Sequences in Pairing Problems

A new approach to reconcile stability with strategy-proofness for elementary matching problems

Jens Gudmundsson
Department of Economics, Lund University, Box 7082, SE-222 07 Lund, Sweden

Abstract
Are any of the negative results on pairing problems overturned when the pairing of agents is allowed to change systematically over time? By alternating between pairings we can “balance” the interest of the agents and thereby provide improvements in terms of welfare and fairness, but can we also pair agents in a stable way and ensure they never lie about their preferences? To answer these questions, we introduce sequences, lists of matchings that are repeated over time, in two-sided (“marriage”) and general pairing (“roommate”) problems.

An agent compares sequences by dominance in terms of successive sums of ordered frequencies (ssd): she prefers the sequence $\Sigma$ to $\Sigma'$ if she is matched more frequently with her most preferred agent in $\Sigma$ than in $\Sigma'$; more frequently with her two most preferred agents in $\Sigma$ than in $\Sigma'$; and so on. Stable sequences are natural extensions of stable matchings; case in point, we show that a sequence of stable matchings is ssd-stable. In addition, there are general pairing problems that have stable sequences but no stable matching.

No strategy-proof rule always selects stable matchings (Roth, 1982). In contrast, we design a weakly group ssd–strategy-proof rule that selects ssd-stable sequences. We call the rule Promises and Rewards, or CR. In contrast to the Deferred Acceptance rule, CR treats the two sides symmetrically. We say it is side-neutral. For general pairing problems, Generalized CR is ssd–5-stable (cannot be blocked by groups of five or fewer), weakly ssd–strategy-proof, and anonymous. We also show that our positive results are the most one can hope for.

Keywords: Pairing problems, sequences, stability, strategy-proofness, algorithms
JEL: C62, D02, D60

Footnote:
I am especially thankful to Tommy Andersson, Albin Erlanson, and Vikram Manjunath for many helpful discussions. Comments by Lars Ehlers, Ehud Kalai, Thayer Morrill, Michael Ostrovsky, David C. Parkes, Alexander Reffgen, Alexander Teytelboym, William Thomson, M. Utku Ünver, Rajiv Vohra are gratefully appreciated. The paper has also benefited from comments made by participants at seminars at University of Rochester and Lund University, as well as participants at the Workshop on Coalitions and Networks (Montreal) and the meeting of Social Choice and Welfare (Boston). Part of the paper was written while I was visiting Boston College and University of Rochester. I thank the respective economics departments for their hospitality. Remaining errors are my own. Financial support from the Jan Wallander and Tom Hedelius Foundation is gratefully acknowledged.

February 13, 2015
1. Introduction

Are any of the negative results on pairing problems overturned when the pairing of agents is allowed to change systematically over time? By alternating between pairings we can “balance” the interest of the agents and thereby provide improvements in terms of welfare and fairness, but can we also pair agents in a stable way and ensure they never lie about their preferences? To answer these questions, we introduce sequences, lists of matchings that are repeated over time. The main finding of the paper is a non-manipulable rule that selects stable sequences.

Our starting point is that sometimes the solution to a matching problem is not just one particular matching. Consider for instance the following challenge facing physicians worldwide. A doctor’s emotional detachment from her patient has long been considered a necessity to prevent a loss of objectivity and perspective in his treatment \cite{Blumgart1964}. Case in point, the American College of Physicians \cite{Snyder2012} makes the following recommendation:

*Physicians should usually not enter into the dual relationship of physician-family member or physician-friend for a variety of reasons. The patient may be at risk of receiving inferior care from the physician. Problems may include effects on clinical objectivity, inadequate history taking or physical examination, overtesting, inappropriate prescribing, incomplete counseling on sensitive issues, or failure to keep appropriate medical records.*

Arguably, remaining detached gets more difficult over time as the patient and doctor get increasingly familiar. The two connect for instance through small talk during visits and, in some cases, interactions on social media websites \cite{Bossletetal2011}. However, the entire issue is likely less severe if any subpar treatment is quickly detected. For this purpose, consider an arrangement in which a patient primarily sees a “main” doctor but occasionally gets a second opinion from some “reserves.” In our context, this is modelled as a sequence consisting of an oft-occurring default matching that at times is swapped for “check-up” matchings. In this way, doctors monitor each other, making sure the treatment never deteriorates too far.

Imagine next being the owner of some shops that are operated by pairs of employees, say by a chef and a cashier. Surely, the employees need to be comfortable working alongside each other, but this can lead to a negative shirking effect. Essentially, if the perceived likelihood of one getting caught shirking is smaller, one rationally provides less effort (see for instance \cite{Shapirotostiglitz1984} page 439). As the owner, you may therefore wish to construct a varied schedule for your employees to maintain a good rotation. The “stability” of this schedule depends crucially on how you weigh “compromises and rewards” for the agents. You cannot

\footnote{It is reasonable to believe that the better friends you are with your coworker, the lower the risk that she tells on you. In this way, friends can more easily collude to promote their own interest ahead of that of the firm \cite{Trole1986, LaffontTrole1993}. We do acknowledge that friendship can have the opposite effect. If two individuals have to paint a fence, then one shirking implies more work for the other. In this case, friends may be less inclined to shirk. See also \cite{MasMoretti2009}.}
always pair an employee with someone she dislikes (have her “compromise”) as she may then seek employment elsewhere. However, you may be forced to occasionally match her to someone she dislikes. If you do so, you should make sure she is “rewarded” for this by sometimes matching her with someone she likes better. Keep in mind also that a reward for one agent may be a compromise for the partnering agent. Thus, designing a functional schedule is both challenging and important.

Finally, consider the employment of lifeguards at public swimming pools. There often needs to be more than one guard overseeing the pool at all times to ensure swimmers’ safety. Let us restrict attention to pools for which using more than two guards can be ruled out as unnecessary (alternatively, to pool complexes where pairs are responsible of keeping watch over smaller areas). The pairing frequently has to change to prevent the guards from becoming inattentive and “too comfortable” with one another. The American Red Cross (2012, page 48) suggests that lifeguards should rotate stations every 20 to 30 minutes. When constructing the lifeguard schedule, you have to take into account the guards’ preferences: where one sees a relaxed co-worker, another sees a lazy no-good slacker. If some guards keep getting paired unfavorably, they may decide to take their talents elsewhere. For instance, they may apply at a competitor, or even start their own lifeguard venture.

In this paper, we look for sequences from which no agents can benefit from deviating, that is, stable sequences. Besides being desirable solutions to many problems, sequences are interesting to study as they have several nice properties. For instance, a “well-balanced” sequence of matchings can Pareto-dominate a stable matching (Example 1). As just eluded to, sequences allow us to promote different agents at different matchings. Thus, they can provide a more fair pairing over time than any one particular matching (Example 2). Finally, an important difference to matchings is that there are non-manipulable rules that select stable sequences (Example 3).

We analyze two-sided and general pairing problems. Hence, there is a set of agents and we wish to pair them. For the two-sided problem, the agents are divided into two groups and matchings are restricted to be of agents from different groups. Each agent has a preference over whom she is matched to and compares sequences using dominance in terms of successive sums of frequencies (ssd). That is, an agent finds the sequence \( \Sigma \) at least as desirable as the sequence \( \Sigma' \) if she is matched at least as frequently with her most preferred agent in \( \Sigma \) than in \( \Sigma' \); at least as frequently with her two most preferred agents in \( \Sigma \) than in \( \Sigma' \); and so on. There is no money in the model: hence, one cannot trade-off a bad schedule for a pay rise.

---

2 See for instance South Carolina state law act number 159 of year 2012. Alternatively, the Connecticut Office of Legislative Research (2012) presents a survey on http://www.cga.ct.gov/2012/rpt/2012-R-0524.htm. Griffiths (2002) notes that “Monotony leads to boredom, which, in turn, leads to a lack of vigilance, one of the biggest problems in lifeguarding today.”

3 These problems are usually referred to as the “marriage” and the “roommate” problem. We find that the word “marriage” has connotations irrelevant for most real-world situations that can be modelled as “marriage” problems. It is certainly controversial to talk about “sequences” in marriage. Therefore, we choose a neutral name.

4 The two-sided pairing problem with money is the “assignment game” of Shapley and Shubik (1971). The general version is the “partnership formation problem” studied by Talman and Yang (2011), Alkan and Tuncay (2014), and Andersson et al. (2014a,b) among others. Similar to this paper, using compromises to resolve instability in
The agents report their preferences to a central authority which selects a sequence (think of the employer proposing a schedule to her employees). We design desirable rules for making this selection.

The appeal of our rules is intimately connected to the axioms they satisfy. Therefore, perhaps in abundance, we spend some time on explaining the importance of stability and strategy-proofness. At the core of most successful applications of matching theory to real-world problems is the insight nicely summarized by Roth (2002). Namely, rules that select stable matchings tend to stay in use year in, year out, whereas others do not. To see why, consider a procedure that yields an unstable matching. By definition, there are agents who can do better on their own, circumventing the procedure. Even if the procedure has nice properties in terms of, say, efficiency and fairness, these properties will immediately be harmed if some agents do not participate. Even more troubling is that once it becomes clear that one can benefit from bypassing the official system, others may follow suit – possibly leading to a breakdown of the entire procedure.

Thus, a lack of stability can turn an otherwise well-designed rule useless in practice. The property’s appeal can however be in question if it does not come bundled with the following characteristic. As the preferences are not available to us, the agents must themselves provide them. It is fundamental for a stable rule that the reported preferences are the true ones. Otherwise, the rule selects an outcome stable with respect to the wrong preferences, but perhaps unstable with respect to the true ones. Our rules should be strategy-proof: no agent should ever benefit from lying about her preference.

Recall, Roth (1982) shows that no strategy-proof rule always selects stable matchings. Thus, the properties we just argued are essential are incompatible. As a second-best solution, much research has since been focused on the Deferred Acceptance rule of Gale and Shapley (1962), DA for short. This rule’s success is rooted in breaking the symmetry between the two sides of agents, making one side “propose” to the other. The rule is manipulable, but

---

6Stability is also essential in decentralized settings where agents interact repeatedly. Once a stable matching is reached, myopic agents have no incentives to deviate from it. Therefore, it is interesting to know whether individual agents, acting in their own self-interest, can coordinate to reach a stable matching (Roth and Vande Vate, 1990; Abeledo and Rothblum, 1995; Diamantoudi et al., 2004; Inarra et al., 2008). For future research, we may ask whether agents that make plans over a longer time horizon can coordinate to reach a stable sequence. 7See the literature on kidney exchange (Roth et al., 2004), school choice (Abdulkadiroğlu and Sönmez, 2003), and the National Resident Matching Program (Roth and Peranson, 1999). Some surveys of the literature are Roth and Sotomayor (1990), Sönmez and Unver (2010), and Manlove (2013).

8This “unraveling” in matching markets is examined by Niederle and Roth (2003) and Ostrovsky and Schwarz (2010) among others.

9Stability can also be viewed as a fairness property. What a group of agents can achieve on their own ought to be a lower bound on what they are assigned. An outcome is stable precisely when all lower bounds are met.

10Strategy-proofness can also be viewed as a fairness property. As some agents may be more strategic than others, strategy-proofness can “level the playing field” for sincere (non-strategic) and sophisticated (strategic) agents (Pathak and Sönmez, 2008).

11Positive results have been found on restricted preference domains as these limit the possibilities for manipulation, see for instance Alcalde and Barberà (1994) and Akahoshi (2014).
only by agents receiving proposals.\footnote{\textit{A strengthening of} Roth's (1982) result is that strategy-proof rules either are inefficient or not individually rational. Alcalde and Barberà (1994). The DA rule is in a sense a compromise. It allows for a limited amount of manipulation in exchange for efficiency and some fairness.} It selects stable matchings, but favors the proposers in this selection. This may at first sight seem unfair: the choice of proposing agents induces a potentially significant welfare gap between the two sides. However, remember that strategy-proofness is a necessity for any type of fairness. The outcome selected by a manipulable rule designed to be fair may, if agents frequently misreport their preferences, not be very fair. Therefore, we have had to rely on rules that prioritize incentive constraints over fairness in this way.

There are similar incompatibilities for rules that select sequences. If we insist that the rule always selects a sequence of stable matchings, then we have to give up on the very weakest incentive properties. Taking the opposite role, if we require an \textit{ssd–strategy-proof} rule, then we have to give up on the very weakest stability properties. This is summarized in Theorem\footnote{All properties are defined formally in Subsection 2.4.} 1. However, our results also indicate that sequences can provide a different point of view. In Theorem\footnote{A rule is \textit{ssd–k–stable} if no group of at most \textit{k} agents can block its suggested sequence.} 2 we present an \textit{ssd-stable}, \textit{weakly group ssd–strategy-proof}, and \textit{side-neutral} rule for selecting sequences for the two-sided problem.\footnote{Strategy-proof “single-lapping rules” have been studied for coalition formation problems on restricted do-} Hence, not only can we combine stability with strategy-proofness, we can strengthen the latter to protect against collusive behaviour \textit{and} we can do this without treating the agents on the two sides asymmetrically. The rule we develop is the \textit{Compromises and Rewards} rule, \textit{CR} for short. It is based in a novel way on David Gale's \textit{Top Trading Cycles} algorithm \cite{Shapley1974}. We also find that a \textit{weakly ssd–strategy-proof} rule occasionally selects a sequence that contains unstable matchings – even though there always are stable matchings. For the general problem, we first illustrate how stable sequences are natural extensions of stable matchings. More precisely, in Theorem\footnote{Results of a similar nature to Theorem 3 have been found in other papers; there are many} 3 we establish that a sequence of stable matchings is stable. This can be especially useful if we wish to achieve a “fair” pairing as the welfare of the agents may differ significantly at different stable matchings. We then construct an intuitive extension of the \textit{DA} rule. For the original \textit{DA} rule, the two sides of agents are essential. Our extension is handcrafted for sequences and does not require the agents to be divided. In Theorem\footnote{Strategy-proof “single-lapping rules” have been studied for coalition formation problems on restricted do-} 4 we show that the rule is \textit{ssd-stable}, \textit{anonymous}, and \textit{individually rational at all times} on a restricted domain of problems. The domain strictly subsumes the domain of problems that have stable matchings. In Theorem\footnote{Strategy-proof “single-lapping rules” have been studied for coalition formation problems on restricted do-} 5 we show that the \textit{Generalized CR} rule is \textit{ssd–5-stable}, \textit{weakly ssd–strategy-proof}, and \textit{anonymous}. We conjecture that it is (fully) \textit{ssd-stable} and \textit{weakly group ssd–strategy-proof}.

In Section\footnote{Strategy-proof “single-lapping rules” have been studied for coalition formation problems on restricted do-} 7 we describe how all results can be generalized to a larger domain of preferences. More precisely, we provide a condition under which our results still hold if agents have cardinal preferences and compare sequences using “expected utility.” The condition has a natural interpretation and the preferences are complete (in contrast to the ssd-relation). In relation to the existing literature, this is the first paper to examine strategy-proofness specifically for the general pairing problem.\footnote{Strategy-proof “single-lapping rules” have been studied for coalition formation problems on restricted do-}
solution concepts that extend stable matchings. Ours generally does not pinpoint one matching, but rather a group of them. Different matchings need not occur with the same frequency in a sequence. In this way, stable sequences give a more precise prediction than a set-valued solution concept like von Neumann-Morgenstern stable sets (von Neumann and Morgenstern[1947] Klaus et al.[2011]) or absorbing sets (Inarra et al.[2010]) (for these, all matchings are attributed equal importance). However, they are not as “exact” as a farsightedly stable or a stochastically stable matching (see e.g. Klaus et al.[2010]). It does retain a sense of “full stability,” in contrast to “almost stable” (Abraham et al.[2006]) and “maximum stable” matchings (Tan[1990]) which rather are focused on finding the least unstable matchings.  

We mention also that we can interpret a sequence as a lottery over matchings. This opens up the scope of applications even further. For instance, to decide on which students to accept to a course, a lottery can be used to break potential ties. A similar type of random tie breaking is used in some kidney exchange mechanisms (Ashlagi et al.[2013]).

The paper is structured as follows. In Section 2, we present the model. In Section 3, we provide some motivating examples. In Section 4, we present results for two-sided problems. In Section 5, we examine many-to-one problems. In Section 6, we present results for general problem. In Section 7, we discuss fractional and probabilistic matchings as well as a generalization of the preference domain. In Section 8, we conclude. Proofs, auxiliary results, and additional details are postponed to the Appendix.

2. Model and definitions

2.1. Preliminaries

There are n agents N divided into sets M and W of equal size N = M ∪ W, M ∩ W = φ. A matching is μ: N → N such that μ(i) = j ⇐ μ(j) = i for each {i, j} ⊆ N. Matchings are restricted to be between agents of different groups. That is, for each m ∈ M, μ(m) ∈ W ∪ {m}, and for each w ∈ W, μ(w) ∈ M ∪ {w}. If μ(i) = i for some i ∈ N, then i is single at μ. With some abuse of notation, for each S ⊆ N, μ(S) denotes the set of partners of agents in S at μ: μ(S) = {μ(i) : i ∈ S}. We often describe a matching by its graph, that is, μ = {(i, j), (k, m), …} is short for μ(i) = j, μ(k) = m, and so on. The set of matchings is M. A preference for i ∈ N is the binary relation R_i on N such that i finds j at least as desirable as k whenever j R_i k. Preferences are strict; formally, if j R_i k and k R_i j, then j = k. The strict relation is denoted P_i. We do not rule out that i P_i j, that is, matching with some agent j may be less desirable to i than being single. In addition, M-agents prefer W-agents to other M-agents and vice versa.

16 General pairing problems that allow stable matchings are characterized by Tan[1991]. The problem with weak preferences (allowing for indifferences) is considered by Gudmundson[2014].

17 This is for instance the case in the Swedish admission system, see https://www.antagning.se/sv/Ta-reda-pa-mer-/Platsfordelning-och-urval/Vid-lika-meritvarde/[in Swedish].

18 This is without loss of generality. If M and W are not of the same size, we can extend the smaller of them with “null agents” who prefer being single. This only comes into play when defining side-neutrality. It does not imply that every agent will be matched to someone.
Formally, for each \( m \in M, m' \in M \setminus \{ m \}, w \in W, \) and \( w' \in W \setminus \{ w \}, w P_m m' \) and \( m P_w w' \). The set of preferences is denoted \( \mathcal{R} \). A profile of preferences is \( R \equiv (R_i)_{i \in N} \in \mathcal{R}^n \). A two-sided pairing problem, or simply a two-sided problem, is completely described by \( R \in \mathcal{R}^n \).

A group of agents block a matching if they can pair up in a way that makes everyone in the group at least as well off and someone better off. Thus, \( S \subseteq N \) blocks \( \mu \in \mathcal{M} \) if there is \( \mu' \in \mathcal{M} \) such that \( \mu'(S) = S \), for each \( i \in S \), \( \mu'(i) \, R \, \mu(i) \), and for some \( j \in S \), \( \mu'(j) \, P \, \mu(j) \). A matching is stable if no group blocks it. For \( R \in \mathcal{R}^n \), the set of stable matchings is \( \mathcal{E}(R) \). The set \( \mathcal{E}(R) \) has a lattice structure with respect to \( R \) with an \( M \)-optimal matching \( \mu_R^M \) and a \( W \)-optimal matching \( \mu_R^W \) [Gale and Shapley, 1962]. That is, for each \( \mu \in \mathcal{E}(R) \), each \( m \in M \), and each \( w \in W \), \( \mu_R^M(m) R_m \mu(m) \) and \( \mu_R^W(w) R_w \mu(w) \). A matching is Pareto-efficient, or simply efficient, if no other matching makes everyone at least as well off and someone better off. Equivalently, \( \mu \in \mathcal{M} \) is efficient if \( N \) does not block \( \mu \).

To formalize our algorithms in Sections 4 and 6 we require some additional definitions. Consider a directed graph \((V, E)\), where \( V \) is its vertex set and \( E \) its edge set. A cycle in \((V, E)\) is a list of vertices such that each vertex has an outgoing edge to its succeeding vertex. The vertex that succeeds the last vertex is the first vertex. For technical reasons, we describe a cycle by putting its “top-ranked” agent first. This is formalized by an injective function \( r : N \rightarrow \mathbb{N} \); throughout, \( r \) is fixed (think of \( r \) as an exogenously given tie-breaker). Thus, a cycle of length \( m \) in \((V, E)\) is \( C = (c_1, c_2, \ldots, c_m) \) such that, for each \( c_k \in C \), we have \( c_k \in V \), \((c_k, c_{k+1}) \in E \mod m \), and \( r(c_1) \geq r(c_k) \). A loop is a cycle of length 1.

2.2. Sequences

A sequence of matchings, a sequence for short, is a finite list \([\mu_1, \mu_2, \ldots]\) such that \( \mu_i \in \mathcal{M} \) for each \( i \). Denote the set of sequences \( \mathcal{S} \). Each \( \Sigma \in \mathcal{S} \) induces a symmetric \( n \times n \) matrix \( \sigma \), where, for each \( \{ i, j \} \subseteq N \), \( \sigma_{ij} \in [0, 1] \) is the frequency at which \( i \) is matched to \( j \) in \( \Sigma \). For instance, if \( \Sigma = \{\mu_1, \mu_2, \mu_3\} \), \( \mu_1(i) = \mu_3(i) = j \), and \( \mu_2(i) \neq j \), then \( \sigma_{ij} = 3/4 \).

Sequences that induce the same matrices are equivalent and viewed as equal by the agents. This rules out time discounting: the sequences \([\mu_1, \mu_2], [\mu_1, \mu_2, \mu_2] \), and \([\mu_1, \mu_2, \mu_1, \mu_2] \) are for instance equivalent. In addition, there are no matching externalities [21] that agents do not discount over time is a sensible approximation if the agents are patient and the sequences are relatively short compared to the full time horizon (which here is infinite as sequences are repeated indefinitely). We denote by \#\Sigma the length of the shortest sequence equivalent to \( \Sigma \).
Agent $i$’s **preference over sequences** is the binary relation $R_{i}^{ssd}$ on $\mathcal{S}$ induced by $R_i$. For each $\Sigma \in \mathcal{S}$ associated with $\sigma$ and each $\Sigma'\in \mathcal{S}$ associated with $\sigma'$,

$$\Sigma R_{i}^{ssd} \Sigma' \iff \forall k \in N, \sum_{j \in R_i k} \sigma_{i j} \geq \sum_{j \in R_i k} \sigma'_{i j} \footnote{The notation $\sum_{j \in R_i k}$ is short for summing over the set $\{j \in N : j \in R_i k\}$.}$$

In words, agent $i$ finds $\Sigma \in \mathcal{S}$ at least as desirable as $\Sigma' \in \mathcal{S}$ if $i$ is matched at least as frequently with $i$’s most preferred agent in $\Sigma$ as in $\Sigma'$; at least as frequently with $i$’s two most preferred agents in $\Sigma$ as in $\Sigma'$; and so on. The strict relation is $R_{i}^{ssd}$ and the indifference relation is $I_{i}^{ssd}$. Note that $R_{i}^{ssd}$ is an incomplete relation: there are sequences the agent cannot compare.$\footnote{Let $2 P_1 3 P_1 1, \mu_1(1) = 2, \mu_2(1) = 1, \mu'(1) = 3, \Sigma = [\mu_1, \mu_2],$ and $\Sigma' = [\mu']$. Neither $\Sigma R_{i}^{ssd} \Sigma'$ nor $\Sigma' R_{i}^{ssd} \Sigma$ hold.}$

We do not exploit this peculiarity of the preferences to prove any of our positive results. In fact, we strengthen our results by “completing the preferences” in Section $7$. The overlapping terminology should not cause any confusion. We always specify whether it is a matching or a sequence that is blocked. Similarly, stable sequences are easy to distinguish from stable matchings.$\footnote{The notation $\Sigma R_{i}^{ssd} \Sigma'$ is not true, and for some $\Sigma S P_{j}^{ssd} \Sigma$. A sequence is ssd-stable, stable for short, if no group blocks it. For $R \in \mathcal{R}^n$, the set of stable sequences is $C_{ssd}^{\Sigma} (R)$. For $k \leq n$, $\Sigma \in \mathcal{S}$ is ssd–$k$-stable, or simply $k$-stable, if no $S \subseteq N$ such that $\#S \leq k$ blocks $\Sigma$. In settings where it is difficult for large groups to coordinate, $k$-stable sequences for $k < n$ are interesting to study. The set of $k$-stable sequences is $C_{k}^{ssd} (R)$. A sequence is ssd-efficient if no other sequence makes everyone at least as well off and someone better off.}$$

We follow Thomson’s (2013) recommendation. Equivalently, $\Sigma \in \mathcal{S}$ is ssd-efficient if $N$ does not block $\Sigma$. If $\Sigma \in \mathcal{S}$ is stable, then $\Sigma$ is $k$-stable for all $k$ and $\Sigma$ is ssd-efficient. If $\Sigma \in \mathcal{S}$ is ssd-efficient, then each $\mu \in \Sigma$ is efficient.

### 2.3. Three remarks

For a group to block, at least one agent in the group has to be made better off. This is a standard assumption. For the remaining agents in the group, we require that all should be at least as well off. We can weaken this by only requiring that no agent should be worse off (remember: $R_{i}^{ssd}$ is incomplete). Blocking is then made easier and the notion of stability is made stronger. Formally, $S \subseteq N$ **weakly ssd-blocks** $\Sigma \in \mathcal{S}$ if there is $\Sigma' \in \mathcal{S}$ such that $\mu'(S) = S$ for each $\mu' \in \Sigma'$, for each $i \in S$, $\Sigma' R_{i}^{ssd} \Sigma$, and for some $j \in S$, $\Sigma' P_{j}^{ssd} \Sigma$. A sequence is ssd-stable, stable for short, if no group blocks it. For $R \in \mathcal{R}^n$, the set of stable sequences is $C_{ssd}^{\Sigma} (R)$. For $k \leq n$, $\Sigma \in \mathcal{S}$ is ssd–$k$-stable, or simply $k$-stable, if no $S \subseteq N$ such that $\#S \leq k$ blocks $\Sigma$. In settings where it is difficult for large groups to coordinate, $k$-stable sequences for $k < n$ are interesting to study. The set of $k$-stable sequences is $C_{k}^{ssd} (R)$. A sequence is ssd-efficient if no other sequence makes everyone at least as well off and someone better off.$\footnote{Bogomolnaia and Moulin (2001) call this “ordinarily efficient”. We follow Thomson’s (2013) recommendation.}$

We allow agents to block individual matchings without affecting the remainder of the sequence, the set of stable sequences coincides with those that only contain stable matchings (see also Theorem $3$).
only if they can block as above. The underlying idea here therefore bears similarities to folk theorems in game theory that state that cooperation can be sustained in repeated interactions through threats of long-term punishments to deviating agents.

For pairing problems, a matching is stable whenever there is no pair of agents that can block it (that is, whenever it is 2-stable). This makes it straightforward to check the stability of a matching. Here, a sequence is 2-stable whenever no group can block it to a sequence consisting of just one matching (a formal proof of this claim for the general problem is provided in Appendix B, Proposition 1), though 2-stable sequences need not be stable. Hence, it is a more complicated task to determine whether a sequence is stable or not.

2.4. Rules and desirable properties

A rule is a mapping \( \varphi: \mathcal{R}^n \to \mathcal{S} \). A rule \( \varphi \) is \textbf{manipulable} at \( R \in \mathcal{R}^n \) by \( i \in N \) if there is a lie \( R_i' \in \mathcal{R} \) such that \( \varphi(R_i', R_{-i}) P^{ssd}_i \varphi(R) \). A rule \( \varphi \) is \textbf{weakly ssd–strategy-proof} if, for each profile, each agent is at least as well off reporting her true preference, \( \forall R \in \mathcal{R}^n, \forall i \in N, \forall R_i' \in \mathcal{R}, \varphi(R_i', R_{-i}) P_i^{ssd} \varphi(R) \) is not true.

A rule \( \varphi \) is \textbf{ssd–strategy-proof} if, for each profile, each agent is at least as well off reporting her true preference as she is lying,

\[
\forall R \in \mathcal{R}^n, \forall i \in N, \forall R_i' \in \mathcal{R}, \varphi(R) R_i^{ssd} \varphi(R_i', R_{-i}).
\]

This is a strengthening as \( R_i^{ssd} \) is incomplete. A rule \( \varphi \) is \textbf{state-wise strategy-proof} if, for each profile, the worst partner an agent pairs with when reporting her true preferences, \( T_{\min} \), is at least as desirable as the best partner she pairs with when telling a lie, \( L_{\max} \),

\[
\forall R \in \mathcal{R}^n, \forall i \in N, \forall R_i' \in \mathcal{R}, T_{\min} R_i L_{\max},
\]

where \( T_{\min} \) is the minimizer of \( R_i \) in \( \{ \mu(i) : \mu \in \varphi(R) \} \) and \( L_{\max} \) is the maximizer of \( R_i \) in \( \{ \mu(i) : \mu \in \varphi(R_i', R_{-i}) \} \). A rule \( \varphi \) is \textbf{state-wise manipulable} at \( R \in \mathcal{R}^n \) by \( i \in N \) through \( R_i' \in \mathcal{R} \) if the best partner \( i \) gets when reporting her true preferences, \( T_{\max} \), is not as desirable as the worst partner \( i \) gets when telling a lie, \( L_{\min} \),

\[
\exists R \in \mathcal{R}^n, i \in N, R_i' \in \mathcal{R} \text{ such that } L_{\min} P_i T_{\max},
\]

where \( T_{\max} \) is the maximizer of \( R_i \) in \( \{ \mu(i) : \mu \in \varphi(R) \} \) and \( L_{\min} \) is the minimizer of \( R_i \) in \( \{ \mu(i) : \mu \in \varphi(R_i', R_{-i}) \} \). A rule \( \varphi \) is \textbf{group-manipulable} at \( R \in \mathcal{R}^n \) by \( S \subseteq N \) if there is \( R_S' \equiv (R_i'_{i \in S}) \in \mathcal{R}^S \) such that, for each \( i \in S \), \( \varphi(R_S', R_{-S}) R_i^{ssd} \varphi(R) \), and for some \( j \in S \), \( \varphi(R_S', R_{-S}) P_j^{ssd} \varphi(R) \). A rule \( \varphi \) is \textbf{weakly group ssd–strategy-proof} if \( \varphi \) never is group-manipulable.

\[\text{This is not saying that it is “difficult” for } S \subseteq N \text{ to know whether they block a sequence. For each of the } S \text{ agents there are } S \text{ linear inequalities that need to be satisfied. By introducing slack variables, this boils down to solving a linear program where the number of inequalities grows quadratically in the number of agents. Whether a sequence is stable or not is computationally more difficult as the number of groups } S \text{ grows exponentially.}\]

\[\text{Except for } i \text{, agents have the same preferences at } R \text{ as at } (R_i', R_{-i}). \text{ Agent } i \text{ has changed her preference to } R_i'.\]

\[\text{For } S \subseteq N, (R_S', R_{-S}) \text{ denotes a similar change of preference but for all agents in } S.\]
The incentive properties are logically related as follows.

\[ \varphi \text{ is state-wise strategy-proof} \implies \varphi \text{ is ssd-strategy-proof} \implies \varphi \text{ is weakly ssd-strategy-proof} \implies \varphi \text{ is not state-wise manipulable.} \]

A rule \( \varphi \) is \textbf{ssd-stable} if it always selects a stable sequence:

\[ \forall R \in \mathcal{R}^n, \varphi(R) \in \mathcal{C}_{\text{seq}}(R). \]

A rule \( \varphi \) is \textbf{stable at all times} if it always selects a sequence of stable matchings:

\[ \forall R \in \mathcal{R}^n, \forall \mu \in \varphi(R), \mu \in \mathcal{C}(R). \]

If \( \varphi \) is \textit{stable at all times}, then \( \varphi \) is \textit{ssd-stable} \cite{Manjunath2013} Proposition 3). A rule \( \varphi \) \textbf{respects mutual best} if it always pairs agents who prefer one another to everyone else:

\[ \forall R \in \mathcal{R}^n, \forall \{i, j\} \subseteq N, \{\forall k \in N, j R_i k \text{ and } i R_j k\} \implies \forall \mu \in \varphi(R), \mu(i) = j. \]

For \( k \leq n \), \( \varphi \) is \textbf{ssd-\( k \)-stable} if it always selects a \( k \)-stable sequence: for all \( R \in \mathcal{R}^n \), \( \varphi(R) \in \mathcal{C}_{\text{seq}}^k(R) \).

If \( \varphi \) \textit{respects mutual best}, then \( \varphi \) is \textit{ssd-1-stable}. In the literature, \textit{ssd-1-stable} rules are usually referred to as \textit{individually rational.} We can strengthen this as follows. A rule \( \varphi \) is \textbf{individually rational at all times} if it always matches agents that find each other at least as desirable as being single:

\[ \forall R \in \mathcal{R}^n, \forall i \in N, \forall \mu \in \varphi(R), \mu(i) R_i i. \]

A rule \( \varphi \) is \textbf{ssd-efficient} if it always selects an ssd-efficient sequence. A rule \( \varphi \) is \textbf{efficient at all times} if it always selects a sequence of efficient matchings.

The stability properties are logically related as follows.

\[
\begin{array}{ccc}
\text{stable at all times} & \implies & \text{ssd-stable} \\
\downarrow & & \downarrow \\
\text{individually rational at all times} & \implies & \text{ssd-efficient} \\
\downarrow & & \downarrow \\
\text{ssd-1-stable} & \implies & \text{efficient at all times} \\
\end{array}
\]

Let \( \pi : N \rightarrow N \) be a bijection such that, for each \( m \in M \) and each \( w \in W \), \( \pi(m) \in W \) and \( \pi(w) \in M \). In simpler terms, \( \pi \) “swaps the sides.” Let \( \Pi \) be the set of such side-swapping bijections. If \( \pi \in \Pi \) is applied to \( R \in \mathcal{R}^n \), we get a new profile \( \tilde{R} = \pi \circ R \) defined as follows. For all \( \{i, j, k\} \subseteq N \), \( j R_i k \Rightarrow \tilde{j} \tilde{R}_i \tilde{k} \), where \( \tilde{i} \equiv \pi(i) \), \( \tilde{j} \equiv \pi(j) \), and \( \tilde{k} \equiv \pi(k) \). If \( \pi \in \Pi \) is applied to

\[ \text{Weak group ssd-strategy-proofness also implies weak ssd-strategy-proofness.} \]
\(\mu \in \mathcal{M}\), we get a new matching \(\tilde{\mu} \equiv \pi \circ \mu\) defined as follows. For all \(i \in N\) and \(j = \mu(i)\), \(\tilde{\mu}(\tilde{i}) = \tilde{j}\), where \(\tilde{i} = \pi(i)\) and \(\tilde{j} = \pi(j)\). For \(\Sigma \in \mathcal{S}\), \(\pi \circ \Sigma = [\pi \circ \mu : \mu \in \Sigma]\) is the sequence obtained when applying \(\pi \in \Pi\) to each matching in \(\Sigma\). A rule is side-neutral (in welfare terms) if neither side receives any “special treatment:"

\[\forall R \in \mathcal{R}^n, \forall \pi \in \Pi, \forall i \in N, \varphi(\pi \circ R) = \varphi(R)\]

A side-neutral rule is invariant to swapping the sides. Simply, its selection is unaffected if instead \(M\) have the preferences of \(W\) and \(W\) have the preferences of \(M\). Taking this one step further, permuting an already permuted problem swaps the sides back, but the identities of the agents within the sides may have changed. A rule that is invariant to changing the identities within the sides is anonymous. This captures that no agent receives any “special treatment.”

3. Three motivating examples

We next present some examples that highlight potential advantages of sequences over matchings. In the first, we use a sequence to Pareto-improve a stable matching.31

Example 1: Pareto-improving matchings through a sequence. Consider a scenario in which each agent cares much more about their most preferred partner than they care about the others. Intuitively, an agent may then be willing to exchange some time with a “middle-ranked” partner for more time with a lower ranked partner and more time with their most preferred partner. The main point of this example is not to convince you that these are generic and natural preferences. Rather it is to highlight that in the event that agents happen to have these tastes – a possibility that cannot a priori be ruled out – sequences add options not present if we focus only on single matchings. For this example, the agents are \(M = \{m_1, m_2\}\) and \(W = \{w_1, w_2\}\) with preferences in Table 1.

<table>
<thead>
<tr>
<th></th>
<th>(R_{m_1})</th>
<th>(R_{m_2})</th>
<th>(R_{w_1})</th>
<th>(R_{w_2})</th>
<th>(u)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(w_1)</td>
<td>(w_2)</td>
<td>(m_2)</td>
<td>(m_1)</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>(\emptyset)</td>
<td>(\emptyset)</td>
<td>(\emptyset)</td>
<td>(\emptyset)</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>(w_2)</td>
<td>(w_1)</td>
<td>(m_1)</td>
<td>(m_2)</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Preferences for Example 1. The \(\emptyset\) symbol represents being single. The column \(u\) displays (cardinal) utility levels compatible with the story presented.

At the unique stable matching \(\mu_0\) every agent is single, \(\mu_0(i) = i\) for all \(i \in N\). The sequence \([\mu_0]\) – that is, spending all time matched according to \(\mu_0\) – is stable. However, there are additional stable sequences. Indeed, this holds with few exceptions: there are generally many more stable sequences than there are stable matchings. For the sake of argument, consider the sequence \([\mu_1, \mu_2]\), where \(\mu_1 = \{(m_1, w_1), (m_2, w_2)\}\) and \(\mu_2 = \{(m_1, w_2), (m_2, w_1)\}\).

31The example can also be found in Manjunath (2013).
To be sure, neither $\mu_1$ nor $\mu_2$ is stable – so why don’t $w_1$ or $w_2$ block $\mu_1$? Recall that once a sequence is interrupted, it has to be replaced by a new sequence. Agents cannot change a matching in the sequence and expect the future plans to remain intact. In this case, if $w_1$ were to block $\mu_1$, then $\mu_2$ may not be formed tomorrow. This would be upsetting for $w_1$. Keep in mind also that sequences are repeated indefinitely. Agents $m_1$ and $m_2$ have no desire to block $\mu_2$ as the perceived sequence at $\mu_2$ is $[\mu_2, \mu_1]$. Strikingly, those agents who can block $\mu_1$ are matched to their first choice at $\mu_2$ and vice versa. As agents find their most preferred agent much better than the others, each agent prefers this alternating “compromise” and “reward” to always being single.

Next, we show a fairness issue that arises when focusing only on single matchings. Namely, stable matchings can favor some agents at the expense of others. To be sure, a matching is not stable because everyone is happy – it is stable because those unhappy cannot convince their preferred agents to match with them rather than with their current partners.

**Example 2: Unbalancedness of stable matchings.** Consider the simplest of examples that showcases the opposing interests of $M$ and $W$. The two-sided problem consists of agents $M = \{m_1, m_2\}$ and $W = \{w_1, w_2\}$ with preferences in Table 2.

<table>
<thead>
<tr>
<th>$R_{m_1}$</th>
<th>$R_{m_2}$</th>
<th>$R_{w_1}$</th>
<th>$R_{w_2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_1$</td>
<td>$w_2$</td>
<td>$m_2$</td>
<td>$m_1$</td>
</tr>
<tr>
<td>$w_2$</td>
<td>$w_1$</td>
<td>$m_1$</td>
<td>$m_2$</td>
</tr>
</tbody>
</table>

Table 2: Preferences for Example 2.

There are two stable matchings: $\mu_1 = \{(m_1, w_1), (m_2, w_2)\}$ and $\mu_2 = \{(m_1, w_2), (m_2, w_1)\}$. The agents in $M$ are well off at $\mu_1$, matching to their most preferred agents. The agents in $W$ are worse off, matching to their least preferred agents. The opposite holds for $\mu_2$. As $\mu_1$ and $\mu_2$ are the only efficient matchings, the set of stable sequences contains exactly those that include no matching other than $\mu_1$ and $\mu_2$. For instance, it includes $[\mu_1, \mu_2]$, which arguably is a fair compromise for the agents. Thus, sequences can eliminate welfare gaps that exist only because we restrict ourselves to selecting one particular matching.

As discussed in the introduction, there is no strategy-proof rule that always selects stable matchings. As 2-stable matchings are stable, this is equivalent to that no strategy-proof rule always selects 2-stable matchings. In contrast, in the following example we construct a weakly group ssd-strategy-proof rule that always selects ssd-3-stable sequences.

**Example 3: Some strategy-proofness and stability.** Let $\varphi$ be the rule that, to each $R \in \mathcal{R}^n$, selects the sequence obtained as follows. First, if possible, find a pair $\{i, j\} \subseteq N$ who prefer one another to all other agents. Formally, $j R_i k$ and $i R_j k$ for all $k \in N$. (Here, by “pair”, we

---

32The rule is also side-neutral, but not individually rational at all times.
Figure 1: Let $S = \{m_1, w_1, m_2, w_2, \ldots, m_4, w_4\}$. A solid line from agent $i$ to $j$ indicates that $i$ prefers $j$ to all other agents in $S$. In general, the agents partition into multiple components. Each component contains one cycle, that is, a list of agents such that each points to the agent following her in the list, the last agent pointing to the first. Here, there is a unique cycle $(m_1, w_1, m_2, w_2)$.

Suppose a group of agents can manipulate the rule. In particular, say $m_3$ reports a preference where $w_3$ is preferred. This is indicated by the dashed arrow from $m_3$ to $w_3$. For this misreport to be beneficial for $m_3$, $m_3$ still needs to be matched some time with $w_1$. Then $w_1$ needs to report a preference where $m_3$ is preferred. Similarly, $m_2$ needs to point to (prefer) $w_1$, $w_2$ to $m_2$, $m_1$ to $w_2$. For this to be in the interest of $m_1$, $w_1$ has to report a preference where she prefers $m_1$. However, $w_1$ has to point to $m_3$. Hence, $m_1$ will not benefit from the misreports, and therefore the group cannot manipulate the rule in this way.

also consider one agent who prefers being single.) We match $i$ and $j$ at each matching in the sequence. As these agents cannot be made better off in any way, they will not be part of a manipulating or blocking group. Remove the pair and reiterate. We may now find a new pair $(k, m)$ who prefer each other to all agents in $N \setminus \{i, j\}$. We match $k$ and $m$ at each matching in the sequence. The only way for $k$ and $m$ to potentially improve is to match with the agents already dealt with (i and j), but, as we already established that these have no interest in changing partners, $k$ and $m$ cannot be made better off. Repeat until no more such pairs are found.

Label the set of remaining agents $S$ and order the members of $S$ by their rank, $r$. We create one matching “for” each agent in $S$. In particular, at the $k$th matching, match the $k$th agent of $S$ to her most preferred agent in $S$. Leave everyone else in $S$ single at the matching.

One easily finds that no agent in $S$ can benefit from misreporting her preferences. Hence, the rule $\varphi$ is weakly ssd-strategy-proof. That no group of agents jointly can manipulate the rule requires a more involved argument. We refer to Figure 1 for a sketch of the proof for a particular two-sided problem. Lastly, it is immediate that $S$ generally will block the sequence. To be more precise, any group of agents that forms a cycle as described in Figure 1 can block. As such cycles must contain at least four agents, the rule is ssd-$3$-stable.

4. Results on two-sided problems

Though the rule designed in Example 3 has several nice properties, it is inefficient and certainly not ssd-stable. In this section, we first propose an ssd-stable rule by using the DA

33For problems that are $a$-reducible [Alcalde 1995], we can find such pairs in any subset of the agents. On the restricted domain of $a$-reducible problems, $\varphi$ is ssd-stable, even for general pairing problems.

34If there is a loop or a cycle of two agents, then that should have been processed in the previous step. Cycles are of even length as they alternate between $M$- and $W$-agents. Hence, the shortest cycle is of length 4 or more.
mechanism. The rule is \textit{stable at all times}, a property that cannot be attained alongside even weak incentive properties. We detail several such impossibilities in Theorem 1. Thereafter, we introduce an \textit{ssd-stable} and \textit{weakly group ssd-strategy-proof} rule. Finally, we provide three alternatives if we insist on \textit{ssd-strategy-proof} rules.

4.1. The Repeated Deferred Acceptance with Alternating Proposers rule

Recall that the \textit{M-proposing DA} mechanism generates $\mu_R^M$, the \textit{M-optimal stable matching}. The rule that, for each $R \in \mathcal{R}^n$, selects $[\mu_R^M]$ is \textit{stable at all times}, and hence \textit{ssd-stable} \cite{Manjunath2013} Proposition 3). It is also \textit{individually rational at all times} and \textit{anonymous}, but not \textit{side-neutral}. The last property can be added by modifying the rule as follows.

\textbf{Repeated DA with Alternating Proposers.} For each $R \in \mathcal{R}^n$, $\text{RDAAP}(R) = [\mu_R^M, \mu_R^W]$. This is an improvement over the rule in Example 3 in terms of \textit{efficiency} and \textit{fairness}, though not in terms of incentives. Agents occasionally can manipulate $\text{RDAAP}$. In general, if a rule is not strategy-proof, some agents may attempt to manipulate it even if they do not have enough information about the others’ preferences to surely benefit from their misreport.\footnote{For empirical evidence of this claim and experimental results, see \cite{Braun2010} and \cite{Pais2011}.}

When this is the case, the sequence selected by the rule is stable with respect to the true ones. The following example, familiar from \cite{Roth1982} and \cite{Alcalde1994}, highlights that $\text{RDAAP}$ is manipulable, as well as provides an interesting impossibility.

\textbf{Example 4: Rules that are stable at all times are manipulable.} Consider the two-sided problem with agents $M = \{m_1, m_2\}$ and $W = \{w_1, w_2\}$ with preferences in Table 3. Define the matchings $\mu_1 = \{(m_1, w_1), (m_2, w_2)\}$ and $\mu_2 = \{(m_1, w_2), (m_2, w_1)\}$. Then $\phi^\text{seq}(R) = \{\mu_1, \mu_2\}$, $\phi^\text{seq}(R_{m_1}^t, R_{-m_1}) = \{\mu_1\}$, and $\phi^\text{seq}(R_{w_1}^t, R_{-w_1}) = \{\mu_2\}$.

\begin{table}[h]
\centering
\begin{tabular}{cccccccc}
\hline
$R_{m_1}^t$ & $R_{m_1}$ & $R_{m_2}$ & $R_{w_1}^t$ & $R_{w_1}$ & $R_{w_2}$ \\
\hline
$w_1$ & $w_1$ & $w_2$ & $m_2$ & $m_2$ & $m_1$ \\
$\emptyset$ & $w_2$ & $w_1$ & $\emptyset$ & $m_1$ & $m_2$ \\
\hline
\end{tabular}
\caption{Preferences for Example 4.}
\end{table}

Let $\phi$ be \textit{stable at all times}. Then $\phi(R_{m_1}^t, R_{-m_1}) = [\mu_1]$, $\phi(R_{w_1}^t, R_{-w_1}) = [\mu_2]$, and, for each $\mu \in \phi(R)$, $\mu \in \{\mu_1, \mu_2\}$. If $\mu_1 \in \phi(R)$, then $w_1$ state-wise manipulates $\phi$ at $R$ through $R_{w_1}^t$. If $\mu_2 \in \phi(R)$, then $m_1$ state-wise manipulates $\phi$ at $R$ through $R_{m_1}^t$. Therefore, $\phi$ is state-wise manipulable at $R$.

We can strengthen this negative finding. For this purpose, we present three impossibilities. The first two show that, if we insist on \textit{individual rationality at all times} and require the weakest form of incentive constraints, that the rule is not \textit{state-wise manipulable}, then we are going to have to give up on all forms of efficiency and stability. The third shows that, if we insist on \textit{ssd-strategy-proofness}, then we have to give up on the weakest form of stability, respecting \textit{mutual best}.\footnote{For empirical evidence of this claim and experimental results, see \cite{Braun2010} and \cite{Pais2011}.}
Theorem 1. The following holds for pairing problems.

1. A rule that is individually rational at all times and efficient at all times is state-wise manipulable.
2. A rule that is individual rational at all times and respects mutual best is state-wise manipulable.
3. No ssd–strategy-proof rule respects mutual best.

4.2. The Compromises and Rewards rule

In this subsection, we design a weakly group ssd–strategy-proof rule that selects stable sequences. Example 5 explains how the algorithm which defines the rule is designed.

Example 5: Introducing Algorithm 1. For concreteness, consider the two-sided problem with agents $M = \{m_1, m_2, m_3, m_4\}$ and $W = \{w_1, w_2, w_3, w_4\}$ with preferences in Table 4.

<table>
<thead>
<tr>
<th>$R_{m_1}$</th>
<th>$R_{m_2}$</th>
<th>$R_{m_3}$</th>
<th>$R_{m_4}$</th>
<th>$R_{w_1}$</th>
<th>$R_{w_2}$</th>
<th>$R_{w_3}$</th>
<th>$R_{w_4}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_1$</td>
<td>$w_2$</td>
<td>$w_1$</td>
<td>$w_4$</td>
<td>$m_2$</td>
<td>$m_1$</td>
<td>$m_3$</td>
<td>$m_3$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\phi$</td>
</tr>
</tbody>
</table>

Table 4: Preferences for Example 5. Whenever only partial preferences are provided, missing agents can be ranked arbitrarily below the provided ones.

We will construct a sequence $[\mu_1, \mu_2]$. As a first step, we create a directed graph; see the leftmost graph in Figure 2. In it, each agent is represented by a vertex and has exactly one outgoing edge, namely to the agent’s most preferred agent. Here, as $w_1$ is $m_1$’s most preferred agent, there is an edge $(m_1, w_1)$.

The graph must contain either a loop or a cycle as each agent has an outgoing edge and there is a finite set of agents. For a two-sided problem, there can be no cycles of odd length as each cycle must alternate between a member of $M$ and a member of $W$. Here, the unique cycle $C = (m_1, w_1, m_2, w_2)$ is marked in gray. At $\mu_1$, we match the first agent of the cycle to the one she is pointing to (the second); the third agent to the fourth; and so on. At $\mu_2$, we match the second agent to the third; the fourth to the fifth; ...; and the $m$th to the first. In other words, agents in $C$ are set to alternate between their neighbours as indicated by the matchings along the edges in Figure 2. The agents in $C$ are then removed and a new graph is
created for $N \setminus C$; see the second graph of Figure 2. It contains the cycle $C' = (m_3, w_3)$. We match $m_3$ and $w_3$ at $\mu_1$ and at $\mu_2$. The agents in $C'$ are then removed and a new graph is created for $m_4$ and $w_4$. In it, there is a loop $(w_4)$. Here, $w_4$ is set to be single at $\mu_1$ and at $\mu_2$. Finally, $m_4$ is the only remaining agent and is also set to be single all the time. Therefore, $\mu_1 = \{ (m_1, w_1), (m_2, w_2), (m_3, w_3), (m_4, w_4) \}$ and $\mu_2 = \{ (m_1, w_2), (m_2, w_1), (m_3, w_3), (m_4, w_4) \}$, and $CR(R) = \{ \mu_1, \mu_2 \}$.

Compromises and Rewards. For each $R \in R^n$, $CR(R)$ is the sequence $[\mu_1, \mu_2]$ obtained when applying Algorithm 1 to $R$.

---

**Algorithm 1: Compromises and Rewards**

1. Define the vertex set $V := N$
2. As long as $V$ is not empty,
3. Define the edge set $E := \{ (i, j) \in V \times V : \forall k \in V, \ j R_i k \}$
4. Find a cycle $C$ in the directed graph $(V, E)$, label it $C = (c_1, c_2, \ldots, c_m)$
5a. If $m = 1$, set $\mu_1(c_1) = \mu_2(c_1) = c_1$
5b. If $m = 2$, set $\mu_1(c_1) = \mu_2(c_1) = c_2$
5c. If $m > 2$, set $\mu_1(c_1) = c_2, \mu_1(c_3) = c_4, \ldots, \mu_1(c_{m-1}) = c_m$
\hspace{1cm}$\mu_2(c_2) = c_3, \mu_2(c_4) = c_5, \ldots, \mu_2(c_m) = c_1$
6. Set $V := V \setminus C$ and repeat Step 2

We are now ready to state the main result on two-sided problems. The proof can be found in the Appendix.

**Theorem 2.** For two-sided pairing problems $R \in R^n$, the Compromises and Rewards rule is ssd-stable, weakly group ssd–strategy-proof, and side-neutral.

An auxilliary result is Proposition 2 (see Appendix); its main implication is that Algorithm 1 is “path-independent.” To be more precise, the sequence obtained is invariant to the order in which the cycles are processed. This is something of a relief: we pointed out that an issue with DA is that the potentially arbitrary choice of proposer has severe welfare implications. Here, a similarly arbitrary choice does not have any implications for sequence. Moreover, this allows us to impose any heuristic we wish for the order in which cycles are chosen. For instance, for computational reasons it might be desirable to prioritize longer cycles over shorter ones.

The $CR$ rule is not **individually rational at all times**. Hence, occasionally there is an agent $i$ who is matched to an agent $j$ such that $i \succ_i j$. In some settings, this may not be feasible. Modifying Algorithm 1 by requiring that agent $i$ points to her most preferred available agent among those who prefer $i$ to being single is not a good way of dealing with this issue. It is easy to construct examples in which one can manipulate the rule by declaring some acceptable agent unacceptable. This was already emphasized in Theorem 1 insisting on individual

---

36Formally, $m_3$ alternates between his neighbours in the cycle, which “both” are $w_3$. 16
rationality at all times and any type of non-manipulability comes at a large cost. Assuming
\( j R_j i \) whenever \( i R_i j \) only worsens the issue as the preference domain then no longer is
“rectangular” (i’s domain of preference should be independent of j’s reported preference).
Strategy-proofness is then not well-defined. Suppose instead we interpret \( i \) and \( j \) being im-
possible to match as something predetermined outside the model. Then agents are only able to
misreport their preference regarding acceptable agents. This requires potentially different
preference domains \( \mathcal{R}_i \) for each \( i \in N \) such that \( i R_i j \) implies \( i R'_i j \) for all \( R_i \in \mathcal{R}_i \) and \( j R_j i \)
for all \( R_j \in \mathcal{R}_j \). Our positive findings, Theorems 2, 4, and 5, hold also when we modify the
model and consider only problems in \( (\mathcal{R}_i)_{i \in N} \subset N^n \) restricted like this.

4.3. What can we get if we insist on ssd–strategy-proofness?

In this subsection we examine three rules, two of which are state-wise strategy-proof. First,
consider the singles rule: it always keeps agents single. It is side-neutral, individually rational
at all times, and state-wise strategy-proof, but both inefficient and unstable.

Singles. For each \( R \in \mathcal{R}^n \), \( S(R) = \{ \mu_0 \} \), where, for each \( i \in N \), \( \mu_0(i) = i \).

At the expense of side-neutrality and individual rationality at all times, we can regain ef-
ficiency. For the next rule, imagine the agents standing in a queue. We first “serve” the first
agent of the queue, who leaves with her favorite agent. Then, we serve the next agent of the
queue, who leaves with her favorite agent among those that remain. This continues until no
agent remains.

Formally, take as given \( R \in \mathcal{R}^n \) and describe the ordering of the agents by \( \omega: \{1, 2, \ldots, n\} \rightarrow N \).
Let \( \omega^{-1} \) denote the inverse of \( \omega \), in the sense that \( \omega^{-1}(i) \) is agent i’s position in the order \( \omega \). The
“sequential priority with respect to \( \omega^{-1} \)”-matching is \( \mu_2^\omega \) and constructed as follows. Initially, let
\( N_1 \coloneq N \), \( i_1 \coloneq \omega(1) \), and \( j_1 \in N_1 \) be such that, for all \( x \in N_1 \), \( j_1 R_i x \). We set \( \mu_2^\omega(i_1) = j_1 \) and
\( N_2 \coloneq N_1 \setminus \{i_1, j_1\} \). In general, at step \( k \), let \( i_k \in N_k \) be such that, for all \( x \in N_k \), \( \omega^{-1}(i_k) \preceq \omega^{-1}(x) \).
Let \( j_k \in N_k \) be such that, for all \( x \in N_k \), \( j_k R_{i_k} x \). We set \( \mu_2^\omega(i_k) = j_k \) and \( N_{k+1} \coloneq N_k \setminus \{i_k, j_k\} \).

ω-Sequential Priority. For each \( R \in \mathcal{R}^n \), \( SP^\omega(R) = \{ \mu_2^\omega \} \).

This rule is ssd-efficient and state-wise strategy-proof. Let \( \Omega \) denote the set of orderings.
The following rule takes all of them into account. It is side-neutral, efficient at all times, and
ssd–strategy-proof.

Random Sequential Priority. For each \( R \in \mathcal{R}^n \), \( RSP(R) = \{ \mu_R^\omega : \omega \in \Omega \} \).

Example [11] (Appendix) shows that \( RSP \) is not ssd-efficient.

5. A short look at many-to-one matching

A many-to-one matching problem is one where agents on one side can match to many
agents on the other. Formally, a problem is described by \( (F, W, R, q) \), where \( F \) and \( W \) are the
agents, \( R \) their preferences, and \( q = (q_f)_{f \in F} \) is a vector of capacities. For concreteness, think
of \( f \in F \) as a firm that hires \( q_f \in \mathbb{N} \) workers among those in \( W \). We can embed the problem
Table 5: A summary of our findings for two-sided problems. Minus signs indicate incompatibilities, checkmarks compatibilities. Minus signs are inherited by stronger axioms as indicated by parentheses. Plus signs are inherited by weaker axioms. Arrows go from stronger to weaker axioms. Axioms in column 1 are met by RDAAP, column 5 by CR, column 6 by Singles, column 7 by SPω, and column 8 by RSP. The incompatibilities in columns 2, 3, and 4 are found in Theorem 1.

in the one-to-one setting as follows. Each \( f \in F \) with capacity \( q_f \) is represented by the agents \( f^1, f^2, \ldots, f^{q_f} \), each of them having the same preference over \( W \) as \( f \) has. That is, \( R_{f^1} = R_{f^2} = \cdots = R_{f^{q_f}} = R_f \). Moreover, \( w \in W \) prefers \( f^k \in F \) to \( g^m \in F \) whenever \( f P_w g \) or, in case \( f = g \), when \( k < m \). It is then straightforward to apply Algorithm 1 to this problem. The CR rule is still ssd-stable; the proof is essentially the same as in Theorem 2. However, as a firm can be part of multiple cycles, it may be able to manipulate the rule.

Example 6: Many-to-one manipulation of CR. Consider the many-to-one matching problem with agents \( F = \{ f_1, f_2, f_3 \} \) and \( W = \{ w_1, w_2, \ldots, w_5 \} \), quotas \( q_{f_1} = 2, q_{f_2} = q_{f_3} = 1 \), and preferences in Table 6.

Table 6: Preferences for Example 6. The preference \( \tilde{R}_{f_i} \) is reported by \( f_i \) to manipulate the CR rule.
The CR rule selects \( [\mu_1, \mu_2] \), where

\[
\mu_1 = \{(f_1^1, w_1), (f_2^2, w_4), (f_2, w_2), (f_3, w_3), (w_5)\}
\]
\[
\mu_2 = \{(f_1^1, w_2), (f_2^2, w_3), (f_2, w_1), (f_3, w_4), (w_5)\}.
\]

Consider the joint manipulation by \( f_1^1 \) and \( f_1^2 \) where they rank \( w_1 \) over \( w_2 \) at the top of their preference lists; that is, they report \( \bar{R}_{f_i} \). The resulting sequence is \( [\mu'_1, \mu'_2] \).

\[
\mu'_1 = \{(f_1^1, w_1), (f_2^2, w_3), (f_2, w_2), (f_3, w_4), (w_5)\}
\]
\[
\mu'_2 = \{(f_1^1, w_2), (f_2^2, w_3), (f_2, w_1), (f_3, w_4), (w_5)\}.
\]

This is an improvement for \( f_1^1 \), though not for \( f_1^2 \). That is to be expected as CR is weakly group ssd-strategy-proof. However, it is an improvement for \( f_1 \) as a whole. When reporting its true preference, \( f_1 \) matches to \( \{w_1, w_4\} \) and \( \{w_2, w_3\} \). When misreporting its preference, \( f_1 \) matches to \( \{w_1, w_5\} \) and \( \{w_2, w_2\} \).

6. Extending to the general pairing problem

For general pairing problems, there is a set of agents \( N \). In contrast to two-sided problems, the agents are not divided into two groups. A (general) matching is a mapping \( \mu : N \rightarrow N \) such that \( \mu(i) = j \iff \mu(j) = i \) for all \( \{i, j\} \subseteq N \). The set of general matchings is \( \mathcal{M}^* \). All concepts defined regarding matchings in Section 2 can be redefined with respect to \( \mathcal{M}^* \) rather than \( \mathcal{M} \).

Previously, a preference was restricted in the sense that \( M \)-agents preferred \( W \)-agents to other \( M \)-agents. We remove this restriction and denote the set of general preferences \( \mathcal{R}^* \). A profile is \( R \in (\mathcal{R}^*)^n \). The set of sequences of general matchings is \( \mathcal{F}^* \). We extend all concepts previously defined by substituting \( \mathcal{F}^* \) for \( \mathcal{F} \). This is straightforward as no concepts besides matchings, preferences, and side-neutrality refer to the sets \( M \) and \( W \). Side-neutrality is not applicable for the general problem, but its implication, anonymity, is. Let \( \Pi^* \) denote the set of bijections on \( N \). A rule is anonymous (in welfare terms) if it is invariant to any permutation \( \pi \in \Pi^* \):

\[
\forall R \in (\mathcal{R}^*)^n, \forall \pi \in \Pi^*, \forall i \in N, \varphi(\pi \circ R) f_i^{\text{ssd}} \pi \circ \varphi(R).
\]

In contrast to two-sided problems, general problems need not have stable matchings. The problem examined in Example 7 has no stable matching, but does have stable sequences.

Example 7: A general problem without a stable matching. Consider the general problem with agents \( N = \{1, 2, 3\} \) with preferences in Table 7.

For each \( \mu \in \mathcal{M}^* \), at least one agent \( i \) is single. Then \( i \) and her second most preferred agent block \( \mu \). Hence, there is no stable matching. In contrast, \( \Sigma = [\mu_1, \mu_2, \mu_3] \) is a stable sequence, where \( \mu_1 = \{(1, 2, 3)\}, \mu_2 = \{(1, 3), (2)\} \), and \( \mu_3 = \{(1, 2), (3)\} \).

Note that \( \Sigma \) is “minimal” in the sense that if a matching is removed, then the sequence is no longer stable. For instance, \( \{1, 3\} \) block \( [\mu_1, \mu_2] \). Intuitively, agent 1 can “guarrantee” himself agent 3 as 3 is always willing to block through \( [\mu_2] \). At \( \mu_1 \), agent 1 therefore makes a compromise by being single. However, agent 1 is not rewarded for this at \( [\mu_1, \mu_2] \), in the
sense that 1 does not get to match with 2. Therefore, 1 and 3 block $\mu_1, \mu_2$. In contrast, $\Sigma$ is a sequence that balances the “compromises” and “rewards” for all agents.\footnote{An immediate consequence is the following. For general pairing problems, if a rule $\varphi$ is ssd–3-stable, then $\#\varphi(R) > 2$ for some $R \in (\mathcal{R}^*)^n$. This is different from what we found in Section 4 where we presented two ssd-stable rules for two-sided problems that never selected sequences of more than two matchings.}

### Table 7: Preferences for Example 7

<table>
<thead>
<tr>
<th></th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td></td>
</tr>
</tbody>
</table>

6.1. The (Extended) All-Proposing Deferred Acceptance rule

As the agents no longer are divided into two groups, it is not clear how to use the DA mechanism. We therefore propose a modified mechanism to extend RDAAP. The rule is defined on a restricted domain. The problem in Example 7 is for instance not covered. In contrast, all two-sided problems are. Formally, we define its domain $\mathcal{E}$ as follows. For $R \in (\mathcal{R}^*)^n$, $R \in \mathcal{E}$ if and only if, when applying Algorithm 2 to $R$, every cycle at Step 3 is of even length.

**All-Proposing DA.** For each $R \in \mathcal{E}$, $APDA(R)$ is the sequence $[\mu_1, \mu_2]$ obtained when applying Algorithm 2 to $R$.

**Algorithm 2: All-Proposing Deferred Acceptance**

1. Create a graph with vertex set $V := N$. An edge $(i, j)$ in the edge set $E$ is associated with the proposal from agent $i$ to $j$. Initially, each agent proposes (adds an edge in $E$) to her most preferred agent.
2. Each agent $i$ rejects all proposals but the one made by her most preferred agent $j$ among those proposing (pointing) to her. If $i$ $P_i j$, then $j$’s proposal to $i$ is also rejected. If a proposal is rejected, its associated edge is removed from $E$. If no proposals are rejected, continue to Step 3. Otherwise, each rejected agent proposes (add an edge) to her most preferred agent that has not yet rejected her, and Step 2 is repeated.
3. For each cycle $C$ in the directed graph $(V, E)$, labeled $C = (c_1, c_2, \ldots, c_m)$,
   - If $m = 1$, set $\mu_1(c_1) = \mu_2(c_1) = c_1$
   - If $m = 2$, set $\mu_1(c_1) = \mu_2(c_1) = c_2$
   - If $m > 2$, set
     \[
     \begin{align*}
     \mu_1(c_1) &= c_2, \\
     \mu_1(c_3) &= c_4, \\
     \ldots, \\
     \mu_1(c_{m-1}) &= c_m \\
     \mu_2(c_2) &= c_3, \\
     \mu_2(c_4) &= c_5, \\
     \ldots, \\
     \mu_2(c_m) &= c_1
     \end{align*}
     \]

For the rule to be well-defined, each agent has to be part of exactly one cycle at Step 3. If an agent is part of two or more cycles, then some agent must have multiple outgoing edges.
But that cannot happen, as the agent only makes a new proposals when her previous one is rejected. If an agent is not part of any cycles, then some agent must have multiple incoming edges. But that cannot happen, as the agent always rejects all but at most one proposal. Hence, the rule is well-defined.

**Remark 1: Relation to stable matchings and partitions.** The domain $E$ is logically independent of the set of problems that have stable matchings. The following six-agent examples also show that the graph in Algorithm 2 is fundamentally different from those that arise from Tan’s (1991) stable partitions (see Appendix C).

(i) Preferences in Table 8(i). The matching $\mu = \{(1, 2), (3,4), (5,6)\}$ is stable. The problem is not in $E$ but in $\hat{E}$. (ii) Preferences in Table 8(ii). There is no stable matching. The problem is in $E$. The APDA rule selects $\left[\{(1, 2), (3,4), (5,6)\}, \{(1,6), (2,3), (4,5)\}\right]$.

<table>
<thead>
<tr>
<th>$R_1$</th>
<th>$R_2$</th>
<th>$R_3$</th>
<th>$R_4$</th>
<th>$R_5$</th>
<th>$R_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Table 8(i): Preferences for Remark 1(i).*

<table>
<thead>
<tr>
<th>$R_1$</th>
<th>$R_2$</th>
<th>$R_3$</th>
<th>$R_4$</th>
<th>$R_5$</th>
<th>$R_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>6</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Table 8(ii): Preferences for Remark 1(ii).*

The APDA rule is *ssd-stable, anonymous, and individually rational at all times* for problems in $E \subset (\mathcal{R}^*)^n$ (proof is given for a more general result in Theorem 4). We can easily extend APDA to a larger domain of problems. Define $\hat{E}$ by including all problems that have stable matchings, so $\hat{E} = E \cup \{ R \in (\mathcal{R}^*)^n : \mathcal{C}(R) \neq \emptyset \}$. Let $C(R)$ denote the sequence that attributes equal weight to each stable matching, that is, $C(R) = [\mu : \mu \in \mathcal{C}(R)]$.

**Extended APDA.** For each $R \in \hat{E}$,

$$EAPDA(R) = \begin{cases} 
APDA(R) & \text{if } R \in E, \\
C(R) & \text{otherwise}.
\end{cases}$$

This extension is motivated by the following result. It extends Manjunath’s (2014) Proposition 3 from two-sided to general pairing problems (using a similar proof technique).

**Theorem 3.** For general pairing problems $R \in (\mathcal{R}^*)^n$, a sequence of stable matchings is stable.

**Theorem 4.** For general pairing problems $R \in \hat{E}$, the Extended All-Proposing Deferred Acceptance rule is *ssd-stable, anonymous, and individually rational at all times*.

### 6.2. The Generalized Compromises and Rewards rule

Next we present a rule defined on the full preference domain $(\mathcal{R}^*)^n$. Algorithm 3, that defines the rule, coincides with Algorithm 1 in special cases, for instance for two-sided problems. For general pairing problems, it extends Algorithm 1 by being able to handle a certain type of odd cycles.
Example 8: Introducing Algorithm 3. To illustrate the algorithm, consider the general problem with agents $N = \{1, 2, \ldots, 6\}$ with preferences in Table 9. The problem has no stable matching and is not in the domain of APDA.

To keep track of who’s matched with whom, we use a tree $T$ that initially only contains its root $\rho$ (Figure 3). To each node of $T$, we associate a list of pairs and a weight (for the final full list, see Table 11). An agent is available at node $\nu$ if she is not paired along the path from $\rho$ to $\nu$. Collect the agents available at $\nu$ in $V_\nu$; note that $V_\rho = N$. If $V_\nu \neq \emptyset$, we find children of $\nu$ as follows. Create a directed graph in which each agent in $V_\nu$ points to her most preferred agent in $V_\nu$. In the graph, select a cycle $C$. Here, at $\rho$, the graph is the leftmost of Figure 4. It has a unique cycle $C = (1, 2, 3)$.

In general, if we select an odd cycle of length $m$, we add $m$ children of $\nu$. At the $k$th child, the $k$th agent of $C$ is not matched to someone else in $C$. Here, we add nodes $a$ (where agents 2 and 3 are matched), $b$ (1 and 3), and $c$ (1 and 2). A node’s weight is one over its number of siblings; $a$, $b$, and $c$ have weight $1/3$. At $a$, the graph is at the middle of Figure 4 and has an odd cycle $(1, 4, 5)$. We pair agents 4 and 5 at node $d$, 1 and 5 at $e$, and 1 and 4 at $f$. Each of these nodes has weight $1/3$. At $c$, the graph is the rightmost of Figure 4 and has an even cycle $(3, 4, 5, 6)$. When we select an even cycle of at least four agents, we add two children. We pair agents 3 and 6, 4 and 5 at node $i$; agents 3 and 4, 5 and 6 at node $j$. Each of these nodes has weight $1/2$. When we select a cycle of at most two agents, we add one child.

A leaf $\lambda$ of $T$ is a node at which all agents are matched, $V_\lambda = \emptyset$. A branch is a collection of connected nodes, starting at the root and ending at a leaf without repeating any nodes. To

---

Table 9: Preferences for Example 8.

<table>
<thead>
<tr>
<th>$R_1$</th>
<th>$R_2$</th>
<th>$R_3$</th>
<th>$R_4$</th>
<th>$R_5$</th>
<th>$R_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>4</td>
<td>6</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 3: The tree $T$ referred to in Example 8.

Figure 4: The three graphs referred to in Example 8.

---

38 The domain is not rectangular. Strategy-proofness is therefore not well-defined.
Table 11: On the left are the nodes of the tree $T$ referred to in Example 8 (Figure 3) with their associated pairs and weights; on the right are the branches of $T$ with their associated matchings and weights.

<table>
<thead>
<tr>
<th>Node</th>
<th>Pairs</th>
<th>Weight</th>
<th>Node</th>
<th>Pairs</th>
<th>Weight</th>
<th>Branch $\beta$</th>
<th>Matching $\mu_\beta$</th>
<th>Weight $\omega_\beta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\rho$</td>
<td>$\emptyset$</td>
<td>1</td>
<td>$g$</td>
<td>${(2,5)}$</td>
<td>1</td>
<td>1 : $\rho$, $a$, $d$, $j$</td>
<td>${(1,6), (2,3), (4,5)}$</td>
<td>$1/9$</td>
</tr>
<tr>
<td>$a$</td>
<td>${(2,3)}$</td>
<td>$1/3$</td>
<td>$h$</td>
<td>${(3,6), (4,5)}$</td>
<td>$1/2$</td>
<td>2 : $\rho$, $a$, $e$, $k$</td>
<td>${(1,5), (2,3), (4,6)}$</td>
<td>$1/9$</td>
</tr>
<tr>
<td>$b$</td>
<td>${(1,3)}$</td>
<td>$1/3$</td>
<td>$i$</td>
<td>${(3,4), (5,6)}$</td>
<td>$1/2$</td>
<td>3 : $\rho$, $a$, $f$, $l$</td>
<td>${(1,4), (2,3), (5,6)}$</td>
<td>$1/9$</td>
</tr>
<tr>
<td>$c$</td>
<td>${(1,2)}$</td>
<td>$1/3$</td>
<td>$j$</td>
<td>${(1,6)}$</td>
<td>1</td>
<td>4 : $\rho$, $b$, $g$, $m$</td>
<td>${(1,3), (2,5), (4,6)}$</td>
<td>$1/3$</td>
</tr>
<tr>
<td>$d$</td>
<td>${(4,5)}$</td>
<td>$1/3$</td>
<td>$k$</td>
<td>${(4,6)}$</td>
<td>1</td>
<td>5 : $\rho$, $c$, $h$</td>
<td>${(1,2), (3,6), (4,5)}$</td>
<td>$1/6$</td>
</tr>
<tr>
<td>$e$</td>
<td>${(1,5)}$</td>
<td>$1/3$</td>
<td>$l$</td>
<td>${(5,6)}$</td>
<td>1</td>
<td>6 : $\rho$, $c$, $i$</td>
<td>${(1,2), (3,4), (5,6)}$</td>
<td>$1/6$</td>
</tr>
<tr>
<td>$f$</td>
<td>${(1,4)}$</td>
<td>$1/3$</td>
<td>$m$</td>
<td>${(4,6)}$</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Each branch $\beta$, we associate a matching $\mu_\beta$ and a weight $\omega_\beta$. For instance, branch 1 through $a$ and $d$ to $j$ corresponds to the matching $\mu_1 = \{(1,6), (2,3), (4,5)\}$. The weight is the product of the nodes’ weights. For instance, branch 5 through $c$ to $h$ has weight $1 \cdot 1/3 \cdot 1/2 = 1/6$. Finally, find the smallest integer $x \geq 0$ such that, for each branch, the product of $x$ and the branch’s weight is integer. Here, $x = 18$. For each branch $\beta$, $\mu_\beta$ is included $\omega_\beta \cdot x$ times in the sequence.

For this problem $R$, we have

$$GCR(R) = [\{ \mu_1, \mu_2, \mu_3, \mu_3, \mu_4, \mu_4, \mu_4, \mu_4, \mu_5, \mu_5, \mu_6, \mu_6 \}]$$

**Generalized Compromised and Rewards.** For each $R \in (\mathcal{R}^*)^n$, $GCR(R)$ is the sequence obtained when applying Algorithm 3 to $R$.

The further down the tree we move, the smaller the problem we have to solve. Hence, it is clear that Algorithm 3 terminates in finite time. Because an agent is available at a node if and only if she has not yet been matched, the procedure generates well-defined matchings and the rule selects a well-defined sequence. All statements made in Proposition 2 are valid for Algorithm 3. Hence, the order in which cycles are processed is irrelevant for the sequence ultimately chosen. For computational reasons, it is in favorable to process even cycles before odd ones.

**Theorem 5.** For general pairing problems $R \in (\mathcal{R}^*)^n$, the Generalized Compromises and Rewards rule is ssd–5-stable, weakly ssd–strategy-proof, and anonymous.

In addition, one can easily show that $GCR$ always selects sequences of efficient matchings. However, this does not necessarily imply that the sequence is ssd-efficient. See Example 11 in the Appendix, building on a similar finding by Bogomolnàï and Moulin (2001). We do however conjecture that we can strengthen the two first properties as follows.

**Conjecture 1.** For general pairing problems $R \in (\mathcal{R}^*)^n$, the Generalized Compromises and Rewards rule is ssd-stable and weakly group ssd–strategy-proof.
Algorithm 3: Generalized Compromises and Rewards

1. Initialize a tree \( T \) rooted at \( \rho \)
2. Use Function 1 to add children of \( \rho \), children of the children of \( \rho \), ...
3. Denote the branches of \( T \) by \( B \). For each \( \beta \in B \), union the pairs along \( \beta \) to get the graph of a matching \( \mu_\beta \). Its weight \( \omega_\beta \) is the product of its nodes' weights.
4. Define \( x > 0 \) as the smallest integer such that, for each \( \beta \in B \), \( \omega_\beta \cdot x \) is integer
5. Define \( GCR(R) \) as the sequence that, for each \( \beta \in B \), contains \( \mu_\beta \) exactly \( \omega_\beta \cdot x \) times and no other matching

Function 1: Add children of node \( \nu \)

1. Define the vertex set \( V_\nu \) as all agents not paired along the path from \( \rho \) to \( \nu \) and define the edge set \( E_\nu = \{ (i, j) \in V_\nu \times V_\nu : \forall k \in V_\nu, j R_k i \} \)
2. Find a cycle \( C \) in the directed graph \((V, E)\), label it \( C = (c_1, c_2, \ldots, c_m) \)
3a. If \( m = 1 \), add one child of \( \nu \) with weight 1. Associate with it \( (c_1) \)
3b. If \( m = 2 \), add one child of \( \nu \) with weight 1. Associate with it \( (c_1, c_2) \)
3c. If \( m > 2 \) is even, add two children of \( \nu \) with weights 1/2
   Associate with the \( k \)th \( (c_k, c_{k+1}), (c_{k+2}, c_{k+3}), \ldots, (c_{k-1}, c_k) \)
3d. If \( m > 2 \) is odd, add \( m \) children of \( \nu \) with weights \( 1/m \)
   Associate with the \( k \)th \( (c_{k+1}, c_{k+2}), (c_{k+3}, c_{k+4}), \ldots, (c_{k-2}, c_{k-1}) \)

7. Discussion

7.1. Probabilistic and fractional matchings

A sequence can be reinterpreted as a lottery over matchings (a probabilistic matching). The support of the lottery coincides with the set of matchings included in the sequence. The probability assigned to a matching in the lottery is the frequency in which the matching appears in the sequence. For instance, \( \Sigma = [\mu, \mu, \mu'] \) corresponds to the lottery that assigns probability \( 2/3 \) to \( \mu \) and \( 1/3 \) to \( \mu' \).

The relation between probabilistic and fractional matchings is most apparent when we examine the matrix \( \sigma \) that is induced by the sequence \( \Sigma \). Formally, \( \sigma \) is a symmetric doubly stochastic matrix. Indeed, this is all that is required for \( \sigma \) to represent a fractional matching. For probabilistic matchings, there is an additional requirement for odd groups of agents. For convenience, define \( \mathcal{O} = \{ S \subseteq N : \#S \geq 3, \#S \text{ is odd} \} \).

Fractional and probabilistic matchings. A fractional matching \( f \) satisfies, for all \( \{ i, j \} \subseteq N, f_{ij} = f_{ji} \in [0, 1] \), and for each \( i \in N, \sum_{j \in N} f_{ij} \leq 1 \). A probabilistic matching \( \sigma \) is a fractional matching such that, for each \( S \in \mathcal{O}, \sum_{i \in S} \sum_{j \in S \setminus \{i\}} \sigma_{ij} \leq \#S - 1 \).

Denote the set of fractional matchings by \( \mathcal{F}^* \). In special cases, for instance for all two-sided problems, fractional matchings coincide with probabilistic matchings (Birkhoff, 1946; von Neumann, 1953). For more recent work on fractional matchings, see Abeledo and Rothblum (1994), Biró and Fleiner (2010), Chiappori et al. (2014), Manjunath (2014), and Budish.
et al. (2013). An analysis of probabilistic matchings in the two-sided problem can be found in Manjunath (2013). The following two applications highlight the difference between fractional and probabilistic matchings.

**Example 9: Fractional and probabilistic matchings.** Suppose agents 1, 2, and 3 are hired to work on a project. For the first part of the project, 1 and 2’s skills are needed; for the second, 1 and 3’s; for the third, 2 and 3’s. The fractional matching $f$ pairs 1 half the time with 2, 1 half the time with 3, and 2 half the time with 3. Note that $f$ cannot be interpreted as a lottery over matchings. During the time that 2 and 3 are matched, we require that 1 is single. However, $f_{23} > 0$ and $f_{11} = 0$.

$$f = \begin{pmatrix} 0 & 1/2 & 1/2 \\ 1/2 & 0 & 1/2 \\ 1/2 & 1/2 & 0 \end{pmatrix} \quad \sigma = \begin{pmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{pmatrix}$$

Next, 1, 2, and 3 are having a break from work and consider playing chess. To find out who will play against whom, they draw straws. If 1 and 2 end up playing (being paired), 3 is single. The expected outcome of this lottery is described by $\sigma$. This is the probabilistic matching familiar from Example 7.

7.2. Completing the preference: expected utility preferences

In this subsection, we define preferences over sequences that are complete. Importantly, for some of these, our main results still hold.

The **utility function** $u_i : N \to \mathbb{R}_+$ represents $R_i \in \mathcal{R}^*$ if, for each $\{i, j, k\} \subseteq N$, $j R_i k \Leftrightarrow u_i(j) \geq u_i(k)$. A **profile of utility functions** is $u \equiv (u_i)_{i \in N}$. For each $R \in (\mathcal{R}^*)^n$, let $\mathcal{U}(R)$ denote the collection of profiles of utility functions such that, for each $i \in N$, $u_i$ represents $R_i$. At $u \in \mathcal{U}(R)$, the **expected utility** for $i \in N$ of $\Sigma \in \mathcal{S}^*$ is

$$U_i(\Sigma) = \sum_{j \in N} \sigma_{ij} u_i(j).$$

For $R \in (\mathcal{R}^*)^n$ and $u \in \mathcal{U}(R)$, define the binary relation $R^u_i$ on $\mathcal{S}^*$ such that, for each $\{\Sigma, \Sigma'\} \subseteq \mathcal{S}^*$,

$$\Sigma R^u_i \Sigma' \Leftrightarrow U_i(\Sigma) \geq U_i(\Sigma').$$

In contrast to $R^\text{ssd}_i$, the relation $R^u_i$ is complete. Define **$u$-blocking** groups and **$u$-stable** sequences by replacing $R^\text{ssd}_i$ by $R^u_i$ in the definitions of ssd-blocking and ssd-stability. At $R \in (\mathcal{R}^*)^n$, if $S \subseteq N$ weakly blocks $\Sigma \in \mathcal{S}^*$, then $S u$-blocks $\Sigma$ for all $u \in \mathcal{U}(R)$. In addition, if there is $u \in \mathcal{U}(R)$ such that $S u$-blocks $\Sigma \in \mathcal{S}^*$, then $S$ ssd-blocks $\Sigma$.

In Example 7 [\(\mu_1, \mu_2, \mu_3\)] is $u$-stable for some $u \in \mathcal{U}(R)$. The idea is similar to that contemplated in Example 1. For agents to be willing to compromise, the rewards need to be sufficiently big. If each of them find their most preferred agent much better than their second most preferred, they cannot block the sequence.
Example 10: Continuing Example 7. Sequences only containing \( \mu_1, \mu_2, \) and \( \mu_3 \) can be represented in the three-dimensional simplex. Figure 5 illustrates \([\mu_1, \mu_2, \mu_3] \) together with the upper and lower contour sets of \( R_{ssd}^1 \) at \( \Sigma \), \( U(R_{ssd}^1, \Sigma) \) and \( L(R_{ssd}^1, \Sigma) \), where

\[
U(R_{ssd}^1, \Sigma) = \{ \Sigma' \in \mathcal{S} : \Sigma'^{r_{ssd}} \Sigma \} \quad \text{and} \quad L(R_{ssd}^1, \Sigma) = \{ \Sigma' \in \mathcal{S} : \Sigma R_{ssd}^1 \Sigma' \}.
\]

The figure also illustrates the line of indifference (generally: hyperplanes) that intersect the contour sets for various preferences \( R_u^1 \). More specifically, let \( u \in \mathcal{U}(R) \) be such that, for each \( i \in N, u_i(i + 1) = 1, u_i(i - 1) = 1/\alpha \equiv \beta \in (0, 1), \) and \( u_i(i) = 0. \) The higher the value of \( \beta \), the steeper the line of indifference. Intuitively, what matters most to agent 1 is finding a partner. In contrast, a low value indicates that it is crucial that the partner is agent 3.

So when does there exist a stable sequence? For each \( i \in N, U_i(\Sigma) = (1 + \beta)/3. \) Take a generic pair \( \{i, i + 1\}. \) This pair can block to \( \Sigma' = [\mu_{i-1}] \). As we have \( U_i(\Sigma') = \beta \) and \( U_{i+1}(\Sigma') = 1 > \beta, \) the pair blocks if \( \beta \geq (1 + \beta)/3, \) that is, if \( \beta \geq 1/2. \) Therefore, \( \Sigma \) is stable whenever \( \beta < 1/2. \) Equivalently, the fraction \( u_i(i + 1)/u_i(i - 1) = 1/\beta = \alpha \) should exceed 2.

In the proofs of our theorems, we do not exploit that the relation \( R_{ssd}^1 \) is incomplete. Neither do we use the discontinuous nature of the relation. Indeed, we can generalize the results by allowing for a larger class of preferences.

\alpha-proportional utility functions.\ For each \( \alpha \geq 1, \) each \( R \in (\mathbb{R}^+)^N, \) and each \( u \in \mathcal{U}(R), \)

\[
u \in \mathcal{U}_\alpha(R) \Leftrightarrow \forall \{i, j, k\} \subseteq N, j P_i k \Rightarrow u_i(j) > \alpha u_i(k).
\]

Example 10 shows that it is necessary to remove some profiles from \( \mathcal{U}(R) \equiv \mathcal{U}_1(R) \) if we wish to find a stable rule – even if it is just 2-stable. In particular, in terms of the parameter \( \alpha, \)
Figure 6: An illustration of some domains of profiles of utility functions for Example 10. The full set is $\mathcal{U}(R)$. The domains are nested: if $\alpha \geq \alpha'$, then $\mathcal{U}_\alpha(R) \subseteq \mathcal{U}_{\alpha'}(R)$. The dashed line contains the profiles of symmetric utility functions examined in Example 10. We argue that the maximal domain with respect to the parameter $\alpha$ for which there always are 2-stable sequences is $\mathcal{U}_2(R)$. This is not to say that this is the maximal domain with respect to set inclusion. For instance, there is a profile $u \in \mathcal{U}_{1.5}(R)$ for which there is a 2-stable sequence. However, for our statement, if we include $u$, then we have to include $\mathcal{U}_{1.5}(R)$ in its entirety. But then there is $u' \in \mathcal{U}_{1.5}(R)$ for which there is no 2-stable sequence. There is no domain larger than $\mathcal{U}_2(R)$ for which there always are 2-stable sequences. See Figure 6 for an illustration. As it turns out, this domain is indeed the largest that always have 2-stable sequences. In this way, we can extend our main theorems as follows.

Theorem 1*: For two-sided pairing problems $R \in \mathbb{R}$ with profiles of utility functions $u \in \mathcal{U}_2(R)$, the Compromises and Rewards rule is $u$-stable and weakly group $u$-strategy-proof.

Theorem 3*: For general pairing problems $R \in \mathbb{E}^*$ with profiles of utility functions $u \in \mathcal{U}_2(R)$, the Extended All-Proposing Deferred Acceptance rule is $u$-stable.

Theorem 4*: For general pairing problems $R \in (\mathbb{R}^*)^n$ with profiles of utility functions $u \in \mathcal{U}_\alpha(R)$ for large enough $\alpha \in \mathbb{R}$, the Generalized Compromises and Rewards rule is $u$-5-stable and $u$-strategy-proof.

8. Conclusion

We study two-sided (“marriage”) and general pairing (“roommate”) problems. We introduce “sequences,” lists of matchings that are repeated in order. Stable sequences are natural extensions of stable matchings; case in point, we show that a sequence of stable matchings is stable. In addition, stable sequences can provide solutions to problems for which stable matchings do not exist. In a sense, they allow us to “balance” the interest of the agents at different matchings. In this way, sequences can be superior to matchings in terms of welfare and fairness.

A seminal result due to Roth [1982] is that no strategy-proof rule always selects stable matchings. In contrast, we show that there is a weakly group ssd–strategy-proof rule that selects stable sequences. We call it the Compromises and Rewards rule, CR for short. We show that the CR rule satisfies two appealing fairness axioms: anonymity and side-neutrality. For the general problem, the Generalized CR rule, GCR for short, is ssd–5-stable (cannot be
blocked by groups of five or fewer agents), *weakly ssd-strategy-proof*, and *anonymous*. In addition, the *Extended All-Proposing Deferred Acceptance* rule is *ssd-stable*, *anonymous*, and *individually rational at all times* on a restricted domain. The domain includes all problems that have stable matchings and some that do not. We provide also two negative findings. Namely, rules that are *stable at all times* are *state-wise manipulable*. Moreover, *ssd-strategy-proof* rules do not *respect mutual best*.

There are still many open questions that are interesting to study, we list only the ones we feel are most important. Clearly, proving Conjecture 1 is one of them (that is, proving that the *GCR* rule is *ssd-stable* and *weakly group ssd-strategy-proof*). Another is the question of how to extend the *APDA* rule to be defined on a larger, perhaps even the full, domain. Next is the question of how to generalize and complete the preferences over sequences. In a sense, we already provide one answer to this in Subsection 7.2 though only verbally. Similarly, we describe only a sketch of the non-cooperative foundation for stable sequences; this would be interesting to formalize. Lastly, all of our results are of the style that a particular rule satisfies certain axioms. An interesting question is to detail which axioms our rules are *the only* to satisfy.
References


Appendix A. Proofs

Appendix A.1. Proof of Theorem

Part 1: Let $M = \{m_1, m_2\}$, $W = \{w_1, w_2\}$, and refer to preferences in Table 12(i). Define $\mu_1 = \{(m_1, w_1), (m_2, w_2)\}$ and $\mu_2 = \{(m_1, w_2), (m_2, w_1)\}$. Then $\mu_1$ and $\mu_2$ are the only efficient matchings at $R$.

Let $\varphi$ be individually rational at all times and efficient at all times. To obtain a contradiction, suppose $\varphi$ is not deterministically manipulable. Suppose $\mu_1 \in \varphi(R)$. By individual rationality at all times, for each $\mu \in \varphi(R'_{w_1}, R_{w_1})$, $\mu(w_1) \neq m_1$. As $\mu$ is not state-wise manipulable, there is $\mu \in \varphi(R'_{w_1}, R_{w_1})$ such that $\mu(w_1) = w_1$, which, by efficiency at all times, implies $\mu(m_2) = w_2$. By efficiency at all times, $\varphi(R'_{w_1}, R'_{w_2}, R_{\{w_1,w_2\}}) = [\mu_2]$. This is a contradiction, as $w_2$ then state-wise manipulates $\varphi$ at $\varphi(R'_{w_2}, R_{w_2})$ through $R'_{w_2}$. Hence, $\mu_1 \notin \varphi(R)$. By repeating the exercise with consecutive misreports by $m_1$ and $m_2$, $\mu_2 \notin \varphi(R)$. This contradicts efficiency at all times.

Part 2: Let $M = \{m_1, m_2, m_3\}$, $W = \{w_1, w_2, w_3\}$, and refer to preferences in Table 12(ii). Define $\mu_1 = \{(m_1, w_1), (m_2, w_2), (m_3, w_3)\}$ and $\mu_2 = \{(m_1, w_2), (m_2, w_1), (m_3, w_3)\}$.

Let $\varphi$ be individually rational at all times and respect mutual best. To obtain a contradiction, suppose $\varphi$ is not state-wise manipulable. By individual rationality at all times, for each $\mu \in \varphi(R''_{m_1}, R_{m_1})$, $\mu(m_1) \neq w_2$. By respecting mutual best, for each $\mu \in \varphi(R''_{m_1}, R''_{w_2}, R_{\{m_1,w_2\}})$, $\mu(w_2) = m_2$. By respecting mutual best, for each $\mu \in \varphi(R''_{m_1}, R''_{w_2}, R_{\{m_1,w_1,w_2\}})$, $\mu(w_1) = m_1$. 

<table>
<thead>
<tr>
<th>$R_{m_1}$</th>
<th>$R_{m_2}$</th>
<th>$R_{m_3}$</th>
<th>$R_{w_1}$</th>
<th>$R_{w_2}$</th>
<th>$R_{w_3}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_1$</td>
<td>$w_2$</td>
<td>$w_1$</td>
<td>$m_2$</td>
<td>$m_1$</td>
<td>$m_1$</td>
</tr>
<tr>
<td>$w_2$</td>
<td>$w_1$</td>
<td>$m_1$</td>
<td>$m_2$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$w_3$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$R''_{m_1}$</th>
<th>$R''_{m_2}$</th>
<th>$R''_{w_1}$</th>
<th>$R''_{w_2}$</th>
<th>$R''_{w_3}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$w_1$</td>
<td>$m_2$</td>
<td>$m_2$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$w_3$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\phi$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 12(i): Preferences for Part 1.

Table 12(ii): Preferences for Parts 2 and 3.
By individual rationality at all times, for each \( \mu \in \varphi(R'_{m_1}, R''_{m_1}, R''_{m_2}, R_{-(m_1,w_1,w_2)}) \), \( m \mu(m_3) = m_3 \) and \( \mu(w_3) = w_3 \). Therefore, \( \varphi(R'_{m_1}, R''_{m_1}, R''_{m_2}, R_{-(m_1,w_1,w_2)}) = [\mu_1] \). If there is \( \mu \in \varphi(R'_{m_1}, R''_{m_2}, R_{-(m_1,w_1,w_2)}) \) such that \( \mu(w_1) \neq m_1 \), then \( w_1 \) state-wise manipulates \( \varphi \) at \( (R'_{m_1}, R''_{m_2}, R_{-(m_1,w_1,w_2)}) \) through \( R''_{m_2} \). Hence, \( \varphi(R'_{m_1}, R''_{m_2}, R_{-(m_1,w_1,w_2)}) = [\mu_1] \). If there is \( \mu \in \varphi(R'_{m_1}, R''_{m_1}) \) such that \( \mu(w_2) \neq m_2 \), then \( w_2 \) state-wise manipulates \( \varphi \) at \( (R'_{m_1}, R_{-m_1}) \) through \( R''_{m_1} \). Hence, \( \varphi(R'_{m_1}, R_{-m_1}) = [\mu_1] \).

By symmetric arguments, \( \varphi(R'_{w_1}, R_{-w_1}) = [\mu_2] \). But then \( \varphi \) is state-wise manipulable at \( R \). If \( \varphi(R) \neq [\mu_1] \), then \( m_1 \) state-wise manipulates \( \varphi \) at \( R \) through \( R'_{m_1} \). If \( \varphi(R) \neq [\mu_2] \), then \( w_1 \) state-wise manipulates \( \varphi \) at \( R \) through \( R''_{w_1} \). This is a contradiction.

**Part 3:** We show that if \( \varphi \) is ssd–strategy-proof and respects mutual best, then \( \varphi \) is individually rational at all times. To obtain a contradiction, suppose there is \( R \in \mathbb{R}^n \), \( \mu \in \varphi(R) \), and \( i \in N \) such that \( i \) \( P_i \mu(i) \). Let \( R'_i \) be such that, for each \( j \in N \), \( i R'_i j \). As \( \varphi \) respects mutual best, for each \( \mu \in \varphi(R'_i, R_{-i}) \), \( \mu(i) = i \). Then telling the truth \( R_i \) is not at least as desirable to \( i \) as telling the lie \( R'_i \). This contradicts ssd–strategy-proofness. Therefore, \( \varphi \) is individually rational at all times. The incompatibility now follows from Part 1.

**Appendix A.2. Proof of Theorem 2**

**Stability:** Let \( R \in \mathbb{R}^n \) denote a typical problem. To obtain a contradiction, suppose \( \Sigma \equiv CR(R) \) is not stable. Assume \( S \subseteq N \) block \( \Sigma \) to \( \Sigma' \), and that no \( T \subset S \) blocks \( \Sigma \). Let \( C \) be the first cycle containing an agent from \( S \), say agent \( i \). Label \( C = (i, i+1, \ldots, i+m) \). Define \( M \) to contain all agents except those that are part of cycles processed prior to \( C \). For all \( j \in M \), \((i + 1) P_i j \). Moreover, no agent in \( N \setminus M \) is part of \( S \) as \( i \) is the first to be. Note that if \#C \( \leq 2 \), then \( \sigma_i,i+1 = 1 \) and \( i \) is matched the entire time with her most preferred available agent. Agent \( i \) is not able to improve upon that. Additionally, \( S \setminus \{i, i+1\} \subseteq S \) can then also block, a contradiction. For \( i \) to block, we require therefore \((i + 1) \in S \) and \( \sigma'_i,i+1 \geq \sigma_i,i+1 = 1/2 \). For \( i+1 \) to block, we require \((i + 2) \in S \) and \( \sigma'_{i+1,i+2} \geq \sigma_{i+1,i+2} = 1/2 \). As \( \sigma'_{i+1,i+1} + \sigma'_{i+1,i+2} = 1 \), we have \( \sigma'_{i+1,i+1} = \sigma'_{i+1,i+2} = 1/2 = \sigma_{i+1,i+2} + \sigma_{i+1,i+2} = 1/2 \). Repeat the argument for the rest of \( C \). We find that \( C \subseteq S \). Moreover, each agent in \( C \) is matched under \( \Sigma' \) as she is under \( \Sigma \). Then no agent in \( C \) is better off. Hence, \( S \setminus C \) is matched entirely among themselves and includes someone better off under \( \Sigma' \). Then \( S \setminus C \subseteq S \) can block, a contradiction as no \( T \subset S \) was assumed to be able to block.

**Group–strategy-proofness:** To obtain a contradiction, suppose the \( CR \) rule is group-manipulable. Assume \( S \subseteq N \) can manipulate \( CR \) at \( R \in \mathbb{R}^n \) through \( R'_S \subseteq \mathbb{R}^S \), and that no \( T \subset S \) can manipulate. Denote \( \Sigma \equiv CR(R) \) and \( \Sigma' \equiv CR(R'_S, R_{-S}) \). Let \( C \) be the first cycle containing an agent from \( S \), say agent \( i \). Label \( C = (i, i+1, \ldots, i+m) \). We also use the labelling \( C = (i, i-m \equiv i+1, \ldots, i-1 \equiv i + m) \). Define \( M \) to contain all agents except those from cycles processed prior to \( C \). For all \( j \in M \), \( i+1 P_i j \). From Proposition 2, no matter in which order the cycles are processed, the final outcome is the same. In particular, no matter \( i \)’s reported preference, each cycle processed prior to \( C \) is still a cycle. Agent \( i \) can therefore not be matched to an agent in \( N \setminus M \). Note that if \#C \( \leq 2 \), then \( i \) is matched the entire time with her most preferred available agent. Agent \( i \) is not able to improve upon that. Additionally, \( S \setminus \{i, i+1\} \subseteq S \) can then also manipulate, a contradiction. For \( i \) to manipulate, we need therefore \( i+1 \in S \) and \( \sigma'_{i,i+1} \geq \sigma_{i,i+1} = 1/2 \).
Assume next \( S \cap C = \{ i \} \), that is, that \( i \) is the only manipulating agent in \( C \). Note that, no matter \( i \)'s reported preference, \( i - 1 \) will point to \( i \) as long as \( i \) is available. Similarly, \( i - 2 \) will point to \( i - 1 \) as long as \( i - 1 \) is available, which is as long as \( i \) is. Repeating the argument, \( i + 1 \) points to \( i + 2 \) as long as \( i \) is available. Therefore, no matter \( i \)'s report, the only way for \( i \) to be matched to \( i + 1 \) is by pointing to \( i + 1 \). This completes the cycle \( C \) and \( i \) is not better off. As above, \( S \setminus \{ i \} \subset S \) then can manipulate, a contradiction. Hence, there are multiple agents in \( S \cap C \).

Recall, for \( i \) to manipulate, \( \sigma'_{i,i+1} \geq 1/2 \). Suppose \( i \) is matched to \( i + 1 \) through \( i \) pointing to \( i + 1 \). Note that \( i + 2 \) is \( i + 1 \)'s most preferred available agent. Given that \( \sigma'_{i,i+1} \geq 1/2 \), \( i + 1 \) can do no better than reporting his preference truthfully (as then \( \sigma'_{i+1,i+2} = 1/2 = \sigma_{i+1,i+2} \)). However, now we can repeat the argument for all agents in \( C \). Each of them is matched under \( \Sigma' \) as under \( \Sigma \). Then \( S \setminus C \subset S \) can manipulate, a contradiction. Hence, for \( i \) to manipulate, it must be that \( i \) is matched to \( i + 1 \) through \( i + 1 \) pointing to \( i \). Then \( i + 1 \) is part of the manipulating group, hence \( i + 1 \in S \). As \( \sigma_{i+1,i+2} = 1/2 \), \( i + 2 \) needs to match to \( i + 1 \) the remaining time. Hence, \( i + 2 \) points to \( i + 1 \), and \( i + 2 \in S \). When we repeat the argument, we find that all agents in \( C \) point in the opposite direction. All of them are matched exactly as if they reported their true preference. Again, \( S \setminus C \subset S \) can then manipulate, a contradiction.

Side-neutrality and anonymity are both immediate. No agent nor any side receives any “special” treatment. \( \Box \)

Appendix A.3. Proof of Theorem 3

We wish to first show that, for all \( \{ i, j \} \subseteq N \),

\[
\sum_{kP_{i,j}} \sigma_{ik} + \sum_{kP_{j,i}} \sigma_{jk} + \sigma_{ij} \geq 1, \tag{1}
\]

where \( \sigma \) is the matrix representation of \( \Sigma \). Suppose, to obtain a contradiction, for some \( \{ i, j \} \subseteq N \) the inequality is not true. Then,

\[
\sum_{kP_{i,j}} \sigma_{ik} + \sum_{kP_{j,i}} \sigma_{jk} + \sigma_{ij} < 1,
\]

and therefore there exists some \( \mu \in \Sigma \) such that neither \( \mu(i) P_j \), \( \mu(j) P_j i \), nor \( \mu(i) = j \). But then \( \{ i, j \} \) block \( \mu \), a contradiction as \( \mu \) is stable.

By contradiction, suppose \( S \subseteq N \) block \( \Sigma \) to \( \Sigma' \in \mathcal{S} \) with matrix representation \( \sigma' \). For all \( \{ i, j \} \subseteq S \), as \( \Sigma' R^\text{sd}_i \Sigma \) and \( \Sigma' R^\text{sd}_j \Sigma \),

\[
\sum_{kP_{i,j}} \sigma'_{ik} \geq \sum_{kP_{i,j}} \sigma_{ik} \text{ and } \sum_{kP_{j,i}} \sigma'_{ik} + \sigma'_{ij} \geq \sum_{kP_{j,i}} \sigma_{ik} + \sigma_{ij}.
\]

Adding these inequalities and using (1),

\[
\sum_{kP_{i,j}} \sigma'_{ik} + \sum_{kP_{j,i}} \sigma'_{ik} + \sigma'_{ij} \geq \sum_{kP_{i,j}} \sigma_{ik} + \sum_{kP_{j,i}} \sigma_{ik} + \sigma_{ij} \geq 1.
\]

33
Let $i \in S$ be such that $\Sigma' P^{ssd}_i \Sigma$. Then $\sigma'_{ij} > \sigma_{ij}$ for some $j \in S$. Therefore,

$$\sum_{kP_j} \sigma'_{ik} + \sum_{kP_j} \sigma'_{jk} + \sigma'_{ij} > \sum_{kP_j} \sigma'_{ik} + \sum_{kP_j} \sigma'_{j1} + \sigma_{ij} \geq \sum_{kP_j} \sigma_{ik} + \sum_{kP_j} \sigma_{jk} + \sigma_{ij} \geq 1$$

(2)

In parallel, consider the problem $(S, R_S)$. For each $\mu \in \Sigma'$, denote the projection of $\mu$ onto $S$ by $\mu_S$. To be more precise, for all $i \in S$, $\mu_S(i) = \mu(i)$. As $S$ block $\Sigma$ to $\Sigma'$, $\mu(i) \in S$ for all $i \in S$. Hence $\mu_S$ is a well-defined matching in the problem $(S, R_S)$. Let $\Sigma'_S \equiv \{ \mu_S : \mu \in \Sigma' \}$ with matrix representation $\sigma'_S$. Note that, for all $(i, j) \subseteq S$,

$$\sum_{kP_j} \sigma'_{Sij} = \sum_{kP_j} \sigma'_{ij} \text{ and } \sigma'_{Sij} = \sigma'_{ij}.$$  

As $\sigma'_{ij} > \sigma_{ij} \geq 0$, $\mu(i) = j$ for some $\mu \in \Sigma'$, and hence $\mu_S(i) = j$ for some $\mu_S \in \Sigma'_S$. By Lemma 3 applied to $(S, R_S)$ and $\Sigma'_S$,

$$\sum_{kP_j} \sigma'_{Sik} + \sum_{kP_j} \sigma'_{Sjk} + \sigma'_{Sij} = 1,$$

where the left hand side equals

$$\sum_{kP_j} \sigma'_{ik} + \sum_{kP_j} \sigma'_{jk} + \sigma'_{ij} = 1,$$

This contradicts (2). Hence, $S$ cannot block $\Sigma$, and $\Sigma$ is therefore stable.

Appendix A.4. Proof of Theorem

Stability: To obtain a contradiction, suppose $S \subseteq N$ at $R \in \mathcal{E}$ block $\Sigma \equiv \Lambda PDA(R)$ through $\Sigma' \in \mathcal{P}^*$. Assume $S$ is minimal, in the sense that no $T \subset S$ can block $\Sigma$. Let $i$ be an arbitrary agent in $S$. Suppose $i$ is part of the cycle $C$. Label $C = (i, i + 1, \ldots, i + m \equiv i - 1)$. Hence, $\sigma_{i,i+1} = \sigma_{i,i-1} = 1/2$. For each agent $j$ preferred by $i$ to both $i + 1$ and $i - 1$, $j$ rejected $i$’s proposal. Moreover, $j$ did not propose to $i$. Hence, at $\Sigma$, $j$ is matched only to agents preferred to $i$. Therefore $i$ cannot match to $j$ at $\Sigma' (\text{then } j \in S, \text{ but } j \text{ would not find } \Sigma' \text{ at least as good as } \Sigma)$.

Case 1: Suppose $i + 1 P_i i - 1$. For $\Sigma' R^{ssd}_{i} \Sigma$, we require $\sigma'_{i,i+1} \geq \sigma_{i,i+1} = 1/2$. Therefore $i + 1 \in S$. From Lemma 1(i), $i + 2 P_{i+1}i$. As $i$ is chosen arbitrarily, the argument applies to $i + 1$ as well. Hence, $i + 1$ cannot match to some agent preferred to $i + 2$ at $\Sigma'$. For $\Sigma' R^{ssd}_{i+1} \Sigma$, we require $\sigma'_{i,i+1} \leq \sigma_{i,i+1}$. Hence, $\sigma'_{i,i+1} = 1/2$. Moreover, we require $\sigma'_{i+1,i+2} \geq \sigma_{i+1,i+2} = 1/2$. Repeat the argument for $i + 2$. We find that $i + 1$ is matched identically under $\Sigma'$ as under $\Sigma$. Moreover, by repeating the argument for each agent in $C$, we find that $C \subseteq S$. Importantly, each agent in $C$ is matched in the same way under $\Sigma'$ as under $\Sigma$. Hence, no agent in $C$ is better off. Then $S \setminus C \subset S$ can block $\Sigma$. This is a contradiction.

Case 2: Suppose $i - 1 P_i i + 1$. Instead we make use of part (ii) of Lemma 1. Otherwise, the proof is as in Case 1.

Anonymity and individual rationality at all times are immediate.
Appendix A.5. Proof of Theorem

Strategy-proofness: The statement is basically proven by repeating the arguments presented in the section on the two-sided problem. The difference is here that a supposedly manipulating agent \( i \) can be part of multiple cycles due to the splits of the procedure, rather than just one. However, for each of those cycles, by the same logic as before, \( i \) cannot gain by misreporting his preference.

To obtain a contradiction, say \( i \) can manipulate. Let \( C = (i, i + 1, \ldots, i + m \equiv i - 1) \) be an arbitrary cycle that includes \( i \). No matter \( i \)'s reported preference, all cycles \( C' \) processed prior to \( C \) are cycles. (We can find a “path” of cycles chosen from the start of the algorithm to the point when \( C \) is chosen. It can be helpful to have the picture of the timeline in mind here.) Hence, \( i \) cannot match with the agents that are unavailable when \( C \) is chosen.

\( i \) can misreport his preference in such a way that he is taken as part of a different cycle \( D = (i, i' + 1, \ldots, i' + m' \equiv i' - 1) \) such that \( i' - 1 \ P_i i - 1 \) and \( i + 1 \ P_i i' + 1 \). However, to compensate for this loss of time spent with \( i + 1 \), \( i \) would have to be taken as part of a different cycle at a different occasion as well. At that occasion, he would similarly swap the agent pointing to him at the expense of time with his, at the time, most preferred agent. This requires yet a different cycle where \( i \) changes. However, as there is a finite number of cycles, \( i \) will in the end not be able to make up for the loss of time with his, at the time, most preferred agent. Hence, \( i \) cannot manipulate.

2-stability: Let \( \hat{\Sigma} \) be the sequence constructed. Clearly, no agent \( i \) can block \( \hat{\Sigma} \) on her own, as if \( i \) always is single she cannot strictly improve upon \( \hat{\Sigma} \), and otherwise \( i \) must at some point be matched to some \( j \ P_i i \). To obtain a contradiction, suppose \( \{i, j\} \subseteq N \) block \( \hat{\Sigma} \) to \( \Sigma' \). It is without loss to assume \( \Sigma' = [\mu] \) such that \( \mu(i) = j \). Then, \( \Sigma' R_i^ssd \hat{\Sigma} \) implies \( j R_i \mu(i) \) for each \( \mu \in \hat{\Sigma} \). Moreover, say \( i \) is chosen as part of a cycle in the algorithm no later than \( j \) is. Hence, \( j \) is available at the time. If \( i \) points to some \( k \neq j \), then \( l \ P_i j \) and there exists \( \mu \in \hat{\Sigma} \) such that \( \mu(i) = k \ P_i j \), a contradiction. Hence, \( i \) points to \( j \). Then, \( i \) and \( j \) are chosen in the same cycle. Applying the same argument to \( j \), if \( j \) were to point to someone else than \( i \), we obtain a contradiction. Hence, the cycle chosen contains only \( i \) and \( j \). But then the same argument applies to the next time, if any, \( i \) is chosen as part of a cycle. Hence, \( i \) and \( j \) must always be matched, and hence neither \( i \) nor \( j \) prefers \( \Sigma' \) to \( \hat{\Sigma} \). This is a contradiction.

5-stability: By Lemma 2, it is immediate that groups \( S \) of size 1, 2, and 3 cannot block. The first cycle needs to contain at least three agents, though there must also be agents in \( S \) that are not in the cycle. Suppose \( S = \{1, 2, 3, 4\} \) can block, and the “first” cycle is \( (1, 2, 3) \). Then \( 1 \) still needs to be matched a third with his most preferred agent, \( 2 \), when blocking. The same goes for \( 2 \) with \( 3 \) and \( 3 \) with \( 1 \). The remaining time \( 1 \) needs to be matched with agent 4 when blocking. Hence, \( 1 \) cannot be taken in a cycle with agents he prefers to 4. The same goes for agents \( 2 \) and \( 3 \) in the other splits. Therefore agent 4 cannot be matched with anyone 4 prefers to 1, 2 and 3. But then no one can be strictly better off when blocking. This is a contradiction. The proof that \( S = \{1, 2, 3, 4, 5\} \) cannot block is similar, but we need to consider more cases. For now, it is available upon request.
Appendix B. Additional results

**Proposition 1.** Let \( R \in \mathcal{R}^n \) be a general pairing problem. The sequence \( \Sigma \in \mathcal{S}^* \) is 2-stable if and only if there is no \( \mu \in \mathcal{M}^* \) and \( S \subseteq N \) such that \( S \) block \( \Sigma \) through \( [\mu] \).

**Proof.** If \( S \) block \( \Sigma \) through \( [\mu] \), then there exists \( i \in S \) and \( j \equiv \mu(i) \) such that \( [\mu] \) \( P^{ssd}_i \) \( \Sigma \) and \( [\mu] \) \( R^{ssd}_j \) \( \Sigma \). Then \( \{i, j\} \) block \( \Sigma \) through \( [\mu] \). Hence \( \Sigma \) is not 2-stable.

Assume for each \( \mu \in \mathcal{M}^* \), there exists no \( S \subseteq N \) that blocks \( \Sigma \) through \( [\mu] \). We wish to show that (i) no \( \{i\} \subseteq N \) and (ii) no \( \{i, j\} \subseteq N \) can block \( \Sigma \). For case (i), if \( i \) blocks through \( \Sigma' \), then \( i \) is single at each matching in \( \Sigma' \). Let \( \mu \in \Sigma' \) be an arbitrary matching. Then \( i \) blocks \( \Sigma \) through \( [\mu] \), a contradiction. For case (ii), if \( \{i, j\} \) block through \( \Sigma' \), then either they are either single or matched together at each matching in \( \Sigma' \). If \( i \) \( P_j \) \( j \), then \( i \) can block on her own, a contradiction. Hence \( j \) \( P_j \) \( i \). Likewise, we must have \( i \) \( P_j \) \( j \). But then \( i \) and \( j \) can block through \( [\mu] \) for any \( \mu \in \Sigma' \) such that \( \mu(i) = j \). This is a contradiction. \( \square \)

**Proposition 2.** Algorithm 1 has the following properties.

(i) At each step, each \( i \in N \) is part of at most one cycle.

(ii) If \( C' \) is a cycle when cycle \( C \) is chosen, \( C' \) remains a cycle.

(iii) If \( D \) becomes a cycle after \( C \) is chosen, and \( D' \) becomes a cycle after \( C' \) is chosen, then \( D \cap D' = \emptyset \).

**Proof.** (i) If there is \( i \in C \cap C' \), then there is \( j \in C \cap C' \) such that \( j \) is followed by \( k \) in \( C \) and \( k' \neq k \) in \( C' \), requiring \( j \) to point to both \( k \) and \( k' \), a contradiction.

(ii) Each \( i \in C' \) points to \( (i + 1) \in C' \) such that \( (i + 1) \notin C \) as \( C \cap C' = \emptyset \) by part (i). Then \( i \) points to \( i + 1 \) after agents in \( C \) are removed as well.

(iii) First removing \( C \) leaves \( C' \) by part (ii) and \( D \) by assumption. Then removing \( C' \) leaves \( D \) by part (ii). Switching the order of \( C \) and \( C' \) leaves \( D' \), but the remaining agents are the same, hence \( D \) and \( D' \) are cycle when \( C \cup C' \) are removed. By part (i), \( D \cap D' = \emptyset \). \( \square \)

**Lemma 1.** Let \( C = (1, 2, \ldots, m) \) be a cycle encountered in Algorithm 2. (i) If \( 2 P_1 m \), then \( k + 1 \equiv k \) \( P_k k - 1 \) \( k \) \( (mod \ m) \) for all \( k = 1, 2, \ldots, m \). (ii) If \( m P_1 m \), then \( k - 1 \equiv k + 1 \) \( P_k k \) \( (mod \ m) \) for all \( k = 1, 2, \ldots, m \).

**Proof.** (i) Assume \( 2 P_1 m \). To obtain a contradiction, suppose \( 1 P_2 3 \). Prior to proposing to \( 3 \), \( 2 \)’s proposal to \( 1 \) was rejected. But then \( 1 \) should also have rejected \( m \)’s proposal. This is a contradiction. Hence, \( 3 P_2 1 \). To complete the proof, apply the same argument to agents \( 3, 4, \ldots, m \).

(ii) Assume \( m P_1 2 \). To obtain a contradiction, suppose \( 1 P_m m - 1 \). Prior to proposing to \( 2 \), \( 1 \)’s proposal to \( m \) was rejected. But then \( m \) should also have rejected \( m - 1 \)’s proposal. This is a contradiction. Hence, \( m - 1 P_m 1 \). To complete the proof, apply the same argument to agents \( m - 1, m - 2, \ldots, 2 \). \( \square \)

**Lemma 2.** Suppose \( S \subseteq N \) is a minimal group that can block the sequence selected by the General Compromises and Rewards rule. Consider a step of the algorithm where (a) all agents of \( S \) are available and (b) the cycle chosen, call it \( C \), includes members of \( S \). Then
• $C$ contains only members of $S$
• $C$ contains an odd number ($\geq 3$) of agents
• $C$ does not contain all members of $S$.

If there has been a split prior to the step, there may be multiple “first” cycles. Then, if $C$ is a cycle as described above for some part of the split and $D$ for another, $C$ and $D$ are agent-disjoint.

**Proof.** Assume $i \in S$ is a member of $C$, and $i$ points to $j$. To obtain a contradiction, suppose $j \not\in S$. Then $j P_t k$ for all $k \in S$. But then $i$ cannot be better off if $i$ has to be matched only within $S$. This is a contradiction, hence $j \in S$. Now, reapply the argument for $j$. By the finiteness of $N$ (and hence of $S$ and $C$), eventually we complete a cycle only within $S$. That is, $C$ contains only members of $S$.

To obtain a contradiction, suppose $C$ contains one or two agents. These agents get to match entirely with their most preferred agent of $S$. They cannot do better when $S$ blocks. Hence, $S \setminus C \subset S$ can block, a contradiction to $S$ being minimal. Suppose instead $C$ is even of length 4 or more. Then each agent in $C$ gets to spend half their time with their most preferred agent of $S$. When $S$ blocks, each agent in $C$ therefore will be matched in this way. We reach the same contradiction. Therefore, $C$ contains an odd number of agents.

To obtain a contradiction, suppose $C$ contains every member of $S$. Then each agent in $S$ gets to match $(\#S - 1)/2\#S$ with his most preferred agent of $S$. When blocking, each agent in $S$ needs to be single $1/\#S$ of the time. This cannot be an improvement over the sequence selected by the GCR rule. This again is a contradiction, hence $C$ does not contain all members of $S$.

Finally, suppose $C$ and $D$ share some agent, say $i$. Then $i$ will point to the same agent in both $C$ and $D$, say $j$. This is because $j$ is $i$’s most preferred agent of $S$. Repeat for $j$ and the rest of the agents of $C$ and $D$. We reach the conclusion that the cycles coincide if they overlap. Hence, if there are different “first” cycles, then they share no agents.

**Lemma 3.** Consider a generalized pairing problem with agents $N$ with preferences $R \in \mathcal{R}^n$. Let $\Sigma \in \mathcal{S}^*$ be such that, for all $\{i, j\} \subseteq N$,

$$\sum_{k \not\in i} \sigma_{ik} + \sum_{k \not\in j} \sigma_{jk} + \sigma_{ij} \geq 1.$$ 

Then, for all $\{i, j\} \subseteq N$ such that $\mu(i) = j$ for some $\mu \in \Sigma$,

$$\sum_{k \not\in i} \sigma_{ik} + \sum_{k \not\in j} \sigma_{jk} + \sigma_{ij} = 1.$$ 

**Proof.** The result can be deduced from Theorem 4.5 in Abeledo and Rothblum [1994].

**Proposition 3.** The CR rule is not ssd–strategy-proof.
Proof. Consider the two-sided problem with agents \( N = \{ m_1, m_2, m_3 \} \) and \( W = \{ w_1, w_2, w_3 \} \) with preferences in Table B.13.

In the sequence \( CR(R) \), \( m_1 \) matches half the time with \( w_1 \) and half the time with \( w_2 \). In the sequence \( CR(R', R-1) \), \( m_1 \) always matches with \( w_3 \). Telling the truth therefore is not better than telling a lie (though neither is telling a lie better than telling the truth).

Example 11: A sequence of efficient matchings need not be ssd-efficient. Here, we show that, if each matching in \( \Sigma \in \mathcal{S}^* \) is efficient, \( \Sigma \) may not be ssd-efficient. The result follows from modifying an example in Bogomolnaia and Moulin (2001). The agents are \( M = \{ m_1, m_2, m_3, m_4 \} \) and \( W = \{ w_1, w_2, w_3, w_4 \} \) with preferences in Table B.14. Consider \( RSP(R) \). There are 8! different orderings in \( \Omega \). The induced matrix is given by \( \sigma \) below. In it, the \( m \)th refers to agent \( m \in M \) and the \( w \)th column refers to agent \( w \in W \).

\[
\sigma = \begin{pmatrix}
15/32 & 1/32 & 15/32 & 1/32 \\
14/32 & 2/32 & 14/32 & 2/32 \\
1/32 & 15/32 & 1/32 & 15/32 \\
2/32 & 14/32 & 2/32 & 14/32 \\
\end{pmatrix}
\]

To the right is the corresponding matrix associated to \( \Sigma' = [\mu_1, \mu_2] \), where

\[
\begin{align*}
\mu_1 &= \{ (m_1, w_1), (m_2, w_3), (m_3, w_2), (m_4, w_4) \} \\
\mu_2 &= \{ (m_1, w_3), (m_2, w_1), (m_3, w_4), (m_4, w_2) \}.
\end{align*}
\]

As \( \Sigma' \) is a Pareto-improvement over \( \Sigma \), \( \Sigma \) is not ssd-efficient.

Appendix C. Tan’s (1991) stable partitions

A partition of \( N \) is \( A^1, A^2, \ldots \) such that, for all \( i \neq j \), \( A^i \cap A^j = \emptyset \) and \( \cup_i A^i = N \). As a special case, \( \mu \in \mathcal{M}^* \) induces a partition of \( N \) into pairs and singletons, \( \{1, \mu(1)\}, \{2, \mu(2)\}, \) and so on.
Tan (1991) considers also larger partition sets. A \emph{ring} is a list of agents $x_1, x_2, \ldots, x_m$ such that, for each $x_i$, $x_{i+1} \ P x_i \ x_{i-1} \ (\text{mod} \ m)$. A \emph{stable partition} $A^1, A^2, \ldots$ is such that (i) each partition set $A^i$ is either a single agent, a pair of agents, or corresponds to a ring, and (ii) for each $x_i \in A^k$ and each $y_j \in A^m$ such that $y_j \neq x_{i+1}$,

$$y_j \ P x_i \ x_{i-1} \Rightarrow y_{j-1} \ P y_j \ x_i.$$ 

If $A^k = \{x_i\}$, then $x_{i-1}$ refers to $x_i$. If $A^k = \{x_i, x_{i+1}\}$, then $x_{i-1}$ refers to $x_{i+1}$. Tan (1991) shows that every general pairing problem has a stable partition.