# OR 3: Chapter 15 - Matching games

## Recap

In the [previous chapter](Chapter_14_Stochastic_games.docx):

* We defined stochastic games;
* We investigated approached to obtaining Markov strategy Nash equilibria.

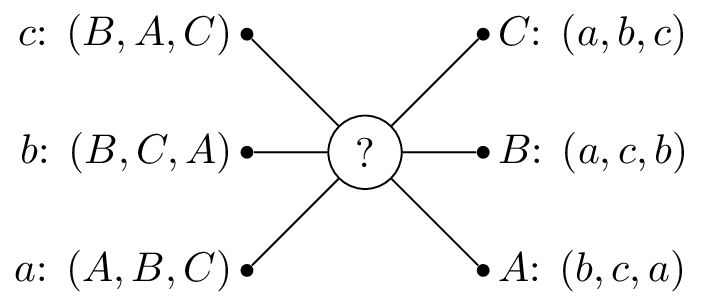
In this chapter we will take a look at a very different type of game.

## Matching Games

Consider the following situation:

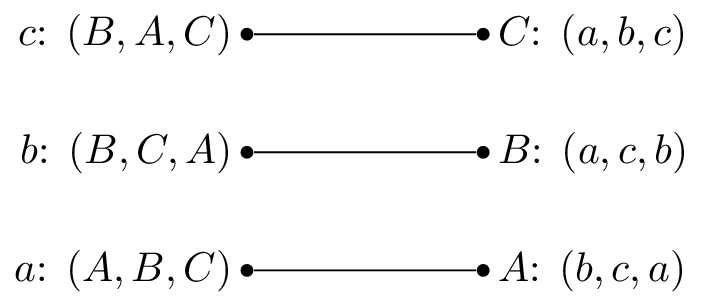
"In a population of suitors and reviewers. We allow the suitors and reviewers to rank their preferences and are now trying to match the suitors and reviewers in such a way as that every matching is stable."

If we consider the following example with suitors: and reviewers: with preferences shown.



A matching game.

So that would prefer to be matched with , then and lastly . One possible matching would be is shown.



A simple matching.

In this situation, and are getting their first choice and their second choice. However actually prefers so that matching is unstable.

Let us write down some formal definitions:

### Definition of a matching game

A matching game of size is defined by two disjoint sets and or suitors and reviewers of size . Associated to each element of and is a preference list:

A matching is a any bijection between and . If and are matched by we denote:

### Definition of a blocking pair

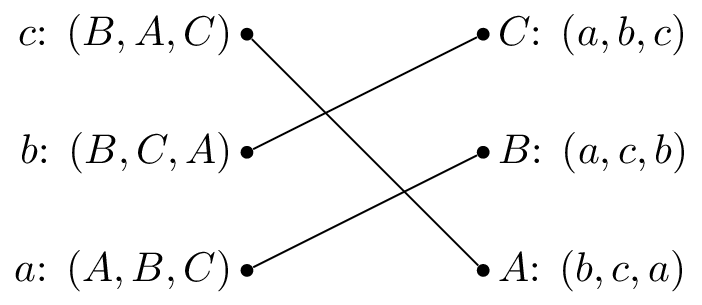
A pair is said to **block** a matching if but prefers to and prefers to .

In our previous example blocks the proposed matching.

### Definition of a stable matching

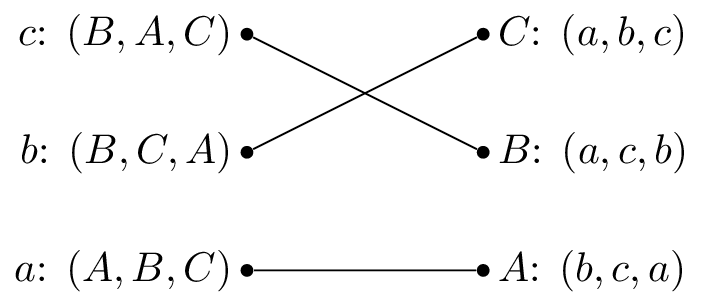
A matching with no blocking pair is said to be stable.

A stable matching is shown.



A stable matching.

The stable matching is not unique, the matching shown is also stable:



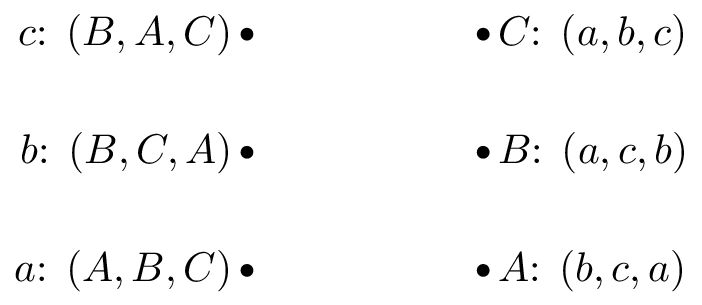
Another stable matching.

## The Gale-Shapley Algorithm

Here is the Gale-Shapley algorithm, which gives a stable matching for a matching game:

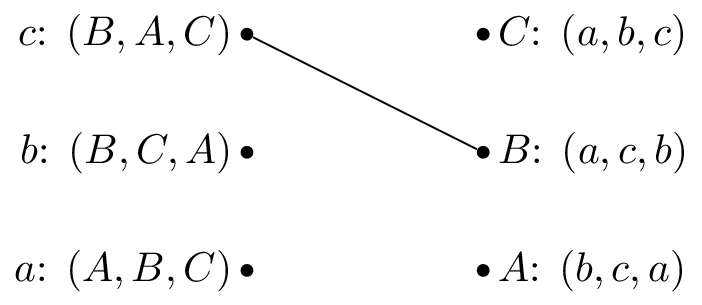
1. Assign every and to be unmatched
2. Pick some unmatched , let be the top of 's preference list:
   * If is unmatched set
   * If is matched:
     + If prefers to then set
     + Otherwise remains unmatched and remove from 's preference list.
3. Repeat step 2 until all are matched.

Let us illustrate this algorithm with the above example.



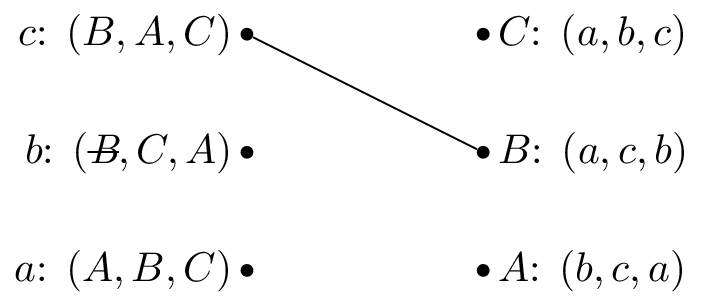
Base example.

We pick and as all the reviewers are unmatched set .



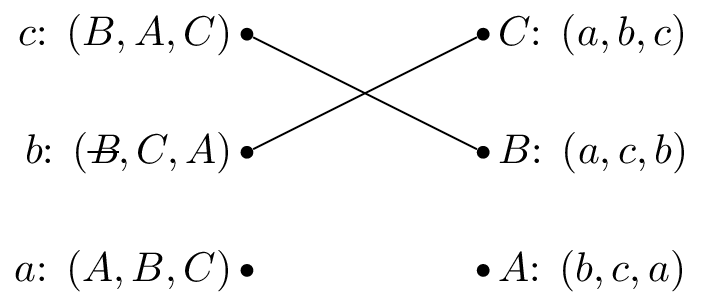
Setting .

We pick and as is matched but prefers to we cross out from 's preferences.



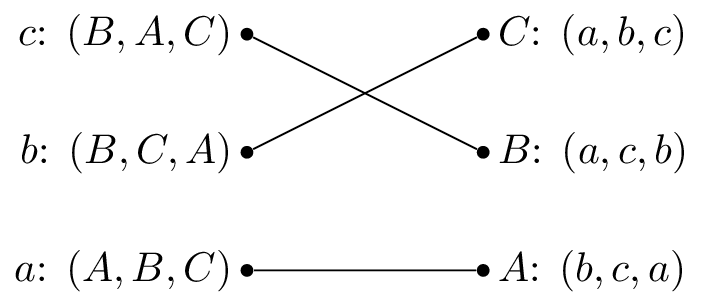
Removing from 's preference list.

We pick again and set .



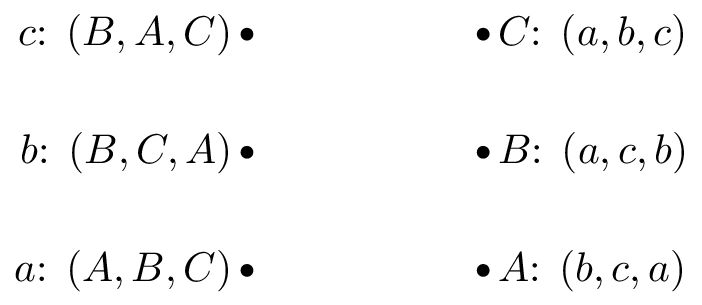
Setting .

We pick and set .



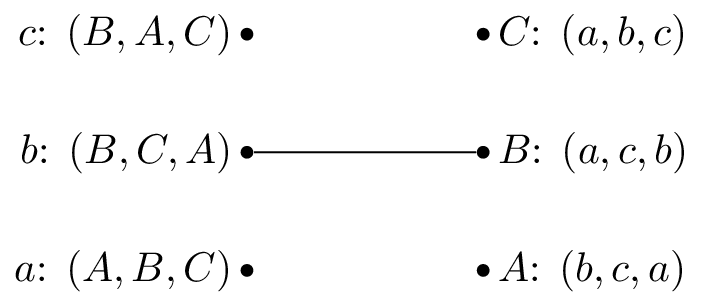
Setting .

**Let us repeat the algorithm but pick as our first suitor.**



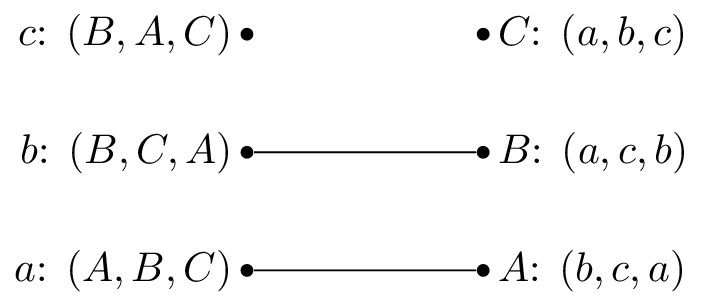
Base example.

We pick and as all the reviewers are unmatched set .



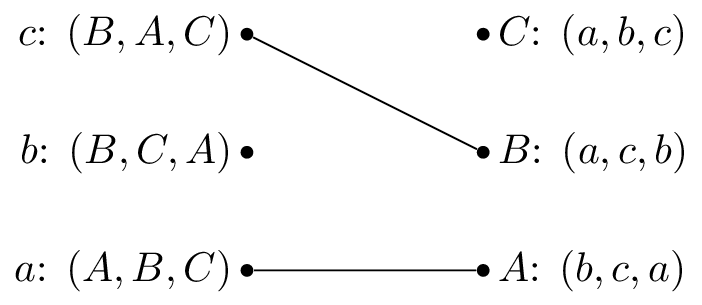
Setting .

We pick and as is unmatched set .



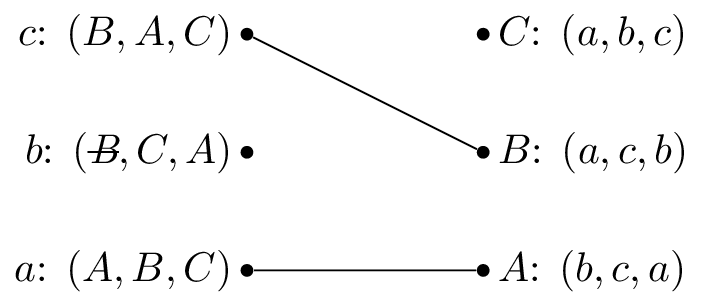
Setting .

We pick and is matched but prefers to , we set .



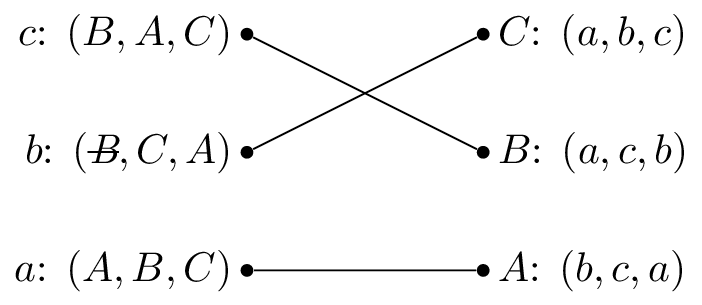
Setting .

We pick and as is matched but prefers to we cross out from 's preferences:



Removing from 's preference list.

We pick again and set .



Setting .

Both these have given the same matching.

### Theorem guaranteeing a unique matching as output of the Gale Shapley algorithm.

All possible executions of the Gale-Shapley algorithm yield the same stable matching **and** in this stable matching every suitor has the best possible partner in any stable matching.

### Proof

Suppose that an arbitrary execution of the algorithm gives and that another execution gives such that such that prefers to .

Without loss of generality this implies that during must have rejected . Suppose, again without loss of generality that this was the first occasion that a rejection occured during and assume that this rejection occurred because . This implies that has no stable match that is higher in 's preference list than (as we have assumed that this is the first rejection).

Thus prefers to so that blocks . Each suitor is therefore matched in with his favorite stable reviewer and since was arbitrary it follows that all possible executions give the same matching.

We call a matching obtained from the Gale Shapley algorithm *suitor-optimal* because of the previous theorem. The next theorem shows another important property of the algorithm.

### Theorem of reviewer sub optimality

In a suitor-optimal stable matching each reviewer has the worst possible matching.

### Proof

Assume that the result is not true. Let be a suitor-optimal matching and assume that there is a stable matching such that such that prefers to . This implies that blocks unless prefers to which contradicts the fact the has no stable match that he prefers in .