{x_i}, u_{S_c}^{x_i}$ and their respective closest chains are equal to 2.

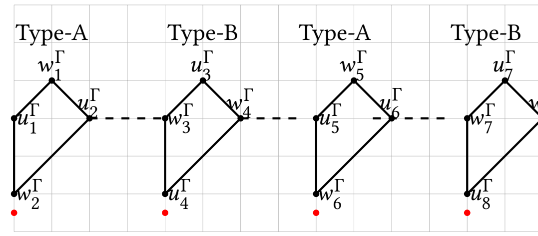


Figure 1: A chain gadgets Γ with $|\Gamma| = 4$. There are $\hat{d}$ pairs of auxiliary agents in each red point, where $\hat{d}$ is an integer that is set equal to the regret score bound t. Note that the basic component only has 4 agents, the lines between the agents are just to highlight that the agents are from the same component.

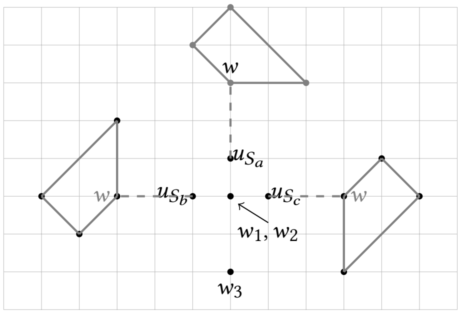
Reception gadget. For each $S_j \in \mathcal{S}$, we create four reception gadgets to form an “S_j-square” with one for each side, denoted as $\Lambda_0^{S_j}$ and $\Lambda_{x_i}^{S_j}$ with $x_i \in S_j$. Each reception gadget has 7 pairs of agents. See Figure 2b for an illustration. Given a reception gadget $\Lambda_{x_i}^{S_j}$ with $S_j \in \mathcal{S}$ and $x_i \in S_j$, there are three chains that link $\Lambda_{x_i}^{S_j}$ to the selection gadget of x_i. The distances between the chains and their respective closest agents from $\Lambda_{x_i}^{S_j}$ are equal to 2, as illustrated in Figure 2b. We use “YES-chain” to denote the chain in the middle, and use “NO-chains” to denote the other two chains.

Crossover gadget. A crossover gadget does not introduce new agents and consists of two basic components of each of two chains which cross each other. Given two chains Γ_1 and Γ_2 that cross with each other, we let $\Gamma_1[i], \Gamma_1[i+1], \Gamma_2[j], \Gamma_2[j+1]$ be the four components forming the crossover gadget. Here, we need to reset the positions of agents in $\Gamma_1[i+1]$ and $\Gamma_2[j+1]$, that is, exchange the positions of $w_{2(i+1)}^{\Gamma_1}$ and $w_{2(i+1)-1}^{\Gamma_1}$ if $\Gamma_1[i+1]$ is Type-A, or exchange the positions of $u_{2(i+1)}^{\Gamma_1}$ and $u_{2(i+1)-1}^{\Gamma_1}$ if $\Gamma_1[i+1]$ is Type-B. The positions of agents of $\Gamma_2[j+1]$ can be reset in a similar way. See Figure 2c for an illustration.

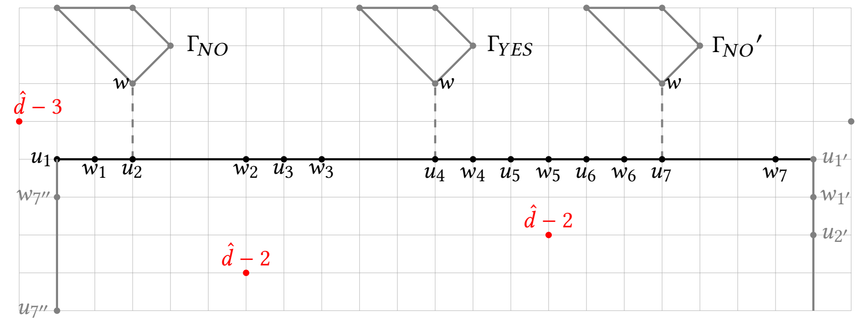
Turn gadget. See Figure 2d for an illustration.

Broadcast gadget. Broadcast gadget consists of 9 pairs of agents, which form a “square”. Each side of the “square” is linked by a chain to either selection gadgets or reception gadgets. The distances between the chains and their respective closest agents in the broadcast gadgets are all equal to 2. We use “IN-chain” to denote the chain that links broadcast gadget to selection gadgets, and “OUT-chain” to denote the chains to reception gadgets. A broadcast gadget has exactly one IN-chain and three OUT-chains. See Figure 2e for an illustration.

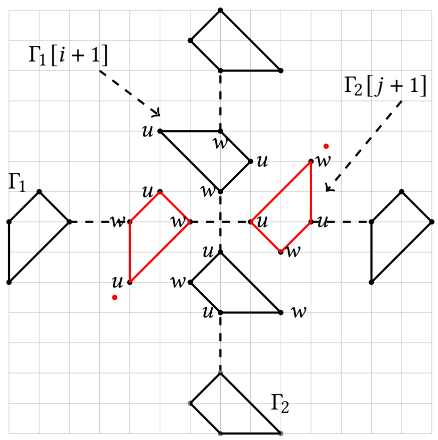
For each element $x_i \in \mathcal{X}$, we create one selection gadget and three broadcast gadgets, denoted as Δ^{x_i} and $B_{S_a}^{x_i}, B_{S_b}^{x_i}, B_{S_c}^{x_i}$ with $x_i \in$



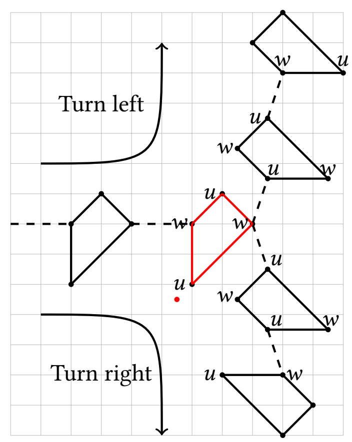
(a) A selection Gadget for $x_i \in \mathcal{X}$ with $x_i \in S_a, S_b, S_c$.



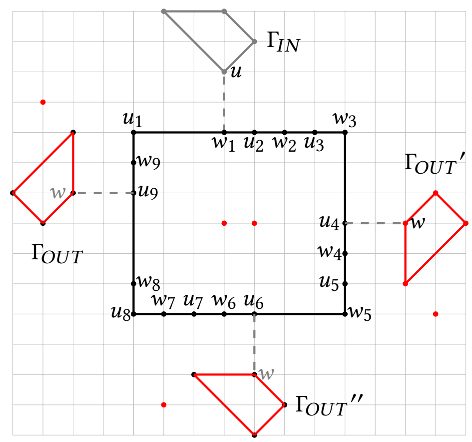
(b) A reception gadget for $S_j \in \mathcal{S}$. Note that a set needs four such gadgets to form an "S_j-square". For each red point there are $\hat{d}-3$ or $\hat{d}-2$ pairs of auxiliary agents. the gray agents are created by reception gadgets from other sides.



(c) A crossover gadget formed by $\Gamma_1[i], \Gamma_1[i+1], \Gamma_2[j], \Gamma_2[j+1]$ with Γ_1 and Γ_2 being two chains. Note that the positions of some agents of $\Gamma_1[i+1]$ and $\Gamma_2[j+1]$ have been reset. In addition, the positions of auxiliary agents of red components have been reset.



(d) Turn left and turn right gadget. We remove the auxiliary agents of the red component. Moreover, we add $\hat{d}-3$ auxiliary agents to the red points.



(e) Broadcast gadget. We remove the auxiliary agents of the red component and add $\hat{d}-2$ auxiliary agents to each red points.

Figure 2: Five Gadgets used in the construction. $\hat{d}$ is set equal to the regret score bound t. The auxiliary agents of chains are omitted if not stated explicitly. We also omit the superscripts of agents in all five figures to make the figures clearer.

S_a, S_b, S_c and $S_a, S_b, S_c \in \mathcal{S}$. For each broadcast gadget $B^{x_i}_{S_k}$ with $k \in \{a, b, c\}$, we create a chain $\Gamma(u^{x_i}_{S_k}, B^{x_i}_{S_k})$ with $u^{x_i}_{S_k}$ being the man created in Δ^{x_i}, and let it be the IN-chain of $B^{x_i}_{S_k}$.

For each $S_j \in \mathcal{S}$, we create four reception gadgets, denoted as $\Lambda^{S_j}_{x_\alpha}, \Lambda^{S_j}_{x_\beta}, \Lambda^{S_j}_{x_\gamma}$ and $\Lambda^{S_j}_0$ with $S_j = \{x_\alpha, x_\beta, x_\gamma\}$.

For each pair of $(B^{x_i}_{S_j}, \Lambda^{S_k}_{x_i})$ with $x_i \in S_j, S_k$, we create a chain $\Gamma(B^{x_i}_{S_j}, \Lambda^{S_k}_{x_i})$. The chain is an OUT-chain of $B^{x_i}_{S_j}$, and is a YES-chain of $\Lambda^{S_k}_{x_i}$ if $S_j = S_k$ (or a NO-chain of $\Lambda^{S_k}_{x_i}$ if $S_j \neq S_k$). See Figure 3 for a concrete example.

Next, we set $\hat{d} = t = 4$ with t being the regret score bound, and create auxiliary agents for each gadget as we show in Figure 1 and Figure 2.

Finally, for each agent a created in the construction, we compute a preference list based on the distances between a and opposite gender agents.

3.2 The correctness proof

Before showing the equivalence of the instances, we prove several properties of the constructed instance. Here, M is the solution matching of I' with I' being the instance created as shown before.

Lemma 3.2. *Given a chain Γ, the following hold. Here, $1 \le i \le |\Gamma|$, and assume that both $\Gamma[1]$ and $\Gamma[|\Gamma|]$ are Type-A.*

(1) *Each component $\Gamma[i]$ can only be matched in two ways, that is*
  (a) $\{(u^\Gamma_{2i-1}, w^\Gamma_{2i-1}), (u^\Gamma_{2i}, w^\Gamma_{2i})\} \subseteq M$, *"Forward"-matching,*
  (b) $\{(u^\Gamma_{2i-1}, w^\Gamma_{2i}), (u^\Gamma_{2i}, w^\Gamma_{2i-1})\} \subseteq M$, *"Backward"-matching.*
(2) *If $\Gamma[i]$ is matched as "Forward"-matching, then $\Gamma[i+1]$ must be matched as "Forward"-matching. If $\Gamma[i+1]$ is matched as "Backward"-matching, then $\Gamma[i]$ must be matched as "Backward"-matching.*
(3) *Given a matching M, if there is a woman w_+ with $\sqrt{2} < \delta(w_+, u^\Gamma_1) \le 2$ and $\delta(w_+, M(w_+)) > \delta(w_+, u^\Gamma_1)$, then Γ must be matched as "Forward"-matching. Here, we say Γ is matched*

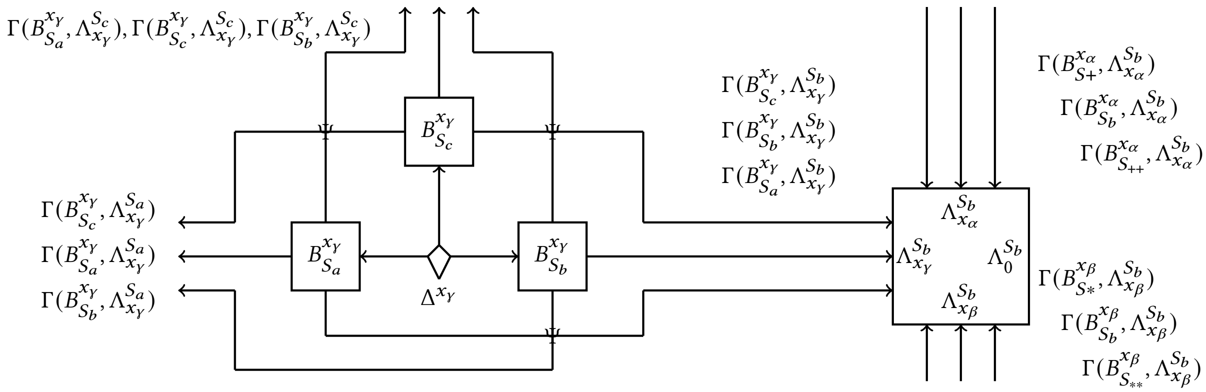


Figure 3: Gadgets created for x_γ and S_b with $x_\gamma \in S_a, S_b, S_c$ and $S_b = \{\alpha, \beta, \gamma\}$. Here, $x_\alpha \in S_b, S_+, S_{++}$ and $x_\beta \in S_b, S_*, S_{**}$.

as "Forward"-matching if all components of Γ are matched as "Forward"-matching. The "Backward"-matching of Γ is defined in a similar way.

(4) *Given a matching M, if there is a woman w_- with $\sqrt{2} < \delta(w_-, u^\Gamma_{2|\Gamma|}) \leq 2$ and $\delta(w_-, M(w_-)) > \delta(w_-, u^\Gamma_{2|\Gamma|})$, then Γ must be matched as "Backward"-matching.*

Proof. We first prove (1). Assume that $\Gamma[i]$ is Type-A. Throughout the construction, there is no man u_* with $\delta(u_*, w^\Gamma_{2i-1}) \leq \sqrt{2}$ even in crossover and turn gadgets. Thus, w^Γ_{2i-1} prefers u^Γ_{2i-1} and u^Γ_{2i} to other men. The distances between u^Γ_{2i-1} (or u^Γ_{2i}) and all women except for w^Γ_{2i} are greater than $\sqrt{2}$, which implies that u^Γ_{2i-1} and u^Γ_{2i} prefer w^Γ_{2i-1} to other women. Thus, w^Γ_{2i-1} can only be matched to either u^Γ_{2i-1} or u^Γ_{2i}. The first $\hat{d}$ men in w^Γ_{2i} can only be members in $\{u^\Gamma_{2i-1}, u^\Gamma_{2i}, u^{\Gamma'}_{2j-1}\} \cup \{u | u \text{ is an auxiliary man}\}$ with Γ' being another chain and $1 \leq j \leq |\Gamma'|$. Thus, like w^Γ_{2i-1}, w^Γ_{2i} can only be matched to either u^Γ_{2i-1} or u^Γ_{2i}. Considering M is a perfect matching, we have either $\{(u^\Gamma_{2i-1}, w^\Gamma_{2i-1}), (u^\Gamma_{2i}, w^\Gamma_{2i})\} \subseteq M$, or $\{(u^\Gamma_{2i-1}, w^\Gamma_{2i}), (u^\Gamma_{2i}, w^\Gamma_{2i-1})\} \subseteq M$.

For (2), if $\{(u^\Gamma_{2i-1}, w^\Gamma_{2i-1}), (u^\Gamma_{2i}, w^\Gamma_{2i})\} \subseteq M$, then $w^\Gamma_{2(i+1)-1}$ can only be matched to $u^\Gamma_{2(i+1)-1}$ since $\sqrt{2} < \delta(u^\Gamma_{2i}, w^\Gamma_{2(i+1)-1}) \leq 2 \leq 2 + \epsilon = \delta(w^\Gamma_{2(i+1)-1}, u^\Gamma_{2(i+1)})$, Here, $\Gamma[i+1]$ is Type-B since we assume that $\Gamma[i]$ is Type-A. The other case can be proved in a similar way. (3) and (4) can be seen as a simple extension of (2), and can be proved in a similar way. □

Lemma 3.3. *Given a selection gadget Δ^{x_i} with $x_i \in \mathcal{X}$ and $x_i \in S_a, S_b, S_c$, the following holds.*

(1) *$u^{\Delta^{x_i}}_{S_a}, u^{\Delta^{x_i}}_{S_b}, u^{\Delta^{x_i}}_{S_c}$ can only be matched to $w^{\Delta^{x_i}}_1, w^{\Delta^{x_i}}_2, w^{\Delta^{x_i}}_3$. Let $S_j \in \{S_a, S_b, S_c\}$, we say Δ^{x_i} matched as "S_j-Yes"-matching if $(u^{\Delta^{x_i}}_{S_j}, w^{\Delta^{x_i}}_3) \in M$.*

(2) *If Δ^{x_i} is matched as "S_j-Yes"-matching, then $\Gamma(\Delta^{x_i}, B^{x_i}_{S_j})$ must be matched as "Forward"-matching.*

Proof. Both $w^{\Delta^{x_i}}_1$ and $w^{\Delta^{x_i}}_2$ prefer $u^{\Delta^{x_i}}_{S_a}, u^{\Delta^{x_i}}_{S_b}, u^{\Delta^{x_i}}_{S_c}$ to other men and vice versa. So $w^{\Delta^{x_i}}_1, w^{\Delta^{x_i}}_2$ can only be matched to the three men. The first $\hat{d}$ men in the preference list of $w^{\Delta^{x_i}}_3$ is $u^{\Delta^{x_i}}_{S_a}, u^{\Delta^{x_i}}_{S_b}, u^{\Delta^{x_i}}_{S_c}$ and u^Γ_2 with $\Gamma = \Gamma(\Delta^{x_i}, B^{x_i}_{S_j})$. The last man cannot be matched to $w^{\Delta^{x_i}}_3$ by Lemma 3.2. Thus, $w^{\Delta^{x_i}}_3$ can only be matched to $u^{\Delta^{x_i}}_{S_a}, u^{\Delta^{x_i}}_{S_b}, u^{\Delta^{x_i}}_{S_c}$. For (2), we have $\delta(\Delta^{x_i}, B^{x_i}_{S_j}) = 2$. Then $\Gamma(\Delta^{x_i}, B^{x_i}_{S_j})$ must be matched as "Forward"-matching by Lemma 3.2. □

Lemma 3.4 (*). *Given a broadcast gadget $B^{x_i}_{S_j}$ with $x_i \in S_j$, $x_i \in \mathcal{X}$ and $S_j \in \mathcal{S}$, the following hold. Here, let $B^* = B^{x_i}_{S_j}$.*

(1) *B^* can only be matched in two ways, either $\{(u^{B^*}_k, w^{B^*}_k) | 1 \leq k \leq 9\} \subseteq M$, or $\{(u^{B^*}_k, w^{B^*}_{k-1}) | 1 \leq k \leq 9\} \subseteq M$ with $w^{B^*}_0 = w^{B^*}_9$. We say the first matching is "Forward"-matching of B^* and the second is "Backward"-matching of B^*.*

(2) *If the IN-chain of B^* is matched as "Forward"-matching, then B^* and the three OUT-chains must be matched as "Forward"-matchings.*

(3) *If there is an OUT-chain of B^* matched as "Backward"-matching, then B^* and the IN-chain of B^* must be matched as "Backward"-matchings.*

Lemma 3.5 (*). *Given a reception gadget $\Lambda^{S_j}_{x_\gamma}$ with $x_\gamma \in \mathcal{X}$, $S_j \in \mathcal{S}$ and $S_j = (x_\alpha, x_\beta, x_\gamma)$, the following holds. Here, we assume that $\Lambda^{S_j}_{x_\gamma}$ is on the top side of the "S_j-square", and $\Lambda^{S_j}_{x_\alpha}, \Lambda^{S_j}_{x_\beta}$ are on the left and right side, respectively. We let $\Lambda^* = \Lambda^{S_j}_{x_\gamma}$, $\Lambda^- = \Lambda^{S_j}_{x_\alpha}$.*

(1) *Λ^* can only be matched in two ways, either $\{(u^{\Lambda^*}_k, w^{\Lambda^*}_k) | 1 \leq k \leq 7\} \subseteq M$, or $\{(u^{\Lambda^*}_k, w^{\Lambda^*}_{k-1}) | 1 \leq k \leq 7\} \subseteq M$ with $w^{\Lambda^*}_0 = w^{\Lambda^-}_7$. We say the first matching is "Yes"-matching of Λ^* and the second is "No"-matching of Λ^*.*

(2) *If the YES-chain of Λ^* is matched as "Forward"-matching, then Λ^* must be matched as "Yes"-matching, and the two NO-chains must be matched as "Backward"-matching.*

(3) *If at least one NO-chain of Λ^* is matched as "Forward", then Λ^* must be matched as "No"-matching, and the YES-chain must be matched as "Backward"-matching.*

(4) *Reception gadgets of the four sides of S_j-square must be matched in the same way.*

By Lemma 3.3, Lemma 3.4, and Lemma 3.5, we can get the following corollary.

Corollary 3.6. *Given a matching M, let $x_\gamma \in \mathcal{X}$ be a member of $S_a, S_b, S_c \in \mathcal{S}$, and $S_b = x_\alpha, x_\beta, x_\gamma$ with $x_\alpha, x_\beta \in \mathcal{X}$. The following hold.*

(1) *If Δ^{x_γ} is matched as "S_b-Yes"-matching, then $\Lambda^{S_b}_{x_\gamma}$ must be matched as "Yes"-matching and $\Lambda^{S_a}_{x_\gamma}$, and $\Lambda^{S_c}_{x_\gamma}$ must be matched as "No"-matchings.*
(2) *If $\Lambda^{S_b}_{x_\gamma}$ is matched as "Yes"-matching, then Δ^{x_α}, Δ^{x_β} and Δ^{x_γ} must be matched as "S_b-Yes"-matchings. If $\Lambda^{S_b}_{x_\gamma}$ is matched as "No"-matching, then none of Δ^{x_α}, Δ^{x_β}, Δ^{x_γ} is matched as "S_b-Yes"-matching.*

Lemma 3.7 (*). *Given a crossover gadget formed by $\Gamma_1[i]$, $\Gamma_1[i+1]$, $\Gamma_2[j]$, and $\Gamma_2[j+1]$ with Γ_1 and Γ_2 being two chains. We have either Γ_1 is matched as "Forward"-matching and Γ_2 is matched as "Backward"-matching, or Γ_1 is matched as "Backward"-matching and Γ_2 is matched as "Forward"-matching.*

Now, we have all tools to prove the equivalence of the instance. "⇒": Given a solution of $I = (\mathcal{X}, \mathcal{S})$ of PC-X3C, we can construct a matching of $I' = (U, W, L, t)$ as follows. For each $S_j \in \mathcal{K}$ with $S_j = \{x_\alpha, x_\beta, x_\gamma\}$, we let

(1) $\Lambda^{S_j}_{x_\alpha}, \Lambda^{S_j}_{x_\beta}, \Lambda^{S_j}_{x_\gamma}, \Lambda^{S_j}_0$ match as "Yes"-matchings,
(2) $\Delta^{x_\alpha}, \Delta^{x_\beta}, \Delta^{x_\gamma}$ match as "S_j-Yes"-matching.

For each $x_i \in S_j, S_k, S_\ell$ with Δ^{x_i} matched as "S_j-Yes"-matching, we let

(1) $B^{x_i}_{S_j}$ match as "Forward"-matching.
(2) $B^{x_i}_{S_k}, B^{x_i}_{S_\ell}$ match as "Backward"-matchings
(3) $\Gamma(u^{x_i}_{S_j}, B^{x_i}_{S_j})$, $\Gamma(B^{x_i}_{S_j}, \Lambda^{S_j}_{x_i})$, $\Gamma(B^{x_i}_{S_j}, \Lambda^{S_k}_{x_i})$, $\Gamma(B^{x_i}_{S_j}, \Lambda^{S_\ell}_{x_i})$ match as "Forward"-matchings.
(4) $\Gamma(u^{x_i}_{S_k}, B^{x_i}_{S_k})$, $\Gamma(B^{x_i}_{S_k}, \Lambda^{S_j}_{x_i})$, $\Gamma(B^{x_i}_{S_k}, \Lambda^{S_k}_{x_i})$, match as "Backward"-matchings.
(5) $\Gamma(u^{x_i}_{S_\ell}, B^{x_i}_{S_\ell})$, $\Gamma(B^{x_i}_{S_\ell}, \Lambda^{S_j}_{x_i})$, $\Gamma(B^{x_i}_{S_\ell}, \Lambda^{S_\ell}_{x_i})$, match as "Backward"-matchings.
(6) $\Gamma(B^{x_i}_{S_k}, \Lambda^{S_\ell}_{x_i})$ match as "Forward"-matching and $\Gamma(B^{x_i}_{S_\ell}, \Lambda^{S_k}_{x_i})$ match as "Backward"-matching.

Note that $\Gamma(B^{x_i}_{S_k}, \Lambda^{S_\ell}_{x_i})$ and $\Gamma(B^{x_i}_{S_\ell}, \Lambda^{S_k}_{x_i})$ cross each other, so we let one of them be "Forward"-matching and let the other be "Backward"-matching. Obviously, the constructed matching is stable by Lemma 3.2 to Lemma 3.5, Corollary 3.6 and Lemma 3.7.
"⇐": Given a matching M, construct a $\mathcal{K}$ as follows. For each $x_i \in \mathcal{X}$ with $x_i \in S_a, S_b, S_c$, we let S_j be a member of $\mathcal{K}$ if $\Lambda^{S_j}_{x_i}$ is matched as"Yes"-matching. We claim that only one of $\Lambda^{S_a}_{x_i}$, $\Lambda^{S_b}_{x_i}$, $\Lambda^{S_c}_{x_i}$ is matched as"Yes"-matching. Suppose there are two of $\Lambda^{S_a}_{x_i}$, $\Lambda^{S_b}_{x_i}$, $\Lambda^{S_c}_{x_i}$ matched as"Yes"-matching, for instance, both $\Lambda^{S_a}_{x_i}, \Lambda^{S_b}_{x_i}$ are matched as "Yes"-matching. Then both $\Gamma(B^{x_i}_{S_a}, \Lambda^{S_a}_{x_i})$ and $\Gamma(B^{x_i}_{S_b}, \Lambda^{S_a}_{x_i})$ are matched as "Forward"-matchings. Then, an agent in $\Gamma(B^{x_i}_{S_b}, \Lambda^{S_a}_{x_i})$ will form a blocking pair with an agent in $\Lambda^{S_a}_{x_i}$, and M is not a stable matching. Suppose none of $\Lambda^{S_a}_{x_i}, \Lambda^{S_b}_{x_i}, \Lambda^{S_c}_{x_i}$ is matched as "Yes"-matching, then we have $B^{X_i}_{S_a}, B^{X_i}_{S_b}, B^{X_i}_{S_c}$ are matched as "Backward"-matching. Thus, at least one blocking pair is formed by them with Δ^{x_i} by Lemma 3.3, which implies that M is not stable. □

4 RESULTS IN 1-EUCLIDEAN SPACE

In this section, we study 1-Euclid-π-SMT, that is, that agents in $U \cup W$ are embedded into a line. Without loss of generality, we assume that the line is horizontal. Under this setting, 1-Euclid-Reg-SMT and 1-Euclid-Forced-SMT can be solved in polynomial time. If preference lists are incomplete, 1-Euclid-π-SMT is NP-hard for all $\pi \in$ {Reg, Egal, Forced, Max}.

4.1 1-Euclid-Reg-SMT

We introduce some notations that will be used in the algorithms.
Free-agent and reselecting-agent. Given a line which U and W are embedded into, a man $u_0 \in U$ and two women w_1, w_2 with w_1 is on the left side of u_0 and w_2 is on the right side of u_0, and a stable matching $M_1 \subseteq U \times W$ with $\{u_0, w_1\} \in M_1$, we say u_0 is a *free-man* between w_1 and w_2 in M_1 if $\delta(u_0, w_1) = \delta(u_0, w_2)$ and $\delta(u_0, w_2) < \delta(M_1(w_2), w_2)$. We can define *free-woman* in a similar way, and we call both free-men and free-women as free-agents. Given U, W, a stable matching M_1, a free-man $u_0 \in U$ between two different women $w_1, w_2 \in W$ in M_1, the following hold.

Lemma 4.1. *If $(u_0, w_1) \in M_1$, then there exists another stable matching M_2 with $(u_0, w_2) \in M_2$.*

Proof. Since $\delta(u_0, w_2) < \delta(M_1(w_2), w_2)$, w_2 must prefer u_0 to $M_1(w_2)$. (u_0, w_1) is not a blocking pair since $\delta(u_0, w_1) = \delta(u_0, w_2)$. We can construct a matching M_2 as follows. Let $u_2 = M_1(w_2)$. First, let $M_2 = M_1 \setminus \{(u_0, w_1), (u_2, w_2)\} \cup \{\{u_0, w_2\}, \{u_2, w_1\}\}$. Then, we check whether there is a man u_3 with $w_3 = M_2(u_3)$, such that there is no man u_4 with $w_4 = M_2(u_4)$ with

(1) $\delta(u_3, w_1) < \delta(u_2, w_1)$,
(2) $\delta(u_3, w_1) < \delta(u_3, w_3)$,
(3) $\delta(u_4, w_1) < \delta(u_4, w_4)$,
(4) $\delta(u_4, w_1) < \delta(u_3, w_1)$.

If so, let $M_2 = M_2 \setminus \{(u_3, w_3), (u_2, w_1)\} \cup \{\{u_3, w_1\}, \{u_2, w_3\}\}$. Repeat until there is no man satisfying the requirement. Then, we do a similar process to u_2, that is, finding the closest woman who prefers u_2 to her partner, matching them together, and repeat this process until no woman satisfies the requirement. Thus, we can claim that M_2 is stable, since there is no pair of agents who prefer each other to their partners matched by M_2. □

We say u_0 reselects his partner from M_1 to M_2, and say u_0 is a *reselecting-agent.*
Vanish-set and newborn-set. Let $M_{\text{old}} = M_1 \setminus (M_1 \cap M_2)$ and $M_{\text{new}} = M_2 \setminus (M_1 \cap M_2)$, we have

- $|M_{\text{old}}| = |M_{\text{new}}|$.
- $U_{\text{old}} = U_{\text{new}}$ and $W_{\text{old}} = W_{\text{new}}$.

Here, $U_{\text{old}} = \{u_i | u_i \in U, w_j \in W, (u_i, w_j) \in M_{\text{old}}\}$. The other three sets can be defined in a similar way. We say M_{old} is the *vanish-set* caused by u_0, and say M_{new} is a *newborn-set* caused by u_0.
Rematch-chain. Let $(u_{\text{end}}, w_{\text{end}})$ be the farthest pair in $M_{\text{old}} \cup M_{\text{new}}$, that is, $\delta(u_{\text{end}}, w_{\text{end}}) > \delta(u_i, w_j)$ with $(u_i, w_j) \in M_{\text{old}} \cup M_{\text{new}} \setminus \{u_{\text{end}}, w_{\text{end}}\}$. We create a $(|U_{\text{new}}| + |W_{\text{new}}|)$-size sorted set of agents, denoted as C, and set it as follows. First, let w_{end} be $C[1]$ with $C[i]$ being the i-th agent in C. Then, we construct $C[2]$ to $C[n]$ in two ways. If $\{u_{\text{end}}, w_{\text{end}}\} \in M_{\text{old}}$, let $M_2(w_{\text{end}})$ be $C[2]$ and $M_1(M_2(w_{\text{end}}))$, denoted as w', be $C[3]$. Then let $M_2(w')$ be

$C[4]$ and $M_1(M_2(w'))$ be $C[5]$, and so on. If $\{u_{\text{end}}, w_{\text{end}}\} \in M_{\text{new}}$, let $M_1(w_{\text{end}})$ be $C[2]$ and $M_2(M_1(w_{\text{end}}))$, denoted as w', be $C[3]$. Then let $M_1(w')$ be $C[4]$ and $M_2(M_1(w'))$ be $C[5]$, and so on. We say C is a *rematch-chain* caused by u_0. Assume that u_0 is $C[i]$. Recall that u_0 is a free-agent and w_1, w_2 are the two partners that u_0 can reselect. We can get the following observation based on the definition of C, that is, $C[j]$ prefers $C[j+1]$ to $C[j-1]$ if $j < i$, named the "men-happy" side of C, and $C[j]$ prefers $C[j-1]$ to $C[j-1]$ if $j > i$, named the "women-happy" side C.

Given an agent $a \in U \cup W$ and a pair (u, w) with $u \in U, w \in W$, we say a is between (u, w) if a is between u and w. We say a pair $(u_{\text{end}}, w_{\text{end}})$ is the farthest pair of a set P if $\delta(u_{\text{end}}, w_{\text{end}}) > \delta(u_i, w_j)$ with $(u_i, w_j) \in P \setminus \{u_{\text{end}}, w_{\text{end}}\}$. Given two stable matchings M_1, M_2 with M_2 being the resulting matching after a free-agent $a_0 \in U \cup W$ reselects his/her partner from M_1, let $M_{\text{old}}, M_{\text{new}}$ be the vanish- and newborn-sets of a_0, and C be the rematch-chain caused by a_0, and a_0 is between a pair $(u_{\text{arc}}, w_{\text{arc}}) \in M_1$.

Lemma 4.2. *Let $(u_{\text{end}}, w_{\text{end}})$ be the farthest pair in $M_{\text{old}} \cup M_{\text{new}}$. Let $a_{\text{in}} \in U \cup W$ be an agent who is between $(u_{\text{end}}, w_{\text{end}})$ with $a_{\text{in}} \notin (U_{\text{old}} \cup W_{\text{old}})$, and $a_{\text{out}} \in U \cup W$ be an agent who is not between $(u_{\text{arc}}, w_{\text{arc}}) \in M_1$, the following holds*

(1) *Either $(u_{\text{end}}, w_{\text{end}}) \in M_1$, or $(u_{\text{end}}, w_{\text{end}}) \in M_2$.*
(2) *If $C[i]$ is on the left side of $C[i+1]$, then $C[i-1]$ is on the left side of $C[i]$.*
(3) *a_0 is between u_{end} and w_{end}.*
(4) *$\{a_{\text{out}}\} \cap (U_{\text{old}} \cup W_{\text{old}}) = \emptyset$.*

Proof. (1) is true since $M_{\text{old}} \cap M_{\text{new}} = \emptyset$. For (2), if $C[i]$ is on the left side of a_0, we have $C[i]$ prefers $C[i+1]$ to $C[i-1]$. if $C[i-1]$ is between $C[i]$ and $C[i+1]$, then $C[i]$ must prefer $C[i-1]$ to $C[i+1]$, a contradiction. If $C[i]$ is on the right side of a_0, we have $C[i]$ prefers $C[i-1]$ to $C[i+1]$, and $C[i-1]$ prefers $C[i-2]$ to $C[i]$, and so on. If $C[i-1]$ is between $C[i]$ and $C[i+1]$, then $C[i-2]$ must between $C[i-1]$ and $C[i+1]$, then $C[i-3]$ must between $C[i-2]$ and $C[i+1]$, that is, each $C[j-1]$ must between $C[j]$ and $C[i+1]$ with $j < i$, which implies that u_0 must be in the right side of $C[i]$, a contradiction. In the definition, we always let w_{end} be $C[1]$ and u_{end} be $C[n]$, thus, (3) is true by (2). For (4), assume that a_out is in the woman happy side. If $\{a_{\text{out}}\} \cap (U_{\text{old}} \cup W_{\text{old}})$, a_out must have an index greater than u_{end} in C by (2), which implies that $(u_{\text{end}}, w_{\text{end}})$ is not the farthest pair. □

Given a line which U, W are embedded into, we can safely match and remove a man-woman pair if the distance of them is less than all other pairs. We call the process, that matching and removing the closet man-woman pair repeatedly until no such pair exists, as *Update.* We say a man $u_0 \in W$ between $w_1, w_2 \in W$ is a *chosen-man* if $\delta(u_0, w_1) = \delta(u_0, w_2)$ and $\delta(u_0, w_1)$ is less than all distances of other man-woman pairs. We can define *chosen-woman* in a similar way. Let U^L be the men on the left side of u_0, and let $U_1^L \subseteq U^L$ be the resulting set of U^L after removing u_0, w_1 and updating, let $U_2^L \subseteq U^L$ be the resulting set of U^L after removing u_0, w_2 and updating. Define W^L, W_1^L, and W_2^L analogously. If $||U_1^L| - |W_1^L|| \leq ||U_2^L| - |W_2^L||$, we say w_1 is the *better-partner* of u_0; otherwise, we say w_2 is the *better-partner* of u_0.

Theorem 4.3. *1-Euclid-Reg-SMT can be solved in polynomial time when preference profile is complete.*

Proof. Given an instance I of 1-Euclid-Reg-SMT with an input line that U and W are embedded into, let $M = \emptyset$. We do the following process repeatedly until $|M| = |U|$:

(1) Update the line and set $M = M \cup M_{update}$ with M_{update} being the pairs matched through the update process.
(2) Find a chosen-agent and match him/her to the better-partner, denoted as (u_c, w_c). Set $M = M \cup (u_c, w_c)$ and remove u_c and w_c from the line.

The correctness of the algorithm follows from Lemma 4.1 and Lemma 4.2. Let (u_c^i, w_c^i) be the i-th chosen-pair added to M. Here, we say (u_c^i, w_c^i) is a chosen-pair if one of u_c^1 and w_c^1 is a chosen-agent. Given a stable matching M_0 with $(u_c^1, w_c^1) \notin M_0$, there must be another matching M_1 with $(u_c^1, w_c^1) \in M_1$ by Lemma 4.1, since the chosen-agent is obviously a free-agent of M_0. Since the chosen-agent, i.e., u_c^1, does not match with his better-partner, we have the men on the left side of u_c^1, denoted as U^L, are not equal to the women on the left side of u_c^1, denoted as U^R. Thus, at least one agent $a_{\text{arc}} \in U^L \cup U^R$ must be matched to an agent b_{arc} who is on the right side of u_c^1. By Lemma 4.2, all agents in the rematch-chain are between a_{arc} and b_{arc}. Thus, M_2 is a solution matching of I if M_1 is a solution of I, since $(a_{\text{arc}}, b_{\text{arc}}) \in M_1$. Similarly, there must be another matching M_2 with $(u_c^2, w_c^2) \in M_2$, if $(u_c^2, w_c^2) \notin M_1$. Thus, M is a solution matching of I. □

4.2 1-Euclid-Forced-SMT

Let M_1, M_2, M_3 be three stable matchings with M_2 (or M_3) being the resulting matching after a free-agent a_1(or a_2) $\in U \cup W$ reselects his/her partner from M_1 (or M_2). Given a pair $\{u_{\text{arc}}, w_{\text{arc}}\} \in M_1$ with a_1, a_2 being between u_{arc} and w_{arc}, the following holds.

Lemma 4.4. *Given an agent a_{out} who is not between u_{arc} and w_{arc}, we have a_{out} is neither a member of C_1 nor a member of C_2, where C_1 and C_2 are rematch-chains caused by a_1 and a_2, respectively.*

Proof. a_{out} is not a member of C_1 by Lemma 4.2.(4).

If $(u_{\text{arc}}, w_{\text{arc}}) \in M_2$, then $a \notin C_2$ by Lemma 4.2.(4). If $(u_{\text{arc}}, w_{\text{arc}}) \in M_2$, then there is an index i with $1 < i < |C_1|$, such that a_2 is between $C_1[i]$ and $C_1[i+1]$. If $(C_1[i], C_1[i+1]) \in M_2$, then a_{out} is not a member of C_2 since $(u_{\text{arc}}, w_{\text{arc}}) \in M_2$. If $(C_1[i], C_1[i+1]) \in M_1$, then a_2 is also a free-agent in M_1. Thus, let a_2 reselect his/her partner in M_1 and get a resulting matching M_2'. We have a_1 is still a free-agent in M_2', and $(u_{\text{arc}}, w_{\text{arc}}) \in M_2'$. Then let a_1 reselect his/her partner in M_2' and denote the resulting matching as M_3'. Obviously, $M_3 = M_3'$. Let C_1' denote the rematch-chain caused by a_1 from M_2' to M_3'. Then, a_{out} is not a member of C_1', since a_1 is between $(u_{\text{arc}}, w_{\text{arc}}) \in M_2'$. □

Lemma 4.5. *Given a matching M, let $(u_1, w_1), (u_2, w_2)$ be two pairs matched in M. If one of u_2, w_2 is between $\{u_1, w_1\}$ and the other is not, then M is not a stable matching.*

Proof. Assume that u_2 is between u_1, w_1 and w_1 is not. If the order of the four agents from left to right is u_1, u_2, w_1, w_2, then u_2 and w_1 form a blocking pair. If the order is w_2, u_1, u_2, w_1, then u_1 and w_2 form a blocking pair. □

Let M_1, M_2, M_3 be three stable matchings with M_2 (or M_3) being the resulting matching after a free-agent a_1(or a_2) $\in U \cup W$ reselecting his/her partner from M_1 (or M_2). In addition, for each pair $(u, w) \in M_1$, either both a_1, a_2 are between u, w, or neither is between u, w. Given a pair $(u_{\text{arc}}, w_{\text{arc}}) \in M_1$ with a_1, a_2 not between u_{arc} and w_{arc}, the following holds.

Lemma 4.6. *Given an agent a_{in} who is between u_{arc} and w_{arc}, we have a_{in} is neither a member of C_1 nor C_2, where C_1 and C_2 are rematch-chains caused by a_1 and a_2, respectively.*

Proof. First, we claim that $a_{\text{in}} \notin C_1$. If it is not true, M_2 is not a stable matching by Lemma 4.5. If $u_{\text{arc}}, w_{\text{arc}} \notin C_1$, then $a_{\text{in}} \notin C_2$ can be proved in a similar way as above. If $u_{\text{arc}}, w_{\text{arc}} \in C_1$, we have two cases. Let (u_{far}, w_{far}) be the farthest pair in $M^1_{\text{new}} \cup M^1_{\text{old}}$, with $M^1_{\text{new}}, M^1_{\text{old}}$ being the newborn- and vanish-sets caused by a_1, respectively. Obviously, a_{in} is between u_{far} and w_{far}. If a_2 is not between u_{far} and w_{far}, then $a_{\text{in}} \notin C_2$ can be proved as above. If a_2 is between u_{far} and w_{far}, then a_2 is between a pair $(u, w) \in M_2$, while a_{in} is not. We can prove $a_{\text{in}} \notin C_2$ by Lemma 4.4. □

Before showing the algorithm, we give a formal definition of *cross-score*, which is used in the algorithm.

Cross-score. Given a matching M and a pair $(u, w) \in W$, the *cross-score* of (u, w), denoted as $\xi(u, w)$, is equal to $|A_{(u,w)}|$, where $A_{(u,w)}$ is the set of agents who are between u and w (and $u, w \in A_{(u,w)}$). The cross-score of M, denoted as $\xi(M)$, is set as $\xi(M) = \sum_{(u,w)\in M} \xi(u, w)$. Let M_1, M_2 be two stable matchings with M_2 being the resulting matching after a free-agent $a_0 \in U \cup W$ reselecting his/her partner. Let $(u_{\text{end}}, w_{\text{end}})$ be the farthest pair in $M_{\text{old}} \cup M_{\text{new}}$ with M_{old} and M_{new} being the vanish- and newborn-sets caused by a_0, respectively. The following hold.

Lemma 4.7. *(1) If $(u_{\text{end}}, w_{\text{end}}) \in M_1$, then $\xi(M_1) > \xi(M_2)$. (2) If $(u_{\text{end}}, w_{\text{end}}) \in M_2$, then $\xi(M_1) < \xi(M_2)$.*

Proof. By the definition of cross-score, we have $\xi(M_2) = \xi(M_1 \cup M_{\text{new}} \setminus M_{\text{old}})$. Thus, if $\xi(M_{\text{new}}) > \xi(M_{\text{old}})$, then $\xi(M_2) > \xi(M_1)$. If $(u_{\text{end}}, w_{\text{end}}) \in M_1$, then $\xi((u_{\text{end}}, w_{\text{end}})) > \sum_{(u,w)\in M_{\text{new}}} \xi(u, w)$, since there are no two pairs $(u_1, w_1), (u_2, w_2) \in M_{\text{old}}$, such that u_1, w_1 are between u_2 and w_2. Similarly, If $(u_{\text{end}}, w_{\text{end}}) \in M_2$, then $\xi((u_{\text{end}}, w_{\text{end}})) > \sum_{(u,w)\in M_{\text{old}}} \xi(u, w)$. □

Let I be an instance of 1-Euclid-Forced-SMT and M be a stable matching of I. Given two agents $a_L, b_R \in U \cup W$, let $A_{(a_L,b_R)}$ denote the set of agents between a_L and b_R (and $a_L, b_R \in A_{(a_L,b_R)}$). We say $A_{(a_L,b_R)}$ has a closed-matching if $M(a_i) \in A_{(a_L,b_R)}$ for all $a_i \in A_{(a_L,b_R)}$. Denote the matching of $A_{(a_L,b_R)}$ in M as $M_{(a_L,b_R)}$, and let $M'_{(a,b)}$ be the closed-matching of $A_{(a_L,b_R)}$, which is stable and has the minimum cross-score among all closed-matchings of $A_{(a_L,b_R)}$. Let $M' = M \cup M'_{(a,b)} \setminus M_{(a,b)}$, the following holds.

Lemma 4.8 (*). *M' is a stable matching of I.*

Theorem 4.9. *1-Euclid-Forced-SMT can be solved in polynomial time, if preference profile is complete.*

Proof. Given a line that U and W are embedded into, $A_{i,j} \subseteq U \cup W$ consists of agents between a_i and a_j (including a_i and a_j) with a_i, a_j being the i-th and j-th agents. The basic idea of the algorithm is, given a subset $A_{i,j}$, we can always divide it into three disjoint parts, that is, $\{a_i, a_k\}$, $A_{i+1,k-1}$ and $A_{k+1,j}$. By enumerating all possible k, we can get the optimal matching of $A_{i,j}$, that is, the closed-matching with the minimum cross-score on $A_{i,j}$, as long as we have all optimal matchings of $A_{i+1,k-1}$ and $A_{k+1,j}$ with $i < k < j$. Thus, we can use a dynamic programming to calculate the optimal matching of $A_{i,j}$ from $j - i = 1$ to $j - i = 2n - 1$. Let $M[i, j]$ be the optimal matching of $A_{i,j}$, that is, $M[i, j]$ has the minimum cross-score among all feasible-matchings of $A_{i,j}$. Here, a matching M is a feasible-matching if and only if M is stable and does not break any pair in F. Here, a matching M breaks a forced pair $(u, w) \in F$, if u is matched by M but $M(u) \neq w$. Let $D(i, j)$ be the cross-score of $M[i, j]$.

Initialization. We initialize $M[i, i + 1], D[i, i + 1]$ as follows.

$$M[i, i + 1] = \{a_i, a_{i+1}\}$$
$$D[i, i + 1] = \xi(a_i, a_{i+1})$$

Recursive formulas. Then we calculate $D[i, j]$ from $j - i + 1 = 4$ to $j - i + 1 = 2n$.

$$D[i, j] = \min_{k\in(i,j)} E[i, j, k]$$

Here, $E[i, j.k] = \xi(a_i, a_k) + D[i + 1, k - 1] + D[k + 1, j]$ if $\{a_i, a_k\} \cup M[i + 1, k - 1] \cup M[k + 1, j]$ is a feasible matching; otherwise, $E[i, j.k] = +\infty$. If $D[i, j] \neq +\infty$, we let $M[i.j]$ be the corresponding optimal matching. This process can be done in $O(n^2)$ time. If $D[i, j] \neq +\infty$, then the solution matching of I is $M[i, j]$ by Lemma 4.8. □

4.3 Other Results

Theorem 4.10 (*). *If preference lists are incomplete, d-Euclid-π-SMT is NP-hard even with $d = 1$. Here, $\pi \in$ {Max, Reg, Egal, Forced}.*

Theorem 4.11 (*). *Given a preference profile L, it is polynomial-time decidable whether L can be embedded into a line.*

5 CONCLUSION

In this paper, we study π-Stable Marriage with Ties (π-SMT) in the d-Euclidean space. If the preference profile is complete, π-SMT with $\pi \in$ {Reg,Egal,Forced} is NP-hard even with $d = 2$. With $d = 1$, there exist polynomial-time algorithms solving Reg-SMT and Forced-SMT. If the preference profile is incomplete, π-SMTI with $\pi \in$ {Reg,Egal,Forced,MAX} is NP-hard even with $d = 1$. In Table 1, we left an open problem, that is, can 1-Euclidean-Egal-SMT be solved in polynomial time? For the future work, (1) it might be interesting to check other variants of Stable Marriage in d-Euclidean space, such as Balanced Stable Marriage [18], Sex-equal Stable Marriage [25], Man-Exchange Stable Marriage [30], etc. (2) Another possible direction is to combine d-Euclidean space with Stable Marriage with strict preference orders, such as Pairwise Stable Marriage [2, 4] and Stable Matching with General Preferences [17]. (3) Are there any fixed-parameter algorithms or approximation algorithms for the NP-hard cases? (4) The incompleteness of preference list has an alternative interpretation, that is, non-listed alternatives are further away than listed ones. It might be interesting to check how the results would be different when $d = 1$.

ACKNOWLEDGMENTS

We thank the AAMAS-23 reviewers for their constructive comments. All authors are supported by the National Natural Science Foundation of China (Grants No.6207050204, 61761136017 and 62072275.).

REFERENCES

[1] Atila Abdulkadiroğlu, Parag A. Pathak, and Alvin E. Roth. 2005. The New York city high school match. *American Economic Review* 95, 2 (2005), 364–367.

[2] David J. Abraham, Ariel Levavi, David F. Manlove, and Gregg O'Malley. 2008. The Stable Roommates Problem with Globally Ranked Pairs. *Internet Mathematics* 5, 4 (2008), 493–515.

[3] Esther M. Arkin, Sang Won Bae, Alon Efrat, Kazuya Okamoto, Joseph S. B. Mitchell, and Valentin Polishchuk. 2009. Geometric stable roommates. *Inform. Process. Lett.* 109, 4 (2009), 219–224.

[4] Haris Aziz, Péter Biró, Tamás Fleiner, Serge Gaspers, Ronald de Haan, Nicholas Mattei, and Baharak Rastegari. 2022. Stable matching with uncertain pairwise preferences. *Theoretical Computer Science* 909 (2022), 1–11.

[5] Haris Aziz, Jiayin Chen, Serge Gaspers, and Zhaohong Sun. 2018. Stability and pareto optimality in refugee allocation matchings. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems.* 964–972.

[6] Giuseppe Di Battista, Giuseppe Liotta, and Francesco Vargiu. [n.d.]. Spirality and optimal orthogonal drawings. *SIAM J. Comput.* 27, 6 ([n. d.]), 1764–1811.

[7] Péter Biró. 2017. Applications of matching models under preferences. (2017).

[8] Anna Bogomolnaia and Jean-François Laslier. 2007. Euclidean preferences. *Journal of Mathematical Economics* 43, 2 (2007), 87–98.

[9] Jiehua Chen and Sven Grottke. 2021. Small one-dimensional Euclidean preference profiles. *Social Choice and Welfare* 57, 1 (2021), 117–144.

[10] Jiehua Chen, Rolf Niedermeier, and Piotr Skowron. 2018. Stable marriage with multi-modal preferences. In *Proceedings of the 19th ACM Conference on Economics and Computation.* 269–286.

[11] Jiehua Chen, Kirk Pruhs, and Gerhard J. Woeginger. 2017. The one-dimensional Euclidean domain: finitely many obstructions are not enough. *Social Choice and Welfare* 48, 2 (2017), 409–432.

[12] Jiehua Chen and Sanjukta Roy. 2022. Multi-dimensional stable roommates in 2-dimensional Euclidean space. In *Proceedings of 30th Annual European Symposium on Algorithms.* 36:1–36:16.

[13] Ágnes Cseh and Klaus Heeger. 2020. The stable marriage problem with ties and restricted edges. *Discrete Optimization* 36 (2020), 100571.

[14] Ágnes Cseh and David F. Manlove. 2016. Stable marriage and roommates problems with restricted edges: complexity and approximability. *Discrete Optimization* 20 (2016), 62–89.

[15] Vânia M. F. Dias, Guilherme Dias da Fonseca, Celina M. H. de Figueiredo, and Jayme Luiz Szwarcfiter. 2003. The stable marriage problem with restricted pairs. *Theoretical Computer Science* 306, 1-3 (2003), 391–405.

[16] Edith Elkind and Piotr Faliszewski. 2014. Recognizing 1-Euclidean preferences: an alternative approach. In *Proceedings of Algorithmic Game Theory - 7th International Symposium.* 146–157.

[17] Linda Farczadi, Konstantinos Georgiou, and Jochen Könemann. 2016. Stable marriage with general preferences. *Theoretical Computer Science* 59, 4 (2016), 683–699.

[18] Tomás Feder. 1995. *Stable Networks and Product Graphs.* Stanford University Press.

[19] David Gale and Lloyd S Shapley. 1962. College admissions and the stability of marriage. *The American Mathematical Monthly* 69, 1 (1962), 9–15.

[20] M. R. Garey and David S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman.

[21] Dan Gusfield. 1987. Three fast algorithms for four problems in stable marriage. *SIAM J. Comput.* 16, 1 (1987), 111–128.

[22] Dan Gusfield and Robert W. Irving. 1989. *The Stable marriage problem - structure and algorithms.* MIT Press.

[23] Robert W. Irving. 1994. Stable marriage and indifference. *Discrete Applied Mathematics* 48, 3 (1994), 261–272.

[24] Robert W. Irving, Paul Leather, and Dan Gusfield. 1987. An efficient algorithm for the "optimal" stable marriage. *Journal of the ACM* 34, 3 (1987), 532–543.

[25] Akiko Kato. 1993. Complexity of the sex-equal stable marriage problem. *Japan Journal of Industrial and Applied Mathematics* 10, 1 (1993), 1–19.

[26] Vicki Knoblauch. 2010. Recognizing one-dimensional Euclidean preference profiles. *Journal of Mathematical Economics* 46, 1 (2010), 1–5.

[27] Dmitry Lebedev, Fabien Mathieu, Laurent Viennot, Anh-Tuan Gai, Julien Reynier, and Fabien De Montgolfier. 2007. On using matching theory to understand P2P network design. In *Proceedings of International Network Optimization Conference.*

[28] David F. Manlove, Robert W. Irving, Kazuo Iwama, Shuichi Miyazaki, and Yasufumi Morita. 2002. Hard variants of stable marriage. *Theoretical Computer Science* 276, 1-2 (2002), 261–279.

[29] David F. Manlove and Gregg O'Malley. 2012. Paired and altruistic kidney donation in the UK: algorithms and experimentation. In *Proceedings of 11th International Symposium on Experimental Algorithms.* 271–282.

[30] Eric McDermid, Christine T. Cheng, and Ichiro Suzuki. 2007. Hardness results on the man-exchange stable marriage problem with short preference lists. *Inform. Process. Lett.* 101, 1 (2007), 13–19.

[31] Matthias Mnich and Ildikó Schlotter. 2017. Stablem marriage with covering constraints a complete computational trichotomy. In *Proceedings of the 10th International Symposium of Algorithmic Game Theory.* 320–332.

[32] Cristopher Moore and J. M. Robson. 2001. Hard Tiling Problems with Simple Tiles. *Discrete and Computational Geometry* 26, 4 (2001), 573–590.

[33] Dominik Peters. 2017. Recognising multidimensional Euclidean preferences. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence.* 642–648.

[34] Yinghui Wen, Aizhong Zhou, and Jiong Guo. 2022. Position-based matching with multi-modal preferences. In *Proceedings of 21st International Conference on Autonomous Agents and Multiagent Systems.* 1373–1381.