# Stable Matching Beyond Bipartite Graphs

Jie Wu
Department of Computer and Information Sciences
Temple University, USA
jiewu@temple.edu

***Abstract*—Binary matching in bipartite graphs and its extensions have been well studied over the decades. A stable matching (or marriage) seeks to establish a stable binary pairing of two genders, where each member in a gender has a preference list for the other gender. In addition, all members are paired (also called perfect matching), with one selection made from each gender. Gale and Shapley showed that a stable marriage exists for any set of preferences in any complete and balanced bipartite graphs. However, stable matching may not exist in a non-bipartite graph represented by a complete graph (i.e. single gender), as shown in the stable roommates problem. In this paper, we study binary matching in complete and balanced $k$-partite graphs with an even number of members, and show that except when $k = 2$, preference lists always exist under which a stable binary matching does not exist. We use a solution for the stable roommates problem to illustrate the matching identification process if one exists. Under a natural extension of pairwise stability for binary matching to $k$-ary matching, we show that stable $k$-ary matching exists for any preference lists in any complete and balanced $k$-partite graphs. Here, we assume that there is a strict preference order of the members over all individual members from different genders, as opposed to preference order over a combination of members from different genders used in existing multi-dimensional extensions. An extended Gale-Shapley algorithm is introduced, together with various implementations, to determine such a stable $k$-ary matching. A parallel implementation is also proposed to speed up the matching process. In the extension, we further propose a weakened unstable condition for $k$-ary matching and show the existence of stable $k$-ary matching.**

***Index Terms*—**Binary matching, bipartite graphs, equivalence relation, $k$-ary matching, $k$-partite graphs, parallel implementation, stable marriage problem, stable roommates problem.

## I. Introduction

Binary matching is well-studied in graph theory. It involves pairing two nodes in a given graph, such that each node appears in one and only one pair. Such pairings are also called *perfect matching*. Binary matching usually seeks some objectives subject to several constraints. For example, in maximum-weighted bipartite matching [1], the objective is to maximize the total utility (or happiness) of matching pairs, represented as the total edge weight of matching pairs.

In this paper, we focus on stable matching based on a notion of stability, as defined in the *stable marriage problem* (SMP) [2]. In the SMP, two genders, men and women, are represented in a complete and balanced bipartite graph, where a matching process pairs all members from different genders. 'Complete' means that any man can be matched with any woman[1]. 'Balanced' represents that there are an equal number of men ($m$) and women ($w$). The SMP finds a stable matching between these two sets of members, given a preference list of the opposite gender for each member. A matching is stable whenever it is *not* the case that there exists two pairs where

1) $m$ of the first pair prefers $w$ of the second pair over the member to which $m$ is already matched, and
2) $w$ also prefers $m$ over the member to which $w$ is already matched.

That is, in a stable marriage, no man would say to another woman, "I love you more than my wife, let us run away" and the woman agrees, i.e., she also likes the man more than her husband. Gale and Shapley proved that, for any equal number ($n$) of men and women with their preferences, it is always possible to solve the SMP and make all marriages stable. They also provided a distributed algorithm, where men propose to women iteratively. It is shown that the SMP is solved in at most $n^2$ accumulative proposals.

There are many extensions of SMP to cases with multiple genders. Three-dimensional stable matching was originally proposed by Knuth [3], where each family consists of three members, with each member coming from each gender. In [4], the preference order is defined as one gender against the combination of all the remaining genders. For example, in the three-dimensional stable matching problem, there are $n$ members in each of three genders, each member of a gender has a preference order for all combination of the other two genders, which have $n^2$ combinations. In another variation [4], the preference rating is cyclic among genders. It has been proved that the existence of stable matching in both models, including some variations [5], are all NP-complete.

In this paper, we examine the multiple-dimensional stable matching problem through the use of $k$-partite graphs. For example, when $k = 3$, we have three types of genders in the SMP: man ($m$), woman ($w$), and undecided ($u$)[2]. The preference lists are still binary from each member of one gender to each of the other $k - 1$ genders, although separate orders are maintained for different genders, one for each gender. The binary matching allows a pair matching from any two different genders. If each gender has $n$ members (i.e., balanced), each member of a gender has $2n$ entries for two preference orders, one for each of the other two genders in a three-dimensional matching. We then propose a $k$-ary

[1] We will drop the word 'complete' in the subsequent discussion, assuming that all graphs, bipartite and later multipartite, are complete.

[2] 'Undecided' here corresponds to a third gender. Like a man and woman, any two undecided members cannot be paired. A more general model will be discussed later, in which two undecided members can be paired as well.

matching in balanced $k$-partite graphs. For example, when $k = 3$, ternary matching includes three elements, one from each of the three genders. A $k$-ary matching is a set of $n$ $k$-tuples (or families) such that each tuple has one member from $k$ genders and each member appears in exactly one tuple.

Given a set of $k$-tuples (current families), a new $k$-tuple (family) is called a *blocking family* if each member in this new family strictly prefers each of the members in the new family to the corresponding one of the current family. A $k$-ary matching is unstable if a blocking family exists; otherwise, it is stable.

In this paper, we first study the traditional stable binary matching in balanced $k$-partite graphs and extensions of these graphs. Then, we focus on the stable $k$-ary matching in balanced $k$-partite graphs. We have the following results:

1) Except for $k = 2$, there always exist preference lists under which stable binary matching does not exist in balanced $k$-partite graphs with an even number of elements.
2) Given $k$ different genders in which each gender has the same number of members, it is possible to derive a stable $k$-ary matching in balanced $k$-partite graphs.

For the first result, we discuss one potential application of stable family formation in a society with multiple gender groups. We then use the solution for *stable roommates problem* [6] to find a stable binary matching in balanced $k$-partite graphs if such a matching exists.

To find a stable $k$-ary matching based on the second result, we propose a $(k-1)$-round binary matching among $k$ genders. This matching, also called *binding process*, forms a spanning tree of $k$ genders. We show that any such spanning tree ensures a stable $k$-ary matching. In addition, we demonstrate that any more than $k-1$ bindings (that can "strengthen" the family tie) may not always exist and any less than $k-1$ bindings (that can "losen" the family tie) may cause instability. Therefore, the number $k-1$ is tight.

To enhance the second result, we weaken the unstable condition. We assume that there is a strict priority order among $k$ genders. Note that members of a blocking family must come from $k'$ ($2 \leq k' \leq k$) current families in a matching. Therefore, there are $k'$ *lead members* based on the given gender priority, one from each of these $k'$ current families. In the weakened blocking family, the condition "each member" is replaced by another condition "each lead member of a current (sub)family in the new family", which makes $k$-ary stable matching harder. We show that stable $k$-ary matching still exists through a priority-aware binding process.

The remainder of the paper is organized as follows. Section 2 reviews the Gale-Shapley algorithm, and then, defines $k$-partite graphs, $k$-ary matching, and extended stable matching. Section 3 shows the non-existence of stable binary matching in balanced $k$-partite graphs, except for $k = 2$. We use a solution for the stable roommate problem to illustrate the solution identification process if stable binary matching exists. Section 4 gives the result of stable $k$-ary matching in balanced $k$-partite graphs. An extended Gale-Shapley algorithm based on $k-1$ binary binding is given to determine such an extended stable matching. Section 5 offers the extension of the proposed model and implementation of the binding process, including a parallel implementation. Section 6 provides a short overview of related work, including gender and reproduction in the animal world that is related to $k$-ary matching. Section 7 concludes the paper and discusses directions for future work.

## II. $k$-Partite Graphs and Extended Matching

In this section, we first give a quick overview of the Gale-Shapley algorithm, followed by $k$-partite graphs, $k$-ary matching, and extended stable matching.

### A. The Gale-Shapley (GS) Algorithm

The Gale-Shapley (GS) algorithm [2] involves a number of iterations of men proposing to women in a bipartite graph. In the first round, each unengaged man first proposes to the woman he prefers most, and then each woman replies "maybe" to her suitor she most prefers, and additionally replies "no" to all other suitors. She is then provisionally "engaged" to the suitor she most prefers so far, and that suitor is likewise provisionally engaged to her. In each subsequent iteration, each unengaged man first proposes to the most-preferred woman to whom he has not yet proposed (regardless of whether the woman is already engaged), and then each woman replies "maybe" to her suitor she most prefers, and rejects the rest. Each engagement becomes final (i.e. marriage) when there is no more new proposals.

The provisional nature of engagements preserves the right of an already-engaged woman's prospects to get better and better (in terms of her preferences), while a man gets worse and worse choices. However, overall, the GS algorithm still favors men over women in terms of preferential happiness [7]. The runtime complexity of this algorithm is $O(n^2)$. The GS algorithm guarantees that everyone gets married (also called perfect matching)[3] and that all marriages are stable. In the subsequent discussion, we simply use matching to represent perfect matching, requiring that all men and women are paired.

***Example 1***: Consider a case of two men and two women $M = \{m, m'\}$ and $W = \{w, w'\}$ with the following preferences:

$m$ ranks $w$ higher than $w'$, $m'$ ranks $w$ higher than $w'$,
$w$ ranks $m'$ higher than $m$, $w'$ ranks $m'$ higher than $m$

When $m$ and $m'$ both propose to $w$, $m$ will be rejected as $w$ favors $m'$ over $m$. $m$ will then propose to $w'$ to form a stable matching: $(m', w)$ and $(m, w')$, although neither $m$ nor $w'$ is happy. If we change the preferences to the following:

$m$ ranks $w$ higher than $w'$, $m'$ ranks $w'$ higher than $w$,
$w$ ranks $m'$ higher than $m$, $w'$ ranks $m$ higher than $m'$

The GS algorithm will generate one stable matching: $(m, w)$ and $(m', w')$ in favor of men, after $m$ and $m'$ propose to $w$ and $w'$, respectively. The other stable matching, $(m, w')$ and $(m', w)$, in favor of women, is not generated by the GS algorithm, representing unfairness towards women in this case.

[3]Perfect matching is so called because everyone is paired with a member of the opposite gender, although it does not necessarily represent a happy marriage in terms of preferences.

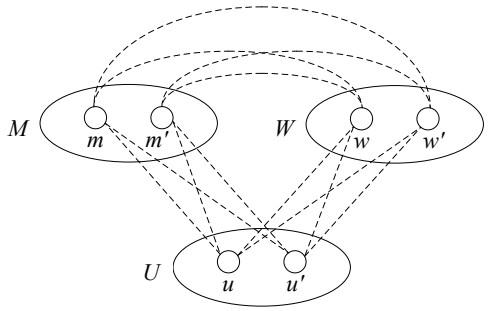


Fig. 1. An example of a balanced tripartite graph.

### B. $k$-Partite Graphs

A *$k$-partite graph* is a graph $G$ whose nodes can be partitioned into $k$ disjoint sets $G_i$, $i = 1, 2, ..., k$, so that no two vertices (members) within the same set are adjacent. When $k = 2$, it is a bipartite graph as in the stable marriage problem, and when $k = 3$, it is called a tripartite graph. A $k$-partite graph is balanced if each disjoint set (or simply set) has the same number of elements.

***Example 2***: Consider a balanced tripartite graph with three sets: $W = \{w, w'\}$, $M = \{m, m'\}$, and $U = \{u, u'\}$, as shown in Figure 1. Each node in a set connects to all four nodes in the other two sets, and provides a ranking of these four nodes. There exists a total of eight pairing choices:

$\{(m, w), (m', u), (w', u')\}$ $\{(m, w), (m', u'), (w', u)\}$
$\{(m, w'), (m', u), (w, u')\}$ $\{(m, w'), (m', u'), (w, u)\}$
$\{(m, u), (m', w), (w', u')\}$ $\{(m, u), (m', w'), (w, u')\}$
$\{(m, u'), (m', w), (w', u)\}$ $\{(m, u'), (m', u), (u', w)\}$

The question is the following: if a (binary) matching exists in a given $k$-partite graph with an even number of nodes, is stable (binary) matching guaranteed? This question will be answered in Section 3.

### C. $k$-ary Matching and Extended Stable Matching

Now we consider a $k$-ary matching in a balanced $k$-partite graph, i.e., each disjoint set has exactly $n$ nodes. In Example 2, there are four possible 3-ary matchings, as follows:

$\{(w, m, u), (w', m', u')\}$ $\{(w, m', u), (w', m, u')\}$
$\{(w, m, u'), (w', m', u)\}$ $\{(w, m', u'), (w', m, u)\}$

Again, we ask if a stable ($k$-ary) matching exists, with each member in a $k$-ary matching coming from different sets of the $k$-partite graph? This question will be further addressed in Section 4.

Next we define a notion of *extended stable matching* in $k$-ary matching, given $k$ different disjoint sets in a $k$-partite graph. A $k$-ary matching is not stable if a blocking family exists. A $k$-tuple is called a *blocking family* (to a set of existing $t$-tuples, or families) if each member in the family strictly prefers each member of that family to the each member of his or her current family. Note that this blocking family differs from the traditional blocking family in which only one preference value (such as the summation of priorities) is defined on the remaining family members of all genders. In the blocking family of this paper, the preference is defined for members of each individual gender.

Note that members of a blocking family must come $k'$ ($2 \leq k' \leq k$) existing families in a matching. For example, we assume that the first and second 3-tuple in a matching are $(m, w, u)$ and $(m', w', u')$, respectively. The following unstable matching occurs, where $m$ prefers both $w'$ and $u'$ to $w$ and $u$ of the current matching, respectively. Also, both $w'$ and $u'$ prefer $m$ to $m'$ of the current matching. Therefore, $(m, w', u')$ forms a blocking family, coming from two existing families. With more members in each gender, a blocking family can come from three existing families, one for each gender in this example.

## III. Stable Binary Matching in $k$-Partite Graphs

In this section, we first show the existence condition for a stable binary matching in $k$-partite graphs, and then use a solution for stable roommates problem to find a stable and fair binary matching if such a matching exists.

### A. Existence of a Stable Binary Matching in $k$-Partite Graphs

We consider stable binary matching in a $k$-partite graph with an even number of nodes. Let us start with the balanced tripartite graph example of Figure 1: $W = \{w, w'\}$, $M = \{m, m'\}$, and $U = \{u, u'\}$. Each node ranks four other nodes from different sets. Suppose nodes $u$ and $u'$ are the two lowest ranked by $w$, $w'$, $m$, and $m'$. It is clear that two matching nodes for $u$ and $u'$ cannot both come from $W$ (and $M$), as it will leave the remaining two nodes unmatched due to being the same gender. Without loss of generality, we assume that these two matching nodes are $w$ and $m$ (for $u$ and $u'$). However, both $w$ and $m$ rank each other higher than $u$ and $u'$, which corresponds to an unstable matching. $(m, w)$ is a blocking family (pair) for the current matching.

Next we show that for each given balanced $k$-partite graph, with $k > 2$, there exists a set of preference lists under which there exist no stable binary matching, although there is a perfect matching.

***Theorem** 1:* For a given balanced $k$-partite graph with an even number of nodes and $k > 2$, there exist preference lists under which there exist no stable binary matching, although there is a perfect matching.

*Proof*: Suppose we define such preference lists that (1) one node $u$ in a disjointed set $G_i$ is ranked as the lowest by all the other nodes, and (2) in the other $k - 1$ disjoint sets, each node is ranked the top by exactly another node from a different set among these $k-1$ sets. Suppose $m$ is a matched node from $G_j$ with $u$. Based on the above definition, there exists a node $w$ from a third disjoint set that ranks $m$ as the highest, and clearly $(m, w)$ will cause instability, i.e., $(m, w)$ is a blocking pair for the current matching. Therefore, there is no stable matching.

Now we show that a perfect matching always exists. Suppose that $k$ is even, and we can easily pairwise two disjoint

sets to construct a total of $(k/2)n$ pairs. When $k$ odd, as there is an even total number of nodes, $n$ must be even. We can equally divide $G_i$ into $G_i^{'}$ and $G_i^{''}$. A perfect matching can be constructed as the following pairings: $(G_1^{'}, G_2^{''})$, $(G_2^{'}, G_3^{''})$, ... , $(G_{k-1}^{'}, G_k^{''})$, $(G_k^{'}, G_1^{''})$, where each pair includes matching members of two disjoint sets to form $n/2$ pairs. □

It is well known that when $k = 2$, stable binary matching exists for any type of balanced bipartite graphs, as in the stable marriage problem.

What will happen if we allow certain disjointed sets in a $k$-partite graph to perform self-matching? For example, in the tripartite graph shown in Figure 1, nodes in part $U$ can be paired with nodes in $U$ itself, as well as with nodes in $W$ and $M$. The answer is negative as well, as in the case of $W = \{w, w'\}$, $M = \{m, m'\}$, and $U = \{u, u'\}$, where $w$, $w'$ $m$, $m'$, and $u$ are ranked as the top by $m$, $m'$, $u$, $w$, and $w'$, respectively. $u'$ being paired with anyone will be unstable; this is irrelevant to where the matched element comes from, inside or outside $U$. However, the stable roommate solution can still be used to determine a matching if one exists.

One potential application of the above result is in sociology. In a traditional society with two-parent families, stable marriage is ensured in a society two genders in an ideal environment. In an ideal environment, the population is static with no new comers. In addition, each person's preferences over others remain the same. The result of this subsection shows that in a society with multiple genders, stable marriage is not guaranteed. In next section, we will study a special formation of family with $k$-parent in a society with $k$ genders.

### *B. Searching for Stable and Fair Binary Marching in $k$-Partite Graphs*

Now we use the solution for the stable roommates problem to detect the existence of stable binary matching in $k$-partite graphs, and find one if one exists. This problem can be considered as a special case of the stable roommates problem [6] with incomplete preference lists (i.e., a person can exclude some members) and with some minor twists. The solution consists of two phases.

In phase one, each participant proposes to all the others based on the preference list, and rejects a proposal if he holds a proposal from someone he prefers. The proposal is unidirectional (i.e., a person can hold a proposal from another person, yet can make his own proposal to the third person). Once a proposal is rejected, it is bidirectional for both parties. That is, if $w$ removes $m$ from her list, it also means $m$ removes $w$ from his list. A *pruning process* is applied to remove all the lowly ranked nodes. For example, if $m$ receives a proposal from $w$, he will remove all persons, $u$, ranked lower than $w$. In addition, $m$ will be removed from $u$'s preference list based on the bidirectional removal rule. The resulting reduced set of preference lists is called a *reduced list*. If a reduced list is empty, then there is no stable matching; if all reduced lists have exactly one element, it has a stable matching; otherwise, go to phase two for loop removal.

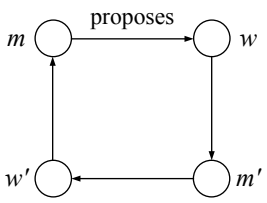


Fig. 2. A deadlock situation within the circular proposal involving $m$, $w$, $m'$, and $w'$.

In phase two, we try to find a loop of alternating first and second preferences among reduced lists. Each participant involved in the loop will reject his first preference and goes with his second preference. The pruning process is applied again to further reduce the reduced set. The above process is repeated until no such loop exists. Again, if one reduced list is empty, there is no stable matching; otherwise, all reduced lists will have one member left, which corresponds to a stable matching.

Suppose we have the following two cases, left hand side and right hand side, of preference lists for Example 2. The list after ":" corresponds to the preference list for members before ":".

| | |
|---|---|
| $m : u'ww'u$ | $m : w'u'uw$ |
| $m'$: $u'wuw'$ | $m'$: $w'wuu'$ |
| $w$ : $mm'u'u$ | $w$ : $m'muu'$ |
| $w'$ : $m'muu'$ | $w'$ : $mm'uu'$ |
| $u$ : $mm'w'w$ | $u$ : $mm'ww'$ |
| $u'$ : $mww'm'$ | $u'$ : $mw'wm'$ |

In the left hand side example above, suppose the requests follow the sequence below:

| | | |
|---|---|---|
| $w \to m$ | $m$ holds $w$ | removes $m : w'u, w' : m, u : m$ |
| $m \to u'$ | $u'$ holds $m$ | removes $u' : ww'm', w : u',$ $w' : u'$, $m' : u'$ |
| $u' \to m$ | $m$ holds $u'$ | rejects $w$; removes $m : w, w : m$ |
| $w \to m'$ | $m'$ holds $w$ | removes $m' : uw', u : m', w' : m'$ |
| $w' \to u$ | $u$ holds $w'$ | removes $u : w, w : u$ |
| $u \to w'$ | $w'$ holds $u$ | |
| $m' \to w$ | $w$ holds $m'$ | |

In the above,$w \to m$ represents a proposal from $w$ to $m$. $m : uw'$ represents removing $u$ and $w'$ from $m$'s list. Note that once $m : u$ and $m : w'$ are removed, $u : m$ and $w' : m$ are removed as well, based on the bidirectional removal rule. Eventually, each reduced list includes one element. The final matching is $(m, u')$, $(m', w)$, and $(w', u)$.

In the right hand side example above, we consider the following case with a focus on actions related to $u$. $m' \to w'$ (removes $w' : u$ and $u : w'$), $m \to w'$ (rejects $m'$), $m' \to w$ (remove $w : u$ and $u : w$), $w' \to m$ (removes $m : u$ and $u : m$), $u \to m'$ ($m'$ holds $u$), and $w \to m'$ ($m'$ rejects $u$, remove $m' : u$ and $u : m'$). $u$'s reduced list is empty. Therefore, there is no stable matching.

Next, we use the stable roommate solution to solve the stable marriage problem to address the gender unfairness issue in the original GS algorithm. The difference is that both men

and women can propose at the same time. After phase one, we have the following reduced sets, representing a circular waiting situation: $m \to w$, $w \to m'$, $m' \to w'$, and $w' \to m$:

$$\begin{aligned} m &: ww' \\ m' &: w'w \\ w &: m'm \\ w' &: mm' \end{aligned}$$

In the above example, each participant goes after his/her first choice. These four participants become involved in a "deadlock" after phase one, as shown in Figure 2. In phase two, one of the two loops needs to be broken in the preference lists. For example, $m$ and $m'$ are involved in a loop with $w$ and $w'$ as their alternating first and second choices. Both $m$ and $m'$ reject $w$ and $w'$, and they accept their second choices, respectively, to form a woman-optimal stable matching: $(m, w')$ and $(m', w)$. If we remove the loop involving $w$ and $w'$, we have a man-optimal stable matching, $(m, w)$ and $(m', w')$. By alternating man-oriented and woman-oriented loop breaking in phase two, we can obtain a procedural fairness among men and women.

## IV. Stable $k$-ary Matching in $k$-Partite Graphs

In this section, we extend the traditional binary matching in a bipartite graph to matching in multipartite graphs. We first show the existence of stable $k$-ary matching in $k$-partite graphs and propose a matching process through iterative binding, followed by performance analysis and an parallel implementation. The section ends with an extension that relax the unstable condition to derive a stronger result.

### A. Existence of Stable $k$-ary Matching in $k$-Partite Graphs

We first denote a $k$-ary matching in balanced $k$-partite graphs: $(u_1, u_2, ..., u_k)$, where $u_i \in G_i$ corresponds to a member in the $i$-th disjoint set. We assume that $|G_i| = n$ for all $i$, as it is balanced. $G$ is the union of $G_i$, i.e., a set of all members. In the subsequent discussion, a balanced $k$-partite graph is simply called a $k$-partite graph.

Let $I = \{1, 2, ..., k\}$ be the gender set for disjoint sets in $k$-partite graphs. A blocking family comes from $k'$ ($2 \le k' \le k$) existing families in a matching. In a *blocking family*, members from the same existing family form a same-family group. It is required that each member of the same-family group agrees on the decision, i.e., each prefers new members from other groups (i.e., from different families) than its existing ones, but there is no need to compare members from the same-family group.

Our matching solution is based on extending the Gale-Shapley (GS) algorithm to $k$-partite graphs through iterative applications of binary matching to pairwisely "bind" all disjoint sets through a spanning tree. Specifically, we denote one application of the GS algorithm on $i$ and $j$ in $I$, GS$(i, j)$, for a binary binding. Relation $i - j$ indicates matching (binding) of $G_i$ and $G_j$. This approach is applied $k - 1$ times on a pair of genders in $I$ in such a way that a spanning binding tree $T$ is eventually constructed on $I$. In Algorithm 1, $V(T)$ and $E(T)$ denote the node set and edge set of $T$, respectively.

As shown in Algorithm 1, we define an equivalence relation $(-, -)$ "in the same matching tuple" based on the matching pairs $P$ to convert binary binding to $k$-ary binding (i.e. $k$-tuples). Such a relation satisfies (1) reflexivity, $(u, u)$ is clearly in the same matching tuple; (2) symmetry, if $(u, v)$, then $(v, u)$; and (3) transitivity, if $(u, v)$ and $(v, w)$, then $(u, w)$. Finally, equivalence classes are derived, which corresponds to all matching $k$-tuples. In each class, it is a set of nodes that are assigned to the same matching tuple based on the equivalence relation.

**Algorithm 1** Iterative Binding GS Algorithm

/* $I$ is a gender set with $|I| = k$ */;
$T$ (binding tree) and $P$ (matching pairs) are empty;
**while** $T$ is not a spanning tree on $I$ **do**
  Find $i$, $j \in I$: $(i, j)$ does not cause a cycle in $T$;
  $V(T) = V(T) \cup \{i, j\}$; $E(T) = E(T) \cup \{(i, j)\}$;
  $P = P \cup \text{GS}(i, j)$;
Derive $E$, equivalence classes from equivalence relation $(-, -)$ "in the same matching tuple" on $P$;
**return** $E$ (matching $k$-tuples)

| | $m$ | $m'$ | $w$ | $w'$ | $u$ | $u'$ |
|---|---|---|---|---|---|---|
| $m$ | - | - | 1 | 2 | 2 | 1 |
| $m'$ | - | - | 2 | 1 | 1 | 2 |
| $w$ | 1 | 2 | - | - | 1 | 2 |
| $w'$ | 2 | 1 | - | - | 2 | 1 |
| $u$ | 1 | 2 | 1 | 2 | - | - |
| $u'$ | 1 | 2 | 2 | 1 | - | - |

Fig. 3. Preference lists of nodes (in the left column) in the balanced tripartite graph of Figure 1.

Let us consider the ternary graph example with three disjoint sets: $M = \{m, m'\}$, $W = \{w, w'\}$, and $U = \{u, u'\}$. Each node's preference list is given in Figure 3. Entry "1" corresponds to a higher rank than entry "2", since each gender has two members, only two ranks are used[4]. For example, both $u$ and $u'$ rank $m$ higher than $m'$, although $m$ ranks $u'$ higher and $m'$ ranks $u$ higher. Assume that the binding process is $M - W$ and $W - U$. The former binds $m$ with $w$ (and $m'$ with $w'$), and the latter binds $w$ with $u$ (and $w'$ and $u'$) to form ternary matchings $(m, w, u)$ and $(m', w', u')$ based on the derived equivalence classes.

The following theorem shows that the proposed binding algorithm constructs a stable $k$-ary matching.

***Theorem** 2:* The iterative binding GS algorithm constructs a stable $k$-ary matching.

*Proof.* We first prove that Algorithm 1 generates a perfect matching, i.e., each node appears in a $k$-tuple once and only once. This is because an application of the GS algorithm on genders $i$ and $j$ forms a perfect matching between $G_i$ and $G_j$. That is, nodes in $i$ and $j$ are paired, one to each of the

[4]Note that these total orders form a global partial order which can be converted into a global total order in various ways

tuples under the equivalence relation $(-,-)$ "the same $k$-ary matching tuple". Spanning binding tree $T$ is constructed in such a way that each link corresponds to a binding between genders $i$ and $j$. $P$ clearly includes all nodes in $G$. The equivalence relation partitions $G$ into $n$ classes, in which each equivalence class has exactly $k$ elements, one from each disjoint set $G_i$.

Next, we prove that the $k$-ary matching is stable. Suppose there exists a blocking family, then genders can be partitioned into several same-family groups based on the current $k$-ary matching. As spanning binding tree $T$ connects all genders, there exists two connected same-family groups, $S$ and $S'$, and there are two genders $i$ and $j$, one each from $S$ and $S'$, such that $(i,j)$ is an edge in the spanning binding tree $T$. Based on the extended stable condition, all elements in $S$ of one matching and all elements in $S'$ agree on a new matching (to form a blocking family). That is, such an instability would involve two $k$-tuple: $(...u_i...u_j...)$ and $(...u_i^{'}...u_j^{'}...)$, with the properties that (1) $u_i$ ranks $u_j^{'}$ higher than $u_j$, and (2) $u_j^{'}$ ranks $u_i$ higher than $u_i^{'}$. These properties imply that $u_i$ and $u_j^{'}$ form a blocking pair for a matching $GS(i,j)$ between $G_i$ and $G_j$. This contradicts the fact that $GS$ algorithm produces a binary stable matching between $G_i$ and $G_j$. □

Consider a futuristic family with $k$-parent, one from each of the $k$ different genders in a society with $k$ genders. The above result shows that stable $k$-ary marriage is guaranteed.

### B. Algorithm Analysis

Although the iterative binding GS algorithm will produce a stable $k$-ary matching, different bindings may generate different stable $k$-ary matchings. For example, bindings $M-U$ and $U-W$ will generate a stable matching of $(m,w',u')$ and $(m',w,u)$, while bindings $M-U$ and $M-W$ will generate a stable matching of $(m,w,u')$ and $(m',w',u)$. Based on Cayley's formula [8], there are $k^{k-2}$ different binding trees to bind $k$ genders in $k$-partite graphs.

We have the following results on time complexity for the iterative binding process. The theorem is obvious from the $k-1$ rounds of applications of the GS algorithm, assuming each matching process corresponds a "proposer" (a man in the G-S algorithm) to a "responder" (a woman).

***Theorem** 3:* The iterative binding GS algorithm takes at most $(k-1)n^2$ iterations of the matching process.

Next we show that $k-1$ rounds of the binding process is tight. More rounds than $k-1$ may not be possible for stable binary matchings, and fewer rounds than $k-1$ will cause instability. Note that more binary bindings will strengthen the family tie. However, when there are $k$ or more binary bindings, there must exist a loop. For example, if a loop occurs in a ternary matching among genders $M$, $W$, and $U$, suppose we have the following preference orders: $m$ prefers $w$ overs $w^{'}$ (simply denoted as $m$: $w$, without causing confusion), $m^{'}$ : $w$, $w$: $m$, $w^{'}$ : $m^{'}$, $w$ : $u$, $w^{'}$ : $u$, $u$ : $w$, $u^{'}$ : $w^{'}$, $m$ : $u$, $m^{'}$ : $u$, $u$ : $m^{'}$, $u^{'}$ : $m^{'}$, it is impossible to perform three binary bindings and maintain their stability. This example can be easily extended to the general $k$-ary case. On the other hand, if $k-2$ or fewer binary bindings are applied, the gender set is partitioned into several groups that correspond to components in a graph. It is easy to show that any matching that involves components without any binding will cause instability by assigning appropriate preference orders among members from different components.

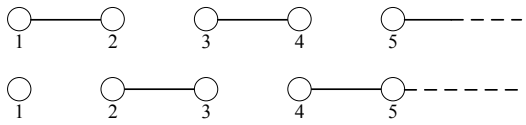


Fig. 4. An even-odd matching process.

***Theorem** 4:* The $k-1$ rounds of the binding process in Algorithm 1 is tight for $k$-ary stable matching.

### C. Parallel Implementation

In this subsection, we will consider a construction of binding tree $T$ to speed up the matching process. It is well known that pairwise matching in the original GS algorithm is difficult to parallelize. Indeed, no parallel algorithm is known for the stable marriage problem with better than $O(n^2)$ worst-case complexity [9]. However, parallelization at the binding tree level is feasible. Assume that $\Delta$ is the maximum node degree of $T$ in the binding process. We call one round a binary matching using the GS algorithm. Under the EREW PRAM[5] model using $k-1$ processors, each gender can be involved in a binary matching, one at a time. We have the following result, which states that the bottleneck is the gender with the maximum node degree in the binding tree $T$.

***Corollary** 1:* Using the EREW PRAM model for parallel processing, the iterative binding GS algorithm takes at most $\Delta n^2$ iterations of the matching process.

The tree with minimum $\Delta$ is a linear line, with $\Delta = 2$. Through parallel even-odd pairings as shown in Figure 4, we only need to apply the GS algorithm twice to form a stable $k$-ary matching. That is, in the first round, all even-labeled genders: $2i$, $i = 1,2,3,...$ are paired with genders: $2i-1$, i.e., odd-labeled genders to their left. In the second round, all even-labeled genders are paired with genders: $2i+1$, i.e., odd-labeled genders to their right.

***Corollary** 2:* Using a linear line as $T$, the iterative binding GS algorithm takes two rounds of the matching process.

Suppose in a round, each node can take the role of "men" (proposers) once (i.e., pairing with another gender in the role of men) and the role of "women" (responders) once. After one round of data duplication, with one copy for proposers and the other for responders, the iterative binding would take one round to complete. Under the CREW PRAM[6] model, as each gender can be read in parallel, a node can be bind through binary binding simultaneously with other genders. With $\log\Delta$ rounds of data replication, where $\Delta$ is the maximum node

[5]EREW stands for exclusive read and exclusive write. PRAM is parallel random-access machine.

[6]CREW stands for current read and exclusive write.

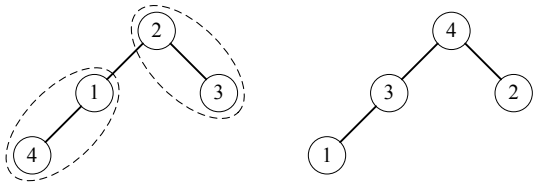


Fig. 5. (a) An unstable binding tree, (b) stable binding tree.

degree of $T$, EREW PRAM can emulate CREM PRAM as each of $\Delta$ copies through $\Delta$ rounds of replication can be read simultaneously.

### *D. Extension of Unstable Condition*

In this subsection, we relax the unstable condition, as it is right now too strong to break a matching. To enhance our result, we weaken the unstable condition. We assume that there is a strict priority order among $k$ genders. Note that members of a blocking family must come from $k'$ ($2 \leq k' \leq k$) existing families in a matching. Therefore, there are $k'$ *lead members* based on the given gender priority, one from each of these $k'$ families. In the weakened blocking family, the condition "each member" is replaced by the condition "lead member of the corresponding families", which makes $k$-ary stable matching harder.

Without loss of generality, we use a numerical number to represent both the identify of a member and the priority of a member's gender. The higher the number, the higher its priority. Suppose a blocking family in a 4-ary matching comes from two families, one family has members from genders 1 and 4 and the other family has members from genders 2 and 3. In the weakened blocking family, we only require that members from lead genders 3 and 4 prefer other members over the existing match. However, a binding tree can no longer guarantee a $k$-ary stable matching. As shown in the example of Figure 5 (a), an unstable situation can occur if a family member in gender 4 likes members of genders 2 and 3 from another family, and a family member in gender 3 likes members of genders 1 and 4 from another family. This is because the binary binding in the binding tree does not bind 3 and 4 directly to members outside their own families. The question is how to construct such a binding tree, so that at least one lead member is bounded to a member outside its family to ensure stability. Figure 5 (b) shows another binding for the same example that ensures stability.

We now propose a priority-aware iterative binding GS algorithm as shown in Algorithm 2, where $V(T)$ is initialized to a set with one element $i_{max}$, the highest priority gender. A sequence is bitonic if it monotonically increases and then monotonically decreases. The increase or decrease phase may or may not exist. For example, the sequences (1, 3, 4, 2), (4, 3, 2, 1), and (1, 2, 3, 4) are bitonic, but (4, 1, 2, 3) is not bitonic. Given a tree in which each node has a distinct priority ID, it is called *bitonic tree* if any two nodes in the tree is connected through a path that is a bitonic sequence. The bitonic tree can

**Algorithm 2** Priority-Based Iterative Binding GS Algorithm

/* $I$ is a gender set with $|I| = k$ and $i_{max}$ is the highest priority gender */;
$V(T) = \{i_{max}\}$, $E(T) = P = \{\}$, and $I = I - \{i_{max}\}$;
**while** $I$ is not empty **do**
  Select an $i$ in $V(T)$ and $j$ in $I$ with the highest priority;
  $V(T) = V(T) \cup \{j\}$; $E(T) = E(T) \cup \{(i, j)\}$;
  $I = I - \{j\}$;
  $P = P \cup \text{GS}(i, j)$;
Derive $E$, equivalence classes from equivalence relation $(-, -)$ "in the same matching tuple" on $P$;
**return** $E$ (matching $k$-tuples);

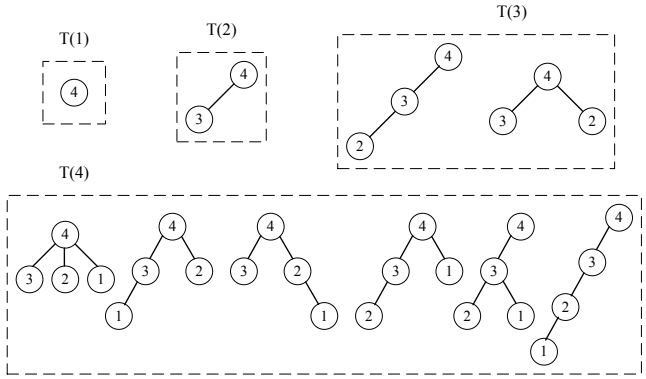


Fig. 6. A construction of growing binding tree in a 4-ary matching.

be generated through growing a tree following a decreasing priority order, as shown in the priority-aware iterative binding GS algorithm.

By the way of construction, we can easily show that the resultant tree is bitonic. The following theorem shows that if a binding tree is bitonic, it will prevent an instability.

***Theorem** 5:* If a binding tree is bitonic, any weakened blocking family will be prevented in $k$-ary matching.

*Proof*: We prove by contradiction. Suppose a weakened blocking family exists; based on definition, this family must come from at least two existing families. Assume $i$ and $j$ are two lead genders from two existing families. Based on the definition of bitonic sequence and bitonic tree, $i$ and $j$ are connected through a bitonic sequence path in the binding tree. Either $i$ or $j$ is directly connected to a higher priority node $k$ (which can be $i$, $j$ or another node) that does not belong to the same family. That is, either $(i, k)$ or $(j, k)$ forms a blocking pair to bring a contradiction similar to the proof of Theorem 2. □

Figure 6 shows a construction of growing a binding tree in a 4-ary matching. Suppose $|T(k)|$ corresponds to the number of priority-based binding trees in a $k$-ary matching. We can easily show that $T(k) = (k-1)T(k-1)$, $T(2) = T(1) = 1$. So $T(k) = (k-1)!$ When $k = 4$, there are 3! = 6 distinct priority-based binding trees. This is because $T(k)$ is constructed by attaching one new node as a neighbor to each node in $T(k-1)$.

## V. Related Work

We start with an overview of related work of stable marriage problem (SMP), with a focus on multi-dimensional extensions. We then discuss matching in the animal world.

### A. Related Work of SMP

In three-dimensional or higher dimensional SMP extensions, in addition to combination or cyclic preferences among genders, Huang [5] considered a preference model in which each gender is rated individually, while group preference is determined by the sum of the individual members in the group. Huang also studied another preference model where indifference is allowed (i.e., a tie situation is allowed). Four variations are considered: weak, strong, super, and altra stable matchings. He showed that all the above cases are NP-complete. A network application of three-dimensional stable matching was discussed in [10]. In [11], an extended stable roommate problem is studied by placing three persons in each room; however, there is only one gender. The problem is showed to be NP-complete.

Many other extensions and applications of the SMP exist. For example, the hospitals/residents problem [12], also known as the college admission problem, is such an extension and application where a hospital (college) can take multiple residents (students), and in this example, a hospital (college) can accept multiple applicants. This problem has been further extended by including an additional constraint in which couples must be assigned together. However, the extended problem of determining whether there is a stable solution and finding it, if it exists, has been proven to be NP-complete [13]. Other extensions of the SMP can be found in [3, 7, 9]. A survey of recent work can be found in [14, 15].

### B. Matching in the Animal World

Many human societies have multiple genders. More recently, UK parliament has approved the creation of babies with DNA from two women and one man, in an historic move for "three-person babies" [16]. Gender is a cultural construction of societal roles. Sex is the biological manifestation of reproduction. Here, we use gender to represent both gender and sex. Binary matching is very common in the animal world. Asexual reproduction [17] is a mode of reproduction by which offspring arises from a single organism, and inherits the genes of that parent only. However, a complete lack of sexual reproduction is relatively rare among multicellular organisms, especially in the animal world.

A majority of animals are sexually dimorphic [18] between a male and a female (i.e., fits for a binary matching in a bipartite graph), and not between two hermaphrodites (i.e., fits for a binary matching in a complete graph). Hermaphrodites have both reproductive parts. Some lower animals, such as worms, are hermaphrodites. $k$-ary matching does occur in the animal world for reproduction; certain harvester ants have three genders, or even four. These ants live in colonies, each of which has a queen. For the ants to be fruitful and multiply, she needs to mate with two different strains of male for future queens and future workers.

## VI. Conclusions

In this paper, we extended the classic stable matching problem in a balanced $k$-partite graph. We showed that there always exists a set of preference lists, under which stable binary matching does not guarantee for any $k$, except when $k = 2$. Under a natural extension of pairwise stability for binary matching to $k$-ary matching, we proved that stable $k$-ary matching exists for any set of preferences in any balanced $k$-partite graphs. An extended Gale-Shapley (GS) algorithm is introduced to find such a stable $k$-ary matching.

Our future direction includes examining other possible weakened blocking families to find stable $k$-ary matching in a $k$-partite graph. One possibility is to explore quorum-based approaches to relax unstable conditions used in the extended stable matching. In addition to seeking the possibility of binary matching in a specific $k$-partite group, we plan to study a more general $k$-ary matching in $k'$-partite graphs, where $k < k'$ and $ck = nk'$ for some constant $c$.


## Acknowledgement

This work is supported in part by NSF grants CNS 149860, CNS 1461932, CNS 1460971, CNS 1439672, CNS 1301774, ECCS 1231461, ECCS 1128209, and CNS 1138963. Special thanks are given to classmates of the WeChat group from Shanghai Guling No.1 Elementary School. The discussion on match making in the group inspires this work.


## References


[1] M. Mucha and P. Sankowski, "Maximum matchings via gaussian elimination," in *Proceedings of FOCS 2004*, pp. 248–255.

[2] D. Gale and L. S. Shapley, "College admissions and the stability of marriage," *American Mathematical Monthly*, pp. 9–15, 1962.

[3] D. E. Knuth, *Stable marriage and its relation to other combinatorial problems: An introduction to the mathematical analysis of algorithms*. American Mathematical Society, 1997, vol. 10.

[4] C. Ng and D. S. Hirschberg, "Three-dimensional stable matching problems," *SIAM Journal on Discrete Mathematics*, vol. 4, no. 2, pp. 245–252, 1991.

[5] C.-C. Huang, "Two's company, three's a crowd: Stable family and threesome roommates problems," in *Proceedings of ESA 2007*, pp. 558–569.

[6] R. W. Irving, "An efficient algorithm for the stable roommates problem," *Journal of Algorithms*, vol. 6, no. 4, pp. 577–595, 1985.

[7] J. Kleinberg and É. Tardos, *Algorithm design*. Pearson Education India, 2006.

[8] A. Cayley, "A theorem on trees," *Quart. J. Math*, vol. 23, pp. 376–378, 1889.

[9] D. Gusfield and R. W. Irving, *The stable marriage problem: structure and algorithms*. MIT press Cambridge, 1989.

[10] L. Cui and W. Jia, "Cyclic stable matching for three-sided networking services," *Computer Networks*, vol. 57, no. 1, pp. 351–363, 2013.

[11] K. Iwama, S. Miyazaki, and K. Okamoto, "Stable roomates problem with triple rooms," in *Proceedings of WAAC 2007*, pp. 105–112.

[12] D. Gale, "The two-sided matching problem: origin, development and current issues," *International Game Theory Review*, vol. 3, no. 02-03, pp. 237–252, 2001.

[13] E. Ronn, "NP-complete stable matching problems," *Journal of Algorithms*, vol. 11, no. 2, pp. 285–304, 1990.
[14] K. Iwana and S. Miyazaki, "A survey of the stable marriage problem and its variants," *International Conference on Informatics Education and Research for Knowledge*, pp. 131–136, 2008.
[15] D. F. Manlove *et al.*, "Algorithmics of matching under preferences," *Bulletin of EATCS*, vol. 1, no. 112, 2014.
[16] http://www.bbc.com/news/health-31069173.
[17] K. J. Dawson, "The advantage of asexual reproduction: When is it two-fold?" *Journal of theoretical biology*, vol. 176, no. 3, pp. 341–347, 1995.
[18] http://en.wikipedia.org/wiki/Sexual%5Fdimorphism.