Local Matching Dynamics in Social Networks

Martin Hoefer *

Abstract

We study stable marriage and roommates problems under locality constraints. Each player is a vertex in a social network and strives to be matched to other players. The value of a match is specified by an edge weight. Players explore possible matches only based on their current neighborhood. We study convergence of natural better-response dynamics that converge to *locally stable matchings* – matchings that allow no incentive to deviate with respect to their imposed information structure in the social network. If we have global information and control to steer the convergence process, then quick convergence is possible and for every starting state we can construct in polynomial time a sequence of polynomially many better-response moves to a locally stable matching. In contrast, for a large class of oblivious dynamics including random and concurrent better-response the convergence time turns out to be exponential. In such distributed settings, a small amount of random memory can ensure polynomial convergence time, even for many-to-many matchings and more general notions of neighborhood. Here the type of memory is crucial as for several variants of cache memory we provide exponential lower bounds on convergence times.

Keywords: Stable Marriage, Stable Matching, Best-Response Dynamics, Convergence

*Department of Computer Science, RWTH Aachen University, 52074 Aachen, Germany. mhoefer@cs.rwth-aachen.de. Tel.: +49 241 80 21104, Fax: +49 241 80 22216.

1 Introduction

Matching problems are at the basis of many important assignment and allocation tasks encountered in economics and computer science. A prominent model for these scenarios are *stable matching* problems [13], as they capture the aspect of rationality and distributed control that is inherent in many assignment problems today. A variety of allocation problems in markets can successfully be analyzed within the context of two-sided stable matching, e.g., the assignment of jobs to workers [5, 14], organs to patients [19], or general buyers to sellers. In addition, stable marriage problems are an interesting approach to model a variety of distributed resource allocation problems in networks [3, 12, 18].

In this paper, we examine a dynamic variant of stable matching for collaboration in (social) networks without central coordination. Players are rational agents that are looking for partners for a joint activity or relationship, such as, e.g., to do sports, write a research paper, share an office, exchange data etc. Such problems are of fundamental interest in economics and sociology, and they serve as basic coordination tasks in computer networks with distributed control. We can capture these problems within the stable roommates problem, an extension of stable marriage that allows arbitrary pairs of players to be matched.

Traditionally, in the stable marriage problem we have sets of men and women, and each man (woman) has a full preference list over all women (men). Each man (woman) can be matched to exactly one woman (man), where all players would rather be matched than unmatched. A *blocking pair* is a pair of a man and a woman such that both prefer this match to their currently assigned partners (if any), and a *stable matching* is a matching that allows no blocking pair. It is well known that in this bipartite case a stable matching always exists and can be found in polynomial time using the classic Gale-Shapley algorithm [11]. This does not hold for the extension to the non-bipartite stable roommates problem, where simple examples without stable matching exist.

A crucial aspect of ordinary stable marriage and roommates problems is that every player knows the complete player set and can match arbitrarily. In contrast, for many of our examples above, we would not expect a player to be able to, e.g., write a research paper with any other player instantaneously. Instead, there are often restrictions in terms of knowledge and information that allow certain players to match up easily, while others need to get to know each other first before they can engage in a joint project. We incorporate this aspect by assuming that players are vertices in a static social network of existing *social links*. Each player strives to build a *matching edge* to another player. The network defines a dynamic information structure over the players, where we use a standard idea from social network theory called *triadic closure*: If a knows b and b knows c, then a and c are likely to meet and thus can engage in a joint project. When matching a to the 2-hop neighbor c, both players engage in a joint project and thus become more familiar with each other. In this case, triadic closure suggests that a learns about all direct neighbors of c, which can allow him to find a better matching partner among those players. More formally, at any point in time, we assume that each player can match only to *accessible* players, that is, players in the 2-hop neighborhood in the graph composed of social links and currently existing matching edges.

In this paper, we will for the most part focus on the prominent class of *correlated* [1, 2] (also called acyclic [18]) preferences or weighted matching, where each possible match yields a single benefit value for both incident players, and preference lists are ordered according to edge benefits. For the ordinary correlated stable roommates problem, existence of a stable matching is guaranteed, a stable matching can be computed in polynomial time [1]. Furthermore, these games are potential games and a variety of dynamics converge to a stable matching in polynomial time [2].

Contribution For many of the assignment and matching tasks in (social) networks that motivate our study, there is an inherent lack of information and coordination. Hence, we are interested in distributed dynamics that allow players with locality restrictions to reach a stable matching quickly. We consider convergence to *locally stable matchings* – matchings that allow no blocking pair of accessible players – for variants of sequential *best-response* and *better-response dynamics*, where in each step a blocking pair of accessible players (or *local blocking pair*) is allowed to deviate to establish their joint edge. It follows directly from results in, e.g., [1, 2] mentioned above that our games are potential games, and thus all such dynamics are guaranteed to converge to a locally stable matching. This holds even for more general variants, in which each player can build up to $k > 1$ matching edges, or each player has access to all players within $\ell > 2$ hops in the graph.

After a formal definition of our model in Section 2, we show in Section 3 that in our basic game with $k = 1$ matching edges and lookahead $\ell = 2$ per player we can always achieve fast convergence: For every game and every starting state there is a sequence of polynomially many better-response moves to a locally stable matching. While this shows that locally stable matchings are achievable, the sequence heavily relies on global knowledge of the graph. In contrast, oblivious dynamics without such knowledge like random or concurrent better-response need an exponential number of steps. When we turn to more general games with $k > 1$ or $\ell > 2$, there are even games and starting states such that *every* sequence of better-response moves to a locally stable matching is exponentially long. For the special case of stable marriage, we show that for general preference relations there are states, from which convergence might never be achieved. We also show improved results for special cases of the problem, especially for one-sided social networks and the job-market model of [5].

Perhaps surprisingly, instead of the usual structural aspects in social networks such as, e.g., low diameter or power-law degree distribution, a natural aspect resulting in polynomial-time convergence is memory. In Section 5 we consider the case that every polynomial number of steps each player remembers (uniformly at random) one of the players he had been matched to before. This random memory is natural, e.g., when we assume there is a random process bringing players together for a limited amount of time. For instance, imagine a research conference where players have the chance to meet some of their previous co-authors, which gives them a temporary chance to renew their collaboration. This seemingly small but powerful adjustment allows to show polynomial-time convergence for a variety of dynamics, for arbitrary $k \geq 1$ and $\ell \geq 2$. We also touch upon the case when memory is considered as an addressbook or cache in which a bounded number of good or recent matches are stored. Here we can again show an exponential lower bound for standard eviction strategies such as FIFO or LRU. This shows that convergence in distributed settings does not only rely on the presence of memory but on the right *type* of memory. The strength of random memory is that players never completely forget previous matches. Essentially, they just have to wait until by chance they happen to meet the right partner again. In contrast, for cache memory we can construct instances in which matching edges that are crucial for progression of the dynamics are forgotten because memories are filled with a large number of unnecessary matches. Then they have to be rediscovered through the structure of the social network, and this can take exponential time.

Related Work There has been an enormous research interest in stable marriage and roommates problems over the last decades, especially in many-to-one matchings and preference lists with ties [6, 9, 14, 20]. For a general introduction to the topic, see standard textbooks [13, 20].

Theoretical work on convergence issues in ordinary stable marriage has focused on better-response dynamics, in which players sequentially deviate to blocking pairs. It is known that for stable marriage these dynamics can cycle [17]. On the other hand, there is always a sequence of polynomially many moves to a stable matching [21]. However, if the ordering of moves is chosen uniformly at random at each step, convergence time is exponential [2]. A prominent case with numerous applications [1,4,12,18], in which fast convergence is achieved even by random dynamics, is correlated or weighted stable matching where each matched pair generates a benefit (or edge weight) for the incident players and the preferences of the players are ordered with respect to edge benefit [1,18].

Local aspects of stable matching are of interest in distributed computing, e.g., communication complexity of distributed algorithms [16]. A localized version of stable marriage is analyzed in [10], where men and women are vertices in a graph and can only match to adjacent women or men, respectively. Each player can only exchange messages with their neighbors and the goal is to design a local algorithm that computes an "almost" stable matching. Similar approaches to almost stable matchings in decentralized settings include, e.g., [7]. In addition, there exist online [15] and parallel algorithms [8] for stable marriage.

Our model of locality is similar to Arcaute and Vassilvitskii [5], who consider locally stable matchings in a specialized case of stable marriage. In their job-market game, there are firms that strive to hire workers. Social links exist only among workers, and each firm can match to k workers, but each worker only to one firm. They show that best-response dynamics converge almost surely and show several characterization results for locally stable matchings and the number of isolated firms after a run of a local variant of the Gale-Shapley algorithm. In this paper we greatly extend their results on convergence of dynamics.

2 Model and Initial Results

We consider *stable matching games* with social context and correlated preferences as follows. We are given a social *network* $N = (V, L)$, where V is a set of n *players* and $L \subseteq V \times V$ is a set of undirected and fixed social *links*. In addition, we have a set E of undirected *potential matching edges*, where we denote $m = |E|$. Each edge $e \in E$ has a *benefit* $b(e) > 0$. A *state* $M \subseteq E$ of the game contains for each player at most k incident matching edges, i.e., each player can be involved in up to $k \geq 1$ matching edges simultaneously. Unless specified otherwise we will assume throughout that $k = 1$. The *utility* or *welfare* of a player u in state M is $\sum_{\{u,v\} \in M} b(\{u, v\})$ if he is matched to at least one player and 0 otherwise. The restriction of E to a subset of all edges will be mostly for convenience and clarity of presentation. Most of our lower bounds can be adjusted to allow every pair as matching edge using only minor technical adjustments. We will indicate the details of this construction in the proofs below.

In a state M two players u and v are *accessible* if there is a path of length at most ℓ in the *access graph* $G = (V, L \cup M)$. We call ℓ the *lookahead* of the game and, unless stated otherwise, focus on the case of triadic closure and $\ell = 2$. An edge $e = \{u, v\} \in E$ is called a *local blocking pair* for M if u and v are accessible, and if for each of u and v either (a) the player has less than k matching edges in M or (b) at least one incident edge $e' \in M$ has $b(e') < b(e)$. Hence, for a local blocking pair the accessible players both strictly increase their welfare by either adding e or replacing e' by e. In the latter case, the replaced edges are removed from M. We call $b(e)$ the benefit of the local blocking pair. A *locally stable matching* is a state M for which there is no local blocking pair.

We consider iterative improvement dynamics that lead to locally stable matchings. In a *local improvement move* we pick one local blocking pair and allow the involved players to deviate to their joint edge, thereby potentially removing other edges. We consider sequential processes that in every step implement one local improvement move. For a single step, we denote by $B \subseteq E$ be the set of all current local blocking pairs. For *(random) best-response dynamics*, we pick in each step deterministically (uniformly at random) one local blocking pair that has globally the maximum benefit, i.e., we choose one pair from the set $\{e \in B \mid b(e) = \max_{e' \in B} b(e')\}$. As *(random) better-response dynamics* we term all sequential dynamics that in each step pick one local blocking pair in a deterministic (uniformly at random) way from B. Finally, for *concurrent better-response dynamics* we assume that every player v picks uniformly at random from the local blocking pairs he is involved in, i.e., from $B_v = \{\{u, v\} \in B \mid u \in V\}$. For every local blocking pair that is chosen by both incident players the corresponding edge is built concurrently. For *concurrent best-response dynamics* the choice of player v is uniformly at random from his local blocking pairs with maximum benefit for him, i.e., from $\{e \in B_v \mid b(e) = \max_{e' \in B_v} b(e')\}$.

In addition, we also consider better-response dynamics with memory. For a dynamics with *random memory* we assume that each player at some point recalls a player that he had been matched to before. In particular, let M_v^t be the set of players that $v \in V$ has been matched with at some point during the first t steps of play. We assume that, in expectation, every T steps each player v remembers some player u chosen uniformly at random from M_v^t, and u and v become temporarily accessible in step $t + 1$. For dynamics with *cache memory* we assume that each player has a cache in which he can keep up to r players previously matched to him. A pair of players then is accessible if and only if they are currently at hop distance at most ℓ in G or one player is present in the cache of the other player.

Recall that a local blocking pair is a blocking pair with the additional requirement that players are accessible. Hence, in general every local blocking pair is a blocking pair, but the other direction holds only in special cases. In general, the set of blocking pairs is a superset of the set of local blocking pairs. For this reason, every stable matching is also a locally stable matching for every social network N. In particular, the set of stable matchings is a subset of the set of locally stable matchings, because locally stable matching must only avoid local blocking pairs (whereas stable matchings must avoid all blocking pairs). An ordinary stable matching (and, hence, a locally stable matching) can be computed in time $O(n \log n)$ by a greedy approach that repeatedly adds an edge for a blocking pair with maximum benefit. This algorithm computes a stable matching in polynomial time, for every $k \geq 1$ and every $\ell \geq 2$. For the ordinary correlated stable roommates problem even random best-response dynamics converge in polynomial time [2]. These convergence results, however, do not necessarily translate to locally stable matchings.

3 Convergence in Games with Correlated Preferences

In this section we consider the duration of sequential and concurrent improvement dynamics. Even when we restrict to accessible players, a new edge built due to a local blocking pair destroys only edges of strictly smaller benefit. This implies the existence of a lexicographical potential function. Hence, both sequential and concurrent better-response dynamics are always guaranteed to converge to a locally stable matching. Moreover, for our standard case with $k = 1$ and $\ell = 2$ the first result shows that we can always achieve fast convergence.

Theorem 1. *For any state of a stable matching game with correlated preferences, there is a sequence of $O(n \cdot m^2)$ local improvement moves that leads to a locally stable matching. The sequence can be computed in polynomial time.*

Proof. Consider a game with a given network N and a state specified by a set M of existing matching edges. Suppose there is a local blocking pair $e = \{u, v\} \in E$ for M, so $e \notin M$ and u and v are at hop distance 2 along a path u, w, v. Edge e falls in one of two categories: (1) link $\{u, v\} \in L$ or links $\{u, w\}, \{w, v\} \in L$; (2) one of $\{u, w\}$ and $\{w, v\}$ is a matching edge e', the other one a link in L. Note that a path of length 2 with $\{u, w\}, \{w, v\} \in E$ is impossible, because w would have two incident matching edges.

Suppose e falls in category (2) and let u be the vertex incident to e and e'. If u and v deviate to e, then e' gets destroyed, because u can only be incident to one matching edge. We will think of the edge moving from e' to e. This is the motivation for our main tool in this proof, the *edge movement graph* G_{mov}. The vertex set of this graph is E. Each vertex $\{u, v\} \in E$ has a corresponding vertex weight $b(\{u, v\})$. If u and v are at distance at most 2 in N, their vertex in G_{mov} is called a *starting point*. We denote the set of all starting points by S.

There are two kinds of edges in G_{mov}, *movement edges* and *domination edges*. For every triple of players $u, v, w \in V$ we introduce a directed movement edge in G_{mov} from $\{u, v\}$ to $\{u, w\}$ when $\{u, v\}, \{u, w\} \in E$, $\{v, w\} \in L$ and $b(\{u, v\}) < b(\{u, w\})$. Note that the target $\{u, w\}$ has a strictly higher vertex weight, and hence the movement edges induce a DAG. The intuition for the movement is as follows. If edge $\{u, v\} \in M$, then u and w are accessible. If there is no other matching edge in M, then $\{u, w\}$ becomes a local blocking pair.

In contrast, even if edge $\{u, v\}$ makes u and w accessible and $\{u, w\}$ has strictly higher benefit than $\{u, v\}$, the edge $\{u, w\}$ might not constitute a local blocking pair. This happens if w is currently matched in M with at least as much benefit and is expressed by a domination edge. Formally, for all pairs $\{u, w\}$ and $\{w, v'\}$ in G_{mov} we introduce a directed domination edge from $\{w, v'\}$ to $\{u, w\}$ when $b(\{w, v'\}) \geq b(\{u, w\})$. In this case $\{u, w\}$ is *dominated* by $\{w, v'\}$. If $b(\{w, v'\}) > b(\{u, w\})$, $\{u, w\}$ is *strictly dominated* by $\{w, v'\}$. Note that, in particular, every movement edge has a domination edge in reverse direction. However, domination edges are independent of the social network N. They do not necessarily induce a DAG, because the source of a domination edge has only weakly larger vertex weight than the target.

For an example of the construction of G_{mov} see Fig. 1. The edge of benefit 4 dominates all other edges, it has maximum benefit and both incident players have an incentive to destroy any existing matching edge if this edge is created. Therefore, we introduce a domination edge between the corresponding vertices in G_{mov}. In particular, there is a domination edge between the vertices of weight 4 and 2, even though there is no movement edge, and the edges in the matching game have no direct relation in terms of local information structure and creation of local blocking pairs.

A state M of the game is equivalent to a marking of all those vertices in G_{mov} that correspond to the edges in M. A local improvement move from M can only happen if some marked vertex p has an outgoing movement edge to another vertex p' (as $k = 1$, p' must be unmarked). This represents a feasible local improvement move only if p' is currently *undominated*, i.e., has no incoming domination edge from a marked vertex. We will describe how to migrate the markings along movement edges to reach a locally stable matching in a polynomial number of steps.

The general idea of the proof is to construct a migration pattern as follows. We first improve existing markings in terms of vertex weights without introducing new ones until we reach a subset of markings that is stable. In this subset, no existing marking can be (1) improved by moving it to

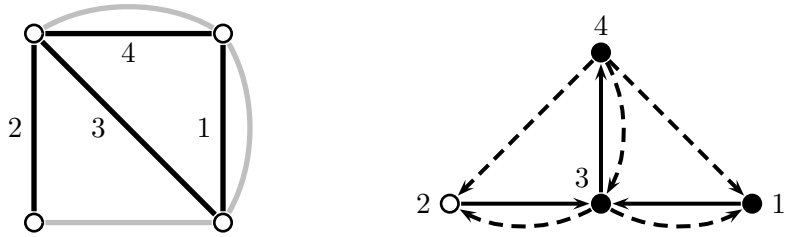

Figure 1: An example for a matching game (left) with the corresponding edge movement graph G_{mov} (right). In the matching game, social links are drawn in gray, possible matching edges are indicated in black. Edge labels show the benefit. In G_{mov} movement edges are solid, domination edges are dashed. Starting points are filled vertices. Vertex labels are weights and correspond to the edge benefits of the matching edge the vertex represents.

an undominated vertex with higher weight along a movement edge or (2) destroyed by creating a marking at a dominating vertex. In particular, this implies for the stable matching game that none of the matching edges represented by the markings can be destroyed neither (1) due to blocking pairs induced by the matching nor due to (2) local blocking pairs that can be constructed as a consequence of creating additional edges at starting points. For the latter, we have to ensure that there is no path of movement edges along undominated vertices from a starting point or an existing marking to a vertex that dominates a marked vertex. This intermediate goal is achieved in Phases 1 and 2. Finally, in Phase 3 we imitate the greedy algorithm for stable matching by iteratively creating a marking at the vertex with largest weight reachable from a starting point via an undominated path. This preserves the invariant that the existing matching edges are locally stable and finishes the proof.

Consider the subgraph of G_{mov} that is reachable from $S \cup M$ via movement edges. If a pair is not in this subgraph, there is no sequence of local improvement moves starting from M that establishes an edge between these players. Without loss of generality, we will prune the graph and consider only the subgraph reachable from $S \cup M$.

Phase 1: In Phase 1 we move existing markings without introducing new ones. As long as it is possible, we arbitrarily move an existing marking along a movement edge to an undominated vertex one at a time. Note that the set of dominated vertices changes in every step. In particular, a marking is deleted if it becomes dominated in the next step (because a dominating neighbor with higher benefit becomes marked).

For example, in Fig. 1 let the starting state contain the edges of benefit 1 and 2 and let the corresponding vertices in G_{mov} be marked. If we improve one of the markings, this means we create the diagonal edge of benefit 3. Now, by marking the corresponding vertex in G_{mov}, we must remove both original markings, because by creating the diagonal edge both original matching edges get destroyed.

Thus, in this process the number of markings only decreases. In addition, for each marking, the vertex weight of the location only increases. Due to acyclicity of movement edges, a particular marking can visit each vertex of G_{mov} only once, thus the phase ends after at most $O(n \cdot m)$ many steps. Note that all of these steps can be found in polynomial time by

examining the markings in G_{mov} and the neighboring vertices.

Phase 2: In Phase 2 we try to improve existing markings even further by introducing new ones and moving them via undominated paths to strictly dominating vertices. In particular, for a marked vertex $\{u, v\}$ in G_{mov} we do the following. We drop all currently dominated vertices from consideration. Now we try to find a path of movement edges from a starting point to a vertex that strictly dominates $\{u, v\}$, which can be done in polynomial time applying, e.g., a DFS from every starting point. If there is such a path, we can introduce a new matching edge via a new marking at the starting point and move it along the path to the dominating vertex. As none of the path vertices are dominated, all the moves are local improvement moves in the game. All markings that become dominated during this process are removed. This also includes in the last step our original marking at $\{u, v\}$. Thus, we can view this process as moving the marking at $\{u, v\}$ to the dominating vertex in $O(m)$ steps. After this, we try to improve all markings by a restart of Phase 1. We keep executing this procedure (move marking to dominating vertex via undominated path from a starting point and restart Phase 1) until this kind of improvement is impossible for every remaining marking.

Phase 2 can be seen as an extension of Phase 1. Overall, we keep decreasing the number of markings, and each surviving marking is increased in terms of vertex weight. However, each such increase might consist of introducing a new marking at a starting point and moving it to a dominating vertex, which requires at most $O(m)$ steps. This increases the number of steps to at most $O(n \cdot m^2)$.

In terms of computation, we can execute this phase as follows. For each marking we examine every possible dominating vertex. For each such vertex, we check the existence of an undominated path from a starting point by dropping all dominated vertices in G_{mov} from consideration and executing a DFS in the remaining DAG of movement edges. All these computations obviously require only polynomial time.

Phase 3: There are only two ways in which an existing marking can be changed – either by moving it along a movement edge to another undominated vertex, or by a sequence of moves that lead to creation of a marking at a dominating vertex. If the latter is possible, Phase 2 obviously has not ended, as this is the type of constellation that we address there. Phase 2 ends with a restart of Phase 1, so the former change is also impossible. Hence, after Phase 2, none of the remaining markings can be (re)moved. The corresponding edges are therefore stable, and we call the incident players *stabilized*. They will not become part of any local blocking pair in the remaining process.

In Phase 3, we now iteratively add edges until we reach a locally stable matching. Our invariant over the iterations will be that no existing matching edge can be moved or removed. In an iteration, we first drop all matched players from the game and adjust G_{mov} by dropping every vertex including at least one matched player and all incident movement and domination edges. We then construct the reachable matching edge of largest possible benefit. In particular, we consider the reduced G_{mov} (which is now completely unmarked) and find a vertex with largest benefit that is reachable from a starting point. We then establish the corresponding edge by moving a new marking along the path. When the marking has reached the end of the path, i.e., the reachable edge with largest benefit has been constructed, there is no path to any edge with strictly larger benefit, and no player will get an incentive to remove this

edge. Furthermore, due to our invariant, the process of moving the marking along the path has not destructed any of the previously existing edges. This implies that after the iteration there are strictly more stabilized players and the invariant continues to hold. Intuitively, this is a translation of the greedy matching algorithm for construction of a stable matching in the ordinary stable roommates problem to our localized environment. Hence, Phase 3 completes the locally stable matching and terminates after $O(n \cdot m)$ steps in total.

Note that pruning of G_{mov} is easily executed in polynomial time. Finding an undominated path to the reachable vertex of maximum vertex weight can be done by dropping dominated vertices from consideration and executing a DFS in the DAG of remaining movement edges. As we need to introduce only a polynomial number of markings in Phase 3, we can implement finding all required improvement moves in polynomial time.

□

The computation of the short sequence in the previous theorem relies heavily on identification of proper undominated paths in the edge movement graph. If we instead consider random or concurrent dynamics that do not rely on such global information, convergence time can become exponential.

Theorem 2. *For every $b \in \mathbb{N}$ there is a stable matching game with $n \in \Theta(b)$ in which (random) best-response and random and concurrent better-response dynamics starting from the state $M = \emptyset$ need $\Omega(1.5^b)$ steps in expectation to converge to a locally stable matching.*

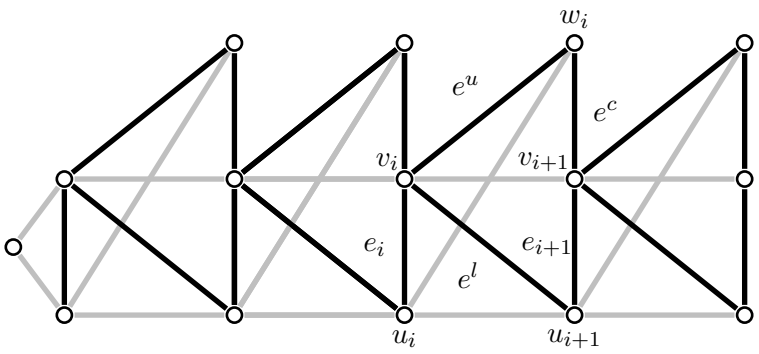

Figure 2: Structure of edge traps in the lower bound of Theorem 2 for $b = 4$. Social links are exactly the edges drawn in gray, possible matching edges are edges drawn in black. The sets of matching edges and social links are disjoint. For the benefits of matching edges we have $b(e_i) < \min\{b(e^u), b(e^l)\}$, $b(e^u) < b(e^c)$, $b(e^l) < b(e_{i+1})$ and $b(e^c) < b(e_{i+1})$. The relation of $b(e^u)$ and $b(e^l)$ can be adjusted to the type of dynamics under consideration.

Proof. We assume that matching edges E and social links L are disjoint. Initially, let $M = \emptyset$. We first prove the theorem for the case $E \subset V \times V$ and consider the case $E = V \times V$ below.

The gadget construction is based on a structure we call an "edge trap". For the full instance we concatenate b edge traps as shown in Fig. 2. In trap i there is a starting edge $e_i \in E$. Our argument follows by lower bounding the number of times that e_i must be created to lead to construction of

e_{i+1}. Upon creation of e_i there are exactly two local blocking pairs that both destroy e_i – either create the *upper edge* e^u or the *lower edge* e^l. Thus, $b(e_i) < \min\{b(e^u), b(e^l)\}$. If e^u is formed, $\{w_i, v_{i+1}\}$ becomes a local blocking pair with e^c destroying edge e^u (i.e., $b(e^u) < b(e^c)$). At this point, both w_i and v_{i+1} are happy with their matching choice. Now suppose e_i is created again, then player w_i will not form e^u. In this case, v_i matches to u_{i+1} via e^l. Now v_{i+1} has an incentive to drop e^c and match with u_{i+1}, because $b(e^c) < b(e_{i+1})$.

By sequential concatenation of edge traps we can make the dynamics simulate a counter with b bits. In particular, we assume there are b traps attached sequentially. In the first trap, there is an additional dummy player u' and two social extra social links to players u_1 and v_1. For the last pair of vertices u_{b+1} and v_{b+1} we assume there is also the final edge e_{b+1}. Bit i is set if and only if edge e^c in trap i is created. We start the dynamics with the empty matching $M = \emptyset$, so the counter is 0. Observe that N is a tree composed of a long path $v_{b+1}, \ldots, v_1, u', u_1, \ldots, u_{b+1}$ with some extra leaf vertices w_i. For $M = \emptyset$ it is straightforward to see that the only local blocking pair is e_1, as this is the only pair of accessible players with a matching edge in E.

First consider best-response dynamics, for which we set $b(e^u) > b(e^l)$ in every trap. Then creation of e_1 implies creation of e^u and then e^c in the first trap, i.e., increase of the counter by 1. At this point, the edge is trapped. Now the only local blocking pair is again e_1, and this leads to improvement steps creating e^l and e_2 that destructs e^c in the first trap. Finally, we create e^c in the second trap, and thus make an increase in the bit counter of 1. Observe that every time M consists of some subset of e^c-edges from the traps, e_1 is the only local blocking pair. Then, a creation of e_1 leads to a state of e^c-edges from the traps that represents an increase of the bit counter by 1. Thus, to create e_{b+1}, the edge of largest benefit, the dynamics needs $\Omega(2^b)$ many creations of e_1. Note that in each step of the dynamics the local blocking pair of maximum benefit is uniquely determined. This proves the lower bound for both deterministic and random best-response dynamics.

For each state reached during the dynamics, only one local blocking pair except (u_1, v_1) can establish their edge. In particular, e^u and e^l cannot be created simultaneously as they have a joint vertex. Hence, as long as the creation of e^u is sufficiently likely in every trap, we can trap enough edges that are subsequently destroyed and show a similar lower bound for other dynamics. We present the argument for random and concurrent better-response dynamics.

Every locally stable matching must contain the edge e_{b+1}. We now bound the number of times we have to create e_1 until e_{b+1} forms. Consider $M = \emptyset$, then e_1 is the unique local blocking pair. Suppose edge e_1 is created and follow this edge moving through the trap. With probability at most $1/2$, the edge moves directly to e^l and arrives at e_2. With probability at least $1/2$, the edge gets stuck in e^c, which implies that a new edge at e_1 is introduced that destroys the edge in e^c and arrives at e_2 with probability 1. Thus, to create a single edge at e_2 we have to create e_1 an expected number of 1.5 times. The same is true for e_i and e_{i+1} in every trap i. Thus, due to the sequential nature of the gadget, we need an expected number of $\Omega(1.5^b)$ creations of e_1 to create edge e_{b+1}. This proves the lower bound for concurrent and random better-response dynamics.

Finally, we note that by assigning the same tiny benefit of ϵ to all matching edges not explicitly mentioned above, we can apply the same arguments when $E = V \times V$. Note that the relevant edges mentioned above all run between one partition composed by vertices v_i and the other partition composed by vertices u_i and w_i. Edges of benefit ϵ will only be established within partitions, and within the set of these edges there evolves no dynamic discovery process as they all have benefit of ϵ. Furthermore, by existing within partitions they cannot speed up the discovery of good edges between partitions. There is only one slight exception to this observation in trap 1, where an edge

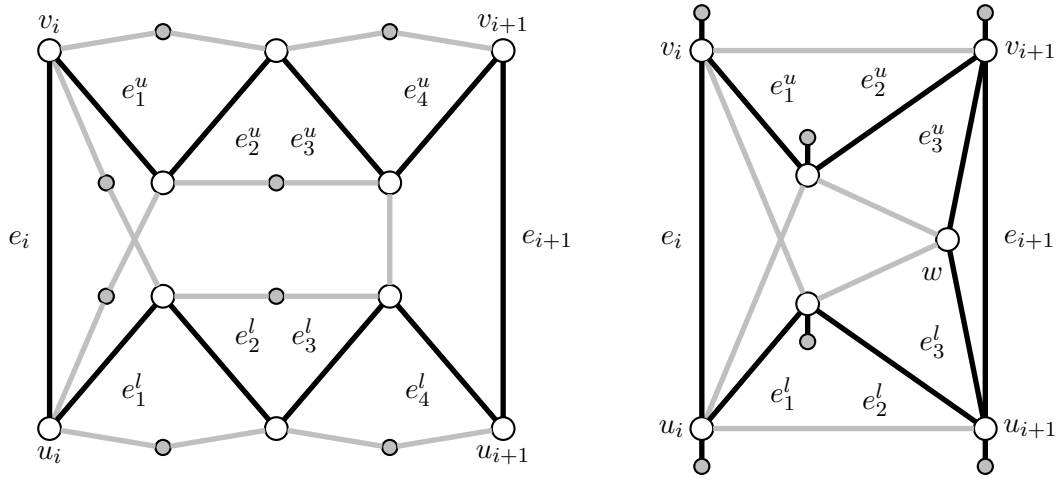

Figure 3: Block structures in the lower bound of Theorem 3. Social links are drawn in gray, matching edges with significant benefit are drawn black. Extension to the complete graph structure by sequential construction as in Fig. 2.

connecting u_2 to the dummy vertex between u_1 and v_1 can lead to discovery of $\{v_1, u_2\}$. Thereby, we might directly create $\{u_2, v_2\}$ and bypass creation of $\{w_1, v_2\}$. This, however, can be avoided by introducing another dummy player u'' that has a social link only to u'. The matching edge between these players is of very large benefit and present in the starting state. This way, no dummy player can be part of any local blocking pair during the dynamics. □

The previous theorem applies similarly to a wide variety of better-response dynamics. The main condition is the choice rule of the dynamics is such that whenever e_i exists and both e^u and e^l are available for creation, then the creation of e^u is always at least as likely as that of e^l. Observe that the construction allows to set $b(e^u) < b(e^l)$ or vice versa. By appropriate choice, we can ensure that the choice rule for a variety of dynamics satisfy the main condition explained above. In particular, this can be ensured for dynamics that make "oblivious" choices by picking local blocking pairs based only on their benefits but not their structural position or the history of the dynamics.

An even stronger lower bound applies when we increase lookahead or matching edges per player.

Theorem 3. *Let $k \geq 2$ or $\ell \geq 3$, then for every $b \in \mathbb{N}$ there is a stable matching game with $n \in \Theta(b \cdot k \cdot \ell)$ and a starting state M such that every sequence of local improvement moves from M to a locally stable matching has length $\Omega(2^b)$.*

Proof. We prove the theorem by examining several cases.

$k = 1$, $\ell = 3$: The construction that provides the exponential lower bound is presented in Fig. 3 (left). The complete network N is divided into b traps and has $\Theta(b)$ vertices.

The construction uses relevant players (large circles) and dummy players (small circles). Dummy players are not involved in any of the blocking pairs and just serve to increase distance between relevant players.

For the relevant players, each block i contains a starting edge e_i between u_i and v_i. When the starting edge is created there are exactly two improving moves by the incident players that lead to creation of a "lower" or an "upper" edge e_1^l, e_1^u, i.e., $b(e_i) < \min\{b(e_1^l), b(e_1^u)\}$. In both cases, the creation leads to a removal of e_i. Each of the edges $e_i^l(e_i^u)$ yield a unique improving move that creates e_{i+1}^l (e_{i+1}^u) and removes e_i^l (e_i^u), respectively (i.e., $b(e_i^u) < b(e_{i+1}^u)$ and $b(e_i^l) < b(e_{i+1}^u)$, for $i = 1, 2, 3$).

Suppose both e_4^l and e_4^u are created, then the outer players u_{i+1} and v_{i+1} are at hop distance 3 and thus create a direct edge e_{i+1} ($b(e_{i+1}) > \max\{b(e_4^u), b(e_4^l)\}$), thereby removing both e_4^l and e_4^r. Hence, to create e_{i+1} we have to create e_4^l and e_4^r, which in consequence means we have to create e_i twice.

Now consider a game with b blocks and recall that the last block has an additional edge e_{b+1}. Initially $M = \emptyset$. For the first block, we assume that u_1 and v_1 are connected via a path of length 3 (involving two dummy players). Upon creation of e_2, both e_2^l and e_2^u from the first block are removed. By repeating this argument, we see that for creation of e_{b+1}, e_1 needs to be created 2^b times. This proves the bound for $k = 1$ and $\ell = 3$.

We can use similar adjustments as in the previous proof when all edges are possible matching edges. This means that a large number of additional matching edges not mentioned above are present in E. The first problem is that these additional edges might block the dynamics, destroy the local blocking pairs described above and lead to early termination of the dynamics. This problem is addressed by giving all additional edges involving non-dummy players a tiny benefit of ϵ. The second problem is that they lead to early discovery of edges with significant benefit and thereby speed up the convergence process. It is easy to address this latter problem for the dummy players. We introduce for each dummy player a second dummy player and connect them via a matching edge of high benefit, which is present in the starting state. In this way, no dummy player is part of any local blocking pair, and, in particular, no matching edge involving a dummy player influences the dynamics. All other additional edges get the same tiny benefit of ϵ. Then, intuitively, during the evolution of the dynamics, additional edges can only be created "along paths in N" and edges with significant benefit are present "between paths in N". More formally, a straightforward inspection of all steps described above allows to observe that no additional edge leads to the discovery of any edge with significant benefit. Thus, they cannot speed up the dynamics, which completes the proof.

$k > 1$, $\ell = 3$: For $k > 1$, we apply the same block construction, and for each relevant player v we join it into a clique with $k - 1$ dummy players and match them all up in our starting state. These edges become the most valuable for all involved players. For the dummy players, there are no other matching edges in E, so they are not part of any blocking pair. For v this construction yields $k - 1$ very valuable matching edges, which he will not remove for the entire time. Thus, he is left with one matching edge to be created, which allows to construct the same dynamics within the blocks as before.

When we allow $E = V \times V$ in this construction, the $k - 1$ dummy players in each clique can create exactly one additional matching edge. These edges could create the same problems as mentioned above, i.e., early termination or shortcuts through early discoveries of relevant edges of the exponential-time dynamics. A simple trick can be used to fix these problems: We set up two copies of the instance described so far. Then connect each dummy vertex from the cliques to his copy with a matching edge of very high benefit. In this way, all dummy players

get their k most beneficial matches in the starting state and are thus never part of any local blocking pair. The two instances are sufficiently separated such that non-dummy players are not able to discover relevant edges. Hence, the exponentially long dynamics within the sets of relevant edges evolve in both instances without influencing each other.

$k > 1$, $\ell > 3$: If $\ell > 3$, we simply add more dummy players in order to keep the relevant players at the correct distance. In particular, instead of one dummy player between relevant players as in Fig. 3 (left), we introduce $(\ell - 2)$ players to get a hop distance of ℓ between endpoints of all e_j^l and e_j^u by the time of their creation, for $j = 1, ..., 4$. The edges that connect e_4^l and e_4^u receive $\ell - 3$ dummy players to keep u_{i+1} and v_{i+1} at distance ℓ when e_2^l and e_2^u exist. The arguments from the previous case with cliques of dummy players and instance copying can similarly applied here to allow $k > 1$ and every edge as matching edge.

$k > 1$, $\ell = 2$: Finally, we also treat the case with $k > 1$ and $\ell = 2$. Here we require only a slight adjustment of the block structure, as Fig. 3 (right) shows for the case $k = 2$. We again add one dummy player for all relevant players with a link and a highly valuable matching edge. The only exception is to the common endpoint w of e_3^l and e_3^r, which gets no associated dummy player, because he must be motivated to create two matching edges to relevant players simultaneously. In this case, we have $b(e_2^l) < b(e_3^l)$ and $b(e_2^r) < b(e_3^r)$. e_{i+1} can only be created when both e_3^r and e_3^l exist, and the creation of e_{i+1} destroys them both. In turn, e_i must be created once for e_3^l and e_3^r, respectively. This implies that one creation of e_{i+1} requires two creations of e_i. Finally, we again attach b blocks sequentially. For $k > 2$ and $\ell = 2$, we use a similar approach of "hard-wiring" matching edges using dummy players in the starting state, thereby leaving one additional matching edge for all relevant players and two for vertex w in each block. This allows to show the result for this case using the arguments made above.

To allow every edge as matching edge we can use similar observations as before. The main intuition in the previous cases is that matching edges with significant benefit are either present in the starting state and never removed (e.g., for dummy players) or their endvertices are at a sufficient hop distance in N such that additional edges of benefit ϵ cannot serve to create local blocking pairs that shortcut the exponential dynamics. In principle, these conditions can be preserved here as well with one exception. In the gadget as shown in Fig. 3 (right) we could potentially create an edge of tiny benefit between u_i and w. This could lead to the discovery of e_1^l and e_1^u without creation of e_i. However, this shortcut can be avoided by enlarging the construction and increasing the number of e_i^l and e_i^u edges, similar to Fig. 3 (left). With this extension the previously outlined constructions can be applied to establish the lower bound also when every edge can be a matching edge.

This proves the theorem. □

By embedding the lower bound structures from the previous proofs into larger graph structures of dummy vertices, we can impose a variety of additional properties on the network N that are often considered in the social network literature. For example, to obtain a small diameter simply add a separate source and connect each vertex from the gadgets via ℓ dummy vertices to the source. We assume that these new vertices have no matching edges, and in this way the lower bounds can be established accordingly for graphs of diameter $\Theta(\ell)$. Note that in graphs with a diameter of exactly ℓ we obtain the ordinary correlated stable roommates problem, because every player knows about every other player in the graph. In this case, polynomial-time convergence is guaranteed.

There also exist simple adjustments using dummy vertices and tiny and extremely large benefits to adjust the results to $E = V \times V$ with all possible matching edges as described above.

Corollary 4. *Theorems 2 and 3 continue to hold even if* $diam(N) \in \Theta(\ell)$.

4 Two-Sided Matching and Stable Marriage

In this section we consider the bipartite case of stable marriage. In accordance with [5] we use the interpretation of a set X of "workers" matching to a set Y of "firms". We denote $n_x = |X|$ and $n_y = |Y|$. In general, the social network $N = (X \cup Y, L)$ can be arbitrary, and the set of possible matching edges is $E = X \times Y$. Each worker (firm) has an arbitrary preference relation over firms (workers).

A variant of this scenario is the *job-market game* considered in [5] that represents the natural special case when the social network $N = (X, L)$ exists only among one partition. Furthermore, in this game each firm $y \in Y$ can build up to k matching edges while each worker can have at most one matching edge.

We first focus on the more general scenario of stable marriage. In contrast to ordinary stable marriage, in our localized variant convergence of any better-response dynamics can be impossible.

Theorem 5. *There are stable marriage games with general preferences and starting states M such that no locally stable matching can be reached by any sequence of local improvement moves from M, even if N is connected.*

Proof. Consider a game with $X = \{x_1, x_2, x'\}$ and $Y = \{y_1, y_2, y'\}$. The network N is connected and has links (x_1, x_2), (x_1, x'), (x', y'), (y', y_1) and (y_1, y_2). For the preferences, both x' and y' prefer most to be matched to each other. For worker x_1 the preference is (y_2, y_1), for x_2 it is (y_1, y_2). For firm y_1 the preference is (x_1, x_2), for y_2 it is (x_2, x_1). The starting state is $M = \{(x', y'), (x_1, y_1)\}$. Every pair $(x, y) \in X \times Y$ such that x and y are at distance 2 in the graph composed of social links and matching edges has either $x = x'$ for $y = y'$. However, due to x' and y' being matched to their most preferred partner, no additional matching edge can be introduced into the system. Instead, the only local blocking pair at state M is (x_1, y_2), which replaces edge (x_1, y_1). Next, the only local blocking pair is (x_2, y_2), which replaces (x_1, y_2). Due to the cyclic nature of preferences, we have reached an equivalent state. □

For weighted matching with correlated preferences, Theorem 1 applies and shows existence of a short sequence. A central assumption of [5] is that every worker $x \in X$ has the same preference list over Y, and every firm $y \in Y$ has the same preference list over X. We will refer to this case as *totally uniform preferences*. As a generalization we consider *worker-uniform* preferences, where we assume that only the preferences of all workers are the same, while firms have arbitrary preferences. *Firm-uniform* preferences are defined accordingly. For totally uniform preferences we can number firms and workers increasingly from least to most preferred in their respective global preference list. For edge $e_{ij} = (x_i, y_j)$ we define a benefit $b(e_{ij}) = j \cdot n_x + i$. Intuitively, here best-response dynamics give preference to local blocking pairs of the most preferred firm, which can be changed to worker by using $b(e_{ij}) = i \cdot n_y + j$ throughout. For worker-uniform preferences we let the numbering of workers be arbitrary. For $e_{ij} = (x_i, y_j)$ we define benefit $b(e_{ij}) = j \cdot n_x + i_j$, when worker x_i is ranked at the i_j-th last position in the preference list of firm y_j. For firm-uniform preferences the same idea can be used by exchanging the roles of firms and workers. This shows that all these cases

are classes of correlated stable matching problems. Our first result is that even for totally uniform preferences, convergence of best-response dynamics can be slow.

Theorem 6. *For every $b \in \mathbb{N}$, there is a stable marriage game with totally uniform preferences, $n_x, n_y \in \Theta(b)$ and a starting state M such that (random) best-response dynamics from M to a locally stable matching take (expected) time $\Omega(2^b)$.*

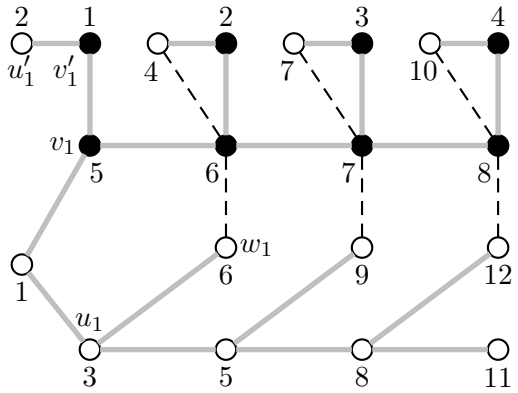

Figure 4: Construction in the lower bound of Theorem 6 for $b = 3$. Filled vertices are firms, empty vertices are workers. Solid edges are social links, every possible edge between a worker and a firm can become a matching edge. Dashed edges indicate the matching edges representing the setting and resetting of bits. Numerical vertex labels are the preference for matching among the respective other player type, higher number implies more preferred. Symbolic labels indicate how the gadget relates to the edge trap in Fig. 2.

Proof. Our game is an adjusted version of the edge trap structure in Fig. 2, see Fig. 4 for an example. In particular, we consider firms and workers numbered in increasing preference and set without loss of generality edge benefits to $b(x_i, y_j) = j \cdot |W| + i$. We treat all vertices v_i as firms and all other vertices u_i, and w_i as workers. The dummy player u' becomes a worker, as well. In addition, for each v_i we assume there is an additional firm v'_i and a worker u'_i. These are connected in N via $\{u'_i, v'_i\}$ and $\{v'_i, v_i\}$. Vertex v'_i is firm y_i and vertex v_i is firm y_{b+1+i}. For the first edge trap, the dummy player u' connects u_1 and v_1, and u' is the lowest ranked worker number 1. The vertices u'_1 and u_1 are workers number 2 and 3. The top vertices in trap i (labeled x_i in Fig. 2) are workers number $3i + 3$. The vertices u'_i and u_i are workers number $3i - 2$ and $3i - 1$, for every $i > 1$. In the starting state we have $M = \{(u'_i, v_i) \mid i = 1, \ldots, b + 1\}$.

Our construction implements a bit counter as follows. Bit i is set if the edge corresponding to e_c in Fig. 2 is created in trap i (i.e., edge $(x_{3(i+1)}, y_{b+i+2})$ in our numbering of workers and firms). Bit i is reset if instead (u'_{i+1}, v_{i+1}) (or (x_{3i+1}, y_{b+i+2}) in our numbering) is created. The dynamics thus starts in a state in which all bits are reset. Best-response dynamics always prefer blocking pairs with the firm ranked highest in the preference order. In the beginning, y_{b+2} connects to x_3 and then to x_6. Then firm y_{b+3} has an incentive to match with x_6, and the first bit is set. Now y_{b+2} again matches with x_3, then deviates to x_5 and x_9. This allows y_{b+3} and y_{b+4} to rewire to x_9, which removes setting of bit 1 and sets bit 2. Then bit 1 is reset by y_{b+3} connecting to x_4, as this is the best worker he currently knows that is willing to be match with him. In general, when

bits 1 to i are set, y_{b+2} matches along the workers x_3 and x_{3j+2} for $j = 1, \dots, i$ to $x_{3(i+2)}$. Then iteratively all y_{b+j+2} remove their bit edges, for $j = 1, \dots, i$, until bit $i+1$ is set by creation of $(x_{3(i+2)}, y_{b+i+3})$. Afterwards, iteratively from firm y_{b+i+2} to y_{b+2} all bits are reset. This shows that the dynamics implement a bit counter with b bits and the theorem is proved. □

A case in which such preferences can still lead to quick convergence is in the *job-market game*. Note that for the following theorem we only need worker-uniform preferences.

Theorem 7. *For every job-market game with worker-uniform preferences, in which each firm can create up to $k \geq 1$ matching edges, (random) best-response dynamics converge to a locally stable matching from every starting state in (expected) time $O(n_x \cdot n_y \cdot k)$, and random and concurrent better-response dynamics converge in expected time $O(n_x^2 \cdot n_y^2 \cdot k)$.*

Proof. In the job-market game, all firms are isolated vertices in N. Thus, no firm-worker pair is at distance at most 2. Thus, if a local blocking pair (x_i, y_j) deviates, this either leads to firm y_j having an additional matching edge or removing an existing one. Also, the existence of a matching edge (x_i, y_j) can only create local blocking pairs involving firm y_j (as with (x_i, y_j) no worker becomes closer to a firm other than y_j).

These insights directly show polynomial convergence time if all firms can only create $k = 1$ matching edge. In this case, no new matching edge can be introduced into the system without removing an existing edge. Each change of a matching edge involves the same firm, thus there are only $|X|$ steps involving a single firm y_j before it is either stabilized or disconnected. Hence, in total there are only $|M| \cdot |X|$ steps until we reach a locally stable matching from starting state M. This holds for arbitrary preference lists among workers and firms.

For $k > 1$, new matching edges can be introduced by a firm that connects to a worker x_i and creates an additional edge to a neighbor of x_i. For this case, fast convergence holds for worker-uniform preferences. In particular, consider the highest ranked firm y. No worker x currently matched to f is part of any blocking pair. Also, if there is a local blocking pairs involving the highest ranked firm y, this is a local blocking pair no matter which matching edges currently exist to lower ranked firms. Thus, after at most $k \cdot |X|$ steps all existing edges of firm y are stabilized and no new ones can be introduced. Hence, we can remove y and all matched workers and apply the argument repeatedly to the remaining instance.

Note that in (random) best-response dynamics at first only local blocking pairs are considered that involve the highest ranked firm y. The dynamics stabilize y by introducing up to k edges and moving each of them monotonically to higher ranked workers before proceeding to the next firm. Applying this insight iteratively, we see that in total these dynamics take at most $O(n_x \cdot n_y \cdot k)$ steps until reaching a locally stable matching. For random and concurrent better-response dynamics, we pick a local blocking pair with maximum benefit with probability at least $1/(n_x \cdot n_y)$. This implies that the dynamics converge in an (expected) number of $O(n_x^2 \cdot n_y^2 \cdot k)$ steps. □

In contrast, if we consider firm-uniform preferences, a lower bound for best-response dynamics can be shown.

Theorem 8. *For every $b \in \mathbb{N}$ and $k \geq 2$, there is a job-market game with firm-uniform preferences, $n_f \in \Theta(b)$, $n_w \in \Theta(b \cdot k)$ and a starting state M such that (random) best-response dynamics from M need $\Omega(2^b)$ steps (in expectation) to converge to a locally stable matching.*

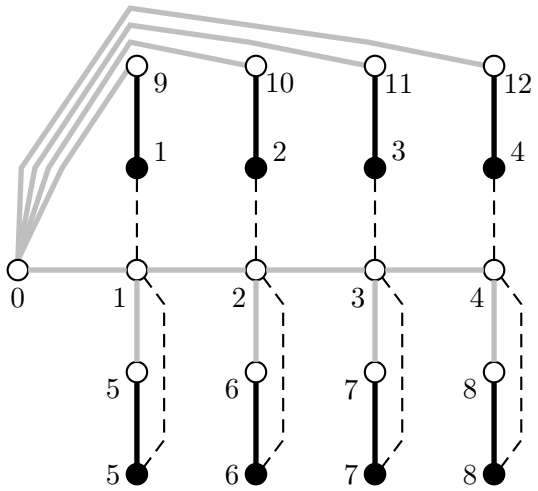

Figure 5: Construction in the lower bound of Theorem 8 for $b = 4$. Filled vertices are firms, empty vertices are workers. Social links are indicated in gray, every possible edge between a worker and a firm can become a matching edge. Bold black edges are "hard-wired" matches never altered during the dynamics. Dashed edges represent setting and resetting of bits. Higher vertex labels at workers indicate higher attractiveness for firms.

Proof. For firm-uniform preferences we consider a structure as follows. We describe the structure for $k = 2$ first. There are $2b$ firms and $3b + 1$ workers. Let the workers $W = \{x_0, x_1, \ldots, x_{3b}\}$ be numbered in increasing preference. The network N is a tree as follows. For every $i = 1, \ldots, b$ there is a link (x_{2b+i}, x_0). In addition, there is a path with links (x_i, x_{i+1}) for $i = 0, \ldots, b-1$. For each worker x_i there is an adjacent worker x_{b+i} connected via edge (x_i, x_{i+b}), for $i = 1, \ldots, b$. Worker x_{2b+i} prefers most to be matched to firm y_i, and for worker x_{b+i} the most preferred firm is y_{b+i}, for all $i = 1, \ldots, b$. These pairs are also the matching edges in our starting state, as well as (x_i, y_{b+i}), i.e., $M = \{(x_{2b+i}, y_i), (x_{b+i}, y_{b+i}), (x_i, y_{b+i}) \mid i = 1, \ldots, b\}$. The preference lists for x_0 to x_b are given as in Table 1. The construction is shown for $b = 4$ in Fig. 5. In the starting state

Worker	Preference List
x_0	$1, 2, \ldots, b, b+1, ...$
x_1	$b, b-1, b-2, \ldots, i, i-1, i-2, i-3, \ldots, 4, 3, 2, 1, b+1, ...$
x_2	$b, b-1, b-2, \ldots, i, i-1, i-2, i-3, \ldots, 4, 3, 2, b+2, 1, ...$
x_3	$b, b-1, b-2, \ldots, i, i-1, i-2, i-3, \ldots, 4, 3, b+3, 2, 1, ...$
x_4	$b, b-1, b-2, \ldots, i, i-1, i-2, i-3, \ldots, 4, b+4, 3, 2, 1, ...$
...	...
x_i	$b, b-1, b-2, \ldots, i, b+i, i-1, i-2, \ldots, 4, 3, 2, 1, ...$
...	...
x_b	$b, 2b, b-1, \ldots, i, i-1, i-2, i-3, \ldots, 4, 3, 2, 1, ...$

Table 1: Preference lists of workers x_i for $i = 0, 1, \ldots, b$. The remaining firms y_j with $j \in [b+1, 2b]$ not explicitly listed are ordered arbitrarily at the end of each list.

workers x_{b+i} for $i = 1, \ldots, 2b$ are matched to their highest ranked firms. Also, they are the most preferred workers for the firms. Hence, there will be no local blocking pair removing any of the edges involving these workers. For best-response dynamics we always prefer local blocking pairs

of the highest ranked workers. Best-response dynamics thus evolve as a bit counter with b bits. Creation of edge (x_i, y_i) corresponds to setting bit i, creation of edge (x_i, y_{b+i}) to resetting of bit i, for $i = 1, \ldots, b$. Thus, in the starting state M all bits are reset.

The dynamics evolve as follows. In the beginning all workers except for worker 0 are matched. He is the only one involved in a blocking pair, namely, in b blocking pairs with firms $1, \ldots, b$. Picking his most preferred choice, we create (x_0, y_1). This means y_1 learns about x_1 and moves to (x_1, y_1). Thus, the first bit is set. In this state x_2 prefers y_{b+2} over y_1, thus, the best local blocking pair is (x_0, y_2). Now we get (x_1, y_2), (thus, removing the setting of bit 1) and (x_2, y_2) (setting bit 2). Again, x_3 still prefers y_{b+3} over y_2. Now the best pair involves x_1, who reconnects with y_{b+1}, thus resetting bit 1. More generally, when bits 1 through i are set, worker x_0 connects to y_{i+1}, which leads to removal of setting of all bits up to i and setting of bit i, as worker $i + 1$ has an interest to remain connected to y_{b+i+1}. Finally, workers i to 1 reconnect iteratively to firms y_i to y_1, thereby resetting the lower bits. As in each step the local blocking pair of largest benefit is unique, the theorem follows also for random best-response dynamics.

For values $k = 3$ and higher we add batches of b highly ranked workers that are initially connected to the firms they like most, such that each firm y_i has $k-1$ matching edges and y_{b+i} has k matching edges in M, for $i = 1, \ldots, b$. This allows to construct the bit counter as before. □

5 Dynamics with Memory in Games with Correlated Preferences

In this section we consider sequential and concurrent better-response dynamics with memory. Our first result is a polynomial-time bound for random memory. Recall that in random memory we assume that, in expectation, every T steps each player v remembers some player u chosen uniformly at random from the set of players v had been matched to before, and u and v become temporarily accessible in the next step.

Theorem 9. *For every $k, \ell \in \mathbb{N}$ and every stable matching game with correlated preferences, in which each player can create up to k matching edges and has lookahead ℓ, (random) best-response dynamics with random memory converge to a locally stable matching from every starting state in (expected) time $O(n^2 \cdot m \cdot k \cdot T)$.*

Proof. The dynamics can rely on the information in the random memory to steer the convergence towards a locally stable matching. Let us consider the dynamics in phases. Phase t begins after the dynamics has created t mutually different matching edges at least once (including the ones in the starting state). Let E_t be the set of edges which have been created at least once when entering phase t. During phase t no new edge is created for the first time. Consider an edge $e \in E_t$ that represents a (global) blocking pair and has maximum benefit. A step in which such an edge is available for creation appears in expectation at most every $n \cdot T$ steps. In such a step, (random) best-response dynamics will establish some (this or some other) edge with maximum benefit from E_t. Such an edge will not be removed in phase t anymore. Thus, by repeating this argument with the remaining edges from E_t, we see that after at most $O(n^2 \cdot k \cdot T)$ steps in expectation either a stable matching has evolved, or phase t has ended by the first creation of a new edge. Note that in total there can be at most m phases, which results in an expected convergence time of at most $O(n^2 \cdot m \cdot k \cdot T)$ for best-response dynamics. □

The result can directly be extended to random and concurrent better-response dynamics.

Corollary 10. *For every $k, \ell \in \mathbb{N}$ and every stable matching game with correlated preferences, in which each player can create up to k matching edges and has lookahead ℓ, random better-response dynamics converge to a locally stable matching from every starting state in expected time $O(n^2 \cdot m^2 \cdot k \cdot T)$, and concurrent better-response dynamics converge in expected time $O(n^4 \cdot m \cdot k \cdot T)$.*

Proof. For random better-response dynamics, we can use essentially the same argument as in Theorem 9. If an edge e corresponding to a blocking pair of currently maximum benefit becomes available in phase t, it is chosen with a probability of at least $\Omega(1/t)$. Hence, the time to discover and resolve a blocking pair of maximum benefit is increased in expectation by at most $O(t)$ over the process in the previous theorem, where we directly resolve blocking pairs with maximum benefit. Note that establishing edges with strictly smaller benefit does not hurt the availability of blocking pairs with maximum benefit. Therefore, the dynamics are simply delayed by a factor t in each phase t in expectation through the creation of lower benefit edges. Thus, the expected time until phase t ends is at most $O(n^2 \cdot k \cdot t \cdot T)$. This means the dynamics take at most $O(n^2 \cdot m^2 \cdot k \cdot T)$ steps to reach a locally stable matching.

For concurrent dynamics, the probability that edge e is chosen is in $\Omega(1/n^2)$. As the process is concurrent, we know that an edge corresponding to a blocking pair of maximum benefit is created after at most $n^3 \cdot T$ steps or stops being a blocking pair. By applying the above argument we can bound the convergence time to $O(n^4 \cdot m \cdot k \cdot T)$. □

These positive results rely on the property that every polynomial number of steps the memory presents the "right" edge for creation. In other words, random memory invokes in each phase a (delayed) execution of a global greedy algorithm that inserts the previously discovered blocking pairs in order of decreasing benefit. This allows to avoid moving through a bit-counter structure of edges with smaller benefit. This property is independent of the structure of N, only the discovery of new edges relies on players being at hop distance ℓ. Thus, the result can be extended to a scenario when N changes over time, which obviously is the case in many social networks when players get more familiar with each other or change the players they interact with over time. We do not have to rely on particular assumptions on the network evolution and can even establish the convergence result under adversarial conditions when an adversary is allowed to manipulate to make convergence time as long as possible.

Corollary 11. *Suppose in each step t, the network N_t among the players is chosen by an adversary. For best-response dynamics with random memory there are at most $O(n^2 \cdot m \cdot k \cdot T)$ steps in expectation between emergence of a new local blocking pair and the first step in which a locally stable matching is reached.*

A similar result as below holds for the better-response dynamics treated in Corollary 10.

Corollary 12. *Suppose in each step t, the network N_t among the players is chosen by an adversary. For random better-response dynamics with random memory there are at most $O(n^2 \cdot m^2 \cdot k \cdot T)$ steps in expectation between emergence of a new local blocking pair and the first step in which a locally stable matching is reached.*

With random memory no previous matching edge can be completely forgotten during the dynamics. Can we allow players to forget some previous matches and still obtain fast convergence with local dynamics? Towards this end, we here consider two natural examples for cache memory and show that delayed forgetting of previous matches can be harmful to convergence. We present a

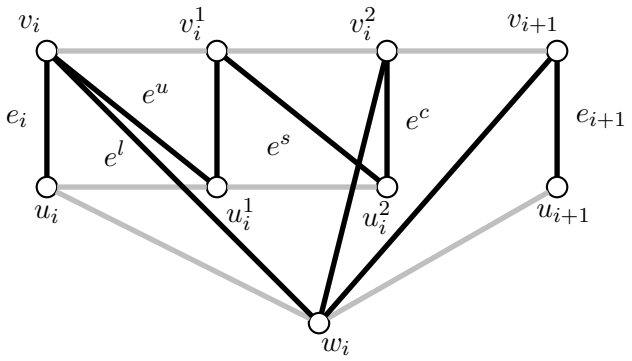

Figure 6: Structure of edge traps in the lower bound of Theorem 14. Social links are indicated in gray, possible matching edges are drawn in black.

lower bound for best-response dynamics with cache memory, largest-benefit and FIFO/LRU eviction strategies. We note that we can easily transform the lower bound constructions below for both caching strategies with our observations above into obtain networks with constant diameter.

Corollary 13. *If each player keeps only the r best matches in his cache, then for every $b \in \mathbb{N}$ there is a stable matching game with correlated preferences and $n \in \Theta(b \cdot r)$, in which best-response dynamics starting from the state $M = \emptyset$ need $\Omega(2^b)$ steps to converge to a locally stable matching.*

Proof. This result is a rather simple corollary of Theorem 2. We first consider $r = 1$. We denote by w an arbitrary player in the edge trap of Fig. 2 and create a side-gadget for every such player. We add three players u_1^w, u_2^w and u_3^w. We add the links $\{u_1^w, u_2^w\}$ and $\{u_2^w, w\}$ and $\{w, u_3^w\}$. In addition, there are two additional matching edges $e' = \{u_1^w, w\}$ and $e'' = \{u_1^w, u_3^w\}$. The benefit $b(e')$ is larger than any benefit of any matching edge in the edge traps, and $b(e'') > b(e')$. Thus, best-response dynamics will first establish edge e', which is recorded in the cache of players w and u_1'. Afterwards, u_1^w and u_3^w switch to their joint edge, which is then updated in their caches. However, the cache of player w still remains set to player u_1^w and will be throughout the dynamics as this is his best incident matching edge. However, this edge remains blocked by e''. In this way, best-response dynamics wrongly initializes the memory of every player in the gadget and the dynamics evolve as before. For larger memory, we simply create more side-gadgets to fill the complete memory with blocked edges before the dynamics evolves as before. □

Theorem 14. *If each player keeps only the r most recent matches in his cache, then for every $b \in \mathbb{N}$ there is a stable matching game with correlated preferences and $n \in \Theta(b^2 \cdot r)$, in which best-response dynamics starting from the state $M = \emptyset$ need $\Omega(2^b)$ steps to converge to a locally stable matching.*

Proof. In our lower bound construction we slightly adjust the edge trap structure from Fig. 2, see Fig. 6 and 7. As previously, we attach b of these edge trap structures sequentially to create a *gadget*. The idea is to create $15br$ gadgets, each of them representing a bit counter with b bits. The edge weights are chosen such that all gadgets are updated simultaneously in a round-robin fashion. The gadgets still have the critical invariant that the creation of a counter edge e^c corresponding to bit i in a gadget is preceded by the movement of the edge through the gadget and deletion of all trapped edges e^c corresponding to lower bits. At this point we now must also reset the *memory*

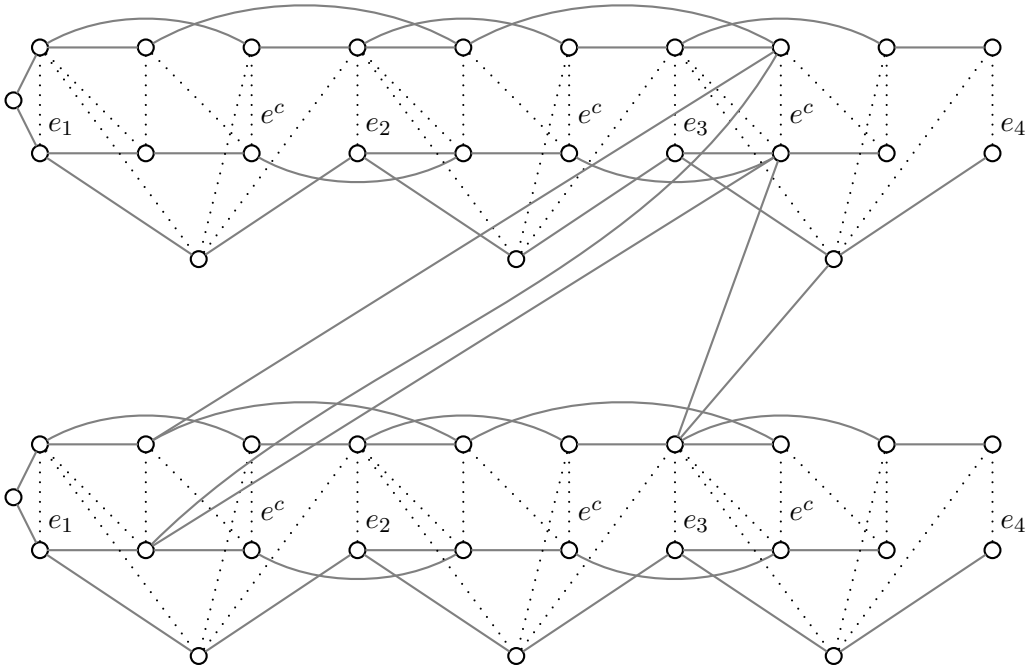

Figure 7: Gadget structure in the lower bound of Theorem 14. We depict two gadgets (α, β) and $(\alpha + 3, \beta + 1)$. Social links are indicated by solid edges. Dotted edges are profitable matching edges among the vertices shown. Vertices from other gadgets can successfully match to the vertices depicted here in the process of resetting memory.

of all vertices in the edge traps for lower bits. This is to ensure that the invariant continues to hold and edges within the gadgets are not created from memory in subsequent steps. To reset the memory of vertices, we must attach and detach each vertex sequentially to at least r other vertices, none of which prove useful in later steps for speeding up the bit-counter dynamics in the gadget.

Formally, we create an array of $15br$ gadgets, each consisting of b sequentially attached edge traps. In each gadget (α, β) with $\alpha = 1, \ldots, b$, $\beta = 1, \ldots, 15r$, the edge between vertices w_i and w_i' is the i-th counter edge. If the edge e^s is created in each gadget, we use the vertices u_i^2 to reset the memory in other gadgets before creating the i-th counter edge e^c. In particular, we let the u_i^2 vertices iteratively create and delete edges to vertices in other gadgets until the memory of all players in the edge traps corresponding to lower bits is filled with vertices u_i^2 from other gadgets. In the end of this process this leads to creation of all the bit-counter edges between u_i^2 and v_i^2 in every gadget. Thus, every memory entry in the edge traps refers to vertices u_i^2 that are matched with higher benefit in a different gadget. This implies that all memory entries are useless and the dynamics unfolds similarly as before in all gadgets. As mentioned earlier, the edge benefits are chosen such that the dynamics essentially evolve simultaneously in all gadgets, i.e., the best-response dynamics is unique and implements a step-by-step update in a round-robin fashion.

To carry out our memory reset, we have to construct social links and matching edges between the gadgets. We describe the construction for a generic gadget (α, β), and our numbering is to be understood in a cyclic fashion (modulo b and $15r$, respectively). We delete the link between v_i^2 and v_i^1. Instead we add a link to vertex w_i in gadget $(\alpha + i, \beta + 4r + 1)$. Hence, before best-

response dynamics can create the i-th bit-counter edge e^c in gadget (α, β), u_i^2 is re-matched to w_i in gadget $(\alpha + i, \beta + 4r + 1)$. Then u_i^2 "sweeps" part of the other gadget, i.e., it creates and deletes edges one by one along the lower path $w_i, u_i, w_{i-1}, u_{i-1}, ...$ in this gadget to the leftmost dummy vertex, and then further along the top path $v_1, v_1^2, v_2, v_2^2, ...$ to vertex v_i. Each of these matching edges has more benefit than the previous one. From v_i, there is a social link to w_i in gadget $(\alpha + 2i, \beta + 4r + 2)$. Then, u_i^2 from gadget (α, β) switches his matching edge along the same path in gadget $(\alpha + 2i, \beta + 4r + 2)$, $(\alpha + 3i, \beta + 4r + 3)$ and so on until he finishes sweeping gadget $(\alpha + ri, \beta + 5r)$. Now u_i^2 is in the memory of all the vertices along these paths in gadgets $(\alpha + i, \beta + 4r + 1), \ldots, (\alpha + ri, \beta + 5r)$. Similarly, we can apply this construction in a cyclic fashion and set the benefits of edges such that best-response dynamics advances the sweeps of vertices u_i^2 from all gadgets in a round-robin fashion. Then each gadget has been covered by sweeps of r vertices u_i^2 from other gadgets and the memories of the involved vertices are fully reset.

However, after this step there are still vertices whose memory is not yet reset. In particular, from vertex v_i in gadget $(\alpha+ri, \beta+5r)$ there is a social link to u_i^1 in gadget $(\alpha+(r+1)i, \beta+5r+1)$. In addition, there is a social link in this gadget between u_i^1 and u_{i-1}^2, for all $i = 2, \ldots, b$. Note that these links do not change the exponential dynamics within the gadget. However, they allow u_i^2 from gadget (α, β) to sweep through u_i^1, u_{i-1}^2, u_{i-1}^1, and so on. Finally, from u_1^1 in gadget $(\alpha + (r + 1)i, \beta + 5r + 1)$ there is a connection to u_i^1 in gadget $(\alpha + (r + 2)i, \beta + 5r + 2)$, in which the same construction is applied. In total, u_i^2 from gadget (α, β) sweeps through gadgets $(\alpha + (r + 1)i, \beta + 5r + 1), \ldots, (\alpha + (2r)i, \beta + 6r)$ in this fashion. Again, each vertex in a gadget becomes attached and detached to exactly r vertices u_i^2 from other gadgets in this procedure, and thus the memories are reset.

Finally, we apply a similar idea for the remaining vertices v_i^1 in the upper parts of the gadgets. Again, we introduce social links from v_i^1 to v_{i-1}^1 in every gadget (α, β) and for every $i = 2, \ldots, b$. Note again that this does not change any of the arguments made above, as profitable matching edges exist again only for selected vertices u_i^2 from other gadgets. In particular, we let our player u_i^2 from gadget (α, β) move his matching edge from v_1^1 in gadget $\beta+6r$ to v_i^1 in gadget $(\alpha+(2r+1)i, \beta+6r+1)$ with the help of a social link between those vertices. Then u_i^2 sweeps the v^1-vertices in gadget $(\alpha + (2r + 1)i, \beta + 6r + 1)$ until he hits v_1^1, continues in gadget $(\alpha + (2r + 2)i, \beta + 6r + 2)$ and so on. Finally, from v_1^1 in gadget $(\alpha + (3r)i, \beta + 7r)$ there is a social link to vertex v_1^2 in gadget (α, β). Hence, at this point the i-th bit-counter edge e^c in gadget (α, β) is finally created.

Verify that at this point, due to the round-robin update of best-response dynamics, in all gadgets all the memories of all vertices in all edge traps for bits $1, ..., i$ have been reset. The only exceptions are u_i^2 and v_i^2 vertices, but these vertices are matched to each other via an edge of higher benefit. Hence, after bit i has been set in all gadgets, the dynamics evolves as before as none of the players in the memories of all vertices in the edge traps of smaller bits is part of any local blocking pair.

For showing correctness, we have to verify that no undesired shortcuts are created in our cyclic construction of the network links. There are two points, at which this problem can occur and where a sufficient spread in our construction is critical. The first point is when vertices from one gadget reset memory in another gadget and vice versa. Instead of sweeping along a path in a gadget, a player u_i^2 might get the opportunity to match to a vertex at the end of the sweep more quickly. Observe, however, that each gadget (α, β) has only links to gadgets $(\alpha + i, \beta + 1)$ and $(\alpha + i, \beta + 4r + 1)$, for all $i = 1, \ldots, b$. Due to the size of our array of $15r$ in the second dimension, there are no two gadgets (α, β) and (γ, δ), where vertices of (α, β) reset the memory in gadget (γ, δ) and vertices from (γ, δ) reset memory in (α, β). The second point where the problem can arise is

when two vertices u_i^2 and u_{i+j}^2 from gadget (γ, δ) sweep the same two gadgets consecutively, say, (α, β) and $(\alpha, \beta + 1)$. In this case, u_{i+j}^2 sweeps more traps than u_i^2. He could use links introduced for the sweep of u_i^2 to avoid sweeping the traps for higher bits and directly enter gadget $(\alpha, \beta + 1)$ at a trap corresponding to a lower bit. However, we assume players u_i^2 and u_{i+j}^2 continue their sweep in gadgets $(\alpha + i, \beta + 1)$ and $(\alpha + i + j, \beta + 1)$, respectively. This allows none of the players to use social ties introduced for other players to shortcut their sweep.

In conclusion, in our construction all shortcuts can be avoided and the dynamics evolve by incrementing all bit-counters in round-robin fashion as desired. □

6 Open Problems

Our work opens up a variety of fascinating issues for further research. Some immediate open problems remain for the special case of the job-market model with $k \geq 2$. For general preference lists, it is simple to extend the lower bound example from [2] to show exponential convergence time with high probability even for random better-response dynamics. Can these dynamics achieve polynomial convergence time for correlated preferences? For dynamics with memory it would be interesting to see if there are forms of memory that allow players to forget about some of their matches and still obtain a good convergence time. What about the convergence time of, say, best-response dynamics and random cache updates? While for this particular combination an exponential lower bound might be obtained along the lines of Theorem 14, nothing is known about other random dynamics and cache eviction strategies. In addition, it would be interesting to see how size, fading, or heterogeneity of memory influences the convergence time of these dynamics.

Acknowledgments

The author would like to thank Siddharth Suri and Heiko Röglin for numerous insightful discussions on the topic. This research is supported by DFG grant Ho 3831/3-1.

References

[1] David Abraham, Ariel Levavi, David Manlove, and Gregg O'Malley. The stable roommates problem with globally ranked pairs. *Internet Math.*, 5(4):493–515, 2008.

[2] Heiner Ackermann, Paul Goldberg, Vahab Mirrokni, Heiko Röglin, and Berthold Vöcking. Uncoordinated two-sided matching markets. *SIAM J. Comput.*, 40(1):92–106, 2011.

[3] Kemal Akkaya, Ismail Guneydas, and Ali Bicak. Autonomous actor positioning in wireless sensor and actor networks using stable-matching. *Intl. J. Parallel, Emergent and Distrib. Syst.*, 25(6):439–464, 2010.

[4] Elliot Anshelevich and Martin Hoefer. Contribution games in networks. *Algorithmica*, 63(1–2):51–90, 2012.

[5] Esteban Arcaute and Sergei Vassilvitskii. Social networks and stable matchings in the job market. In *Proc. 5th Intl. Workshop Internet & Network Economics (WINE)*, pages 220–231, 2009.

[6] Federico Echenique and Jorge Oviedo. A theory of stability in many-to-many matching markets. *Theoretical Economics*, 1(2):233–273, 2006.

[7] Kimmo Eriksson and Olle Häggström. Instability of matchings in decentralized markets with various preference structures. *Int. J. Game Theory*, 36(3–4):409–420, 2008.

[8] Tomás Feder, Nimrod Megiddo, and Serge Plotkin. A sublinear parallel algorithm for stable matching. *Theoret. Comput. Sci.*, 233(1–2):297–308, 2000.

[9] Tamás Fleiner. A fixed-point approach to stable matchings and some applications. *Math. Oper. Res.*, 28(1):103–126, 2003.

[10] Patrik Floréen, Petteri Kaski, Valentin Polishchuk, and Jukka Suomela. Almost stable matchings by truncating the Gale-Shapley algorithm. *Algorithmica*, 58(1):102–118, 2010.

[11] David Gale and Lloyd Shapley. College admissions and the stability of marriage. *Amer. Math. Monthly*, 69(1):9–15, 1962.

[12] Michel Goemans, Li Li, Vahab Mirrokni, and Marina Thottan. Market sharing games applied to content distribution in ad-hoc networks. *IEEE J. Sel. Area Comm.*, 24(5):1020–1033, 2006.

[13] Dan Gusfield and Robert Irving. *The Stable Marriage Problem: Structure and Algorithms*. MIT Press, 1989.

[14] Alexander Kelso and Vincent Crawford. Job matchings, coalition formation and gross substitute. *Econometrica*, 50(1483–1504), 1982.

[15] Samir Khuller, Stephen Mitchell, and Vijay Vazirani. On-line algorithms for weighted bipartite matching and stable marriages. *Theoret. Comput. Sci.*, 127(2):255–267, 1994.

[16] Alexis Kipnis and Boaz Patt-Shamir. On the complexity of distributed stable matching with small messages. *Distributed Computing*, 23(3):151–161, 2010.

[17] Donald Knuth. *Marriages stables et leurs relations avec d'autres problemes combinatoires*. Les Presses de l'Université de Montréal, 1976.

[18] Fabien Mathieu. Self-stabilization in preference-based systems. *Peer-to-Peer Netw. Appl.*, 1(2):104–121, 2008.

[19] Alvin Roth, Tayfun Sönmezc, and M. Utku Ünverd. Pairwise kidney exchange. *J. Econ. Theory*, 125(2):151–188, 2005.

[20] Alvin Roth and Marilda Oliveira Sotomayor. *Two-sided Matching: A study in game-theoretic modeling and analysis*. Cambridge University Press, 1990.

[21] Alvin Roth and John Vande Vate. Random paths to stability in two-sided matching. *Econometrica*, 58(6):1475–1480, 1990.