to hospital residencies and students to elementary schools. The following model and algorithm are directly useful for these and other applications with amazingly few modifications.

10.2.1 The Model

We consider two finite sets V and W of vertices—the "applicants" and the "hospitals"—with equal cardinality. Each vertex has a total ordering over the vertices of the other set. For example, in Figure 10.9, the applicants have a common ranking of the hospitals, while the hospitals have very different opinions about the applicants.

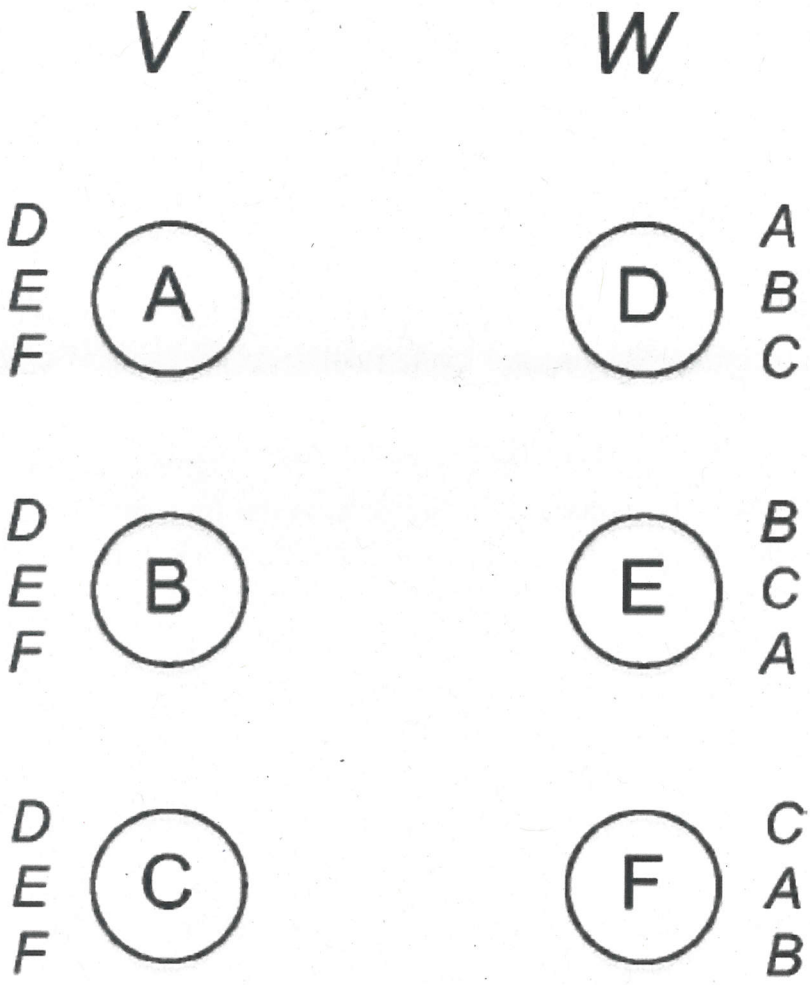

Figure 10.9: An instance of stable matching. Each vertex is annotated with its total ordering over the vertices of the opposite side, with the most preferred vertex on top.

Let M be a perfect matching of V and W, assigning each vertex to one vertex from the other set. Vertices $v \in V$ and $w \in W$ form a *blocking pair for* M if

they are not matched in M, v prefers w to her match in M, and w prefers v to its match in M. A blocking pair spells trouble, because the two vertices are tempted to secede from the process and match with each other. A perfect matching is *stable* if it has no blocking pairs.

10.2.2 The Deferred Acceptance Algorithm

We next discuss the elegant deferred acceptance algorithm for computing a stable matching.

Deferred Acceptance Algorithm

while there is an unmatched applicant $v \in V$ **do**

v attempts to match with her favorite hospital w who has not rejected her yet

if w is unmatched **then**

v and w are tentatively matched

else if w is tentatively matched to v' **then**

w rejects whomever of v, v' it likes less and is tentatively matched to the other one all tentative matches are made final

Example 10.4 (The Deferred Acceptance Algorithm)

Consider the instance in Figure 10.9. Suppose in the first iteration we choose the applicant C, who tries to match with her first choice, D. The hospital D accepts because it currently has no other offers. If we pick the applicant B in the next iteration, she also proposes to the hospital D. Since hospital D prefers B to C, it rejects C in favor of B. If we pick applicant A next, the result is similar: D rejects B in favor of A. A possible trajectory for the rest of the algorithm is: applicant C now proposes to her second choice, E; applicant B then also proposes to E, causing E to reject C in favor of B; and finally, C proposes to her last choice F, who accepts.

We note several properties of the deferred acceptance algorithm. First, each applicant systematically goes through her preference list, from top to bottom. Second, because a hospital only rejects an applicant in favor of a better one, the applicants to whom it is tentatively matched only improve over the course of the algorithm. Third, at all times, each applicant is matched to at most one hospital and each hospital is matched to at most one applicant.

Stable matchings and the deferred acceptance algorithm have an astonishing number of remarkable properties. Here are the most basic ones.

Theorem 10.5 (Fast Computation of a Stable Matching)

The deferred acceptance algorithm completes with a stable matching after at most n^2 *iterations, where n is the number of vertices on each side.*

Corollary 10.6 (Existence of a Stable Matching) *For every collection of preference lists for the applicants and hospitals, there exists at least one stable matching.*

Corollary 10.6 is not obvious a priori. For example, there are some simple variants of the stable matching problem for which a solution is not guaranteed.

Proof of Theorem 10.5: The bound on the number of iterations is easy to prove. Each applicant works her way down her preference list, never trying to match to the same hospital twice, resulting in at most n attempted matches per applicant and n^2 overall.

Next, we claim that the deferred acceptance algorithm always completes with every applicant matched to some hospital (and vice versa). For if not, some applicant must have been rejected by all n hospitals. An applicant is only rejected by a hospital in favor of being matched to a better applicant, and once a hospital is matched to an applicant, it remains matched to some applicant for the remainder of the algorithm. Thus, all n hospitals must be matched at the end of the algorithm. But then all n applicants are also matched at the end of the algorithm, a contradiction.

To prove the stability of the final matching, consider an applicant v and hospital w that are not matched to each other. This can occur for two different reasons. In the first case, v never attempted to match to w. Since v worked her way down her preference list starting from the top, she ends up matched to a hospital she prefers to w. If v did attempt to match to w at some point in the algorithm, it must be that w rejected v in favor of an applicant it preferred (either at the time that v attempted to match to w, or subsequently). Since the sequence of applicants to whom w is matched only improves over the course of the algorithm, it ends up matched to an applicant it prefers to v. ■

*10.3 Further Properties

The deferred acceptance algorithm is underdetermined, leaving open how the unmatched applicant is chosen in each iteration. Do all possible choices lead to the same stable matching? In Figure 10.9 there is only one stable matching, so in that example the answer is yes. In general, however, there can be more than one stable matching. In Figure 10.10, the applicants and the hospitals both disagree on the ranking of the others. In the matching computed by the deferred acceptance algorithm, both applicants get their first choice, with A and B matched to C and D, respectively. Giving the hospitals their first choices yields a different stable matching.