Cambridge University Press & Assessment
978-1-108-83199-4 — Online and Matching-Based Market Design
Edited by Federico Echenique , Nicole Immorlica , Vijay V. Vazirani , Foreword by Alvin E. Roth
Excerpt
More Information

PART ONE

Foundations of Market Design

Cambridge University Press & Assessment
978-1-108-83199-4 — Online and Matching-Based Market Design
Edited by Federico Echenique , Nicole Immorlica , Vijay V. Vazirani , Foreword by Alvin E. Roth
Excerpt
More Information

CHAPTER ONE

Two-Sided Markets: Stable Matching

Federico Echenique, Nicole Immorlica, and Vijay V. Vazirani

1.1 Introduction

The field of matching markets was initiated by the seminal work of Gale and Shapley on stable matching. Stable matchings have remarkably deep and pristine structural properties, which have led to polynomial time algorithms for numerous computational problems as well as quintessential game-theoretic properties. In turn, these have opened up the use of stable matching to a host of important applications.

This chapter will deal with the following four aspects:

1. Gale and Shapley's deferred acceptance algorithm for computing a stable matching; we will sometimes refer to it as the DA algorithm;
2. the incentive compatibility properties of this algorithm;
3. the fact that the set of all stable matchings of an instance forms a finite, distributive lattice, and the rich collection of structural properties associated with this fact;
4. the linear programing approach to computing stable matchings.

A general setting. A setting of the stable matching problem which is particularly useful in applications is the following (this definition is quite complicated because of its generality, and can be skipped on the first reading).

Definition 1.1. Let W be a set of n workers and F a set of m firms. Let c be a *capacity function* $c\colon F \to \mathbb{Z}_+$ giving the maximum number of workers that can be matched to a firm; each worker can be matched to at most one firm. Also, let $G = (W, F, E)$ be a bipartite graph on vertex sets W, F and edge set E. For a vertex v in G, let $N(v)$ denote the set of its neighbors in G. Each worker w provides a strict preference list $l(w)$ over the set $N(w)$ and each firm f provides a strict preference list $l(f)$ over the set $N(f)$. We will adopt the convention that each worker and firm prefers being matched to one of its neighbors to remaining unmatched, and it prefers remaining unmatched to being matched to a non-neighbor.[1] If a worker or firm remains unmatched, we will say that it is matched to $\perp$.

[1] An alternative way of defining preference lists, which we will use in Section 1.3.2 is the following. Each worker w has a preference list over $F \cup \{\perp\}$, with firms in $N(w)$ listed in the preference order of w, followed by $\perp$, followed by $(F \setminus N(w))$ listed in arbitrary order. Similarly, each firm f's preference list is over $W \cup \{\perp\}$.

We wish to study all four aspects stated for this setting. However, it would be quite unwise and needlessly cumbersome to study the aspects directly in this setting. It turns out that the stable matching problem offers a natural progression of settings, hence allowing us to study the aspects gradually in increasing generality.

1. **Setting I.** Under this setting $n = m$, the capacity of each firm is one and graph G is a complete bipartite graph. Thus in this setting each side, consisting of workers or firms, has a total order over the other side. This simple setting will be used for introducing the core ideas.
2. **Setting II.** Under this setting n and m are not required to be equal and G is arbitrary; however, the capacity of each firm is still one. The definition of stability becomes more elaborate, hence making all four aspects more difficult in this setting. Relying on the foundation laid in Setting I, we will present only the additional ideas needed.
3. **Setting III.** This is the general setting defined in Definition 1.1. We will give a reduction from this setting to Setting II, so that the algorithm and its consequences carry over without additional work.

1.2 The Gale–Shapley Deferred Acceptance Algorithm

In this section we will define the notion of a stable matching for all three settings and give an efficient algorithm for finding it.

1.2.1 The DA Algorithm for Setting I

In this setting, the number of workers and firms is equal, i.e., $n = m$, and each firm has unit capacity. Furthermore, each worker and each firm has a total order over the other side.

Notation. If *worker w prefers firm f to f'* then we represent this as $f \succ_w f'$; a similar notation is used for describing the preferences of a firm.

We next recall a key definition from graph theory. Let $G = (W, F, E)$ be a graph with equal numbers of workers and firms, i.e., $|W| = |F|$. Then, $\mu \subseteq E$ is a *perfect matching* in G if each vertex of G has exactly one edge of μ incident at it. If so, μ can also be viewed as a bijection between W to F. If $(w,f) \in \mu$ then we will say that μ *matches w to f* and use the notation $\mu(w) = f$ and $\mu(f) = w$.

Definition 1.2. Worker w and firm f form a *blocking pair* with respect to a perfect matching μ, if they prefer each other over their partners in μ, i.e., $w \succ f \mu(f)$ and $f \succ_w \mu(w)$.

If (w,f) form a blocking pair with respect to perfect matching μ then they have an incentive to secede from matching μ and pair up by themselves. The significance of the notion of stable matching, defined next, is that no worker–firm pair has an incentive to secede from this matching. Hence such matchings lie in the *core* of the particular instance; this key notion will be introduced in Chapter 3. For now, recall from cooperative game theory that the core consists of solutions under which no subset of the agents can gain more (i.e., with each agent gaining at least as much and at least one agent gaining strictly more) by seceding from the grand coalition. Additionally, in Chapter 3 we will also establish that stable matchings are efficient and individually rational.

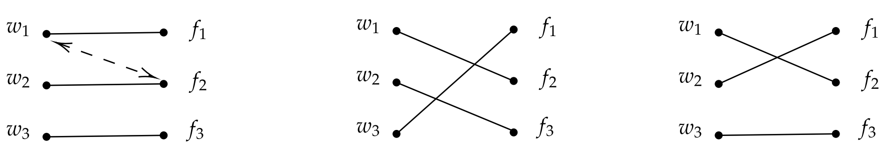
Definition 1.3. A perfect matching μ with no blocking pairs is called a *stable matching*.

It turns out that every instance of the stable matching problem with complete preference lists has at least one stable matching. Interestingly enough, this fact follows as a corollary of the deferred acceptance algorithm, which finds in polynomial time one stable matching among the $n!$ possible perfect matchings in G.

Example 1.4. Let I be an instance of the stable matching problem with three workers and three firms and the following preference lists:

$$\begin{array}{ll} w_1 : f_2, f_1, f_3 & \quad f_1 : w_1, w_2, w_3 \\ w_2 : f_2, f_3, f_1 & \quad f_2 : w_1, w_2, w_3 \\ w_3 : f_1, f_2, f_3 & \quad f_3 : w_1, w_3, w_2 \end{array}$$

Figure 1.1 shows three perfect matchings in instance I. The first matching is unstable, with blocking pair (w_1, f_2), and the last two are stable (this statement is worth verifying).



(a) An unstable perfect matching (b) Stable matching 1 (c) Stable matching 2

Figure 1.1

We next present the deferred acceptance algorithm[2] for Setting I, described in Algorithm 1.8. The algorithm operates iteratively, with one side proposing and the other side acting on the proposals received. We will assume that workers propose to firms. The initialization involves each worker marking each firm in its preference list as *uncrossed.*

Each iteration consists of three steps. First, each worker proposes to the best uncrossed firm on its list. Second, each firm that got proposals tentatively accepts the best proposal it received and rejects all other proposals. Third, each worker who was rejected by a firm crosses that firm off its list. If in an iteration each firm receives a proposal, we have a perfect matching, say μ, and the algorithm terminates.

The following observations lead to a proof of correctness and running time.

Observation 1.5. *If a firm gets a proposal in a certain iteration, it will keep getting at least one proposal in all subsequent iterations.*

[2] The reason for this name is provided in Remark 1.11.

Observation 1.6. *As the iterations proceed, for each firm the following holds: once it receives a proposal, it tentatively accepts a proposal from the same or a better worker, according to its preference list.*

Lemma 1.7. *Algorithm 1.8 terminates in at most n^2 iterations.*

Proof In every iteration other than the last, at least one worker will cross a firm off its preference list. Consider iteration number $n^2 - n + 1$, assuming the algorithm has not terminated so far. Since the total size of the n preference lists is n^2, there is a worker, say w, who will propose to the last firm on its list in this iteration. Therefore by this iteration w has proposed to every firm and every firm has received a proposal. Hence, by Observation 1.5, in this iteration every firm will get a proposal and the algorithm will terminate with a perfect matching. □

Algorithm 1.8. Deferred acceptance algorithm

Until all firms receive a proposal, do:

1. $\forall w \in W$: w proposes to its best uncrossed firm.
2. $\forall f \in F$: f tentatively accepts its best proposal and rejects the rest.
3. $\forall w \in W$: If w got rejected by firm f, it crosses f off its list.

Output the perfect matching, and call it μ.

Example 1.9. The Figures 1.2 shows the two iterations executed by Algorithm 1.8 on the instance given in Example 1.4. In the first iteration, w_2 will get rejected by f_2 and will cross it from its list. In the second iteration, w_2 will propose to f_3, resulting in a perfect matching.

Figure 1.2

Theorem 1.10. *The perfect matching found by the DA algorithm is stable.*

Proof For the sake of contradiction assume that μ is not stable and let (w, f') be a blocking pair. Assume that $\mu(w) = f$ and $\mu(f') = w'$ as shown in Figure 1.3. Since (w, f') is a blocking pair, w prefers f' to f and therefore must have proposed to f' and been rejected in some iteration, say i, before eventually proposing to f. In iteration i, f' must have tentatively accepted the proposal from a worker it likes better than w. Therefore, by Observation 1.6, at the termination of the algorithm, $w' \succ_{f'} w$. This contradicts the assumption that (w,f') is a blocking pair. □

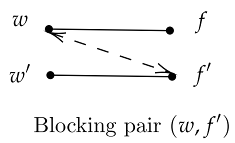


Figure 1.3 Blocking pair (w, f').

Remark 1.11. The Gale–Shapley algorithm is called the *deferred acceptance algorithm* because firms do not immediately accept proposals received by them – they defer them and accept only at the end of the algorithm when a perfect matching is found. In contrast, under the immediate acceptance algorithm, each firm immediately accepts the best proposal it has received; see Chapter 3.

Our next goal is to prove that the DA algorithm, with workers proposing, leads to a matching that is favorable for workers and unfavorable for firms. We first formalize the terms "favorable" and "unfavorable."

Definition 1.12. Let S be the set of all stable matchings over (W, F). For each worker w, the *realm of possibilities* $R(w)$ is the set of all firms to which w is matched in S, i.e., $R(w) = \{f \mid \exists \mu \in S \text{ s.t. } (w,f) \in \mu\}$. The *optimal firm* for w is the best firm in $R(w)$ with respect to w's preference list; it will be denoted by optimal(w). The *pessimal firm* for w is the worst firm in $R(w)$ with respect to w's preference list and will be denoted by pessimal(w). The definitions of these terms for firms are analogous.

Lemma 1.13. *Two workers cannot have the same optimal firm, i.e., each worker has a unique optimal firm.*

Proof Suppose that this is not the case and suppose that two workers w and w' have the same optimal firm, f. Assume without loss of generality that f prefers w' to w. Let μ be a stable matching such that $(w,f) \in \mu$ and let f' be the firm matched to w' in μ. Since $f = \text{optimal}(w')$ and w' is matched to f' in a stable matching, it must be the case that $f \succ_{w'} f'$. Then (w',f) forms a blocking pair with respect to μ, leading to a contradiction. See Figure 1.4. □

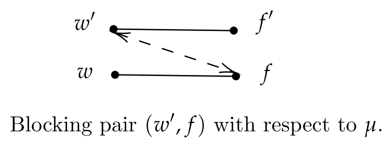


Figure 1.4 Blocking pair (w', f) with respect to μ.

Corollary 1.14. *Matching each worker to its optimal firm results in a perfect matching, say μ_W.*

Lemma 1.15. *The matching μ_W is stable.*

Proof Suppose that this is not the case and let (w,f') be a blocking pair with respect to μ_W, where $(w,f),(w',f') \in \mu_W$. Then $f' \succ_w f$ and $w \succ_{f'} w'$.

Since optimal$(w') = f'$, there is a stable matching, say μ', s.t. $(w',f') \in \mu'$. Assume that w is matched to firm f'' in μ'. Now since optimal$(w) = f$, $f \succ_w f''$. This together with $f' \succ_w f$ gives $f' \succ_w f''$. Then (w,f') is a blocking pair with respect to μ', giving a contradiction. See Figure 1.5. □

(a) Blocking pair (w,f') with respect to μ_W (b) Blocking pair (w,f') with respect to μ'

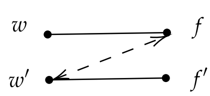
Figure 1.5

Proofs similar to those of Lemmas 1.13 and 1.15 show that each worker has a unique pessimal firm and the perfect matching that matches each worker to its pessimal firm is also stable.

Definition 1.16. The perfect matching that matches each worker to its optimal (pessimal) firm is called the *worker-optimal (-pessimal) stable matching*. The notions of *firm-optimal (-pessimal) stable matching* are analogous. The worker and firm optimal stable matchings will be denoted by μ_W and μ_F, respectively.

Theorem 1.17. *The worker-proposing DA algorithm finds the worker-optimal stable matching.*

Proof Suppose that this is not the case; then there must be a worker who is rejected by its optimal firm before proposing to a firm it prefers less. Consider the first iteration in which a worker, say w, is rejected by its optimal firm, say f. Let w' be the worker that firm f tentatively accepts in this iteration; clearly, $w' \succ_f w$. By Lemma 1.13, optimal$(w') \neq f$ and, by the assumption made in the first sentence of this proof, w' has not yet been rejected by its optimal firm (and perhaps never will be). Therefore, w' has not yet proposed to its optimal firm; let the latter be f'. Since w' has already proposed to f, we have that $f \succ_{w'} f'$. Now consider the worker-optimal stable matching μ; clearly, (w,f), $(w',f') \in \mu$. Then (w',f) is a blocking pair with respect to μ, giving a contradiction. See Figure 1.6. □



Blocking pair (w',f) with respect to μ

Figure 1.6 Blocking pair (w',f) with respect to μ.

Lemma 1.18. *The worker-optimal stable matching is also firm pessimal.*

Proof Let μ be the worker-optimal stable matching and suppose that it is not firm pessimal. Let μ' be a firm-pessimal stable matching. Now, for some $(w,f) \in \mu$, pessimal(f) $\neq w$. Let pessimal(f) $= w'$; clearly, $w \succ_f w'$. Let $w =$ pessimal(f'); then $(w,f'), (w',f) \in \mu'$. Since optimal(w) $= f$ and w is matched to f' in a stable matching, $f \succ_w f'$. Then (w,f) forms a blocking pair with respect to μ', giving a contradiction. □

1.2.2 Extension to Setting II

Recall that in this setting each worker and firm has a total preference order over only its neighbors in the graph $G = (W,F,E)$ and $\perp$, with $\perp$ the least preferred element in each list; matching a worker or firm to $\perp$ is equivalent to leaving it unmatched.

In this setting, a stable matching may not be a perfect matching in G even if the number of workers and firms is equal; however, it will be a maximal matching. Recall that a matching $\mu \subseteq E$ is *maximal* if it cannot be extended with an edge from $E - \mu$. As a result of these changes, in going from Setting I to Setting II, the definition of stability also needs to be enhanced.

Definition 1.19. Let μ be any maximal matching in $G = (W,F,E)$. Then the pair (w,f) forms a *blocking pair* with respect to μ if $(w,f) \in E$ and one of the following holds:

- **Type 1.** w,f are both matched in μ and prefer each other to their partners in μ.
- **Type 2a.** w is matched to f', f is unmatched, and $f \succ_w f'$.
- **Type 2b.** w is unmatched, f is matched to w', and $w \succ_f w'$.

Observe that, since $(w,f) \in E$, w and f prefer each other to remaining unmatched. Therefore they cannot both be unmatched in μ – this follows from the maximality of the matching.

The only modification needed to Algorithm 1.8 is to the termination condition; the modification is as follows. Every worker is either tentatively accepted by a firm or has crossed off all firms from its list. When this condition is reached, each worker in the first category is matched to the firm that tentatively accepted it and the rest remain unmatched. Let μ denote this matching. We will still call this the deferred acceptance algorithm. It is easy to see that Observations 1.5 and 1.6 still hold and that Lemma 1.7 holds with a bound nm on the number of iterations.

Lemma 1.20. *The deferred acceptance algorithm outputs a maximal matching in G.*

Proof Assume that $(w,f) \in E$ but that so far worker w and firm f are both unmatched in the matching found by the algorithm. During the algorithm, w must have proposed to f and been rejected. Now, by Observation 1.5, f must be matched, giving a contradiction. □

Theorem 1.21. *The maximal matching found by the deferred acceptance algorithm is stable.*

Proof We need to prove that neither type of blocking pair exists with respect to μ. For the first type, the proof is identical to that in Theorem 1.10 and is omitted. Assume that (w, f) is a blocking pair of the second type. There are two cases:

Case 1. w is matched, f is not, and w prefers f to its match, say f'. Clearly w will propose to f before proposing to f'. Now, by Observation 1.5, f must be matched in μ, giving a contradiction.

Case 2. f is matched, w is not, and f prefers w to its match, say w'. Clearly w will propose to f during the algorithm. Since f prefers w to w', it will not reject w in favor of w', hence giving a contradiction. □

Notation. If worker w or firm f is unmatched in μ then we will denote this as $\mu(w) = \perp$ or $\mu(f) = \perp$. We will denote the sets of workers and firms matched under μ by $W(\mu)$ and $F(\mu)$, respectively.

Several of the definitions and facts given in Setting I carry over with small modifications; we summarize these next. The definition of the *realm of possibilities* of workers and firms remains the same as before; however, note that in Setting II, some of these sets could be the singleton set $\{\perp\}$. The definitions of *optimal and pessimal firms for a worker* also remain the same, with the change that they will be $\perp$ if the realm of possibilities is the set $\{\perp\}$. Let $W' \subseteq W$ be the set of workers whose realm of possibilities is non-empty. Then, via a proof similar to that of Lemma 1.13, it is easy to see that two workers in W' cannot have the same optimal firm, i.e., every worker in W' has a unique optimal firm.

Next, match each worker in W' to its optimal firm, leaving the remaining workers unmatched. This is defined to be the *worker-optimal matching*; we will denote it by μ_W. Similarly, define the *firm-optimal matching*; this will be denoted by μ_F. Using ideas from the proof of Lemma 1.15, it is easy to show that the worker-optimal matching is stable. Furthermore, using Theorem 1.17 one can show that the deferred acceptance algorithm finds this matching. Finally, using Lemma 1.18, one can show that the worker-optimal stable matching is also firm pessimal.

Lemma 1.22. *The numbers of workers and firms matched in all stable matchings are the same.*

Proof Each worker w prefers being matched to one of the firms that is its neighbor in G over remaining unmatched. Therefore, all workers who are unmatched in μ_W will be unmatched in all other stable matchings as well. Hence for an arbitrary stable matching μ we have $W(\mu_W) \supseteq W(\mu) \supseteq W(\mu_F)$. Thus $|W(\mu_W)| \geq |W(\mu)| \geq |W(\mu_F)|$. A similar statement for firms is $|F(\mu_W)| \leq |F(\mu)| \leq |F(\mu_F)|$. Since the number of workers and firms matched in any stable matching is equal, $|W(\mu_W)| = |F(\mu_W)|$ and $|W(\mu_F)| = |F(\mu_F)|$. Therefore the cardinalities of all sets given above are equal, hence establishing the lemma. □