Matching with Homophily

Antonio Nicolò[*], Arunava Sen[†], Sonal Yadav[‡]

May 3, 2017

Abstract

We study a matching model in which agents have to be matched in pairs to undertake a project. Each agent partitions the set of partners into friends and outsiders, and the set of of possible projects, into good and bad ones (dichotomous preferences). The overall preference ordering on partner, project pairs is separable. Friendship is mutual and preferences over projects among friends exhibit value homophily in the following sense: when comparing two friends, the set of good projects for one individual is included in the set of good projects of her friend. We propose appropriate notions of stability and non-manipulability in this model and propose an algorithm, the minimum demand priority algorithm that generates stable allocations ,satisfies a limited notion of Pareto efficiency called friendship efficiency and has good strategic properties. Finally we show stable allocations may not exist if the value homophily and dichotomous preferences assumptions are relaxed.

1 Introduction

In many situations individuals form teams to perform a task or develop a project. Agents typically have preferences over the project they are asked to work on as well as the partners they are assigned to work with on the project. The "successful" formation of teams clearly depends on preferences over both these dimensions. We aim to study a matching problem in which pairs of agents have to form a team in order to perform a task. We are interested in understanding the conditions under which it is possible to reach stable allocations defined appropriately (see below).

Such partnership problems are common occurences. An interesting example is pair programming. Pair programming is a software development technique in which two programmers work together at one work station. One, the driver writes the code while, the other

[*]University of Padua, Padua, Italy

[†]Indian Statistical Institute, New Delhi, India.

[‡]University of Padua, Padua, Italy.

observer, reviews each line of the code as it is typed in. The two programmers switch roles frequently. A website for pair programming is a centralized mechanism that assigns pairs of programmers to do distinct jobs or tasks. In general, there is a growing computer science literature about the colloborative software business development model. Most of the open source softwares have been developed in this manner, by a team of independent programmers who join and work to develop a program. Examples of colloborative softwares are abundant, but the size of the teams that work on different projects vary substantially.

Another instance can be found in the creation of task forces where a firm assigns workers in teams to perform a task. Police precincts usually send out officers in pairs. In a course, students are assigned in groups to write term papers from a set of possible topics. Different groups of students in a class have different interests. Each group is cohesive and students in a particular group find each other mutually acceptable and want to work on similar projects. The roommate problem where the agents care about their roomate and the room they are assigned to is another example.

In our model, each agent has a preference ordering over possible partner, project pairs. We make several assumptions about this preference ordering. The first is that preferences over each component are *dichotomous* i.e. alternatives in each component are partitioned into acceptable and non acceptable sets. The set of partners possible is partitioned into friends (acceptable partners) and outsiders. Similarly the set of projects is partitioned into good and bad projects. Preferences are *separable* over partners and projects. According to these preferences all partner, project pairs can be placed in one of the four indifference classes : (i) both partner and project are acceptable, (ii) only the partner is acceptable, (iii) only the project is acceptable and (iv) neither partner nor project is acceptable. Seperability implies that (i) is the most preferred equivalence class while (iv) is the least preferred. There are two possible preferences for an agent given her acceptable partner and project sets. The first is the *partner dominant* preference where the agent ranks (ii) above (iii). In this case the agent prefers an unacceptable partner,acceptable project pair over an unacceptable partner, acceptable project pair. The second is the *project dominant* preference where the agent ranks (iii) over (ii) i.e. the agent prefers an unacceptable partner, good project pair over a friend, bad project pair.

We assume that partner acceptability (or friendship) is mutual and transitive. These assumptions induce a partition on the set of agents where each element (friendship component) in the partition is a set of agents who are all acceptable to each other. Agents in the same friendship components are referred to as friends.

Finally we assume *homophily* in the preferences of friends, which is similarity in tastes of friends. Sociological literature has provided ample evidence of the existence of homophily, that is, the tendency of individuals to connect and form close ties with other individuals who are similar to them. Lazarsfeld et al. (1954) distinguished two types of homophily: status homophily, in which similarity is based on informal, formal, or ascribed status, and value homophily, which is based on values, attitudes, and beliefs. These two forms of homophily depend on each other, because individuals tend to have ties with whom they share similar experiences, like school, sports, hobbies, or with those who live in the same neighborhood or with those who belong to the same age group. Hence, we not only tend to build closer ties with individuals which are similar to us[1], but we also become more similar to our friends over time (Bauman and Ennett, 1996). All this amounts to an observed similarity between tastes and friendships, with friends sharing the same preferences for music (Selfhout et al., 2009), food and so on. More relevant for our paper, Ruef et al. (2003) have shown that homophily and network constraints based on strong ties have the most pronounced effect on the composition of entrepreneurial founding teams. Specifically, we assume a strong alignment in the preference for projects among friends: there cannot be a pair of projects a, b and a pair of friends i, j such that i likes a but not b, while j likes b but not a. However i and j need not have the same set of preferred projects, one agent can be fussier than the other and like only a subset of the projects liked by the other agents.

Agents can block an assignment in the following three ways. Note that for any assignment, there is a set of projects which is left unassigned (possibly empty). Any project which is unassigned is available and can be taken up a pair of agents. Thus a pair of agents can leave their current partners in the assignment and come togeher with an *available unassigned* project which makes them strictly better off than before. A pair of agents can also block an assignment via a position swap i.e. the agents in the blocking pair swap places with each other to strictly improve. Blocking via an uassigned project or via a position swap leads to a change in both partner and project for the agents in the blocking pair. The third possibility is that two pairs of agents swap projects and all four agents strictly improve. There is a change of project for each agent (belonging to the two pairs), while the partner for each agent remains unchanged. An assignment is stable at a preference profile if it is immune to blocking via uassigned projects, position swaps and project swaps. Stability obviously depends on the preference profile. Note that for any given set of acceptable partners and projects for an agent, there exists two preference orderings consistent with these acceptable sets i.e. a partner dominant ordering and a project dominant one. Thus for a given configuration of partner and project acceptable sets for all agents, there is a spectrum of preference profiles consistent with this configuration. A preference profile is consistent with the acceptable set configuration if for any agent i, the preference for an agent i is either partner or project dominant with respect to the acceptable partner, project sets for agent i. A particularly attractive property for an assignment is *robust stability* where the assignment is stable at every preference profile consistent with respect to a given acceptable set configuration. It is not clear a priori whether such assignments exist.[2] We propose an algorithm, the minimum demand priority algorithm (MDPA) that generates robustly stable allocations, satisfies a

[1]See Cohen (1977), Kandel (1978), Verbrugge (1983), McPherson et al. (2001), Golub and Jackson (2012).

[2]Note that a stable assignment may not be robustly stable. We provide examples to illustrate this.

limited notion of Pareto efficiency called friendship efficiency and has good strategic properties. There are some difficulties in formulating strategy proofness as homophily imposes a restriction on the preference profile of agents who belong to the same friendship component. However we construct a model which deals with these issues and we show that the MDPA is strategy proof in this sense.

Finally we demonstrate that our assumptions regarding homophily and dichotomous preferences are crucial for the existence of stable assignments. For instance, in the absence of homophily, stable assignments fail to exist in the subdomian where each agent has partner (or project) dominat preferences. This indicates that value homophily is a fundamental assumption for the existence of stable allocations in this class of problems. We show that stable assignments fail to exist in any domain which contains the following preference profile: the preference for any agent satisfies single peakedness with respect to each dimension (partner and project) and the project dimension is lexicographically dominant.

We now briefly discuss our preference domain assumptions. The assumptions on preference dichotomy are indeed restrictive. Each individual has at most four indifference classes depending on whether the partner is a friend or an outsider and whether the project is good or bad. Nonetheless, it is important to point out that no further assumptions are imposed: while the best assignments for each agent are those in which she is assigned to a good project with a friend, and the worst ones those in which she has to work on a bad project with an outsider, no restrictions are imposed on how agents rank assignments in which they share a good project with a bad partner, versus assignments in which they are matched with an outsider on a good project.

The homophily assumption is also critical for our result. If we give up the assumption on preference correlation, the existence of stable allocations is not guaranteed in dichotomous domains, even when we restrict attention to the preference sub-domain where all agents care more about partners than projects, or in the sub-domain where all agents care more about projects than partners. Finally, if we depart from the dichotomous domain, impossibility results obtain even when strong restrictions such as single peakedness on imposed on each dimension.

1.1 Relationship to Existing Literature

The paper is linked to several strands in the matching literature.

The first strand is the literature on the existence of stable outcomes in the roommate problem. The essentials of the roommate problem introduced in Gale and Shapley (1962) is a set of agents, allowed to pair up in couples or stay-single, together with individual-specific preferences over partners.[3] Every agent has a preference over possible partners. Our model

[3]See Tan (1991), Chung (2000), Alcalde (1994) and Gudmundsson (2014) for conditions that grant the

analyses the question of stability when every agent also has a preference over the rooms, in addition to preferences over potential partners. We investigate the existence of stable assignments in situations where individuals not only care about their roommate, but also have a preference over the room that they will receive in the assignment.

The second related strand is the literature on the existence of stable matchings in a two sided matching market with couples (of students) on one side and hospitals on the other side.[4] The couples are fixed, and every agent has an individual preference on the hospitals and a joint preference on the possible hospital duples. An agent in a couple is only concerned about the two hospitals that are assigned to her and her partner. A couple can be viewed as a friendship component consisting of an individual and her partner. So in an assignment either the couple is matched to the same hospital, or it is possible that they are matched alone which means that the two individuals are allocated to different hospitals. This model is a two sided matching model, where the hospital also has preferences over the agents assigned to the hopsital. We consider a model where each agent has a preference over potential partners and an agent is willing to be paired with any other agent. In this sense, there are no *fixed* couples in our model. Thus a stable allocation in this context has to be robust to changes in both partner and projects.

The remainder of the paper is organized as follows. We introduce the model in the next section. Section 3 describes the MDPA algorithm to compute a robustly stable assignment and presents the main result. Section 4 discusses the efficiency properties of the MDPA algorithm and shows that the assignment generated by the algorithm satisfies a weaker notion of Pareto efficiency imposed on a group of friends. Section 5 examines the strategic aspects of the MDPA mechanism. The dichotomous domain without homophily is investigated in Section 6. Section 7 presents an analysis about the non existence of stable assignments in non dichotomous domains and Section 8 concludes. The Appendix provides the provides the proofs.

2 The Model

There is a finite set of agents $N = \{1, 2, \ldots, i, j, \ldots, n\}$ and a finite set of alternatives (projects), $A = \{a, b, c, d \ldots\}$. We assume that $|n| = 2m$ for some integer $m \geq 2$ and that $|A| \geq m$.

An *assignment* σ is a collection of triples (i, j, a) with the interpretation that agent pair (i, j) is assigned project a. In order to ensure feasibility, we require that each agent is paired with one other agent and with one project. In addition, every project is assigned to exactly one pair or left unassigned. We require all agents to be assigned a partner and a project. We

existence of stable allocations in the roommate problem.

[4]See Klaus and Klijn (2005), Klaus and Klijn (2007), Klaus et al. (2007), Khare and Roy (2016) for requirements for the existence of stable allocations in the two sided matchings with couples.

shall say that the triple (i, j, a) is an element of σ, if the pair (i, j) is assigned to the project a. Abusing notation, we shall also let $\sigma(i) = (j, a)$ if (i, j, a) is an element of σ. Feasibility imposes some obvious restrictions on σ which we do not elaborate for convenience. Finally let u^σ be the set of unassigned projects in the assignment σ i.e. it is the set of projects a such that there does not exist an agent i such that $\sigma(i) = (j, a)$ for some j.

2.1 Preferences

Each agent i has a preference ordering[5] $\succ_i$ over partner, project pairs (j, a) where $j \neq i$. There are several restictions on $\succ_i$ which we outline below (the consequences of relaxing some of these restrictions will be discussed in Section (Without Homophily)).

Every agent i has a set of acceptable partners (friends) $P^i \subseteq N \setminus \{i\}$ and acceptable projects $G_i \subseteq A$. We allow for the possibility of $P^i, G^i = \emptyset$. We denote the complement sets of G^i and P^i by $A \setminus G^i$ and $N \setminus P^i$ respectively. The ordering $\succ_i$ satisfies the following properties.

(i) If j, k belong to the same element of the partition $\{P^i, N \setminus P^i\}$ and a, b belong to the same element of the partition $\{G^i, A \setminus G^i\}$, then $(j, a) \sim_i (k, b)$.

(ii) If $a \in G^i$, $b \notin G^i$, then $(k, a) \succ_i (k, b)$ for all $k \neq i$.

(iii) If $k \in P^i$, $l \notin P^i$, then $(k, a) \succ_i (l, a)$ for all $a \in A$.

Preferences are *separable* over partners and projects. In addition, preferences over each component (partners and projects) are *dichotomous* i.e. alternatives in each component are partitioned into acceptable and non acceptable sets.

According to these preferences all partner, project pairs can be placed in one of four indifference classes: (I) both partner and project are acceptable, (II) only the partner is acceptable (III) only the project is acceptable and (IV) neither partner nor project is acceptable.

Separability implies (I) is the most preferred equivalence class and (IV) is the least preferred. Classes (II) and (III) can be ranked either way. We refer to the case where (II) is ranked above (III) as a *partner dominant* preference. In this case agent i prefers an acceptable partner, unacceptable project pair over an unacceptable partner, acceptable project pair.

Similarly, a *project dominant* preference is one where (III) is ranked above (II) i.e. agent i prefers an unacceptable partner, acceptable project pair over an acceptable partner, unacceptable project pair.

[5]An ordering is a binary relation which is complete, reflexive and transitive.

We assume that partner acceptability is mutual and transitive i.e. if $j \in P^i$ then $i \in P^j$. In addition, if $i \in P^j$ and $j \in P^k$, then $i \in P^k$.[6] These assumptions induce a partition on the set of agents N where each element in the partition is a set of agents who are all acceptable to each other. We shall refer to each element of this partition as a *friendship component* and agents' in the same friendship component as friends. These components are labelled $\{F_1, F_2, \ldots, F_q\}$. Note that the number of components which have an odd number of agents must be even.

Finally we assume *homophily* in the preferences for agents in the same friendship component. Specifically if i, j are friends, then either $G^i \subseteq G^j$ or $G^j \subseteq G^i$ holds. Homophily induces a strong alignment in the preference for objects among friends. In other words, there cannot be a pair of projects a, b and a pair of friends i, j such that i likes a but not b, while j likes b but not a. However i and j need not have the same set of prefered projects, one agent can be fussier than the other and like only a subset of the projects liked by the other agent.

We assume that the preference of agent i, $\succ_i$ satisfies all the conditions above. Thus $\succ_i$ is associated with a set of acceptable partners and projects which we denote by $P^i(\succ_i)$ and $G^i(\succ_i)$ respectively. Let Σ^{part} denote the set of all partner dominant preferences , i.e. if $\succ_i \in \Sigma^{\text{part}}$, there exists P^i, G^i such that $\succ_i$ is partner dominant with respect to P^i, G^i. Similarly we define Σ^{proj} as the set of all project dominant preferences. Let $\Sigma = \Sigma^{\text{part}} \cup \Sigma^{\text{proj}}$. A preference profile $\succ$ is an n-tuple $\succ \in \Sigma^n$.

2.2 Blocking and Stability

Our notion of stability differs in an important way from the standard notion of stability in the Gale-Shapley model. There two agents who like each other more than their current partners can come together while leaving their assigned partners. In our model, we assume that both agents who are assigned a project have "rights" over that project. A pair of agents can block an assignment in either of the following two ways: (i) they can leave their current partner and project and come together with an unassigned project which makes them strictly better off than before or (ii) they can swap places with each other and strictly improve. Also two pairs of agents can block the assignment via a project swap which makes all the four agents strictly better off. We shall refer to blocking of the first type as *blocking via unassigned projects* and of the second type as *blocking by a position swap.* We refer to blocking of the third type as *blocking via a project swap.* An assignment is stable if it is immune to blocking of these three types.

Definition 1 *The pair of agents k, l blocks assignment σ via an unassigned project at profile $\succ$ if there exists $c \in u^\sigma$ such that*

[6]These are strong assumptions but they are a convenient starting point for the analysis.

F_1	F_2
$1\ \{a\}$	$3\ \{b\}$
$2\ \{a\}$	$4\ \{b\}$

Table 1: Example 1

1. $(k, l, c) \succ_k (k, i, a)$.

2. $(k, l, c) \succ_l (l, j, b)$.

where $(k, i, a), (l, j, b) \in \sigma$.

Definition **2** *The pair of agents* k, l *blocks assignment* σ *via a position swap at profile* $\succ$ *if*

1. $(k, j, b) \succ_k (k, i, a)$.

2. $(l, i, a) \succ_l (l, j, b)$.

where $(k, i, a), (l, j, b) \in \sigma$.[7]

Definition **3** *The pairs of agents* k, i *and* l, j *block assignment* σ *via a project swap at profile* $\succ$ *if*

1. $(k, i, b) \succ_k (k, i, a)$.

2. $(k, i, b) \succ_i (k, i, a)$.

3. $(l, j, a) \succ_l (l, j, b)$.

4. $(l, j, a) \succ_j (l, j, b)$.

where $(k, i, a), (l, j, b) \in \sigma$.[8]

Examples 1, 2 and 3 illustrate the independence of the various notions of blocking.

Example **1** Let $N = \{1, 2, 3, 4\}$ and $A = \{a, b, c, d\}$. The friendship components are $F_1 = \{1, 2\}$ and $F_2 = \{3, 4\}$. Table 1 provides the acceptable set of projects for the agents.

Consider a preference profile $\succ$ where $\succ_i$ is partner dominant for all i. We claim that the assignment $\sigma = \{(1, 2, b), (3, 4, a)\}$ is immune to blocking via an unassigned project and a position switch. However σ can be blocked via a project switch.

[7]Note that after the swap the project partner pairs are (l, i, a) and (k, j, b).

[8]Note that after the project swap, the resulting partner,project pairs are $(k, i, b), (l, j, a)$.

F_1	F_2
$1\ \{a\}$	$3\ \{b\}$
$2\ \{a\}$	$4\ \{b\}$

Table 2: Example 2

The set of unassigned projects is $u^\sigma = \{c, d\}$. The projects c, d are unacceptable for all agents. Thus blocking via an unassigned project is not possible.

Since the preference of each agent is partner dominant, blocking via a position swap will make the concerned agents worse off. For instance, consider agents 1 and 3. A position swap results in the assignment $\{(3, 2, b)(1, 2, a)\}$. This makes agents 1 and 3 worse off as both agents now have partners not belonging to their respective components and $\succ_i$ is partner dominant for $i \in \{1, 3\}$.

However blocking via a project switch is possible. The two pairs swap their assigned projects and the resulting assignment is $\{(1, 2, a), (3, 4, b)\}$. Note that all four agents strictly improve to the (G, G) class.

Example **2** Let $N = \{1, 2, 3, 4\}$ and $A = \{a, b, c\}$. The friendship components are $F_1 = \{1, 2\}$ and $F_2 = \{3, 4\}$. Table 2 provides the acceptable projects sets for the agents.

Consider a preference profile $\succ$ where $\succ_i$ is partner dominant for all i. We claim that the assignment $\sigma = \{(1, 2, c), (3, 4, b)\}$ is immune to blocking via a position swap and a project swap. But blocking via an unassigned project is possible.

The preference for each agent is partner dominant. Consider two agents who swap positions, say agents $1, 3$. This results in the assignment $\{(3, 2, c), (1, 4, b)\}$ and $1, 3$ are worse off after the position swap. This shows that any two agents who swap positions in σ will be worse off after the swap.

Both agents in the pair $(3, 4)$ have an acceptable project and a project switch will not lead to a strict improvement for 3 and 4. Thus blocking via a project swap is not possible.

However $(1, 2)$ will block σ via the unassigned project a.

Example **3** Let $N = \{1, \ldots, 6\}$ and $A = \{x, y, e, f\}$. The friendship components are $F_1 = \{1, 2, 3\}$, $F_2 = \{4, 5\}$ and $F_3 = \{6\}$. Table provides the acceptable project sets.

Consider a preference profile $\succ$ where $\succ_1$ is project dominant and $\succ_i$ is partner dominant for all $i \in N \setminus \{1\}$. We will show that the assignment $\sigma = \{(1, 2, y), (4, 5, e), (3, 6, x)\}$ is immune to blocking via an unassigned project and a project swaps. However blocking via a position swap is possible.

Note that $u^\sigma = \{e\}$. The agents who can strictly improve by any means given σ are

F_1	F_2	F_3
1 $\{x\}$	4 $\{e, f\}$	6 $\{x\}$
2 $\{x, y\}$	5 $\{e, f\}$	
3 $\{x, y\}$		

Table 3: Example 3

F_1	F_2
1 $\{a\}$	5 $\{a\}$
2 $\{a, b\}$	6 $\{a, b\}$
3 $\{a, b, c\}$	7 $\{a, b, c\}$
4 $\{a, b, c, d\}$	8 $\{a, b, c, d\}$

Table 4: Example 4

agents 1 and 3.[9] Agents 1 and 3 cannot strictly improve by moving to project f, as it is an unacceptable project for both.

There is no project switch possible between two pairs in σ where all four agents strictly improve. For instance, consider the pairs $(1, 2)$ and $(3, 6)$. Since $y \in G^2$ and $x \in G^6$, agents 3 will remain indifferent and agent 6 will be worse off after the project swap. Thus blocking via a project swap is ruled out.

However σ is blocked by agents 1 and 3 by a position swap. The assignment for the two agents after the position swap is $\{(3, 2, y), (1, 6, x)\}$. Agent 1 strictly improves as $x \in G^1$ and $\succ_1$ is project dominant. Agent 3 strictly improves because $\succ_3$ is partner dominant.

Definition 4 *The assignment σ is stable at $\succ$ if it is immune to both blocking via an unassigned project, by a position swap or by a project swap at $\succ$.*

We illustrate the notion of stability with an example.

Example 4 Let $N = \{1, 2, \ldots, 8\}$ and $A = \{a, b, c, d, e\}$. The friendship components are $F_1 = \{1, 2, 3, 4\}$ and $F_2 = \{5, 6, 7, 8\}$. The set of acceptable projects are summarized in Table 4.

Consider a preference profile $\succ$ where $\succ_i$ is project dominant for all i. We claim that the assignment $\sigma = \{(1, 5, a), (2, 6, b), (3, 7, c), (4, 8, d)\}$ is stable at $\succ$. Note that $u^\sigma = \{e\}$ and e

[9]Agents $2, 4, 5$ are in the (G, G) class and 6 is in (B, G) class. Since agent 6 has an empty acceptable partner set, agent 6 has only two indifference classes: (B, G) and (B, B).

is not an acceptable project for any agent. Since each agent is assigned a good project in σ and $\succ$ is project dominant, blocking via an unassigned project is not possible.

Consider blocking via a position swap. In the assignment all agents are paired with partners outside their friendship components. However after any swap, at least one agent is left with an unacceptable project. Since preferences are project dominant, the blocking cannot be strictly improving. For instance if agents 7 and 4 swap positions, the resulting assignment is $(3, 4, c)$, $(7, 8, d)$ and d is an unacceptable project for agent 7.

Consider blocking via a project swap. In the assignment σ, each agent has been assigned an acceptable project. Thus no two pairs of agents can swap their assigned projects such that all agents in the concerned pairs strictly improve after the project swap.

Consider a preference profile $\succ'$ where $\succ'_i$ is partner dominant for all i. The assignment σ is no longer stable at $\succ'$. For instance 1 and 2 can block via the unassigned project e.

Finally note that the assignment $\sigma' = \{(1, 2, a), (3, 4, b), (5, 6, d), (7, 8, c)\}$ is stable at $\succ'$. In this assignment, the agents $1, 2, 3, 4, 7$ and 8 are getting both acceptable partners and projects. Agents 5 and 6 belong to the same friendship component but are getting an unacceptable project. Blocking via the unassigned project e is not strictly improving. Since preferences are partner dominant, it suffices to consider swaps within the same friendship component. However the project assigned to the pair $(7, 8)$ is not acceptable to either 5 or 6. The only pair of agents where both agents can strictly improve by switching projects with another pair is $(5, 6)$. However no such pair of agents exists as the projects assigned to $(1, 2)$, $(3, 4)$ and $(7, 8)$ are acceptable for both agents in the pair. The assignment σ' is immune to blocking by a project swap. Hence σ' is stable at $\succ'$.

Interestingly σ' is also stable at $\succ$. Even though the preferences of 5 and 6 are project dominant, there is no project "available" which will make them both strictly better off. It is not strictly improving for either 5 or 6 to swap with either 7 or 8 (since c is unacceptable to both). Moreover swapping with either 1, 2, 3 or 4 will leave the other agent being swapped strictly worse off. Blocking via a project switch is not possible for the pair $(5, 6)$ as all other pairs have been assigned a project which is acceptable for both agents in the pair.

In fact σ' remains stable at all preference profiles $\succ^*$ where the underlying acceptable sets for partners and projects for all agents remain the same. We shall refer to such an assignment as robustly stable.

EXAMPLE **5** Let $N = \{1, 2, 3, 4\}$ and $A = \{x, y, z\}$. The friendship components are $F_1 = \{1, 2\}$ and $F_2 = \{3, 4\}$. The set of acceptable projects are summarized in Table 5.

Consider a preference profile $\succ$ where $\succ_i$ is partner dominant for all i. We claim that the assignment $\sigma = \{(1, 2, y), (3, 4, x)\}$ is stable at $\succ$. Since $u^\sigma = z$ and z is not an acceptable project for agents 1 and 3, there is no blocking possible via an unassigned project. Note that agents 1 and 3 are the only agents who have been assigned an unacceptable project. Since $\succ_1$ and $\succ_3$ is partner dominant, agents 1 and 3 will not switch positions with each other.

F_1	F_2
$1\ \{x\}$	$5\ \{y\}$
$2\ \{x, y\}$	$6\ \{x, y\}$

Table 5: Example 5

This switch will make both agents worse off. A switch of projects between the pairs $(1, 2)$ and $(3, 4)$ leads to a strict improvement for only two agents (1 and 3). So blocking via a project switch is not possible. Thus σ is stable at $\succ$.

Consider a preference profile $\succ'$ where $\succ'_i$ is project dominant for all i. The assignment σ is not stable at $\succ'$. The agents 1 and 3 will block the assignment via a position switch. Both agents strictly improve by the position switch as $x \in G^1$, $y \in G^3$ and their preferences are project dominant.

Note that the assignment $\sigma' = \{(1, 2, x), (3, 4, y)\}$ is stable at both $\succ$ and $\succ'$. In fact σ' is robustly stable.

These observations motivate the definition below.

The profiles $\succ$ and $\succ'$ are *acceptable set equivalent* if $P^i(\succ_i) = P^i(\succ'_i)$ and $G^i(\succ_i) = G^i(\succ'_i)$ for all i.

Definition 5 *The assignment σ is robustly stable at profile $\succ$ if it is also stable at all profiles $\succ^{'}$ that are acceptable set equivalent to $\succ$.*

A robustly stable assignment remains stable irrespective of whether agent preferences are partner or project dominant. We will show that such an assignment exists and is attainable by the algorithm below.

3 The Algorithm and result

We describe an algorithm below which we refer to as the *Minimum Demand Priority Algorithm* (MDPA).

The agents in N are ordered using the numerical order $\succ^N$. The set of projects A is ordered using the alphabetical order $\succ^A$.

Fix an arbitrary profile $\succ$. This profile induces friendship components $\{F_1, \ldots, F_L\}$ which we arbitrarily order as $F_1, \ldots, F_L$. We can order the agents in any component F_q using $\succ^N$.

Step 0: In each component F_q, where $|F_q|$ is odd, remove the agent with the lowest priority in F_q, say agent i and add agent i to the *Residual set R*. The remaining friendship components are $\{\tilde{F}_1, \ldots, \tilde{F}_L\}$. If $|F_q|$ is even, then $F_q = \tilde{F}_q$.

Step 1: We make partial assignments in $\tilde{F}_1$ in the following manner. The set of available projects is A. We first calculate the demand for every available project.

Consider any $x \in A$. The demand for project x is $D(x) = \sharp \left\{ i \in \tilde{F}_1 : x \in G^i(\succ_i) \right\}$. Let the set of agents who demand project x be $S(x)$. Formally, $S(x) = \{i \in \tilde{F}_1 : x \in G^i(\succ_i)\}$.

We can order the projects from the most demanded to the least demanded one.[10]

Remove all projects with zero demand. Next consider the project (projects) with the least demand. In case there is more than one such project, we break ties using $\succ^A$ by choosing the project with the lowest alphabet. Suppose the project with the lowest demand (after tie breaking) is a.

Step 1.1: Assign project a to an agent or a pair of agents in the following manner.

1. $D(a) = \{1\}$. Then provisionally assign a to agent j where $S(a) = \{j\}$. Put agent,project pair (j, a) in the waiting set W. Update the set of available projects to $A \setminus \{a\}$ and the set of agents to $\tilde{F}_1 \setminus \{j\}$. Proceed to Step 1.2.

2. $D(a) \geq 2$. Arrange the agents in $S(a)$ acccording to the fixed priority defined on N. Assign a to the pair comprising of agents with the highest and second highest priority in $S(a)$. Update the set of available projects to $A \setminus \{a\}$ and the set of agents to $\tilde{F}_1 \setminus \{i, j\}$ where (i, j) is the pair who is assigned project a in this step. Proceed to Step 1.2.

Step 1.2: Repeat Step 1.1 with the following adjustments. Calculate the demand for each available project wih respect to the updated set of agents obtained at the end of Step 1.1.

Remove all projects with zero demand. Consider the project with the least demand. If there is more than one such project, break ties using the alphabetical order on A.

Suppose the project with the least demand in this step is b.

1. $D(b) = 1$ and $W \neq \emptyset$. Let $(i, a) \in W$. and $S(b) = \{j\}$. Then pair (i, j) with project b i.e. $(i, j, b) \in \sigma$. Remove (i, a) from W. Update the set of projects by removing b. Update the set of agents by removing agent j.[11] Add project a to the set Q_1.[12] Proceed to Step 1.3.

2. $D(b) = 1$ and $W = \emptyset$. Assing project b to agent j provisionally. Add (j, b) to W. Update the set of available projects by removing project b. Update the set of agents by removing agent j. Proceed to Step 1.3.

[10]Note that there may be two projects with equal demand.

[11]Note that agent i with $(i, a) \in W$ is not in the set of agents obtained at the end of Step 1.1.

[12]Note that once a project is added to the set Q_1, it is no longer an available project for the component $\tilde{F}_1$. The projects in Q_1 will be available to the next component $\tilde{F}_2$.

3. $D(b) \geq 2$. Arrange the agents in $S(b)$ acccording to the fixed priority defined on N. Assign b to the pair comprising of agents with the highest and second highest priority in $S(b)$. Update the set of available projects by removing b and the set of agents by removing agents i, j where (i, j) is the pair who is assigned project b in this step. Proceed to Step 1.3.

$\vdots$

$\vdots$

$\vdots$

Step 1.q: Calculate the demand for each available project at the end of Step 1.$(q-1)$, using the set of agents remaining at the end of Step 1.$(q-1)$. Remove all projects with zero demand.

Consider the project with the least demand, say project c. If there are several such projects, break ties using the alphabetical order. Repeat Step 1.$(q-1)$ with the project c.

Proceeding in this manner, there exists s^* such that in Step 1.s^*, the demand for every available project is zero. Here three cases can arise.

I. The set of agents is empty and $W = \emptyset$. This means that all agents in $\tilde{F}_1$ have been assigned a partner and a project.

II. There exists an even number of agents in Step 1.s^* who do not have a partner and a project and $W = \emptyset$. Arrange the agents using the priority order on N. Form pairs of consecutive agents proceeding in sequence. Assign an arbitrary project to each pair from the set of available projects.

III. $W \neq \emptyset$ and there exists an odd number of agents in Step 1.s^* who do not have a partner and a project. Note that there is exactly one agent,project pair in W, say $(i, a) \in W$. Arrange the agents (not in W) using the priority order on N. Pair agent i (where $(i, a) \in W$) with the agent who has the highest priority and assign to this pair project a. Then form pairs of consecutive agents proceeding in sequence. Proceed in sequence to assign projects to each pair: Assign to each pair the project with the lowest alphabet (using the order $\succ^A$) from the set of available projects.

Let the set of projects assigned to pairs in $\tilde{F}_1$ be $\Delta(\tilde{F}_1)$. At the beginning of Step 2 (or end of Step 1), the set of available projects is $U(\tilde{F}_2) = [A \setminus \Delta(\tilde{F}_1)] \cup Q_1$. We assign projects in $\tilde{F}_2$ in the same manner as Step 1 with the set of available projects being $U(\tilde{F}_2)$. Proceeding in this manner, at the end of Step L we have assigned partners and projects to all agents in $\tilde{F}_1 \cup \tilde{F}_2 \ldots \cup \tilde{F}_L$.

Step $L+1$: In this step, projects are assigned to agents in the set R. The set of available projects is $U(R) = [U(\tilde{F}_L) \setminus \Delta(\tilde{F}_L)] \cup Q_L$. Without loss of generality (and by suitable relabelling of agents), let $R = \{1, 2, \ldots, 2r\}$ for $r \geq 0$.

Step $(L+1.1)$: In this step, we match agent 1 with the agent with the lowest index $t > 1$ in R such that $G^1(\succ_1) \cap G^t(\succ_t) \cap U(R)$ is non empty. This pair is assigned a project x in $G^1(\succ_1) \cap G^t(\succ_t) \cap U(R)$. In this case, $R_1 = \{i, t\}$. If no such agent t exists, then agent 1 is unmatched at this step and $R_1 = \{1\}$.

The set of projects assigned in this step is denoted by $S_R(1)$ where $S_R(1) = \emptyset$ if $R_1 = \{1\}$ and $\{x\}$ if $R_1 = \{1, t\}$ and x is assigned to $(1, t)$.

The set of agents remaining for the next step is $R \setminus R_1$. The set of projects available for the next step is $U(R) \setminus S_R(1)$.

Step $(L+1.2)$: We repeat Step $(L+1.1)$ with agent 1 being replaced by the agent with the smallest index in $R \setminus R_1$ and the set of projects $U(R)$ replaced by $U(R) \setminus S_R(1)$. This generates possibly another assignment. At the end of this step, we obtain R_2 and the set of available projects $U(R) \setminus \cup_{j=1}^{2} S_R(j)$.

Proceeding in this manner, there will exist a Step $(L+1.r^*)$ where $\cup_{j=1}^{r^*} R_j = R$. The set of projects available for assignment at the end of the completion of Step $(L+1.r^*)$ is $U(R) \setminus \cup_{j=1}^{r^*} S_R(j)$.

There are two possibilities. If all agents in R have been assigned a partner (and project), then move to Step $L+2$.

Otherwise there exists an even number of unmatched agents in R and we move to Step $(L+1.r^*+1)$. Consider all unmatched agents in the previous step and consider them in the same order as they were in R. Pair consecutive agents in sequence. At every step in the sequence, assign the pair an available project which belongs to the acceptable set of the agent with higher priority in the pair (i.e. lower index). If no such project exists, then assign one from the acceptable set of the agent with the lower priority which is available. If this fails, then add this pair to set S.

Finally assign projects to pairs in S: proceed in sequence in S and to each pair, assign the project with the lowest alphabet (using $\succ^A$) from the set of available projects.

THEOREM 1 *The MDPA algorithm generates a robustly stable assignment at every $\succ$. Consequently a robustly stable assignment exists at every profile.*

The proof of Theorem 1 is provided in the Appendix.

4 Friendship Efficiency

We define a weaker version of efficiency that imposes Pareto efficiency among a group of friends.

Definition **6** *Consider an assignment σ and a preference profile $\succ$. Then σ satisfies friendship efficiency with respect to component F_q if for any $(k, i, a), (l, j, b) \in \sigma$ with $k, i, l, j \in F_q$, there does exist an assignment σ' which is constructed by reconfiguring the agents in $\{k, i, l, j\}$ and projects in $\{a, b\} \cup u^\sigma$ such that*

1. *There exists an agent $s \in \{k, i, l, j\}$ such that agent s is strictly better off in σ' at $\succ_s$.*
2. *The remaining agents in the set $\{k, i, l, j\}$ are not worse off in σ'.*

The assignment σ satisfies friendship efficiency if it satisfies this property for every component.

Theorem **2** *For any profile $\succ$, the MDPA generates an assignment which satisfies friendship efficiency.*

Proof: Consider a profile $\succ$. Let $\sigma(\succ)$ be the assignment generated by the MDPA at the preference profile $\succ$. We will show that σ satisfies friendship efficiency with respect to every friendship component.

We will prove the result by contradiction. There exists a profile $\succ$ (where the MDPA assignment is $\sigma(\succ)$), a component F_q, $(k, i, a), (l, j, b) \in \sigma(\succ)$ where $k, i, l, j \in F_q$ and an assignment σ' [13] such that there exists an agent in $\{k, i, l, j\}$ who strictly improves in σ' while the remaining agents are as well off as before.

Let agent k be the agent that strictly improves in σ'. This implies that $a \notin G^k(\succ_k)$ and the project assigned to k in σ' is an acceptable project. Note that this fact is true independent of whether $\succ_k$ is partner dominant or project dominant. This holds because in both assignments, agent k has an acceptable partner.

Since $(k, i, a) \in \sigma(\succ)$ and $a \notin G^k(\succ_k)$, we know that (k, i, a) is formed in Step $q.s^*$ in the algorithm. The demand for each available project in Step $q.s^*$ is zero. Also agent k is an available agent in Step $q.s^*$. This implies that there does not exist a project $x \in u^\sigma$ such that $x \in G^k(\succ_k)$. So project b is assigned to agent k in σ' and $b \in G^k(\succ_k)$. Thus there exists an agent in $\{l, j\}$ who is assigned project a in σ'.

We claim that $b \in G^l(\succ_l) \cap G^j(\succ_j)$. To see this, note that agent k is an available agent in every Step $q.s$ where $s \leq s^*$. Note that in Step $q.s^*$, the demand for each available project

[13]Note that σ' is generated by forming pairs from the set of agents in $\{k, i, l, j\}$ and projects in the set $\{a, b\} \cup u^\sigma$.

is zero. Thus project b is assigned in some step $q.s'$ where $s' < s^*$. [14] We will argue that $D(b) \geq 3$ in Step $q.s'$. Suppose not i.e. $1 \leq D(b) < 3$ in Step $q.s'$.[15] Let $D(b) = 1$. This means that $S(b) = \{k\}$ and the MDPA will form $(k, b) \in W$. Then agent k is assigned an acceptable project in σ: either $(k, ., b)$ or $(k, ., y)$ for some $y \in G^k(\succ_k)$. This contradicts the assumption that agent k strictly improves in σ'. The other possibility is $D(b) = 2$. Here $S(b) = \{k, p\}$ where $p \in \tilde{F}_q$. This results in (k, p, b) in σ and contradicts the assumption that agent k strictly improves in σ'.

So $D(b) \geq 3$ in Step $q.s'$. Since $s' < s^*$, project b is assigned to the two maximal agents in $S(b)$ according to $\succ^N$. Thus $l, j \in S(b)$ in Step $q.s'$ and $b \in G^l(\succ_l) \cap G^j(\succ_j)$.

There are two possibilities: either $a \notin G^i(\succ_i)$ or $a \in G^i(\succ_i)$.

(I) $a \notin G^i(\succ_i)$.

Since $(k, i, a) \in \sigma(\succ)$ and a is an unacceptable project for both agents, project a is available in each step $q.s$ where $s < s^*$. In particular, project a is available in Step $q.s'$ (the step in which project b is assigned).

We know that there exists an agent in $i \in \{l, j\}$ such that agent i is assigned project a in σ'. W.l.o.g let $i = l$. Thus $a \in G^l(\succ_l)$ as agent l is assigned an acceptable project in $\sigma(\succ)$ and all agents remain as well off as before in σ'.

In Step $q.s'$,

1. Agents l, j, k are available.
2. Projects a, b are available.
3. $D(a) > 0$ as $a \in G^l(\succ_l)$.
4. $D(b) \geq 3$.
5. $D(b) \leq D(a)$.

Fact 5 follows from the fact that b is the project which is assigned in Step $q.s'$ and so it is the project with the least demand.

An implication of Fact 5 and homophily is $S(b) \subseteq S(a)$ in Step $q.s'$. To see this, suppose $S(b) \not\subset S(a)$. First consider the case where $D(b) = D(a)$. This means that there exists an agent $p \in S(b)$ such that $p \notin S(a)$. Also since $D(b) = D(a)$, there exists an agent $q \in S(a)$ such that $q \notin S(b)$. Thus we have

(i) $b \in G^p(\succ_p)$ and $a \notin G^p(\succ_p)$.

[14]We know $D(b) \geq 1$ in all steps $q.s$ where $s < s'$. This is because agent k is available in each such step and $b \in G^k(\succ)$.

[15]The demand for b is positive is Step $q.s'$ as project b is assigned in this step and $s' < s^*$.

(ii) $a \in G^q(\succ_q)$ and $b \notin G^q(\succ_q)$.

Facts (i) and (ii) imply that $G^p(\succ_p) \not\subset G^q(\succ_q)$ and $G^q(\succ_q) \not\subset G^p(\succ_p)$. This contradictions the assumption of homophily with respect to agents p, q belonging to the component F_q.

The second case is $D(b) < D(a)$. By assumption, $S(b) \not\subset S(a)$. Thus there exists an agent $p \in S(b)$ such that $p \notin S(a)$. Since $D(b) < D(a)$, there exists an agent $q \in S(b)$ such that $q \notin S(a)$. So there exist agents $p, q \in F_q$ such that $G^p(\succ_p) \not\subset G^q(\succ_q)$ and $G^q(\succ_q) \not\subset G^p(\succ_p)$. This contradicts the homophily assumption.

We have established that in Step $q.s'$, $S(b) \subseteq S(a)$. However by assumption, $a \notin G^k(\succ_k)$ and $b \in G^k(\succ_k)$. So $k \in S(b)$ and $k \notin S(a)$ in Step $q.s'$, resulting in a contradiction.

(II) $a \in G^i(\succ_i)$.

Since $a \notin G^k(\succ_k)$, we know that (k, i, a) is formed in Step $q.s^*$ in the algorithm.[16]

An implication of $a \in G^i(\succ_i)$ is that there exists a step $q.s$ where $s < s^*$ such that

1. $(i, a) \in W$ in Step $q.s$.
2. For all Steps $q.s'$ with $s < s' < s^*$, $W = \{(i, a)\}$.[17]

We know that there exists an agent in $i \in \{l, j\}$ such that agent i is assigned project a in σ'. W.l.o.g let $i = l$. Thus $a \in G^l(\succ_l)$ as $b \in G^l(\succ_l)$ and $(l, j, b) \in \sigma(\succ)$.

We claim that agent l is not an available agent in Step $q.s$ where (i, a) is added to W. To see this, suppose not i.e. assume that agent l is available in Step $q.s$. Then $D(a) \geq 2$ as $l, i \in S(a)$ and (i, a) will not be added to the set W in this step. The algorithm will assign project q to the two maximal agents in $S(a)$ according to $\succ^N$. So l is not an available agent in Step $q.s$.

Since $(l, j, b) \in \sigma(\succ)$ where b is an acceptable project for both agents and agent l is not available in Step $q.s$, we know that $(l, j, b) \in \sigma(\succ)$ is formed in Step $q.s'$ where $s' < s$.

In the Step $q.s'$,

1. Agents l, j, i, k are available.
2. Projects a, b are available.
3. Project a has positive demand as $a \in G^l(\succ_l)$.
4. $3 \leq D(b) \leq D(a)$.

[16]Step $q.s^*$ is the termination step for the component $\tilde{F}_q$ in the MDPA.

[17]This means that for any Step $q.s'$, the least demanded project has demand atleast 2. Thus no new agent,project tuple is assigned to W in any step uptill Step $q.s^*$. So $(i, a) \in W$ is present in the termination step $q.s^*$ where (k, i, a) is formed in $\sigma(\succ)$.

Fact 4 and homophily together imply that $S(b) \subseteq S(a)$ in Step $q.s'$. However we have $k \in S(b)$ and $k \notin S(a)$. This results in a contradiction.

■

5 Strategy Proofness

The homophily assumption imposes a restriction on the profile of announced acceptable sets of agents who belong to the same component. Thus there are some delicate issues in formulating strategy proofness. We assume that friendship is commonly observable. Since friendship is mutual and transitive, no agent can *individually* manipulate and misreport her set of acceptable partners (or friends). However assuming that the acceptable project set of an agent is private information may result in a profile of acceptable project sets which are not consistent with homophily.

So we assume the following: for each component, there is a commonly known linear order $\succ^O$ over the set of projects A. For any $x, y \in A$, if $x \succ^O y$ then all agents in the component who like x also like y.[18] The private information of an agent is a project x, and her acceptable set of projects contains any project y such that $x \succ^O y$ (including x). This assumption ensures that any announced profile of acceptable project sets is consistent with homophily.

We modify the MDPA algorithm, where ties between projects which have the same demand are broken by using $\succ^O$: in any step where there are multiple minimum demanded projects, the project which is highest in the order $\succ^O$ is chosen to be assigned. Note that this is a slight modification to the MDPA, and does not change the basic principles of the algorithm. We show that the MDPA (with this modification) is strategy proof.

Consider an agent $i \in F_q$ for some $q \in \{1, \dots, L\}$. The set of acceptable projects for agent i is G^i and $\succ_i$ is the preference of agent i consistent with G^i.[19] Let G^{-i} denote the acceptable project sets of all agents in $N \setminus \{i\}$ and $\succ^{-i}$ be any preference profile consistent with G^{-i}. Let G_q^{-i} denote the acceptable project sets of all agents in $F_q \setminus \{i\}$.

We now define the assignment mechanism σ. Formally, $\sigma : \Sigma^n \to F$ where F is the feasible set of assignments.[20]

Consider an agent $i \in F_q \subseteq N$ and her acceptable project set G^i. The assignment mechanism is non manipulable for the agent i if for any $\succ_i$ consistent with G^i, for any

[18]Note that it is possible that for some $x, y \in A$, it is the case that the set of agents who like x and y is the same. Here ties can be broken using the alphabetical order on A and thus $x \succ^O y$.

[19]Note that there are two such preferences, namely the partner and the project dominant one.

[20]Note that every element in F is a collection of distinct triples satisfying two properties: each agent is paired with exactly one agent and every pair is assigned a distinct project. Also each agent is assigned a partner and a project.

$\hat{G}^i \neq G^i$; any preference $\hat{\succ}_i$ consistent with $\hat{G}^i$ and any $(G^{-i}, \succ^{-i})$, the assignment $\sigma(\succ_i', \succ_{-i})$ is not strictly preferred to $\sigma(\succ_i, \succ_{-i})$ at $\succ_i$. The assignment mechanism is strategy proof it is non manipulable for each $i \in N$.

THEOREM **3** *The MDPA algorithm is strategy proof.*

Proof: Fix an agent $i \in N$. Let $i \in F_q$ for some q. Let G^i be the set of acceptable projects for agent i and $\succ_i$ be a preference for agent i consistent with G^i. Consider G^{-i} to be the acceptable project sets for the other agents and $\succ_{-i}$ is a preference profile for the agents in $N \setminus \{i\}$ consistent with G^{-i}.

Let $\sigma(\succ_i, \succ_{-i})$ be the assignment generated by the MDPA algorithm at $\succ$ and $(k, i, a) \in \sigma(\succ)$. There are two possibilities.

Case 1: Agent i is in the residual set in $\sigma(\succ)$ i.e. $i \in R$. Note that any misreport by agent i will not change his residual status as the MDPA algorithm always picks the minimum agent in F_q according to $\succ_N$.

The priority order in R is fixed and does not depend on the reports about the acceptable project sets by the agents.

Consider a misreport by agent i, $(\hat{G}^i, \hat{\succ}_i)$. Suppose agent i strictly improves at $\succ_i$ by the misreport $(\hat{G}^i, \hat{\succ}_i)$. This implies that agent i is assigned an acceptable project in $\sigma(\hat{\succ}_i, \succ_{-i}$), say project b. Also the project assigned to agent i in $\sigma(\succ_i, \succ_{-i})$ is not an acceptable project for i i.e. $a \notin G^i$. Note that this fact is independent of whether $\succ_i$ is partner or project dominant as i is a residual agent in both $\sigma(\hat{\succ}_i, \succ_{-i})$ and $\sigma(\succ_i, \succ_{-i})$. Since a is an unacceptable project for k, agent k is unmatched agent in Step $(L + 1.r^*)$. There exists a Step $(L + 1.s)$ where k is the agent who has the highest priority in this step. There are two possibilities with respect to the available projects in this Step.

1. All acceptable projects for k are unavailable in Step $(L + 1.s)$.

2. There is an available acceptable project for k (say x), but there does not exist an agent j who has lower priority than k in Step $(L + 1.s)$ and x is an acceptable project for j.

Since the priority of agents in R is fixed in the MDPA algorithm, any misreport by agent i will leave the two facts above unchanged. Thus agent k will be still be pushed down to Step $(L + 1.r^*)$ for any possible misreport by i.

In Step $(L + 1.r^*)$, agents are still ordered using the priority order in R and consecutive agents are paired together. Thus k, i are consecutively placed in this Step, and any misreport by agent i will leave this pairing unchanged. Let k have lower priority than i. Suppose a is an unacceptable project for both agents (assignment to (k, i) in $\sigma(\succ_i, \succ_{-i})$), there are no available acceptable projects for i at this point. Thus a misreport by i will not change the project assigned to i to an acceptable project. Note that either the project assigned to (k, i)

will be unchanged or will be another unacceptable project for i in $\sigma(\hat{\succ}_i, \succ_{-i})$. Let $a \in G^k$. Since k has higher priority than i in R, any misreport by i leads to no change in the project assigned to (k, i).

Let i have higher priority than k in R. Then a possible misreport by agent i will either not change the project assigned to (i, k) or change it to another unacceptable project for i.

Case 2: Agent i is not a residual agent in $\sigma(\succ)$ i.e. $k, i \in F_q$. Any misreport will result in an assignment where agent i is matched with another agent from her component.

Assume without loss go generality that the set of acceptable projects in F_q is $\{a_1, a_2, \ldots, a_T\}$. Assume also that $a_T \succ^o a_{T-1} \succ^o \ldots \succ^o a_1$, i.e. if an agent in this component finds object a_t acceptable, then she also finds a_{t-1} acceptable, for $t = 2, \ldots, T$.

Consider an arbitrary acceptable object profile $\{G^j\}$, $j \in F_q$. As described earlier, it is determined by a profile of threshold objects, $\{a^j\}$, $j \in F_q$. This profiles generates a demand vector $D(a_t)$, $t = 1, \ldots, T$. By assumption, the acceptable object profile is homophily consistent and $D(a_1) \geq D(a_2) \geq \ldots \geq D(a_T)$. The profile also specifies the sets of agents who demand each object $S(a_1), S(a_2), \ldots S(a_T)$.

According to the MDPA (and as a consequence of homophily) , objects are considered in the sequence $a_T, a_{T-1}, \ldots, a_2, a_1$. Abusing notation slightly, let $D^r(a_t)$ denote the demand for object object a_t when object a_r is considered by the algorithm. In other words, $D^T(a_t) = D(a_t)$ in the demand vector in the first step when object a_T is being considered. When object a_r is being considered, all objects a_t where $t > r$ have already been considered. Note that these objects have not necessarily been allocated to some pair; however for the purposes of the algorithm we can regard $D^r(a_t) = 0$ whenever $t > r$. Let $S^r(a_t)$, $t \leq r$ denote the set of agents who have positive demand for object a_t.

Suppose i's threshold object in the $\{G^j\}$, $j \in F_q$ is a_k, i.e she has positive demand only for objects a_t where $t \geq k$. We begin with an important observation. Agent i can manipulate only if she does receive an acceptable object when being truthful. This implies that $D^t(a_t) \geq 3$ for all $t \leq k$. Moreover there are at least two agents in each each set $S^t(a_t)$ who are ahead of i in the numerical order. Let a_t be an object with $t \leq k$. If i has been allocated an object or is waiting when a_t is being considered, she must be receiving or will receive an acceptable object. Therefore it must be the case that $i \in S^t(a_t)$. If $D^t(a_t) = 1$, agent i will be a waiting pair and by construction of the algorithm, must receive an acceptable object. If $D^t(a_t) = 2$, i gets a_t. Hence $D^t(a_t) \geq 3$ so that object a_t must be allocated when it is considered. There must therefore be two agents who are ahead of i in the numerical order in the set $S^t(a_t)$ who are allocated a_t.

Agent i can attempt to manipulate in only one of two ways - (i) by announcing a threshold a^m where $T > m > k$, i.e by expanding the set of acceptable objects and (ii) by announcing a threshold a^m where $1 \leq m < k$, i.e by contracting the set of acceptable objects.

We consider case (i) first. Let $\hat{D}^r(a_t)$, $t \leq r$ denote the demand of objects when object

a_r is being considered, i.e these are the demands in the various stages of MDPA is run on the profile where i misreports. Similarly $\hat{S}^r(a_t)$ denotes the set of agents who demand a_t at the misreported profile. We will track the object received by i in this profile and show that it cannot belong to the set $\{a_1, a_2, \ldots, a_k\}$.

Consider objects a_r where $r > m$. For such objects, $\hat{D}^r(a_r) = D^r(a_r)$ and $\hat{S}^r(a_r) = S^r(a_r)$. Hence these objects if allocated, are allocated to the same agents in the truthful and misreported profiles. In addition, any agent, object pair who is waiting in one of the profiles is also waiting in the other.

We now turn to the case where a_m is being considered in the misreported profile. By assumption, $i \in \hat{S}(a_m)$. Let $\hat{D}^m_{-i}(a_m) = \hat{D}^m(a_m) - 1$ and $\hat{S}^m_{-i}(a_m) = \hat{S}^m(a_m) \setminus \{i\}$. Note that $D^m(a_m) = \hat{D}^m_{-i}(a_m)$ and $S^m(a_m) = \hat{S}^m_{-i}(a_m)$ by virtue of the argument in the previous paragraph.

There are several cases to consider at this point. If $\hat{D}^m_{-i}(a_m) = 0$. Clearly $(a_m i)$ is a waiting pair. If there already exists a waiting pair, then i is allocated a_m which is, by assumption, an unacceptable object. Otherwise (a_m, i) is a waiting pair and will be a allocated an objected later. Note that $\hat{D}^{m-1}_{-i}(a_{m-1}) = D^{m-1}(a_{m-1})$ and $\hat{S}^{m-1}_{-i}(a_{m-1}) = S^{m-1}(a_{m-1})$.

Suppose $\hat{D}^m_{-i}(a_m) = 1$ or $\hat{D}^m_{-i}(a_m) \geq 2$ and i is one of the two greatest agents according to the numerical order in the set $\hat{S}^m(a_m)$. Then i is allocated a_m which is an unacceptable object. Otherwise, since $S^m(a_m) = \hat{S}^m_{-i}(a_m)$, a_m will be allocated to the same two players who were allocated a_m in the truthful profile. Once again, $\hat{D}^{m-1}_{-i}(a_{m-1}) = D^{m-1}(a_{m-1})$ and $\hat{S}^{m-1}_{-i}(a_{m-1}) = S^{m-1}(a_{m-1})$.

We can therefore conclude the following. If i is allocated an object at this stage, it must be to an unacceptable object; if she is not allocated an object then $\hat{D}^{m-1}_{-i}(a_{m-1}) = D^{m-1}(a_{m-1})$ and $\hat{S}^{m-1}_{-i}(a_{m-1}) = S^{m-1}(a_{m-1})$.

Suppose i is not allocated an object at stage a_m. Consider stage a_{m-1}. Suppose $i \notin S(a_{m-1})$, i.e. $m - 1 > k$ and a_{m-1} is not an acceptable object for i. If $\hat{D}^{m-1}_{-i}(a_{m-1}) = 0$, i remains waiting with a_m. If $\hat{D}^{m-1}_{-i}(a_{m-1}) = 1$, i is allocated a_{m-1} with the other demander for a_{m-1}. Since a_{m-1} is not an acceptable object for i, manipulation fails. Similarly $\hat{D}^{m-1}_{-i}(a_{m-1}) \geq 2$ and i is one of the two highest agents according to the numerical order in $\hat{S}^{m-1}(a_{m-1})$, i is allocated a_{m-1}. The only remaining case is when i is not among the two highest agents $\hat{S}^{m-1}(a_{m-1})$ according to the numerical order. In this case, $\hat{S}^{m-1}_{-i}(a_{m-1}) = S^{m-1}(a_{m-1})$ implies that the same pair of agents are allocated a_{m-1} in the misreported as in the truthful profile.

Summarizing, we have reached the same conclusion in stage a_{m-1} as in stage a_m. Suppose a_{m-1} is not acceptable object for i. If i is allocated an object in stage a_{m-1}, it must be to an unacceptable object. If i has not been allocated an object, it must be true that $\hat{D}^{m-2}_{-i}(a_{m-2}) = D^{m-2}(a_{m-2})$ and $\hat{S}^{m-2}_{-i}(a_{m-2}) = S^{m-2}(a_{m-2})$.

In fact, the same argument can be replicated for all stages a_{m-t}, $t = 0, \ldots, m - k$ till one reaches a_k where the object been considered by the algorithm in the misreported profile is an acceptable object of i. In particular, we can conclude that either i is allocated an unacceptable object or the algorithm reaches stage a_k with $\hat{D}^k_{-i}(a_k) = D^k(a_k)$ and $\hat{S}^k_{-i}(a_k) = S^k(a_k)$.

At stage a_k, we have already argued that $D^k(a_k) \geq 3$ and there are two agents other than i in $S^k(a_k)$ who are ahead of i in the numerical order. Therefore $\hat{D}^k_{-i}(a_k) \geq 2$. Moreover $\hat{S}^k_{-i}(a_k) = S^k(a_k)$ implies that the same pair of agents who were allocated a_k in the truthful profile are also allocated a_k in the misreported profile. The same argument can be applied repeatedly to show that if i has not been allocated at stage a_{k+1}, she cannot be allocated any of the objects $a_1, \ldots a_{k-1}, a_k$. The algorithm then allocates an unacceptable object to i. This can happen in one of two ways. If i is waiting with an object a_l, $k < l \leq T$, after a_1 has been allocated, t i will be allocated a_l. Otherwise i will receive an unacceptable object after allocation of acceptable objects in F_q has been completed. In any case, manipulation does not succeed.

Finally consider case (ii) where i misreports by contracting her acceptable set. Following the earlier argument, it is clear that allocation proceeds in exactly the same manner in the misreported profile as in the truthful profile until stage a_k is reached. This implies that the same agents who were allocated objects $a_1, \ldots a_k$ are not allocated in stages a_{k+1} and earlier. Hence at each stage a_t, $t = k, k-1, \ldots, 1$, $\hat{D}_{-i}(a_t) \geq 2$ and $\hat{S}_{-i}(a_t)$ contains two agents with numerical order greater than i. Therefore these agents are allocated $a_1, \ldots, a_k$ and i does not receive acceptable project. This completes the proof.

■

6 Dichotomous Domain without Homophily

In this section, we demonstrate the centrality of the homophily assumption for the existence of stable assignments. We provide examples to show that stable assignments do not exist for either partner dominant or project dominant preferences.

We retain all the assumption on preferences in Section 2.1 except the homophily assumption.

Example **6** Let $N = \{1,2,3,4\}$ and $A = \{a,b,c,d,e,f\}$. Let $P^1 = \{2,3\}$, $P^2 = \{1,3\}$, $P^3 = \{1,2\}$ and $P^4 = \emptyset$, i.e. F_1 and F_2 are the two friendship components where $F_1 = \{1,2,3\}$ and $F_2 = \{4\}$. The set of acceptable projects for the agents are: $G^1 = \{a,b\}$, $G^2 = \{c,d\}$, $G^3 = \{e,f\}$ and $G^4 = \{f\}$. (See Diagram) Since the acceptable sets of all agents in the component F_1 are non-empty but mutually disjoint, the homophily assumption is violated. Let $\succ$ be the partner dominant profile with acceptable sets described above.

Proposition 4 *There does not exist a stable matching at $\succ$.*

Agent	Acceptable Set
1	$\{a, b\}$
2	$\{c, d\}$
3	$\{e, f\}$
4	$\{g, h\}$
5	$\{x, y\}$
6	$\{z, u\}$

Table 6: Proposition 3

Proof: Let σ be an arbitrary assignment. Two of the three agents in component F_1 must be matched with each other and one must be matched with 4. Since all agents in F_1 are symmetric, assume w.l.o.g that $(1, 2, a) \in \sigma$. Suppose $(3, 4, f) \in \sigma$ (The argument is unchanged if $(3, 4, e) \in \sigma$). However σ is blocked by $(2, 3, c)$. Agent 2 now gets an acceptable project *and* an acceptable partner while 3 gets an acceptable partner and an unacceptable project which she strictly prefers to getting an unacceptable partner and acceptable project by virtue of the partner dominance assumption on $\succ_3$. ■

Example 7 below shows that stable assignments may not exist without homophily when agent preferences are project dominant.

Example 7 Let $N = \{1, 2, 3, 4, 5, 6\}$ and $A = \{a, b, c, d, e, f, g, h, x, y, z, u\}$. The friendship components are F_1 and F_2 where $F_1 = \{1, 2, 3\}$ and $F_2 = \{4, 5, 6\}$. Table 6 specifies the set of acceptable projects for each agents. The acceptable sets of all agents are non-empty and mutually disjoint. Once again, the homophily assumption is violated. In the preference profile $\succ$, each agent's preference is project dominant with respect to the acceptable partner and project sets specified above.

Proposition 5 *There does not exist a stable assignment at profile $\succ$.*

Proof: Let σ be an arbitrary stable assignment at $\succ$. Since there is an odd number of agents in each component, at least one agent in F_1 is matched to an agent in F_2. Assume without lot of generality that 3 and 6 are paired with some project $\alpha \in A$, i.e $(3, 6, \alpha) \in \sigma$. Since the acceptable project sets of 3 and 6 are disjoint, at least one of these agents is getting an unacceptable project. Assume w.l.o.g that agent 3 is such an agent.

In order for 3 not to form a blocking coalition with either 1 or 2 (via an unassigned project) either (i) 1 and 2 are both getting acceptable projects or (ii) there is no unassigned project acceptable to both 1 and 2. However, (ii) is impossible since the number of projects acceptable to either 1 or 2 is four while the total number of assigned projects is three. Hence (i) must be true.

The set of acceptable projects for 1 and 2 are disjoint; in order for (i) to hold, it must be the case that both 1 and 2 are matched to agents in F_2, i.e. to agents 4 and 5. Irrespective of who agents 4 and 5 are matched to in F_1, they must *both* be getting unacceptable projects. Since they are also both getting unacceptable partners (by virtue of getting matched with agents in F_1), they can form a blocking coalition with any unassigned project (there must be at least five such projects). Hence σ is not stable which is a contradiction.

$\blacksquare$

7 Non Dichotomous Preferences

We show the non existence of stable allocations when the preferences of the agents are non dichotomous. In particular, we assume that the preference for an agent i satisfies single peakedness on each component (partner and project) and the project component lexicographically dominates the partner component.[21]

Consider a strict ordering $<_o$ on A and $<_p$ on N. Let $\tau(i)$ denote the project peak of agent i. Let P_i denote a single peaked ordering with respect to $<_o$. Note that P_i is single peaked with respect to $<_o$ if

- for all $a, b \in A$ with $b <_o a <_o \tau(i)$, we have aP_ib and
- for all $a, b \in A$ with $\tau(i) >_o a >_o b$, we have aP_ib.

Let $\pi(i)$ denote the partner peak of agent i.[22] Also let P_i' (a strict ordering over the set $N \setminus \{i\}$) denote a single peaked ordering with respect to $<_p$.

Let $\succ_i$ denote the preference of agent i over all possible partner-project tuples i.e. all tuples in $N \setminus \{i\} \times A$. The preference for agent i satisfies the following properties.

(i) For all $j \in N \setminus \{i\}$ and $a, b \in A$ with aP_ib, we have $(j, a) \succ_i (j, b)$.

(ii) For all $a \in A$ and $j, k \in N \setminus \{i\}$ with $jP_i'k$, we have $(j, a) \succ_i (k, a)$.

(iii) For all $a, b \in A$ and $j, k \in N \setminus \{i\}$ such that aP_ib, we have $(j, a) \succ_i (k, b)$.

Example 8 Let $N = \{1, 2, 3, 4, 5, 6\}$ and $A = \{a, b, c, d, e, f\}$. Let $a <_o b <_o c <_o d <_o e <_o f$ and $1 <_p 2 <_p 3 <_p 4 <_p 5 <_p 6$.

[21]Note that the preference described satisfies *multi dimensional* single peakedness as defined in Barberà et al. (1993).

[22]For any agent i, $<_p$ provides a strict ordering on $N \setminus \{i\}$.

P_1	P_2	P_3	P_4	P_5	P_6
a	b	c	d	e	f
b	a	b	e	f	e
c	c	a	f	d	d
d	d	d	c	c	c
e	e	e	b	b	b
f	f	f	a	a	a

Table 7: Single peaked preferences on projects

P'_1	P'_2	P'_3	P'_4	P'_5	P'_6
2	1	1	1	1	1
3	3	2	2	2	2
4	4	4	3	3	3
5	5	5	5	4	4
6	6	6	6	6	5

Table 8: Single peaked preferences on partners

The project peaks are: $\tau(1) = a$, $\tau(2) = b$, $\tau(3) = c$, $\tau(4) = d$, $\tau(5) = e$ and $\tau(6) = f$. Table 7 provides preferences of all agents on projects. Note that for any agent i, P_i satisfies single peakedness with respect to $<_o$. The agents 2 and 3 are left oriented i.e. any project which lies to the left of the project peak is preferred to a project which lies to the right of the peak. Similarly agents 4 and 5 are right oriented.

The partner peaks for the agents are $\pi(1) = 2$, $\pi(j) = 1$ for all $j \neq 1$. Table 8 provides the preferences of agents on partners.

In the preference profile $\succ$, each agent's preference $\succ_i$ satisfies properties (i), (ii) and (iii) with respect to P_i and P'_i.

Proposition 6 *There does not exist a stable matching at $\succ$.*

Proof: Consider an assignment σ such that σ is stable at $\succ$. For any agents $i, j \in N$ such that $\tau(i) <_o \tau(j)$, the interval $(\tau(i), \tau(j))$ consists of all projects which lie between $\tau(i)$ and $\tau(j)$. The closed interval $[(\tau(i), \tau(j) = (\tau(i), \tau(j)) \cup \{\tau(i), \tau(j)\}$.

LEMMA **1** *If the assignment σ is stable at $\succ$, then for any $i, j \in N$ with $(i, j, x) \in \sigma$, either (i) or (ii) holds. W.l.o.g let $\tau(i) <_o \tau(j)$.*

(i) $x \in [\tau(i), \tau(j)]$.

(ii) $\{\tau(i), \tau(j)\} \cap u^\sigma = \emptyset$.[23]

Proof: We will prove the lemma by contradiction. Let σ be an assignment which is stable at $\succ$. Suppose Lemma 1 is false i.e. there exist agents $i, j \in N$ (with $\tau(i) <_o \tau(j)$) such that 1, 2 and 3 hold.

1. $(i, j, x) \in \sigma$.

2. $x \notin [\tau(i), \tau(j)]$.

3. $\{\tau(i), \tau(j)\} \cap u^\sigma \neq \emptyset$.

Let $\tau(i) \in u^\sigma$. We consider two cases based on the position of x with respect to the projects peaks of agents i, j.

Case I: Let $x <_o \tau(i) <_o \tau(j)$. We have $(j, \tau(i)) \succ_i (j, x)$ and $(i, \tau(i)) \succ_j (i, x)$.[24] Thus the pair (i, j) blocks σ via the unassigned project $\tau(i)$.

Case II: Let $\tau(i) <_o \tau(j) <_o x$. There are two subcases: either $\tau(j) \in u^\sigma$ or $\tau(j)$ is assigned to another pair.

Let $\tau(j) \in u^\sigma$. The pair (i, j) blocks σ via the unassigned project $\tau(j)$. So $\tau(j) \notin u^\sigma$. Thus there exist agents k, l with $(k, l, \tau(j)) \in \sigma$.

We require the following claims to complete the argument for this case.

Claim 1.1: There does not exist an agent $s \in \{k, l\}$ such that $\tau(s) <_o \tau(i)$.

Proof: Suppose not i.e there exists an agent $s \in \{k, l\}$ with $\tau(s) <_o \tau(i)$. We assume w.l.o.g that $s = k$. We know $\tau(i) P_k \tau(j)$. Then by Requirement (iii) about $\succ_k$, $(i, \tau(i)) \succ_k (l, \tau(j)$. Similarly for agent i, $(k, \tau(i)) \succ_i (j, x)$. Since $\tau(i) \in u^\sigma$, the pair (k, i) blocks σ via the unassigned project $\tau(i)$. ■

Thus by Claim 1.1, we have $\tau(i) <_o \tau(s)$ for any $s \in \{k, l\}$.

Claim 1.2: There does not exist an agent $s \in \{k, l\}$ such that $\tau(s) \in (\tau(i), \tau(j))$.[25]

Proof: Suppose not i.e. there exists an agent $s \in \{k, l\}$ with $\tau(i) < \tau(s) < \tau(j)$. Assume w.l.o.g $s = k$.

We claim $\tau(k) \notin u^\sigma$. To show this, let $\tau(k) \in u^\sigma$. Since $\tau(i) < \tau(k) < \tau(j) < x$ and $(i, j, x), (k, l, \tau(j)) \in \sigma$, the pair (i, k) blocks the assignment via the unassigned project $\tau(k)$. So $\tau(k) \notin u^\sigma$. Thus project $\tau(k)$ must be assigned to the two remaining agents, say agents p, q.[26] We have $(p, q, \tau(k)) \in \sigma$ and $\tau(p) \in u^\sigma$.

[23]Note that u^σ is the set of unassigned projects in the assignment σ.

[24]Note that this follows from the fact that the preference for any agent s, $\succ_s$ satisfies single-peakedness with respect to the project component. The partner, project pairs being compared only differ in the projects.

[25]It is sufficient to consider the open interval as all agents have distinct project peaks.

[26]Note that $p, q \in N \setminus \{i, j, k, l\}$.

Next we show that there does not exist an agent $s \in \{p, q\}$ with $\tau(s) <_o \tau(k)$. Suppose the claim is false. Let agent p be such that $\tau(p) <_o \tau(k)$. There are two subcases.

(i) $\tau(p) <_o \tau(i) <_o \tau(k)$. The pair (p, i) blocks σ via $\tau(i)$.

(ii) $\tau(i) <_o \tau(p) <_o \tau(k)$. The pair (i, p) blocks σ via $\tau(p)$.

By a similar argument, we have $\tau(k) <_o \tau(q)$.[27]

We argue that $\tau(p), \tau(q) \notin (\tau(k), x)$. Let us suppose the claim is false. Let agent p be such that $\tau(p) \in (\tau(k), x)$. The pair (i, p) blocks σ via the unassigned project $\tau(p)$. By a similar argument, we have a contradiction if $\tau(q) \in (\tau(k), x)$.

Thus $\tau(p), \tau(q)$ satisfy either (i) or (ii): (i) one of the peaks is equal to x and the other one lies to the right of x; (ii) both peaks lie to the right of x. In both cases, there is atleast one project from $\{\tau(p), \tau(q)\}$ which is unassigned in σ. We know $(p, q, \tau(k)) \in \sigma$ and $\tau(k) <_o \tau(s)$ for any $s \in \{p, q\}$. The pair (p, q) blocks σ via the unassigned project in $\{\tau(p), \tau(q)\}$. This completes the proof of Claim 1.2.

■

Claims 1.1 and 1.2 imply that $\tau(k)$ and $\tau(l)$ must lie to the right of $\tau(j)$. We will show that this fact contradicts the assumption that σ is stable. There are three possibilities: $x \notin \{\tau(k), \tau(l)\}$, $x = \tau(k)$ and $x = \tau(l)$.

Let $x \notin \{\tau(k), \tau(l)\}$. Note that at least one project from $\{\tau(k), \tau(l)\}$ is unassigned in σ. Then σ is not stable as (k, l) blocks it via the unassigned project in $\{\tau(k), \tau(l)\}$. This follows from $(k, l, \tau(j)) \in \sigma$ and $\tau(j) <_o \tau(s)$ for any $s \in \{k, l\}$.

So either $x = \tau(k)$ or $x = \tau(l)$ holds.

Let $x = \tau(k)$. Thus $(i, j, \tau(k)), (k, l, \tau(j) \in \sigma$. The agents j and k block σ via a position swap. The assignments for agents j and k after the position swap are $(j, l, \tau(j))$ and $(i, k, \tau(k))$. The agents j, k obtain their project peaks and strictly improve after the position swap.

Let $x = \tau(l)$. Then $(i, j, \tau(l)), (k, l, \tau(j)) \in \sigma$. The agents j and l block σ via a position switch. This is because both agents obtain their project peaks after the position swap and thus strictly improve. This completes the proof for Case II. ■

Remark: Consider any stable assignment σ. There are three pairs formed in σ. Since all agents in N have distinct project peaks, it is not possible that each pair formed in σ satisfies Requirement (ii) of Lemma 1. Also there cannot exist two pairs formed in σ such that Requirement (ii) of Lemma 1 is satisfied for these pairs. To see this, let us assume that there exist two such pairs, say (i, j) and (k, l). By Requirement (ii), all projects in the set

[27]Note that since all agents have distinct peaks, $\tau(s) \neq \tau(k)$ for any $s \in \{p, q\}$.

$\{\tau(i), \tau(j), \tau(k), \tau(l)\}$ have to be assigned in σ. This is not possible as only three projects will be assigned in σ. So there can exist at most one pair in σ such that Requirement (ii) of Lemma 1 is satisfied for this pair.

Lemma **2** *If σ is stable, then at least one project from $\{\tau(1), \tau(2)\}$ is assigned in σ.*

Proof: We prove the lemma by contradiction. Let σ be a stable assignment with $\tau(1), \tau(2) \in u^{\sigma}$. Also let x and y be the projects assigned to agents 1 and 2 in σ.[28] Also for any $x \in A \setminus \{\tau(1), \tau(2)\}$, $\tau(1) <_o \tau(2) <_o x$. Thus agents $1, 2$ are assigned projects in σ which lie to the right of $\tau(2)$. The pair $(1, 2)$ blocks via $\tau(1)$ or $\tau(2)$, contradicting the assumption that σ is stable. ■

Lemma **3** *If σ is stable, then at least one object from $\{\tau(5), \tau(6)\}$ is assigned in σ.*

Proof: We use similar arguments as in Lemma 2. The only difference is that agents $5, 6$ are assigned projects which lie to the left of their respective peaks. Thus the pair $(5, 6)$ blocks σ via $\tau(5)$ or $\tau(6)$. ■

Lemma **4** *Consider $\sigma = \{(1, 2, x), (3, 4, y), (5, 6, z)\}$ for any $x, y, z \in A$. The assignment σ is not stable at $\succ$.*

Proof: We assume for the sake of contradiction that σ is stable at $\succ$. From Lemma 1 and Lemma 2, we have $x \in \{\tau(1), \tau(2)\}$. Similarly Lemma 1 and Lemma 3 together imply $z \in \{\tau(5), \tau(6)\}$. Since $x, z \notin \{\tau(3), \tau(4)\}$, Lemma 1 implies $y \in \{\tau(3), \tau(4)\}$.

Let $x = \tau(1)$. Then $\tau(2) \in u^{\sigma}$. We claim that if σ is stable at $\succ$, $y = \tau(3)$. This means that agent 3 is assigned her project peak in σ. Suppose not i.e. $y \neq \tau(3)$. Thus $(3, 4, \tau(4)) \in \sigma$. Agent 3 is left oriented (and $\tau(2) P_3 \tau(4)$) and strictly improves by moving to $\tau(2)$. The pair $(2, 3)$ blocks σ via $\tau(2)$.

So $(1, 2, \tau(1)), (3, 4, \tau(3)) \in \sigma$. We claim that $z = \tau(5)$. Suppose not i.e. $\tau(5) \in u^{\sigma}$. Then $(4, 5)$ block the assignment via $\tau(5)$. Agent 4 strictly improves as $\tau(5) P_4 \tau(3)$ (agent 4 is right oriented).

Thus $\sigma = \{(1, 2, \tau(1)), (3, 4, \tau(3)), (5, 6, \tau(5))$ and $\tau(6) \in u^{\sigma}$. The pair $(4, 6)$ blocks σ via $\tau(6)$. Agent 4 strictly improves because $\tau(6) P_4 \tau(3))$ (as agent 4 is right oriented). This contradicts the assumption that σ is stable at $\succ$.

Suppose $x = \tau(2)$. Then stability of σ at $\succ$ implies that agent 3 must be assigned $\tau(3)$. This is because $\tau(1) \in u^{\sigma}$ and agent 3is left oriented. Then $(1, 2, \tau(2)), (3, 4, \tau(3)) \in \sigma$. Since

[28]It is possible that $x = y$ which means that agents $1, 2$ are paired together in σ.

agent 4 is not assigned her project peak in σ and is right oriented, agent 5 must be assigned $\tau(5) \in u^{\sigma}$. Finally we have $(5, 6, \tau(5) \in \sigma$ and $\tau(6) \in u^{\sigma}$. The pair $(4, 6)$ blocks σ via $\tau(6)$. This contradicts the assumption that σ is stable at $\succ$. ■

Lemma **5** *Consider σ such that $(1, 2, x) \in \sigma$ for some $x \in A$. The assignment σ is not stable at $\succ$.*

Proof: We assume that σ is stable at $\succ$. By Lemma 2, atleast one of the projects from $\{\tau(1), \tau(2)\}$ must be assigned. Since agents $1, 2$ are paired together in σ, Lemma 1 implies $x \in \{\tau(1), \tau(2)\}$.

Since σ is stable, Lemma 4 implies that agent 3's partner in σ is either agent 5 or 6. Let $(3, s, y) \in \sigma$ where $s \in \{5, 6\}$ and $y \neq x$. We claim that $y = \tau(3)$. To show this, let $y \neq \tau(3)$. There are two possibilities.

(i) $y <_o \tau(3)$. This implies that $y \in \{\tau(1), \tau(2)\}$. By Lemma 1, $\{\tau(3), \tau(s)\} \cap u^{\sigma} = \emptyset$. However in the assignment at least one of projects in $\{\tau(3), \tau(s)\}$ is unassigned resulting in a contradiction.

(ii) $\tau(3) <_o y$. We claim that at least one project in $\{\tau(1), \tau(2)\}$ is unassigned in σ.[29] Let $\tau(s)$ (where $s \in \{1, 2\}$) be the unassigned project in σ. The pair $(s, 3)$ blocks σ via $\tau(s)$. This follows from the fact that agent 3 is left oriented.

We know $(3, s, \tau(3)) \in \sigma$ where $s \in \{5, 6\}$. W.l.o.g assume that $s = 5$. Then $(4, 6, z) \in \sigma$. Since $x \in \{\tau(1), \tau(2)\}$, $y = \tau(3)$, Lemma 1 and Lemma 3 imply that the pair z is assigned a project in $\{\tau(5), \tau(6)\}$. Thus $\tau(4) \in u^{\sigma}$ and the pair $(5, 4)$ block σ via $\tau(4)$. This contradicts the assumption that σ is stable.

■

Lemma **6** *Consider σ such that $(1, 3, x) \in \sigma$ for some $x \in A$. The assignment σ is not stable at $\succ$.*

Proof: We claim that $x \in [\tau(1), \tau(3)]$. Suppose not i.e. assume $x \notin [\tau(1), \tau(3)]$. By Requirement (ii) of Lemma 1, $\tau(1)$ and $\tau(3)$ have to be assigned in σ. Then Lemma 3 implies that $x \in \{\tau(5), \tau(6)\}$. Clearly, one project from $\{\tau(5), \tau(6)\}$ is unassigned. Also

[29]Lemma 1 implies that either Reqirement (i) or (ii) must hold for the remaining two pairs formed from $\{3, 4, 5, 6\}$. Since $x \in \{\tau(1), \tau(2)\}$, it must be the case that both pairs are assigned projects from their respective intervals.

agent s where $s \in \{5,6\}$ is assigned a project to the left of $\tau(s)$. The pair $(5,6)$ blocks σ via the unassigned project in $\{\tau(5),\tau(6)\}$. Thus $x \in [\tau(1),\tau(3)]$.

(6.1) Let $x = \tau(2)$. Let agent 2's assignment in σ be $(2,s,y)$ where $s \in \{4,5,6\}$. We claim that $y >_o \tau(2)$. To show this, assume $y <_o \tau(2)$ which implies $y = \tau(1)$. Since $\tau(1) \notin [\tau(2),\tau(s)]$ for any $s \in \{4,5,6\}$, Lemma 1 implies that $\tau(s)$ must be assigned in σ. There are three possibilities about the pairs formed in σ: $\{(2,4),(5,6)\}$; $\{(2,5),(4,6)\}$ and $\{(2,6),(4,5)\}$. In all these cases, Lemma 1 is violated for the pair which does not contain agent 2. For instance, consider $\{(2,4),(5,6)\}$. Then $\tau(4)$ is assigned to the pair $(5,6)$ and $\tau(5),\tau(6) \in u^\sigma$. For the pair $(5,6)$, the project assigned in σ does not belong to $[\tau(5),\tau(6)]$ and both peaks are unassigned in σ. Thus $\tau(2) <_o y$.

Let $y = \tau(3)$. So $(1,3,\tau(2)),(2,s,\tau(3)) \in \sigma$ where $s \in \{4,5,6\}$. The assignment σ is not stable at $\succ$ as agents 2 and 3 block it via a position swap. Both agents strictly improve by the position swap by obtaining their respective projects peaks in the new assignment.

So $y >_o \tau(3)$. We claim that $\tau(3) \in u^\sigma$. This follows from Lemma 3 and the fact that there is only one remaining pair (different from $(1,3)$ and $(2,s)$ where $s \in \{4,5,6\}$). Thus this pair is assigned a project from $\{\tau(5),\tau(6)\}$ in σ. Finally the pair $(2,3)$ blocks σ via $\tau(3)$.

(6.2) Let $x \neq \tau(2)$. We claim that agent 2 is assigned $\tau(2)$ in σ. Agent 2's partner in σ is an agent from $\{4,5,6\}$. Lemma 1 and $x \in \{tau(1),\tau(3)\}$ imply that any pair formed in σ from the set $\{2,4,5,6\}$ is assigned a project which lies in the interval of the project peaks. Thus $(2,s,y) \in \sigma$ where $s \in \{4,5,6\}$, $y \in (\tau(2),\tau(s)]$. The last remaining pair is assigned a project from $\{\tau(4),\tau(5),\tau(6)\}$. So $tau(2) \in u^\sigma$. There exists $i \in \{1,3\}$ such that $x \neq \tau(i)$. Then $(i,2,\tau(2))$ is a blocking coalition.[30]

We have $(1,3,x)$ (with $x \in \{\tau(1),\tau(3)\}$) and $(2,s,\tau(2))$ (with $s \in \{4,5,6\}$) belong to σ.

We first consider the case $x = \tau(1)$. Here $\tau(3) \in u^\sigma$. The pair $(3,s)$ (where $s \in \{4,5,6\}$) blocks σ via $\tau(3)$. Agent s strictly improves as $\tau(s) >_o \tau(3) >_o \tau(2)$ for all $s \in \{4,5,6\}$.

The remaining case is $x = \tau(3)$. We have $(2,s,\tau(2) \in \sigma$ where $s \in \{4,5,6\}$. Finally $(k,l,z) \in \sigma$ where $k,l \in \{4,5,6\} \setminus s$ and $z \in [\tau(k),\tau(l)]$. Note that there exists an agent in $\{k,l\}$, say agent k such that $z \neq \tau(k)$ and $\tau(k) \in u^\sigma$. Also for any agent $s \in \{4,5,6\}$, we have $\tau(k) P_s \tau(2)$. This follows from the fact that agents $4,5$ are right oriented and agent 6's project peak is to the right of the peak of all other agents. Then $(s,k,\tau(k))$ is a blocking coalition and hence σ is not stable at $\succ$.

■

LEMMA **7** *Consider σ such that $(1,4,x) \in \sigma$ for some $x \in A$. The assignment σ is not stable at $\succ$.*

[30]Suppose $i = 1$. Then agent 1 strictly improves by blocking as $\tau(2)P_1\tau(3)$. Let $i = 3$. Agent 3 strictly improves by blocking as $\tau(2)P_3\tau(1)$.

Proof: By Lemma 1, we have $x \in [\tau(1), \tau(4)]$. Note that if $x \notin [\tau(1), \tau(4)]$, then Lemma 1 implies that $\tau(1)$ and $\tau(4)$ must be assigned in σ. Note that there can exist exactly one pair in σ such that Requirement (ii) of Lemma 1 is satisfied. Thus we know that the pairs formed from $\{2, 3, 5, 6\}$ must satisfy Requirement (i) of Lemma 1. This is not possible as $\tau(1)$ does not belong to the interval of the peaks of any $i, j \in \{2, 3, 5, 6\}$. In fact any pair formed in σ from $\{2, 3, 5, 6\}$ is assigned a project belonging to the interval of the peaks of the agents in the pair.

So $x \in [\tau(1), \tau(4)]$ and the following three possibilties can arise.

(6.1) $x = \tau(2)$. Then $(1, 4, \tau(2)), (2, k, y) \in \sigma$ where $k \in \{3, 5, 6\}$.

Let $k = 3$. Lemma 1 implies $y = \tau(3)$. Also $(5, 6, z) \in \sigma$ with $z \in [\tau(5), \tau(6)]$. There exists $i \in \{5, 6\}$ with $\tau(i) \in u^\sigma$. The pair $(4, i)$ blocks σ via $\tau(i)$. Agent 4 strictly improves as she is right oriented and $\tau(i)P_4\tau(2)$ (as $\tau(2) <_o \tau(4) <_o \tau(i)$).

Let $k \in \{5, 6\}$. W.l.o.g assume $k = 5$. We claim $\tau(5) \notin u^\sigma$. To see this, suppose $\tau(5) \in u^5$. Then $(4, 5)$ block σ via $\tau(5)$. Also $\tau(3) \notin u^\sigma$. Note that if $\tau(3)$ is unassigned, then $(3, 4, \tau(3))$ is a blocking coalition. Thus $\tau(6) \in u^\sigma$. Finally the pair $(4, 6)$ blocks σ via $\tau(6)$ resulting in a contradiction.

(6.2) Let $x <_o \tau(2)$ and in particular $x = \text{peak}(1)$.

We have $(1, 4, \tau(1)), (2, k, y) \in \sigma$ with $k = \{3, 5, 6\}$. We know $y \in [\tau(2), \tau(k)]$. We claim $y = \tau(2)$. Suppose not i.e. $y \neq \tau(2)$. Then $\tau(2) \in u^\sigma$ and $(4, 2, \tau(2)$ is a blocking coalition. So $(2, k, \tau(2)) \in \sigma$ with $k \in \{3, 5, 6\}$.

Let $k = 3$. Then $\tau(3) \in u^\sigma$ (by Lemma 1) and the pair $(3, 4)$ blocks σ via $\tau(3)$.

Let $k \in \{5, 6\}$. W.l.o.g assume $k = 5$. By Lemma 1, $z \in [\tau(3), \tau(6)]$. Let $z \neq \tau(4)$. Then $\tau(4) \in u^\sigma$ and $(4, 5, \tau(4))$ is a blocking coalition. So $z = \tau(4)$. This implies $\tau(5) \in u^\sigma$ and $(5, 6, \tau(5))$ is a blocking coalition.

(6.3) Let $x >_o \tau(2)$. We claim that agent 2 is assigned $\tau(2)$ in σ.[31]

We have $(1, 4, x)$ with $x \in \{\tau(3), \tau(4)\}$ and $(2, k, \tau(2))$ with $k \in \{3, 5, 6\}$ in the assignment. Note that there exists an agent $i \in \{5, 6\}$ such that $\tau(i) \in u^\sigma$. For instance, consider the case where $k = 5$. Here $(3, 6, z) \in \sigma$ with $z \in [\tau(3), \tau(6)]$. For any possible choice of z, it is always the case that either $\tau(5)$ or $\tau(6)$ is unassigned in σ.

Let $x = \tau(3)$. Since agent 4 is right oriented, $\tau(i)P_4\tau(3)$ for any $i \in \{5, 6\}$. The pair $(4, i)$ blocks σ via $\tau(i)$.

Let $x = \tau(4)$. We claim that stability of σ implies that agent 3 must be assigned $\tau(3)$. To see this, we assume that agent 3 is not assigned $\tau(3)$. By Lemma 1, we know that $\tau(3) \in u^\sigma$. Then $(1, 3, \tau(3))$ is a blocking coalition.

[31] The argument used to prove this claim in (6.2) is still valid.

Thus agent 3 is assigned $\tau(3)$ in σ, implying that $k \neq 3$. Thus $(2, k, \tau(2)), (3, l, \tau(3)) \in \sigma$ with $k, l \in \{5, 6\}$. Also $\tau(5) \in u^\sigma$. The pair $(5, 6)$ blocks σ via $\tau(5)$. This contradicts the assumption that σ is stable at $\succ$.

■

Lemma **8** *Consider σ such that $(1, 5, x) \in \sigma$ for some $x \in A$. The assignment σ is not stable at $\succ$.*

Proof: By Lemma 1, we have $x \in [\tau(1), \tau(5)]$. Note that if $x \notin [\tau(1), \tau(5)]$, then Lemma 1 implies that $\tau(1)$ and $\tau(5)$ must be assigned in σ. Note that there can exist exactly one pair in σ such that Requirement (ii) of Lemma 1 is satisfied. Thus we know that the pairs formed from $\{2, 3, 4, 6\}$ must satisfy Requirement (i) of Lemma 1. This is not possible as $\tau(1)$ does not belong to the interval of the peaks of any $i, j \in \{2, 3, 4, 6\}$. In fact any pair formed in σ from $\{2, 3, 4, 6\}$ is assigned a project belonging to the interval of the peaks of the agents in the pair.

(7.1) Let $x <_o \tau(2)$.

We claim that for any agent $i \in \{3, 4\}$, $\tau(i) \notin u^\sigma$. To see this, assume w.l.o.g $\tau(i) \in u^\sigma$ for some $i \in \{3, 4\}$. Then $(5, i, \tau(i))$ is a blocking coalition. Agent 5 strictly improves as $\tau(i) P_5 x$ for any $i \in \{3, 4\}$.

Thus $\tau(3)$ and $\tau(4)$ are assigned to the two pairs formed from $\{2, 3, 4, 6\}$ in σ. Also $\tau(5) \in u^\sigma$. Finally the pair $(5, 6)$ blocks σ via $\tau(5)$. Agent 6 strictly improves as $\tau(5) P_6 \tau(4) P_6 \tau(3)$. This contradicts the assumption that σ is stable.

(7.2) Let $x = \tau(2)$. So $(1, 5, \tau(2))$ and $(2, k, y)$ with $k \in \{3, 4, 6\}$ belong to σ. By Lemma 1 and $x = \tau(2)$, we have $y \in (\tau(2), \tau(k)]$ where $k \in \{3, 4, 6\}$. Note that for any $k \in \{3, 4, 6\}$, $\tau(k) P_5 \tau(2)$. Thus agents 2 and 5 block σ via a position swap.

(7.3) Let $x >_o \tau(2)$. Then $x \in [\tau(3), \tau(5)]$.

We claim that $\tau(2) \notin u^\sigma$. To see this, let $\tau(2) \in u^\sigma$. The pair $(1, 2)$ blocks σ via $\tau(2)$. Agent 1 strictly improves as $\tau(2) P_1 x$ and 2 strictly improves by moving to her peak.

We show that $\tau(2)$ is assigned to agent 2 in σ. Suppose not i.e. $\tau(2)$ is not assigned to agent 2. Since $\tau(2) \notin u^\sigma$ and $x >_o \tau(2)$, we have $(l, j, \tau(2)) \in \sigma$ with $l, j \in \{3, 4, 6\}$. Note that $\tau(2)$ does not lie in the interval of the peaks of any two agents in $\{3, 4, 6\}$. By Lemma 1, the projects $\tau(l)$ and $\tau(j)$ have to be assigned to the remaining two pairs in σ with an additional contraint: the project assigned to any pair must lie in the interval of the peaks of the agents in the pair. This implies $l, j \in \{3, 4\}$ and $(3, 4, \tau(2)) \in \sigma$.[32] Also

[32] We know that there exists exactly one pair in σ which satisfies Requirement (ii) of Lemma 1. The pair $(l, , \tau(2)$ cannot contain agent 6. This is because $\tau(6)$ will then need to be assigned to two pairs, $(1, 5)$ and $(2, k)$ with $k \in \{3, 4\}$.

$x, y \in \{\tau(3), \tau(4)\}$ and $\tau(5) \in u^\sigma$. The pair $(5,6)$ block σ via $\tau(5)$. Agent 6 strictly improves as $(2,6,y)$ where $y \in \{\tau(3), \tau(4)\}$.

We have $(1,5,x)$ with $x \in [\tau(3), \tau(5)]$ and $(2,k,\tau(2))$ with $k \in \{3,4,6\}$ belong to σ. Also $(3,4,z) \in \sigma$ where $z \in [\tau(3), \tau(4)]$. This follows from the fact that the pair $(3,4)$ must satisfy Requirement (i) of Lemma 1.[33]

We claim that agent 5 is assigned $\tau(5)$ i.e. $x = \tau(5)$. To show this, assume $x \neq \tau(5)$. Then $\tau(5) \in u^\sigma$ as $y = \tau(2)$ and $z \in [\tau(3), \tau(4)]$. The pair $(5,6)$ blocks σ via $\tau(5)$.

We have shown that if σ is stable, then $(1,5,\tau(5))(2,6,\tau(2)) \in \sigma$. Agents 1 and 6 block σ via a position swap. This contradicts the assumption that σ is stable at $\succ$. ■

LEMMA **9** *Consider σ such that $(1,6,x) \in \sigma$ for some $x \in A$. The assignment σ is not stable at $\succ$.*

Proof: We claim that the pair $(1,6)$ formed in σ must satisfy Requirement (i) of Lemma 1. To see this, assume Requirement (i) is violated for $(1,6)$. Then the pair $(1,6)$ must satisfy Requirement (ii) and both projects $\tau(1)$ and $\tau(6)$ have to be assigned in σ. This will imply that any pair formed in σ from $\{2,3,4,5\}$ will violate both Requirement (i) and (ii) of Lemma 1. So $x \in [\tau(1), \tau(6)]$.

(8.1) Let $x \in [\tau(1), \tau(2)]$. Let $x = \tau(1)$. There exists an agent $i \in \{2,3,4,5\}$ such that i is assigned a project different from $\tau(i)$ and $\tau(i) \in u^\sigma$. Then $(i, 6, \tau(i)$ is a blocking coalition. Agent 6 strictly improves as $\tau(i) P_6 \tau(1)$ for any $i \in \{2,3,4,5\}$.

Let $x = \tau(2)$. There exists an agent $i \in \{3,4,5\}$ such that i is assigned a project different from $\tau(i)$ and $\tau(i) \in u^\sigma$. For instance, $(2,3,\tau(3)), (4,5,\tau(4)) \in \sigma$. In this case, $i = 5$. The pair $(6,i)$ blocks σ via $\tau(i)$.

(8.2) Let $x >_o \tau(2)$ i.e. $x \in [\tau(3), \tau(6)]$.

We claim that $\tau(2) \notin u^\sigma$. To see this, assume $\tau(2) \in u^\sigma$. Th pair $(1,2)$ blocks σ via $\tau(2)$.

Next we show that $\tau(2)$ must be assigned to agent 2. We assume for contradiction that $\tau(2)$ is not assigned to agent 2. Since $\tau(2) \notin u^\sigma$, it is assigned to a pair formed from $\{3,4,5\}$. There are two subcases: (a) the pairs $(2,3)$ and $(4,5)$ are formed in σ and (b) the pairs $\{(2,4),(3,5)\}$ or $\{(2,5),(3,4)\}$ are formed in σ.

(a) Here $(4,5,\tau(2)) \in \sigma$. The pair $(4,5)$ is assigned a project which does not belong to $[\tau(4), \tau(5)]$. Then by Requirement (ii) of Lemma 1, both $\tau(4)$ and $\tau(5)$ must be assigned in σ. This will result in the pair $(2,3)$ violating both requirements of Lemma 1.

[33] It is not possible that $(3,4)$ satisfies Requirement (ii) of Lemma 1 as $y = \tau(2)$ in σ. So it is possible to assign exactly one project from $\{\tau(3), \tau(4)\}$.

(b) Here either $(3, 5, \tau(2)$ or $(3, 4, \tau(2)$ belongs to σ. We have $x \in [\tau(3), \tau(6)]$. Let $(3, 5, \tau(2)) \in \sigma$. Agents 1 and 5 block σ via a position swap. Agent 5 strictly improves as $\tau(i)P_5\tau(2)$ for any $i \in \{3, 4, 5, 6\}$.[34] We can argue similarly for the case where $(3, 4, \tau(2)) \in \sigma$. Agents 1 and 4 block σ via a position swap.

Thus we have $(2, k, \tau(2)) \in \sigma$ with $k \in \{3, 4, 5\}$. We first consider the case where $k \in \{4, 5\}$. So $(1, 6, x)$ (with $x \in [\tau(3), \tau(6)]$) and $(2, k, \tau(2))$ (with $k \in \{4, 5\}$) belong to σ. Agents 1 and k block σ via a position swap. For instance, let $k = 4$. We have $xP_4\tau(2)$ for $x \in [\tau(3), \tau(6)]$. This follows from the fact that 4 is right oriented.

The only remaining case is $k = 3$. There are two possibilties: $x = \tau(3)$ or $x \in (\tau(3), \tau(6)]$. Let $x = \tau(3)$. Then agents 1 and 3 block σ via a position swap.

So $x \in (\tau(3), \tau(6)]$. By Lemma 1, $(4, 5, z) \in \sigma$ with $z \in [\tau(4), \tau(5)]$. This implies $\tau(3) \in u^\sigma$ and the pair $(1, 3)$ blocks σ via $\tau(3)$. We have a contradiction to the assumption that σ is stable.

■

This completes the proof of the theorem.

■

8 Conclusion

In this paper, we have investigated a class of matching models where agents' have to be matched in pairs with a project. We provide a restriction on partner,project pairs based on homophily that guarantees the existence of a robustly stable assignment. We provide an algorithm, the MDPA algorithm which generates a robustly stable assignment and satisfies Pareto efficiency imposed on each friendship component and has good strategic properties.

In future, we hope to extend our work to teams of general size. Our current results on pairs do not extend in a straightforward manner to the more general case. We also hope to investigate other notions of capturing homophily in project and team assignment models.

9 Appendix

We show that MDPA algorithm generates a robustly stable assignment at every preference profile.

Proof: Let $\succ$ be an arbitrary profile and σ be the assignment generated by the PBCA algorithm at $\succ$. We will show that σ cannot be blocked via an unassigned project, by a

[34]This is because either agent 5 moves closer to her peak or moves to an alternative to the right of her peak. Agent 5 is right oriented.

position swap or by a project swap at any $\succ'$ which is acceptable set equivalent to $\succ$.

An important initial observation is that the algorithm relies only on the $P^i(\succ_i)$,$G^i(\succ_i)$ sets for all agents. Therefore if two profiles $\succ, \succ'$ induce the same acceptable sets for all agents, the algorithm generates the same assignment.

Case I: Suppose by way of contradiction that σ is blocked via an unassigned project, i.e. there exist $(k, i, a), (l, j, b) \in \sigma$ and $b \in u^\sigma$ such that[35]

1. $(k, l, b) \succ'_k (k, i, a)$.
2. $(k, l, b) \succ'_l (l, j, c)$.

where $\succ'$ is acceptable set equivalent to $\succ$.

We consider various subcases.

I.(i) The agents $k, l \in F_q$ for some q. There are two subcases to consider.
(a) Agents k, l are paired together in σ, i.e. $(k, l, a) \in \sigma$. An application of 1 and 2 above yields $a \notin G^k(\succ'_k)$, $a \notin G^l(\succ'_l)$ and $b \in G^k(\succ'_k) \cap G^l(\succ'_l)$.[36] In the algorithm, it must have been the case that (k, l, a) was formed in Step $q.s^*$ (the termination step for the component F_q) and the demand for each available project in this step is zero. Since $b \in u^\sigma$, b is available in Step $q.s^*$ and $D(b) = 2$ in Step $q.s^*$. This results in a contradiction.

(b) Agents k, l are not paired together in σ, i.e. $(k, i, a), (l, j, c) \in \sigma$ where $i \neq l$ and $j \neq k$. In the algorithm, there is atmost one residual agent from each friendship component. Thus there are three subcases: $i, j \in F_q$; $i \in F_q$, $j \notin F_q$ and $i \notin F_q$,$j \in F_q$. Without loss of generality, we assume that agent l is the residual agent.[37] Thus we can remove the third possibility and we have $k, i \in F_q$. From 1, we have $a \notin G^k(\succ'_k)$ and $b \in G^k(\succ'_k)$. In the algorithm, it must have been the case that agent k is assigned a project and partner in Step $q.s^*$, where the demand for each available project is zero. Since $b \in u^\sigma$, project b is available in Step $q.s^*$. Also $D(b) = 1$ in Step $q.s^*$ as $b \in G^k(\succ'_k)$. This results in a contradiction.

I.(ii) Agents k, l belong to different partitions. Let $k \in F_q$ and $l \in F_p$. There are several possibilities to consider.

(a) $\succ'_k$ and $\succ'_l$ are both partner dominant. Since k and l are strictly improving, we must have $i \notin F_q$ and $j \notin F_p$. Thus k, i, l, j all belong to the residual set R. Moreover from 1 and 2, we have $a \notin G^k(\succ'_k)$, $b \in G^k(\succ'_k)$, $c \notin G^l(\succ'_l)$ and $b \in G^l(\succ'_l)$. Since k and l are being assigned unacceptable projects in σ, we can conclude that (k, i, a) and (l, j, c) are being formed in Step $(L + 1.r^* + 1)$. Suppose agent k has higher priority than l in R. Since b is unassigned,

[35]Note that it is possible that $i = l$ and $j = k$. This means that in σ, agents k, l are paired together with an project i.e. $a = c$.

[36]This fact is independent of whether $\succ'_k$ is partner or project dominant.

[37]Note that the second and third subcases are symmetric.

there will exist a Step $(L+1.q)$ $(q \leq r^*))$ where agents (k, l) could have been paired together with project b. Therefore we have a contradiction.

(b) $\succ'_k$ and $\succ'_l$ are both project dominant.
Suppose one of the pairs (k, i), (l, j) belong to the same friendship component. Without loss of generality, suppose $k, i \in F_q$ for some q. From 1, we have $a \notin G^k(\succ'_k)$ and $b \in G^k(\succ'_k)$. We have a contradiction exactly as in Case I.(i)(b).

Finally suppose neither k, i nor l, j belong to the same component. Thus k, i, l, j belong to the residual set. This reduces to Case I.(ii)(a) which we have dealt with.

(c) One of $\succ'_k$ and $\succ'_l$ is partner dominant while the other is project dominant. Suppose $\succ'_k$ is partner dominant. From 1, i and k cannot belong to the same friendship component. Also $a \notin G^k(\succ'_k)$ and $b \in G^k(\succ'_k)$.

There are two possibilities. Suppose l, j do not belong to the same component. Then agents k, i, l, j are residual agents. Once again, we are back to Case I.(ii).(a). The remaining case is $l, j \in F_p$ for some p. By 2, we have $c \notin G^l(\succ'_l)$ and $b \in G^l(\succ'_l)$. Note that this case is equivalent to case I.(i).(b), where agent k is replaced by l and i is replaced by j.

So this concludes Case I, i.e. σ cannot be blocked via an unassigned project.

Case II: Suppose σ can be blocked by a position swap. Suppose $(k, i, a), (l, j, b) \in \sigma$ and

3. $(k, j, b) \succ'_k (k, i, a)$.

4. $(l, i, a) \succ'_l (l, j, b)$.

where $\succ'$ is acceptable set equivalent to $\succ$.

We will deal with this case by showing that all the agents k, i, l, j belong to the residual set. We are then able to show a contradiction. Claims 1, 2 and 3 show that all four, no three and no pair of agents can belong to the same friendship component respectively.

Claim 1: $\{k, i, l, j\}$ cannot belong to the same friendship component.

Proof: Suppose not i.e. $k, i, l, j \in F_q$ for some q. By homophily, either $G^k(\succ'_k) \subseteq G^l(\succ'_l)$ or $G^l(\succ'_l) \subseteq G^k(\succ'_k)$. We assume w.l.o.g that $G^k(\succ'_k) \subseteq G^l(\succ'_l)$. Since k, i, j are in the same friendship component and k strictly improves in the blocking, it must be true that $a \notin G^k(\succ'_k$ and $b \in G^k(\succ'_k)$. By an analogous argument, $b \notin G^l(\succ'_l)$ and $a \in G^l(\succ'_l)$. This contradicts the homophily assumption. ■

Claim 2: No three agents from the set $\{k, i, l, j\}$ belong to the same friendship component.

Proof: Suppose not. In view of Claim 1, there must exist exactly three agents from the set $\{k, i, l, j\}$ who belong to the same friendship component. The triple is one of the following sets: $\{k, i, l\}$,$\{k, i, j\}$,$\{k, l, j\}$, $\{i, l, j\}$.

Suppose that the agents k, i, l belong to the same friendship component i.e. $k, i, l \in F_q$ for some q. There are two possibilities. Suppose $\succ'_k$ is partner dominant. Since $j \notin F_q$, agent k does not strictly improve by blocking.

The other possibility is $\succ'_k$ is project dominant. Since $j \notin F_q$ and $(l, j, b) \in \sigma$, we know l, j are residual agents and $b \in U(R)$. Also $U(R) \subseteq U(F_q)$ and project b is available when projects are assigned to any pair in F_q. Application of 3 yields $a \notin G^k(\succ'_k)$ and $b \in G^k(\succ'_k)$.

Since $a \notin G^k(\succ'_k)$ and $(k, i, a) \in \sigma$, agent k is assigned a partner and a project in Step $q.s^*$. Note that the demand for each available project in Step $q.s^*$ is zero. However project b is available in Step $q.s^*$ (as $b \in U(R)$). Also $D(b) \geq 1$ as $b \in G^k(\succ'_k)$ in Step $q.s^*$. This is because agent k is available in this Step and $b \in G^k(\succ'_k)$. This results in a contradiction.

Suppose the triple is $\{k, i, j\}$ and $k, i, j \in F_q$ for some q. After the swap, agent k's partner is still acceptable. Therefore 3 implies a is an acceptable project for k, while b isn't. Since $l \notin F_q$ and (l, j) was paired with the project b in σ, $l, j \in R$ and $b \in U(R)$. Also $b \in U(R) \subseteq U(F_q)$ i.e. project b was available when projects were assigned to any pair in F_q. This reduces to the case above where the triple is $\{k, i, l\}$.

The case where $\{k, l, j\}$ is the triple belonging to the same friendship component can be dealt with in the same way as $\{k, i, l\}$.[38] Similarly the $\{i, j, l\}$ case is identical to the $\{k, i, j\}$ case. This completes the proof of the Claim.

■

Claim 3: No two agents in the set $\{k, i, l, j\}$ can belong to the same friendship component.

Proof: Suppose $k, l \in F_q$ for some q. We know from Claims 1 and 2 that $i, j \notin F_q$. Since $(k, i, a), (l, j, b) \in \sigma$, agents k, l are residual agents from the component F_q. This is impossible according to the algorithm. Similarly $\{k, j\}$ and $\{i, j\}$ cannot belong to the same component. The remaining cases are: $\{k, i\}$ and $\{l, j\}$. By symmetry it suffices to consider only one case.

Suppose $k, i \in F_q$ for some q. If $\succ_k$ is partner dominant, agent k cannot strictly improve by blocking since $j \notin F_q$. Suppose therefore $\succ_k$ is project dominant. Using 3, we have $a \notin G^k(\succ'_k)$ and $b \in G^k(\succ'_k)$. There are two subcases possible: $l, j \in F_p$ for some $p \neq q$ or l, j do not belong to the same friendship component.

Let $l, j \in F_p$. If $\succ_l$ is partner dominant, agent l cannot strictly by blocking since $i \notin F_p$. Suppose $\succ_l$ is project dominant. W.l.o.g let $q < p$.[39] This implies $U(F_p) \subseteq U(F_q)$. Since $(l, j, b) \in \sigma$, we have $b \in U(F_p)$. This means that $b \in U(F_q)$ and b was available when projects were assigned to any pair in F_q. Since $a \notin G^k(\succ'_k)$ and $(k, i, a) \in \sigma$, agent k is assigned a partner and a project in Step $q.s^*$. Note that the demand for each available project in Step $q.s^*$ is zero. However project b is available in Step $q.s^*$ (as $b \in U(R)$). Also $D(b) \geq 1$ as

[38] We argue by replacing agent k with l.

[39] Suppose $p < q$. Then $a \in U(F_q) \subseteq U(F_p)$. An implication of 4 is $a \in G^l(\succ'_l)$ and $b \notin G^l(\succ'_l)$. We can now replicate the argument described for the case $p < q$ by replacing agent k with l and agent i with agent j.

$b \in G^k(\succ'_k)$ in Step $q.s^*$. This is because agent k is available in this Step and $b \in G^k(\succ'_k)$. This results in a contradiction.

Let $l, j \in R$. Thus $b \in U(R) \subset U(F_q)$ and b was available when projects were assigned to any pair in F_q. This reduces to the above case where $l, j \in F_p$ and $q < p$.

■

In view of Claims 1, 2 and 3, the only remaining possibility is all agents in $\{k, i, l, j\}$ belong to different friendship components i.e. all these agents belong to the set R. We show below that this leads to a contradiction. Since k, l block, we must have $a \notin G^k(\succ'_k)$,$b \in G^k(\succ'_k)$ and $b \notin G^l(\succ'_l)$, $a \in G^l(\succ'_l)$. This implies that the assignments (k, i, a) and (l, j, b) are made in the Step $(L+1.r^*+1)$ in the algorithm. Assume w.l.o.g. that k has higher priority than l in R. We consider several possibilities based on the priorities of the agents $\{k, i, j\}$ in R.

Suppose k has highest priority amongst $\{k, i, j\}$. Suppose one of the agents in $\{i, j\}$ has b as an acceptable project. Since b is available in Step $(L+1.r^*+1)$, it must also have been available at all steps $(L+1.s)$ where $s \le r^*$. According to the algorithm, there exists a step $(L+1.s')$ where $s' \le r^*$ where there exists an agent with a *common acceptable* project with k and a lower priority than k. According to the algorithm, k must be assigned a partner and a project in Step $(L+1.s')$. Consequently k cannot be an agent in Step $(L+1.r^*+1)$. Suppose such an agent does not exist. Since (k, i, a) in σ, k and i must be consecutive priority in Step $(L+1.r^*+1)$. Since k, i have higher priority than l, j in Step $(L+1.r^*+1)$ and b is assigned to pair (l, j), project b must also be available to the pair (k, i). Also k has higher priority than i by assumption, $a \notin G^k(\succ'_k)$ and $b \in G^k(\succ'_k)$. Hence $(k, i, a) \in \sigma$ is not possible.

Agent j cannot have higher priority than k in Step $(L+1.r^*+1)$. If this were true, then l, j could not be paired in σ. The only remaining case is that agent i has the highest priority in the set $\{k, i, j\}$. Suppose $a \in G^i(\succ'_i)$. It follows from the argument in the previous paragraph that i must have been assigned a partner and a project in a step before $(L+1.r^*+1)$. Finally suppose $a \notin G^i(\succ'_i)$. Since a is not an acceptable project for both agents i, k and b is available when (i, k) is assigned a project in Step $(L+1.r^*+1)$. Hence $(i, k, a) \in \sigma$ is not possible.

Case III: Suppose by way of contradiction that σ is blocked via a project swap i.e. there exists $(k, i, a), (l, j, b) \in \sigma$ such that

5. $(k, i, b) \succ'_k (k, i, a)$.

6. $(k, i, b) \succ'_i (k, i, a)$.

7. $(l, j, a) \succ'_l (l, j, b)$.

8. $(l, j, a) \succ'_j (l, j, b)$.

We will deal with this case by arguing that all the agents k, i, l, j belong to the residual set and then show a contradiction.

Claim 4: $\{k, i, l, j\}$ do not belong to the same friendship component.

Proof: Suppose not i.e. $k, i, l, j \in F_q$ for some q. By homophily, either $G^k(\succ'_k) \subseteq G^l(\succ'_l)$ or $G^l(\succ'_l) \subseteq G^k(\succ'_k)$. We assume w.l.o.g that $G^k(\succ'_k) \subseteq G^l(\succ'_l)$. Since k, i, j are in the same friendship component and k strictly improves in the blocking, it must be true that $a \notin G^k(\succ'_k)$ and $b \in G^k(\succ'_k)$. By an analogous argument, $b \notin G^l(\succ'_l)$ and $a \in G^l(\succ'_l)$. This contradicts the homophily assumption. ■

Claim 5: No three agents from the set $\{k, i, l, j\}$ belong to the same friendship component.

Proof: Suppose not. In view of Claim 4, there must exist exactly three agents who belong to the same friendship component. The triple is one of the following sets: $\{k, i, l\}$, $\{k, i, j\}$, $\{i, j, l\}$ and $\{k, j, l\}$. Note that any triple contains an agent $s \in \{k, l\}$ and her assigned partner in σ. Since the argument relies only on the agent s and her partner in σ, it suffices to consider one case.

Suppose the triple is $\{k, i, l\}$ i.e. $k, i, l \in F_q$ for some q. Since $j \notin F_q$, we conclude l, j are residual agents and $b \in U(R)$. Implications of 5 and 6 are $a \notin G^k(\succ'_k)$, $a \notin G^i(\succ'_i)$ and $b \in G^k(\succ'_k) \cap G^i(\succ'_i)$. According to the algorithm, (k, i, a) is formed in Step $q.s^*$. Step $q.s^*$ is the termination step for the component F_q where the demand for each available project is zero. However project b is available in Step $q.s^*$. Also $D(b) = 2$ in step $q.s^*$. This results in a contradiction. ■

Claim 6: No two agents in the set $\{k, i, l, j\}$ can belong to the same friendship component.

Proof: Suppose $k, l \in F_q$ for some q. We know from Claims 4 and 5 that $i, j \notin F_q$. Since $(k, i, a), (l, j, b) \in \sigma$, agents k, l are residual agents from the component F_q. This is impossible according to the algorithm. Similarly $\{k, j\}$ and $\{i, j\}$ cannot belong to the same component. The remaining cases are: $\{k, i\}$ and $\{l, j\}$. By symmetry it suffices to consider only one case.

Suppose $k, i \in F_q$ for some q. Using 5 and 6, we have $a \notin G^k(\succ'_k)$, $a \notin G^k(\succ'_k)$ and $b \in G^k(\succ'_k) \cap G^i(\succ'_i)$.

There are two subcases possible: $l, j \in F_p$ for some $p \neq q$ or l, j do not belong to the same friendship component.

Let $l, j \in F_p$ and w.l.o.g assume $q < p$. Thus $b \in U(F_p) \subseteq U(F_q)$. Since $(k, i, a) \in \sigma$ and a is not an acceptable project for both agents k and i, we infer that (k, i, a) is formed in Step $q.s^*$. Note that in Step $q.s^*$, the demand for each available project is zero. This contradicts that b is an available project in Step $q.s^*$ and has demand $D(b) = 2$.

The remaining case is l, j do not belong to the same component. Since $(l, j, b) \in \sigma$, l, j are residual agents and $b \in U(R)$. Exactly as in the previous paragraph, b is available in Step $q.s^*$ resulting in a contradiction. ■

In view of Claims 4, 5 and 6, the only remaining possibility is that all agents in $\{k, i, l, j\}$ belong to different friendship components i.e. $k, i, l, j \in R$. Requirements 5 and 6 imply $a \notin G^k(\succ'_k)$, $a \notin G^i(\succ'_i)$ and $b \in G^k(\succ'_k) \cap G^i(\succ'_i)$. Similarly from 7 and 8, b is not an acceptable project for l, j while a is an accepatable project for both agents. Since $(k, i, a), (l, j, b) \in \sigma$, we know that k, i and l, j are consecutive in priority in Step $(L + 1.r^* + 1)$.[40] Also since a is not an acceptable project for both k, i, the pair (k, i) is a waiting pair and is assigned a project in Step $(L + 2)$. Similarly the project assignment for the pair (l, j) is made in Step $(L + 2)$. Note that both projects a, b are available for assignment in Step $(L + 1.s)$ for all $1 \leq s \leq r^* + 1$.

W.l.o.g assume that agent k has higher priority than l in R. Let agent i have higher priority than k in R. According to the algorithm, there exists a Step $s' \leq r^*$ where there exists an agent (here agent k) with a *common acceptable* project with i (project b) and a lower priority than i. According to the algorithm, agent i must be assigned a partner and project in Step $(L + 1.s')$. Thus i cannot be an agent in Step $(L + 1.r^* + 1)$.

Let agent k have higher priority than i in R. Following the arguments in the previous paragraph, we conclude that k cannot be an agent in Step $(L + 1.r^* + 1)$ which is a contradiction.

This completes the proof of the theorem. ■

References

Alcalde, J. (1994): "Exchange-proofness or divorce-proofness? Stability in one-sided matching markets," *Economic Design*, 1, 275–287.

Barberà, S., F. Gul, and E. Stacchetti (1993): "Generalized median voter schemes and committees," *Journal of Economic Theory*, 61, 262–289.

Bauman, K. E. and S. T. Ennett (1996): "On the importance of peer influence for adolescent drug use: Commonly neglected considerations," *Addiction*, 91, 185–198.

Chung, K.-S. (2000): "On the existence of stable roommate matchings," *Games and economic behavior*, 33, 206–230.

Cohen, J. M. (1977): "Sources of peer group homogeneity," *Sociology of Education*, 227–241.

[40]Agents k, i, l, j are present in Step $(L + 1.r^* + 1)$ and the pairs (k, i) and (l, j) are formed in this Step. These pairs are assigned projects in Step $(L + 2)$ since both pairs have been assigned a project which is not acceptable for both agents in the pair.

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

Golub, B. and M. O. Jackson (2012): “How homophily affects the speed of learning and best-response dynamics,” *The Quarterly Journal of Economics*, 127, 1287–1338.

Gudmundsson, J. (2014): “When do stable roommate matchings exist? A review,” *Review of Economic Design*, 18, 151–161.

Kandel, D. B. (1978): “Homophily, selection, and socialization in adolescent friendships,” *American journal of Sociology*, 84, 427–436.

Khare, S. and S. Roy (2016): “Stability in Matching with Groups having Non-Responsive Preferences,” .

Klaus, B. and F. Klijn (2005): “Stable matchings and preferences of couples,” *Journal of Economic Theory*, 121, 75–106.

——— (2007): “Paths to stability for matching markets with couples,” *Games and Economic Behavior*, 58, 154–171.

Klaus, B., F. Klijn, and J. Massó (2007): “Some things couples always wanted to know about stable matchings (but were afraid to ask),” *Review of Economic Design*, 11, 175–184.

Lazarsfeld, P. F., R. K. Merton, et al. (1954): “Friendship as a social process: A substantive and methodological analysis,” *Freedom and control in modern society*, 18, 18–66.

McPherson, M., L. Smith-Lovin, and J. M. Cook (2001): “Birds of a feather: Homophily in social networks,” *Annual review of sociology*, 27, 415–444.

Ruef, M., H. E. Aldrich, and N. M. Carter (2003): “The structure of founding teams: Homophily, strong ties, and isolation among US entrepreneurs,” *American sociological review*, 195–222.

Selfhout, M. H., S. J. Branje, T. F. ter Bogt, and W. H. Meeus (2009): “The role of music preferences in early adolescentsâĂŹ friendship formation and stability,” *Journal of adolescence*, 32, 95–107.

Tan, J. J. (1991): “Stable matchings and stable partitionsâĹŮ,” *International Journal of Computer Mathematics*, 39, 11–20.

Verbrugge, L. M. (1983): “A research note on adult friendship contact: a dyadic perspective,” *Social Forces*, 78–83.