2023 IEEE Symposium Series on Computational Intelligence (SSCI)
Mexico City, Mexico. December 5-8, 2023

Balancing Matching of Two-Sided Agents with Adaptive and Fair Instability

Peash Ranjan Saha, Salimur Choudhury, and Kai Salomaa
School of Computing, Queen's University, Kingston, Ontario, Canada, K7L3N6
Email: p.saha@queensu.ca, s.choudhury@queensu.ca, ksalomaa@cs.queensu.ca

***Abstract*—The concept of stable matching is substantially used in bipartite graphs with individual preferences of the vertices. The existence of stability restricts the weight and size of the matching to be satisfactory. We study the trade-offs in stability, weight and cardinality in a one-to-many capacitated weighted bipartite matching with an edge-weight-oriented preference setting. We establish a stability relaxation framework which is adaptive to the pairing suitability and capacity of the vertices. The purpose of the relaxation is to update the stable matching towards the balance of stability, weight and cardinality in the result. The relaxation preserves fairness by keeping the satisfaction degradation of the vertices with the potential new partner in a desired range. We propose an algorithm to produce a new matching using the stability relaxation framework. Furthermore, we define a novel popularity measurement model of matching based on the edge weight with the multi-voting ability of one-sided vertices. We show the resulting matching is also popular as stable matching. The experimentation performed based on the use case of the homeless placement system complements the claim of improving the weight and cardinality in the matching with marginal and fair relaxation of stability.**

***Index Terms*—adaptive instability, balanced matching, popularity, fairness**

I. Introduction

Consider a bipartite graph $G = (U, V, E)$, where U and V are the two disjoint sets of vertices connected by the set of edges E. A matching M is the subset of E where two endpoints of all edges are distinct. Assume all the vertices of G have preferences over the other set. A matching S is stable if no vertex pair prefers each other over their current placement in S. Such a pair preferring each other is called blocking or unstable. Thus, a stable matching has no unstable pair. The stable matching is first defined in the stable marriage problem introduced by Gale and Shapley [1]. Later on, it was extended for the one-to-many matching problem such as the college admission problem [2]. In a one-to-many matching, multiple edges can share an endpoint on one side. There may have capacity on the number of edges incident on a vertex for that side. Stable matching has been much used in applications that require one-to-many matching of two-sided agents with preferences. The factors fuelling the preferences are crucial for those applications [3].

Let's consider any edge $e \in E$ has a numerical weight and G is redefined as (U, V, E, W) where W is the set of weights of the edges. Assume a vertex preference over the other set of vertices is determined using the decreasing order of the weight of the edges incident on it. We call the scenario the edge-weight-oriented preference *(EWP)* setting of G. There exist real-world problems where the balancing of the stability, weight, and cardinality is desired in a one-to-many bipartite matching with *(EWP)* setting. The homeless placement problem is a well-established example of such a scenario. The weight of an edge between a homeless person and a shelter represents the suitability between them. Here, suitability refers to the likeliness of a favourable outcome of the placement. The higher it is, the higher the quick recovery chances of the homeless person. The stability in the matching guarantees the preferences of the homeless persons and the shelters admit them are preserved. The total weight indicates the overall recovery capability of the solution and the cardinality refers to the number of homeless persons admits in the shelters. Optimizing all three factors are paramount in the resulting placement and a balanced matching is desired.

However, there exist trade-offs among these objectives in a matching. A maximum weighted bipartite matching *MWBM* is the matching of the maximum possible total weight. The *MWBM* might be promising in terms of cardinality but possibly have many unstable pairs. The weight of the stable matching in *(EWP)* setting can be significantly less than *MWBM*. A maximum cardinality bipartite matching *MCBM* is matching with the maximum possible number of edges. The number of edges in the stable matching can be $\frac{1}{2}$ of *MCBM* [4]. This is true in *(EWP)* setting as well. The total weight of the *MCBM* is not satisfying either. We discuss the trade-offs with proofs and examples in section III.

The strategy of generating an initial matching based on a single objective and updating it to balance the others can be implemented. One strategy can be sacrificing the weights to reduce the unstable pairs in a *MWBM*. This might balance stability up to an extent but is adversarial for the size of the matching since two unstable pairs may be replaced by one stable pair (shown in Fig. 1). Similarly, sacrificing the cardinality in a *MCBM* can end up balancing either weight or stability with adverse behaviour to the other. The strategy of relaxing the stability to increase the weight of the matching by replacing stable pairs with unstable ones is not adversarial for the cardinality. Because one stable pair is always replaced by at least one unstable pair, possibly two. Thus, we introduce a stability relaxation framework to bring the desired balance in stability, weight and cardinality in the matching. The framework determines the degree of instability one vertex can consume based on the weight of the edges incident on them and the capacity where applicable. Hence, it is both fair for all vertices and adaptive to the application.

We consider an incomplete bipartite graph for the research problem. No edge between $u \in U$ and $v \in V$ means they are not compatible for pairing. In a stable matching, some vertices from U can remain unmatched even when some vertices from V have empty spots due to pairing incompatibility and the limited capacity of other vertices in V. This can be considered as the capacity waste in G, which is keeping weight and cardinality down in the result. Our proposed algorithm reduces the capacity waste by updating the stable matching using the stability relaxation determined by the defined framework. This eventually brings the desired balance among the objectives.

Furthermore, a significant property of stable matching is its popularity. A matching M is popular if the number of vertices prefers M over any matching M' is not less than the number of vertices prefers M' over M. A vertex prefers M over M' if it is not matched in M', or prefers the partner of M than the partner of M' otherwise. The popular matching is first defined in [19]. There exists no matching popular than the stable matching [4]. However, multiple popular matching may exist. The relaxation of the stability allows generating another popular matching with a larger size if it exists [20]. This motivates us to relax the stability with the aim of improving both the cardinality and weight of the matching. The popularity of any matching is justified using the popularity measurement model. We introduce a novel popularity measurement model for the *(EWP)* setting. The model allows multiple voting of the vertices of V based on the weight of the edges incident on them. Our proposed algorithm is designed in a way such that the resulting matching is popular which can be justified by integrating the popularity measurement model.

The contribution summary of the research: we study the relationship between objectives in a one-to-many capacitated weighted bipartite matching. We show the trade-offs in optimizing stability, weight, and cardinality when the edge-weight is the only preference factor for agents. We propose a novel stability relaxation framework which is adaptive to the edge-weight and the capacity of the vertices. We propose an algorithm to update the stable matching using the fair degree of instability determined for each vertex using the stability relaxation framework. This brings the desired balance in stability, weight and cardinality in the resulting matching. We experiment with a use case of the homeless placement system. The results support the stability relaxation framework concept with the expected improvement of weight and cardinality over the stable matching. We also define a new popularity measurement model with the multi-voting ability of vertices based on the weight of the edges. We verify the resulting matching with the popularity measurement model and show it is also popular as stable matching.

We present the relevant research in the section II. We show the objectives trade-offs in the section III. The stability relaxation framework is presented in section IV. We present our algorithm in the section V. The popularity measurement model and the corresponding proof are shown in section VI. We show the experimental results in section VII. We conclude with the summary and future direction in section VIII.

II. Related Work

There is an increased concern for the problems of matching under preferences due to the rapid growth of such applications. The research on the algorithmic perspective of the matching under preferences also hiked in the recent past [5]. The utilization of the stable matching concept for the bipartite graph with two-sided preferences has long been established after its introduction in the stable marriage problem by Gale and Shapley [1]. One important aspect of the existing works on stable matching is the consideration the preference list variability such as incomplete preference list, preference list with ties etc [6]. We particularly reviewed the previous works with incomplete and strict preference lists for this research.

There exist conflicts between stability and the optimality of other criteria. The stable matching might not admit a fair matching [7]. The relaxation of stability gained attraction for modern applications that need to balance multiple objectives. The authors in [8] introduce a nearly stable matching notion using the mathematical programming for the dynamic ride-sharing system with preferences from both riders and drivers. They consider the weight optimization in the matching and adopted a strategy for a ride-sharing matching based on the cost and reducing the stability with a relaxed blocking pair condition afterwards. In [9], the authors introduced an approach of imposing a cost on the blocking edge for a weighted bipartite graph. They restricted the stability relaxation in terms of the cost of the blocking edges and showed a computationally tractable approach.

The concept of envy-free matching is introduced as a fairness condition [10]. A matching is envy-free if no agent wants to exchange the outcome with another. The authors in [11] showed an envy-free matching exists with both upper and lower quotas for the agents with relaxed stability. They set a limit of matched and unmatched edges to participate in the blocking pair as the stability relaxation. They showed no stable matching instance exists with the lower quota for agents and hence stability is always relaxed. A critical bipartite graph contains critical vertices. A vertex is critical if removing it constitutes a perfect matching [12]. The notion of relaxed stability restricting the edges to participate in the blocking pair [11] is also used in [13] to show a critical matching always exists in such a setting. Critical matching focuses on including edges incident on the critical vertices. The authors in [13] showed that stability and criticality can not co-exist and thus relaxed stability is used to compute a critical matching.

The authors in [14] showed that cardinality improves with the relaxation of the preference rule in a one-to-many stable matching for the taxi-sharing service. They introduced both fixed and variable discount strategies and guarantee the matching size improvement for the former one. The notion of popularity is introduced to balance the cardinality in a stable matching [4]. The relaxation of stability towards other popular matching possibly increases the size of the matching. However, popular matching does not necessarily be optimal in size. The authors in [15] proposed a strategy to compute a maximum cardinality popular matching in the stable marriage setting. However, they have not considered edge weight. The authors in [16] consider computing a maximum cardinality popular matching in a weighted capacitated house allocation problem. They utilize the weight of the agents as an incentive for priority in the preference list. The total weight of such maximum cardinality popular matching is not promising in most cases. There exists no work considering the relaxation of stability with the combined goal of balancing stability, weight and cardinality. We thus introduce a stability relaxation framework equally prioritizing all three criteria and propose an algorithm to produce a balanced matching.

Furthermore, the authors in [21]–[23] introduced the popularity measurement model for one-to-many matching. However, they have not considered the weight of the edges in the voting model. Instead, their model counts the votes based on the differences in the number of edges incident on the vertices of V in two matchings [21]. Hence, we integrate a novel popularity measurement model that provides a multi-voting ability of the vertices of V based on edge weight with our proposed algorithm for the justification of the output.

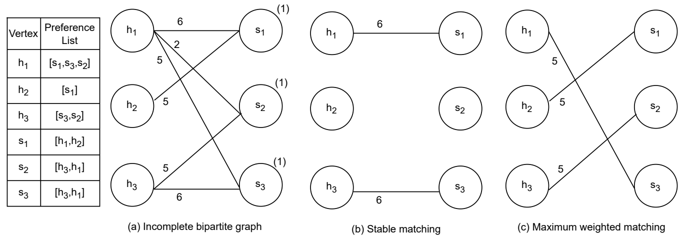
III. Objectives Trade-Offs

A vertex have a strict preference list in the problem we considered in the research. No two vertex have the same preference in a strict preference list. The preference list of a vertex may be incomplete that is excluding some vertices from the list. Usually, an explicit ranking of the agents is maintained in a placement process such as the shelter ranking in the homeless placement system. We assume such ranking is used to break the tie in the preference list when more than one edge has the same weight. A vertex $u \in U$ has a strict preference list over the vertices of V based on the decreasing order of the weight of the edges incident on u. Similarly, a strict preference list for a vertex $v \in V$ is constructed.

The authors in [4] showed that the size of the stable matching can be half of *MCBM*. This clarifies the stability-cardinality trade-off. We show that the total weight of a stable matching can be half of the total weight of *MWBM*.

Theorem 1: The total weight of stable matching is $\frac{1}{2}$ approximation of the maximum weighted bipartite matching.
Proof: Let's assume M_s is the stable matching with size $|M|$. The total weight of M_s is not the maximum possible. Consider M_w as the *MWBM*. There is no edge in M_w which connects two unmatched vertices with respect to M_s. Because such an edge would have been added in M_s. A vertex is unmatched with respect to a matching if no edge from the matching is incident to it. That indicates any edge of M_w either belongs from M_s or shares an endpoint with any one or two edges of M_s. Let's consider the set of edges of M_w from the latter case as N. The weight of any edge of N must not be higher than at least one of the edges from M_s with whom it shares the endpoint, otherwise, such an edge would have constructed an unstable pair. On the other hand, there may exist two edges of N from the two endpoints of an edge of $e \in M_s$ and they both can have the same weight as e. Thus, the weight of the edges in N can be twice the weight of the edges of M_s with whom they share an endpoint. If there is no common edge between M_w and M_s, then the total weight of M_w can be twice the total weight of M_s, which proves the bound.

There might exist unstable pairs in *MWBM*. In worst case, *MWBM* of size n may have $(n-1)$ unstable pairs as shown in Fig. 1. The capacity of every right-hand side vertices is 1 as shown with each vertex in Fig. 1 (a). The stable matching for (a) is shown in (b). The pairs $(h_1, s_1), (h_3, s_3)$ are unstable with respect to *MWBM* shown in (c).



Vertex	Preference List
h_1	$[s_1, s_3, s_2]$
h_2	$[s_1]$
h_3	$[s_3, s_2]$
s_1	$[h_1, h_2]$
s_2	$[h_3, h_1]$
s_3	$[h_3, h_1]$

Fig. 1: Instability in Maximum Weighted Matching

On the other hand, the transformation of the *MWBM* to the *MCBM* is subject to the loss of weights [18]. However, the preferences of agents are not being considered in *MWBM*. Hence, there exists adversariality for the weight and stability in *MCBM*. A matching balancing stability, weight, and cardinality up to an extent are necessary. In the next section, we present our stability relaxation technique which improves the weight and cardinality with fair instability for every vertex without decreasing the popularity of the stable matching.

IV. Stability Relaxation Framework

The partner (or partners) of a matched vertex in a one-to-many stable matching is the most preferred possible in the matching. Stability relaxation is the natural dilution of the preference of the partner (or partners) of a vertex over to stable matching. This allows reducing the capacity waste whenever possible by pairing the vertices with the less preferred partners which eventually improves the cardinality and weight over to the stable matching. However, arbitrary relaxation can make the matching unfair for some vertices. We thus need to use the information associated with the vertices to make the strategy adaptive and fair.

The suitability value of a pair of vertices $(u \in U, v \in V)$ is denoted by the weight of the edge between them. Recall that some pair of vertices $(u \in U, v \in V)$ may not be connected, hence they are not suitable for pairing. In a real scenario, some vertices might be suitable with few vertices.The suitability value of such vertices with their peer is most likely to be higher. On the other hand, some vertices are suitable to make pair with many vertices with a similar suitability value. We define a compatibility level for the vertices of U based on the number of vertices they can be paired with and their suitability value with them. First, we determine a compatibility value of a vertex $u \in U$, which is the summation of the number of vertices from V with whom u is suitable and the average suitability value of u. The number of vertex u is suitable with is defined by n_u and the average suitability s_u is the mean suitability value of u over n_u vertices. The compatibility value of u is defined as c_u and its equation is shown in 1.

$$c_u = n_u + s_u \tag{1}$$

A list of vertices sorted in the increasing order of compatibility values is constructed. The vertices are then classified into different compatibility levels using the sorted list. The number of compatibility levels is assumed to be predefined for the application. Let's assume there are n compatibility levels. The vertices are then divided into n groups based on their position in the sorted list. The number of vertices that falls in each group may not be the same. In the first $(n-1)$ group, it is $\lfloor \frac{(|U|)}{n} + 0.5 \rfloor$. The vertices in the first $\lfloor \frac{(|U|)}{n} + 0.5 \rfloor$ position in the sorted list are added to the first group. The vertices in the next $\lfloor \frac{(|U|)}{n} + 0.5 \rfloor$ position are added in the second group and so on. The remaining vertices after the first $(n-1)$ group formation falls into the n^{th} group.

The compatibility level of the vertices falls into the first group is defined as 1. In general, the compatibility level of the vertices belonging to n^{th} group is n. The compatibility level is the indicator of maximum instability permitted for a vertex to keep the solution fair for the vertex. The larger the compatibility level is, the more relaxation is possible for a vertex. The vertices are able to pair with many vertices considered to have more compatibility. Thus, n_u is considered as a primary factor to calculate c_u as shown in equation 1 to force the vertices with more possible pairs to fall into the higher compatibility level. The vertices highly suitable with fewer vertices would have fallen into the higher compatibility level if only s_u would have been used to calculate c_u. This approach works particularly well for settings where the range of suitability values is not so large.

Similarly, the compatibility level for the vertices of V is determined. However, the average suitability value s_v of $v \in V$ is determined by the mean over the capacity ca_v of v. The minimum value between ca_v and n_v, the number of vertices v is suitable with is used as one of the primary factors in this case as shown in equation 2. Note that, we assume every vertex in the graph can pair with at least one vertex, thus the compatibility value of any vertex is at least 1.

$$c_v = min(ca_v, n_v) + s_v \tag{2}$$

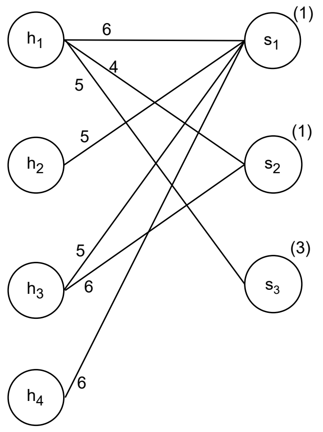
We define a vertex $u \in U$ as first-degree unstable in a matching S' if u's partner is S' is the next preferred than u's partner in the stable matching S. Thus, a vertex u is called α-degree unstable in S' when the preference of its partner in S' is α behind its partner in S. The vertex $v \in V$ is called α-degree unstable in S' when the preference of its least-preferred partner in S' is α behind its least-preferred partner in S.

The stability relaxation framework calculates the degree of instability for vertices based on their compatibility level. A predefined degree of instability for the first compatibility level vertices is considered. This value represents the best case. For example, the first degree of instability for the vertices with the compatibility level 1 means their partner or least-preferred partner in the resulting matching is the next-preferred than their partner or least-preferred partner in the stable matching. We can use the value as a seed to determine the degree of instability for the vertices of other compatibility levels. Let's assume the degree of instability for the vertices with the compatibility level 1 is d_1. Then the relation $d_n > d_{n-1} > .... > d_2 > d_1$ holds where the degree of instability increases by a constant I_n for each level as shown in equation 3.

$$d_n = d_{n-1} + I_n \tag{3}$$

The degree of instability is highly adaptive to the pairing suitability of the vertices and their capacity when applicable. It also reduces the chances of degrading the stability unfairly for any vertex. Our proposed algorithm is presented in the next section which utilizes the degree of instability determined for each vertex in G.

Vertex	Compatibility Value
h_1	15/3 + 3 = 8
h_2	5+1 = 6
h_3	11/2 + 2 = 7.5
h_4	6+1 = 7
s_1	22 + 1 = 23
s_2	10 + 1 = 11
s_3	5+1 = 6



Vertex	Degree of Instability
h_1	2
h_2	0
h_3	2
h_4	1
s_1	2
s_2	1
s_3	0

Fig. 2: Example of Degree of Instability Calculation

One example instance of calculating the degree of instability is shown in Fig. 2. Here, the number of compatibility levels is defined as 3, and i_n is 1 for any level. The capacity of the right-hand side vertices is shown with the vertices. Here, h_2 is the only vertex with compatibility level 1 since $c_{h_2} = 6$, and this is the minimum of all. The degree of instability for the compatibility level 1 is set as 0 here. Consequently, h_4 falls into the compatibility level 2 and the degree of instability for h_4 is 1, h_3 and h_1 falls into the compatibility level 2 and the degree of instability for them is 2. The compatibility levels and the degree of instability of the right-hand side vertices can be verified similarly.

V. Proposed Algorithm

We propose an iterative algorithm to improve the weight and cardinality over the stable matching. The stable matching is first generated using the deferred-acceptance algorithm by Gale and Shapley [1] on the input graph. Then the algorithm which we define as the 'Balanced-Matching' algorithm replaces the stable pairs with the unstable pairs within the limit of the degree of instability determined for each vertex by the stability relaxation framework. We first define some notions and concepts used in the algorithm below.

- $\delta(M)$: The set of matched vertices with respect to the matching M.
- $I(u), I(v)$: The set of edges incident on a vertex $u \in U$, $v \in V$.
- $T(u)$: The proper subset of edges of $I(u)$ traversed by the algorithm at any point.
- Alternating path: An alternating path with respect to M is a set of edges where two consecutive edges, one is from M and another is from $(E-M)$ share an endpoint
- Augmenting path: An augmenting path is an alternating path that starts and ends with an edge from $(E-M)$.
- Balanced augmenting path *(BAP)*: An augmenting path always has an odd number of edges and the minimum length is 3. A *(BAP)* with respect to M is the augmenting path of minimum length where the combined weight of the first and third edge exceeds the weight of the second edge. We define m_a as the second or matched edge of *BAP* and n_a as the set of two unmatched edges of *BAP*. The replacement of m_a by the edges of n_a in M ensures the degree of instability determined for the vertices incident on them is not violated when the vertex $v \in V$ incident on the third edge has an empty spot.
- N: The set of starting edges of any A newly added in M as per the updates.
- $w(e)$: The weight of the edge e.

The steps of the Balanced-Matching algorithm *(BMA)* is shown in Algorithm 1. The stable matching S generated using the deferred-acceptance algorithm is the input of *BMA*. We first initialize a new matching M same as S and mark all the vertices from U not in $\delta(M)$ as unchecked. The *BMA* starts iterating the edges incident on the unchecked vertices which have not been traversed and checks the existences of a *BAP*. It updates M by replacing m_a with the edges of n_a if there exists any *BAP*. The *BMA* then adds the first edge e of *BAP* in both N and $T(u)$ and marks u as checked and stops traversing further edges incidents on u as per line $(4-8)$.

It is possible that an edge replaces the first edge of a *BAP* and still constructs a *BAP* with the rest of the two edges with a better total weight. The *BMA* updates the list N by adding the first edge of any *BAP* found. The purpose is to keep track of the first edges such that they can be replaced by any other edge. The edge e replaces any such first edge e' must be incident on a different vertex of U, otherwise e would have been added in a *BAP* earlier. The *BMA* checks whether the current edge e for which the iteration is going on can replace an edge e' if there is no *BAP* exists starting from e in line 9. If e can replace such e', *BMA* replaces e' with e in M, removes e' from N and the endpoint of e' from U as unchecked again. It also adds e to N and $T(u)$ and marks u where $e \in I(u)$ as checked before stopping checking further edges incident on u as per line $(10-14)$. One such replacement example can be observed in Fig. 2. A *BAP* $(h_2, s_1), (s_1, h_1), (h_1, s_2)$ exists starting from the edge (h_2, s_1) and the matching is updated accordingly. The edge (h_4, s_1) can replace the edge (h_2, s_1) since a *BAP* could be constructed using the edges $(h_4, s_1), (s_1, h_1), (h_1, s_2)$ with better total weight. Thus, the matching is updated again by replacing the edge (h_2, s_1) with (h_4, s_1) in this scenario.

Algorithm 1 Balanced-Matching

```
Input A stable matching $S$ in $G = (U, V, E)$.
    Output An updated balanced matching $M$ in
$G = (U, V, E)$.
1: initialize a new matching $M = S$.
2: Mark all $u \in \delta(M)$ as checked and mark all $u \notin \delta(M)$
   as unchecked.
3: while there exist an unchecked $u$ do
4:    for all ($e \in I(u)$ and $e \notin T(u)$) do
5:       if a BAP exist with respect to $M$ starts by $e$ then
6:          remove edge $m_a$ from $M$, add edges in $n_a$ in $M$
7:          add $e$ in $N$, add $e$ in $T(u)$, mark $u$ as checked.
8:          break
9:       else if $w(e) > w(e')$ where $e, e' \in (N \wedge I(v))$ for
         any $v \in V$ then
10:         replace $e'$ by $e$ in $M$, remove $e'$ from $N$.
11:         add $e$ in $N$, remove $e'$ from $N$, add $e$ in $T(u)$.
12:         mark $u$ as checked.
13:         mark $u' \in U$ where $e' \in I(u')$ as unchecked.
14:         break
15:      else
16:         add $e$ in $T(u)$
17:      end if
18:   end for
19:   if len($T(u)$) = $len(I(u))$ then
20:      mark $u$ as checked
21:   end if
22: end while
```

In the case when neither a *BAP* starts from e nor it can replace any e' from N, *BMA* only adds e into $T(u)$ as per line $(15-16)$. The purpose of constructing $T(u)$ for each u is to keep track of the traversed edges from $I(u)$. The *BMA* marks u as checked whenever the length of $T(u)$ reaches the length of $I(u)$, which indicates the completion of traversal for $u \in U$ and no updates were done for u as shown in line $(19-20)$. The iteration continues until there exists any unchecked vertex from U as per the loop defined in line 3.

The existence of a *BAP* guarantees the improvement of cardinality and weight in the resulting matching. This is due to the fact that one matched edge is being replaced by two unmatched edges with more total weight. It might be possible to update the matching in a similar fashion with an alternating path or augmenting path of any length. However, the former case can not improve the cardinality as the length is even or the path starts with an matched edge when the alternating path is not an augmenting path. The augmenting path of any length can improve the cardinality by 1 same as any *BAP*, but can make the iteration highly complex. Thus, we design *BMA* with the consideration of finding the augmenting path of minimum length to improve both weight and cardinality together for proper balancing of the matching. A *BAP* also ensures the degree of instability restriction for all the vertices is not violated. Thus, the *BMA* is capable of bringing fair balance of stability, weight and cardinality in the resulting matching by finding the *BAP* whenever exists.

VI. Popularity Measurement Model

The popularity of a matching is determined based on the choices of the vertices among all possible matching. The *BMA* is designed in a way such that the matching becomes more balanced with marginal instability and without losing popularity. For a one-to-many matching, the choice of the vertices of U can be determined by comparing the single partner from any two matchings, but the choice of the vertices of V needs the consideration of all the partners. Thus, we define a novel edge-weight-based multi-voting (*EWMV*) popularity measurement model where vertices of V can vote for each of the edges they are incident on.

The *EWMV* model compares the total number of votes between any two matching to determine their popularity over each other. It counts the votes for each edge incident on any vertex based on its weight. Thus the vertices from V can cast multiple votes up to their capacity. Let's consider two matchings M and M' to be compared using the *EWMV* model. The model first determines the number of edges incident on both M and M' for any vertex. It then equalizes the number of edges incident on a vertex in both matching by adding dummy edges with weight 0 in the matching where any vertex has fewer edges incident on it and put the edge in the last preference. It then follows the rules shown in table I below to count the votes for the vertices of U.

TABLE I: Rules for vertices of U in *EWMV* model

M	M'	Reasoning
1	0	$P(M) > P(M')$
0	1	$P(M) < P(M')$
1	1	$P(M) = P(M')$

TABLE II: Rules for vertices of V in *EWMV* model

M	M'	Reasoning
1	1	for the edges in $(M \wedge M')$
1	0	If the edge is not in $(M \wedge M')$ and there exist at least one edge in M' with less preference
0	1	If the edge is not in $(M \wedge M')$ and there exist at least one edge in M with less preference

Here, $P(M)$ indicates the preference of the partner paired with u in matching M. The vertex u votes for matching M if it gets a more preferred partner in M than M' and vice versa. The vertex u votes for both if it is paired with the same partner in both matching. We consider counting a vote for the same edge in both matchings to keep the vote count consistent with the number of edges.

Similar rules for the vertices of V are shown in the table II. The rules for the vertices of V apply to every single edge incident on any $v \in V$. The vertices of V vote for each edge incident on them in the matching. They vote for both matchings for the common edges as shown in the table II. They vote for M for the edge e, if e is not in M' and there exists a less preferred edge than e in M'. Similarly, they vote for M' for the edge e' if there exists a less preferred edge than e' in M. The *EWMV* model discards the edge with weight 0 after it contributes 1 votes to the other matching to avoid casting multiple votes for the same dummy edge. In this way, the *EWMV* model considers every edge of both matchings and cast votes for each of them. One instance of the popularity measurement using the *EWMV* model is shown in Fig. 3.

	$s_1(2)$ $[h_3,h_1]$	$s_2(1)$ $[h_1,h_2,h_3]$
h_1 $[s_2,s_1]$	4	6
h_2 $[s_2]$		5
h_3 $[s_1,s_2]$	5	3

Matching	Edges	Votes
M	$(h_1,s_2)(h_3,s_1)$	$h_1 = 1$, $h_2 = 0$, $h_3 = 1$ $s_1 = 1$, $s_2 = 1$
M'	$(h_1,s_1)(h_2,s_2)(h_3,s_1)$	$h_1 = 0$, $h_2 = 1$, $h_3 = 1$ $s_1 = 2$, $s_2 = 0$

Fig. 3: Example of *EWMV* model implementation

In Fig. 3, the vertices from U and V are shown in the first column and first row of the left table respectively. The preferences and the capacity where applicable for each vertex are provided in the same cell. The corresponding weights of the edges are shown in the designated cell of the left table. No value in a cell means the associated vertices are not suitable for pairing, (h_2, s_1) in Fig. 3. In the right table, the edges of the two matching M and M' are shown in the second column. The votes counted for each of the edges for both M and M' using the rules of the *EWMV* model shown in the last column. We can see vertex s_1 cast two votes for each of the edges of matching M' since the edge (h_3, s_1) is common with M and the edge (h_1, s_1) has better weight than the dummy edges with weight 0 added in M. The other votes can be verified similarly. Next, we show the resulting matching produced by the algorithm 1 is also popular according to the *EWMV* model.

Theorem 2: The matching produced by *BMA* is popular.

Proof: The *BMA* updates the stable matching M only when there exists a *BAP* or an edge with better weight than the first edge of a *BAP* found earlier. In the case of updating with a new *BAP*, the *BMA* replaces the matched edge with two unmatched edges of *BAP* in M. The updated matching M' gets behind by two votes which count for M since both vertices of the matched edge prefer this over the new edges. Otherwise, this edge would not have been added in M. On the other hand, M' gains two votes, one for the first edge and another for the third edge of the *BAP*. This is because the first edge of *BAP* is the incident on an unmatched vertex from U and the third edge of *BAP* fills up an empty space of a vertex from V. In the scenario, when the *BMA* found an edge that can replace the first edge of a *BAP* found earlier, the vote count for M' does not change. Hence, the updated matching M' gets the same number of votes as M, and it is popular.

The *EWMV* model is designed to verify the popularity of any matching in the *EWMP* setting considered in the research. Although the *BMA* is designed in a way such that it retains the popularity of the stable matching, we integrated the *EWMV* popularity measurement model in our experimentation to justify the correctness of the algorithm. We discuss our experimental results in the next section.

VII. Use Case and Experimental Results

The United Nation's 2030 agenda for sustainable development aims to build partnerships among developed and developing countries to integrate economic, social and environmental evolution. The development of sustainable cities with adequate shelters and improved human settlement systems is a major goal for all nations [24]. We consider the problem of placing homeless individuals into shelters under the broader domain of UN goals as a use case for the research.

Consider a bipartite graph $G = (H, S, E, W)$, where H is the set of homeless individuals, S is the set of shelters, E is the set of edges and W is the set of weights of the edges. Recall that the weight of an edge $(h \in H, s \in S)$ indicates the pairing suitability between h and s. No edge between such h and s means they are not suitable for pairing. The pairing suitability depends on multiple factors such as the shelter's location and type, the homeless individual's background etc. We are not discussing these factors in detail as this is not in the scope of the research. The homeless individuals and the shelters both have preferences over the other parties. Such as some homeless individuals want to stay in a particular city, some shelters might look for homeless individuals from a similar background to run a particular support service etc. The objective of the problem is to generate a one-to-many matching of homeless-to-shelter such that the total suitability or likelihood of the placement and the number of homeless individuals placed can be maximized while prioritizing the preferences of both sides. The trade-off shown in section III applies to the problem. Thus, we apply the stability relaxation framework and *BMA* on the random instances prepared for the problem to find the desired balance in the outcome. Next, we discuss the background for the data preparation.

Every year, employment and social development Canada published a report about the current homelessness situation in Canada and the capacity of the shelters [25], [26]. We closely followed the reports to reflect the real shortage scenario as much as possible on the random instances. We consider the scenario where homeless individuals have a singular bed demand. A suitability matrix for the homeless-shelter pairs is generated first where the suitability value ranges between $0 - 6$ as per [27]. Each pair is assigned a random suitability value. The suitability value 0 indicates incompatibility between the pair. The capacity of the shelters varies significantly depending on demands. It starts from 1 considering the fact that many small private organizations may have very few beds. Some shelters are most preferred and they usually have extremely low capacity with respect to demand. We thus consider a few shelters preferred by most homeless individuals and set their capacity to be a random value ranging between 1 and 10 percent of the total number of homeless individuals. The other shelters have higher capacity and thus assigned a random value ranging between 11 to 50 percent of the total number of homeless individuals. However, there are scenarios where some shelters with higher capacity are not preferred by many homeless individuals. Hence, we increase the incompatibility of the homeless individuals to some of the higher capacity shelters by converting their suitability value into 0. We then apply the stability relaxation framework discussed in section IV to determine the degree of instability for each agent.

In the stability relaxation framework, we define three compatibility levels, namely low, moderate and high. We prepare the list of adaptive compatibility values for homeless individuals and shelters. We then classified them into compatibility levels. We set the degree of instability for both the homeless individuals and shelters in the low compatibility level as 1. The degree of instability increment constant I_n is set differently for homeless individuals and shelters based on the number of agents on the other side. Let's assume there are N homeless individuals and M shelters. For the homeless individuals and shelters, I_n is set as $\frac{M}{10}$ and $\frac{N}{10}$ respectively. Thus, the homeless individuals in the moderate and high compatibility levels have $(1 + \frac{M}{10})$ and $(1 + \frac{2M}{10})$ degree of instability respectively which is $(1 + \frac{N}{10})$ and $(1 + \frac{2N}{10})$ for the shelters in the same levels.

TABLE III: Comparison of Total Suitability

No. of Homeless	No. of Shelter	Integer Program	Stable Matching	*BMA*
100	3	253.35	232.65	249.7
200	6	627	575.4	603.1
300	9	1074.3	970.6	1010
400	12	1635.3	1502.3	1560.4
500	15	2428	2170.1	2265.4
600	18	3093.65	2711.15	2809.5
700	21	3901.25	3375.9	3496

TABLE IV: Comparison of Number of Placement

No. of Homeless	No. of Shelter	Integer Program	Stable Matching	*BMA*
100	3	57.3	52.2	56
200	6	128	111.2	119.1
300	9	196.7	174.9	184
400	12	287.5	260.1	273.4
500	15	422	371.3	392.6
600	18	537.8	461.8	481.6
700	21	675.2	569.6	594

We first implemented the deferred-acceptance algorithm [1] using the preference list generated for homeless individuals and shelters based on the suitability matrix prepared. The deferred-acceptance algorithm produces optimal matching for the proposing side. Thus, we consider the homeless individuals as the proposing side to prioritize the survivors to get the best possible partner in the initial matching. We then implemented *BMA* with the degree of instability calculated for each agent.

We perform the coding in Python using Google Colab. We compare the total suitability and size of the matching produced by *BMA* with the stable matching. We also implemented the integer program of maximum weighted matching and maximum cardinality matching using the same data instances in IBM ILOG CPLEX Optimization Studio 22.1.0. We evaluated the performance of the resulting balanced matching with the output of the integer program. The comparison of the total suitability is shown in table III. Each instance is executed 10 times and the average is taken as the final value to cover the real-world variability. The results shown in table III indicate the total suitability improves significantly in the balanced matching produced by *BMA*. The total suitability of the stable matching remains far behind optimal. The *BMA* is able to reduce the gap to a large extent. For the instance of 500 homeless individuals and 15 shelters, the balanced matching has almost 100 unit more suitability than the stable matching. The stable matching is known to produce better results in terms of weight when the data size increases [28]. A notable weight or suitability increment in the matching of *BMA* even for the larger instances is evident as shown in table III.

Similarly, we compare the number of homeless individuals placed in the balanced matching. The balanced matching is able to reduce the capacity waste of the shelters by placing more homeless individuals as shown in table IV. The cardinality of the balanced matching is higher than the stable matching in all instances experimented. We also implemented the popularity measurement model discussed in section VI to justify the claim of retaining stable matching's popularity in the balanced matching. The results support the justification as the number of votes for the balanced matching is always the same as the stable matching in the experimentation. Thus, *BMA* is able to place more homeless individuals into the shelters with a better likelihood of the expected outcome utilizing adaptive and fair stability relaxation for every agent.

VIII. Conclusion and Future Work

Stable matching is a widely used technique for bipartite graphs with two-sided preferences. We study the limitation and perspectives of optimizing stability, weight and cardinality in a matching where the preferences are generated based on the weight of the edges. We introduce a stability relaxation framework which is adaptive to the characteristic of the vertices and fair with an independent degree of instability for them. Our proposed algorithm updates the stable matching using the degree of instabilities and produces a matching with the balance of stability, weight and cardinality. We define a new edge-weighted-multi-voting (*EWMV*) popularity measurement model to consider every edge in the voting process. We show the resulting matching is also popular as stable matching. The experimentation results based on the homeless placement system support the claim of the proposed combination of the stability relaxation framework and the algorithm to bring the desired balance in the matching. The proposed solution can be examined for similar applications with two-sided preferences. The investigation of the performance ratio of the resulting matching in terms of the optimal weight and cardinality can be a valuable extension of the research. The effect of the variable degree of instability among vertices can be analyzed for further improvement of the approach.

References

[1] D. Gale, L. S. Shapley: College Admissions and the Stability of Marriage. The American Mathematical Monthly **1**(69), 9–15 (1962)

[2] Manlove, D.: Hospitals/Residents Problem, Encyclopedia of Algorithms. Springer US, 390–394, Boston, MA (2008)

[3] Biró, Péter: Applications of matching models under preferences. AI Access (2017)

[4] Huang C.,Kavitha T: Popular matchings in the stable marriage problem. Information and Computation (222), 180–194 (2013)

[5] David M: Algorithmics of Matching Under Preferences. EATCS (2013)

[6] K. Iwama, S. Miyazaki: A Survey of the Stable Marriage Problem and Its Variants. Intl Conf on Informatics Education and Research for Knowledge-Circulating Society. 131–136, Kyoto, Japan (2008)

[7] Masarani, F.,Gokturk, S.: On the existence of fair matching algorithms. Theory and Decision **3**(26) (1989)

[8] Xing W., Niels A., Alan E.: Stable Matching for Dynamic Ride-Sharing Systems. Transportation Science 52(4).850–867 (2017)

[9] uri F., Ioannis M., Michalis S., Jay S.: (Un)stable matchings with blocking costs. Operations Research Letters, 49, 655–662 (2021)

[10] Daniel F., Atsushi I., Peter T., Suguru U.,Makoto Y.: Strategyproof Matching with Minimum Quotas. ACM Trans. Econ. Comput (2015).

[11] Krishnaa, P., Limaye, G., Nasre, M., Nimbhorkar, P.: Envy-Freeness and Relaxed Stability: Hardness and Approximation Algorithms. In: Algorithmic Game Theory., Springer, Cham. SAGT (2020)

[12] Y. Zhang,Y. Li: An Algorithm for Determining Critical Bipartite Graphs. International Symposium on Computer, Consumer and Control, Taichung, Taiwan, 64–67, (2012)

[13] Nasre M., Nimbhorkar P., Ranjan K.: Critical Relaxed Stable Matchings with Two-Sided Ties. arXiv (2303.12325) (2023)

[14] Zixuan P., Wenxuan S., Xiaoning Z., Bin Y.: Many-to-one stable matching for taxi-sharing service with selfish players. Transportation Research Part A: Policy and Practice, 160, 255–279 (2022)

[15] Kavitha T: Popularity vs maximum cardinality in the stable marriage setting. In Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete algorithms (SODA), USA, 123—134 (2012)

[16] Colin, T., David, M.: Popular matchings in the weighted capacitated house allocation problem. Journal of Discrete Algorithms (8)(2010).

[17] Abraham, D J., Irving, R W., Kavitha, T, Mehlhorn, K: Popular Matchings. SIAM Journal on Computing **4**(37), 1030–1045 (2007)

[18] S. Pettie: A simple reduction from maximum weight matching to maximum cardinality matching. Information Processing Letters (2012).

[19] Gärdenfors P: Match making: Assignments based on bilateral preferences. Behavioural Sciences (20), 166–175 (1975)

[20] Cseh, Á., Kavitha, T.: Popular Matchings in Complete Graphs. Algorithmica (83), 1493–1523 (2021)

[21] Kavitha G, Nasre M, Nimbhorkar, P, Reddy, T. P: Many-to-One Popular Matchings with Two-Sided Preferences and One-Sided Ties. Computing and Combinatorics, COCOON. 193–205, (2019).

[22] Nasre M, Amit R: Popularity in the generalized Hospital Residents Setting. CoRR (abs/1609.07650), (2016).

[23] Nasre M, Nimbhorkar P: Popular Matching with Lower Quotas. ArXiv (abs/1704.07546), (2017).

[24] Department of Economic and Social Affairs, UN: SDG-11, Sustainable Cities and Human Settlement. https://sdgs.un.org/topics/sustainable-cities-and-human-settlements. (2015)

[25] Employment and Social Development Canada: Shelter Capacity Report (2021). https://www.infrastructure.gc.ca/homelessness-sans-abri/reports-rapports/shelter-cap-hebergement-2021-eng.html

[26] Employment and Social Development Canada: Government of Canada Statistics (2022). https://www150.statcan.gc.ca/n1/pub/11-627-m/11-627-m2022017-eng.htm

[27] P. R. Saha, S. Choudhury and K. Salomaa: A Two-Stage Based Strategy to Optimize Homeless Placement in Shelters. IEEE International Humanitarian Technology Conference (IHTC), Ottawa, ON, Canada (2022)

[28] Lam, D.D., Nguyen, V.T., Le, M.H., Nguyen, M.T., Nguyen, Q.B., Le, T.S.: Weighted Stable Matching Algorithm as an Approximated Method for Assignment Problems. In: Intelligent Information and Database Systems, Springer, ACIIDS (2020)