Chapter 9. Section 2

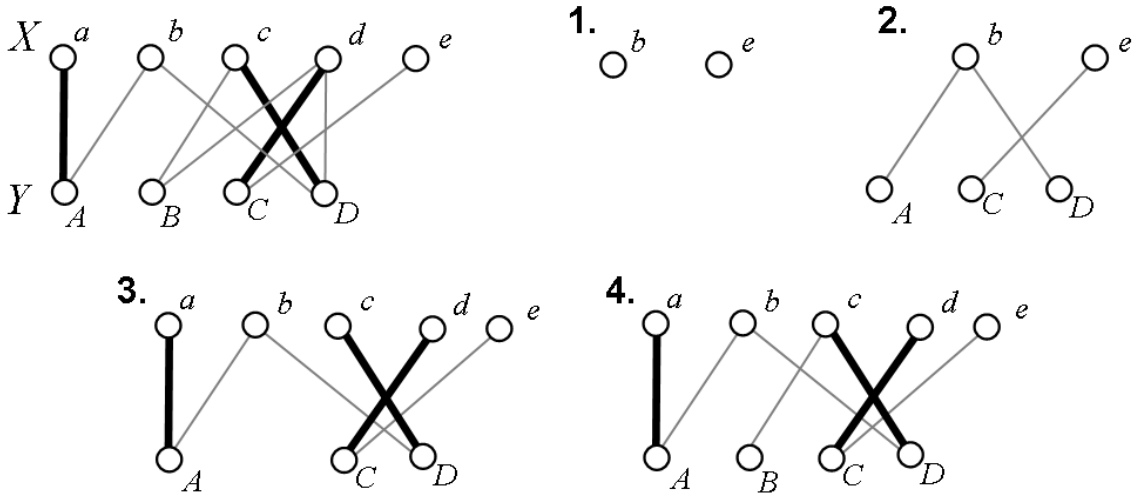
Kőnig's Bipartite Matching Algorithm

As usual, the heart of our method for finding a maximum matching is a tree-growing algorithm. Actually, it grows a forest. The algorithm grows the forest downwards using non-M edges and then back up **in** M:

```
Maximal_Alternating_Forest(G, M)
# Input: bipartite graph G with parts X and Y and matching M
    F := ({M−unsaturated vertices}, ∅)     # F consists of single vertices and no edges
    maximal:=FALSE;
    while not maximal do    # extend the forest
        if there is an edge e ∉ M leaving F from X then
            add all such edges to F     # non-M-edges going down from X
        elif there is an edge e ∈ M leaving F from Y then
            add all such edges to F     # M-edges going back up from X
        else maximal:=TRUE fi
    od;
    return(F)
```

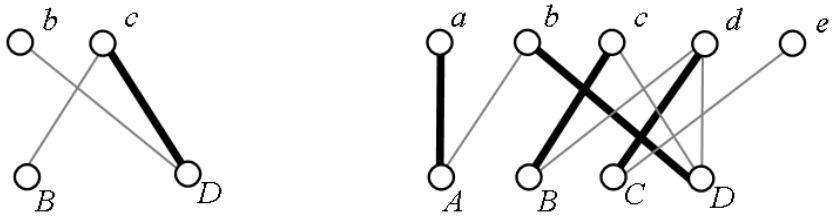
Example

In the graph and matching shown below left F is initialised to consist of the 2 M-unsaturated vertices b and e (Step 1). Then we complete our forest F in two iterations: steps 2 and 3 grow F down and back up again; then step 4 finds one further non-M edge cB.



Note that, in step 4, we **cannot** add the non-M-edges from vertex d since they are not leaving F. Indeed, after step 4 the forest is spanning and so is certainly maximal, so Maximal_Alternating_Forest terminates.

Building a maximal M-alternating forest works because: *either F contains an M-augmenting path or no such path exists in G.* Sure enough in our forest there *is* an M-augmenting path: it is shown below left, and is used to create a bigger matching, below right, as explained in Chapter 8, Section 3.

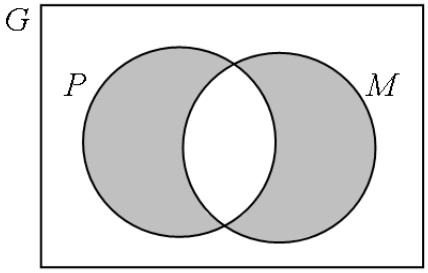


We now specify in pseudocode Kőnig's algorithm which uses Maximal_Alternating_Forest repeatedly to arrive at a maximum matching.

It will be convenient to have a neat notation for the way M and non-M edges are interchanged on an M-augmenting path. If P is such a path then the **symmetric difference** $P\Delta M$ denotes the set of edges containing all of P plus all of M, but with any overlap discarded. Formally:

$$P\Delta M = (P\backslash M) \cup (M\backslash P).$$

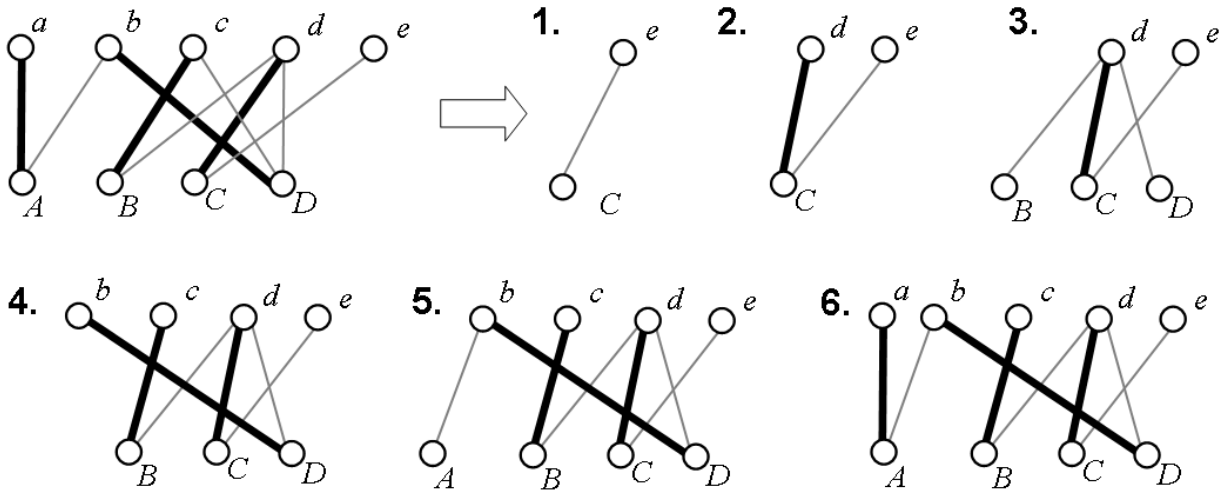
The symmetric difference operation has a standard Venn diagram picture which is worth keeping in mind when visualising $P\Delta M$:



However, it is really just a mathematical device for expressing "swap all the M and non-M edges on the path and keep the rest the same" in precise way.

```
Kőnigs_Bipartite_Matching_Algorithm(G)
# Input: bipartite graph G with parts X and Y
    M := ∅;     # begin with an empty matching
    maximum:=FALSE;
    while not maximum do     # build a new forest
        F :=Maximal_Alternating_Forest(G, M);
        if F contains an M-augmenting path P then M := PΔM
        else maximum:=TRUE
        fi
    od;
    return(M)
```

Our previous example is a snapshot of Kőnigs_Bipartite_Matching_Algorithm in action: having achieved a matching with three edges it builds a new forest, finds an augmenting path, and creates a new matching with four edges. This is clearly maximum since every vertex in the lower part Y is now contained in a matching edge. But the algorithm is not so intelligent — it will run a further iteration:



Sure enough, step 6 gives a maximal M-alternating forest which contains no M-augmenting path — how could it because there is only one M-unsaturated vertex! So this confirms our four-edge matching is maximum.