(b) Show that there are graphs for which the inequality ($\star$) is strict.

6. Suppose $G = (V,E)$ is a bipartite graph with partition U,W. Suppose also that $|U| = |W|$. Let H be the complete bipartite graph with partition U and W (the edges of H are *all* pairs ij where $i \in U$ and $j \in W$). Assign weights to the edges of H as follows: for every edge ij,

$$w_{ij} = \begin{cases} 1 & \text{if } ij \text{ is an edge of } G \\ 0 & \text{otherwise} \end{cases}$$

Recall that y is feasible if for every edge ij of H, $y_i + y_j \geq w_{ij}$.

a) Show that there exists a feasible $0,1$ vector y (i.e. all entries of y are 0 or 1).

b) Show that if y is integer at the beginning of an iteration of the matching algorithm then y will be integer at the beginning of the next iteration.

c) Show that there exists a feasible integer vector y and a matching M of G such that

$$\sum_{i \in V} y_i = |M|.$$

HINT: Look at what you have when the matching algorithm terminates.

d) Show that there exists a feasible $0,1$ vector y and a matching M of G such that

$$\sum_{i \in V} y_i \leq |M|.$$

HINT: Add (resp. substract) the same value α to all $y_i \in U$ (resp. $y_i \in W$).

e) Show that there exists a vertex cover S of G and a matching M of G such that $|M| = |S|$. In particular ($\star$) holds with equality for bipartite graphs.

3.2.6 *Finding perfect matchings in bipartite graphs**

Hall's Theorem 3.12 states that, in a bipartite graph G with bipartition U,W where $|U| = |W|$, there is an efficient algorithm that finds a perfect matching in G if one exists, and otherwise returns a deficient set $S \subseteq U$; i.e., a set S that has fewer than $|S|$ neighbours. In this section, we show how this can be accomplished.

We start by introducing a few more matching related terms and definitions. Consider a (possibly empty) matching $M \subseteq E$; we say that a path P in G is *M-alternating* if its edges

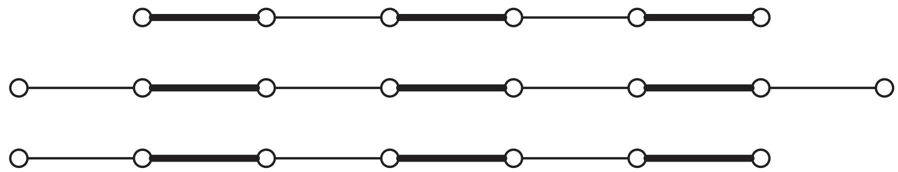


Figure 3.4: M-alternating paths (edges in M are bold).

alternate between matching and non-matching edges, or equivalently if $P \setminus M$ is a matching. Examples of alternating paths are given in Figure 3.4. Given a matching M, a vertex v is *M-covered* if M has an edge that is incident to v, otherwise v is *M-exposed*. In Figure 3.5(i) vertices $v_1, v_2, v_4, v_6, v_7, v_8$ are M-covered and vertices v_3, v_5 are M-exposed. In Figure 3.5(ii) all vertices are M-covered, or equivalently M is a perfect matching. An al-

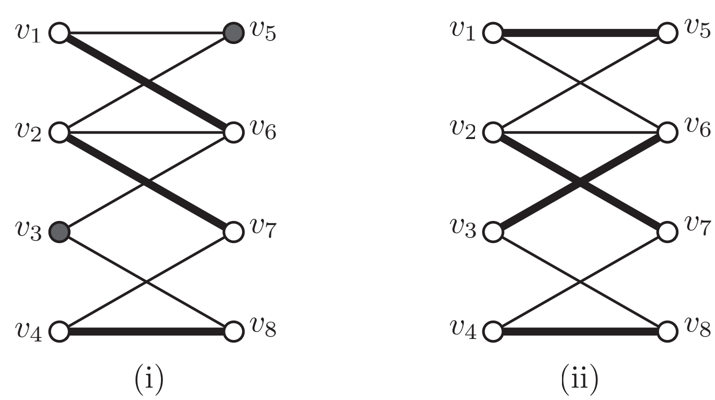


Figure 3.5: M-exposed and M-covered vertices (edges in M are bold).

ternating path P is *M-augmenting* if its endpoints are both M-exposed. In Figure 3.5(i) the path $P_1 = v_5v_1, v_1v_6, v_6v_3$ is M-augmenting as both v_5, v_3 are M-exposed, but the path $P_2 = v_1v_6, v_6v_2, v_2v_7, v_7v_4, v_4v_8, v_8v_3$ is M-alternating but not M-augmenting, as v_1 is M-covered. Observe that the matching M' in Figure 3.5(ii) is obtained from the matching M in Figure 3.5(i), by removing $v_1, v_6 \in P_1$ from M and adding edges $v_5v_1, v_6v_3 \in P_1$. In fact given an M-augmenting path, we can always construct a new matching M' where $|M'| = |M| + 1$. We formalize this operation next. For two sets A and B, we let $A \Delta B$ denote the set of those elements in $A \cup B$ that are not in both A and B; i.e., $A \Delta B = (A \setminus B) \cup (B \setminus A)$. A matching is a *maximum matching* if there is no matching with more edges.

We leave the following remark as an easy exercise,

Remark 3.15. *Let M be matching of a graph G and let P be an M-augmenting path. Then $M' := M\Delta P$ is a matching and $|M'| = |M| + 1$. In particular, M is not a maximum matching.*

For instance in Figure 3.5(i) we had $M = \{v_1v_6, v_2v_7, v_4v_8\}$, an augmenting path $P_1 = v_5v_1, v_1v_6, v_6v_3$ and a matching M' in (ii) where,

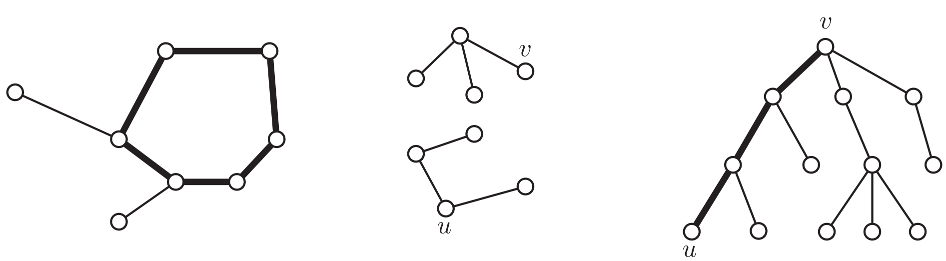
$$M' = \{v_1v_6, v_2v_7, v_4v_8\}\Delta\{v_5v_1, v_1v_6, v_6v_3\} = \{v_5v_1, v_2v_7, v_6v_3, v_4v_8\}.$$

In fact the converse of Remark 3.15 holds (but we will not need it), namely, that if a matching is not maximum then there must exist an augmenting path. Thus a plausible strategy for finding a perfect matching is as follows: start with the empty matching $M = \emptyset$. As long as there exists an M-augmenting path P, replace M by $M\Delta P$. If the final matching is not perfect then no perfect matching exists. Two problems remain however: how do we find an augmenting path, and if no augmenting path exist, how do we find a deficient set? The key to addressing both of these problem is the concept of M-alternating trees , to be defined next.

Let us start with a few definitions. A *cycle* is a sequence of edges,

$$C = v_1v_2, v_2v_3, \ldots, v_{k-1}v_k,$$

where $k \geq 2$ and where $v_1, \ldots, v_{k-1}$ are distinct vertices and $v_k = v_1$. In other words a cycle is what we obtain if we identify the endpoints of a path. For instance in Figure 3.6(i) the bold edges form a cycle of the graph. We say that a graph is *connected* if there exists a



(i) (ii) (iii)

Figure 3.6: Cycles and trees

path between every pair of distinct vertices. For instance in Figure 3.6(ii) the graph is not connected as there is no path with endpoints u and v. A graph is a *tree* if it is connected but has no cycle. In Figure 3.6, the graph in (iii) is a tree, but (i) is not a tree, as it has a cycle, and (ii) is not a tree as it is not connected.

If G is a tree then for any pair of distinct vertices u and v there exists at least one path between u and v (as G is connected). Moreover, it is easy to check that if there existed two distinct paths between u and v we would have a cycle, thus

Remark 3.16. *In a tree there exists a unique path between every pair of distinct vertices.*

Given a tree T with distinct vertices u, v, we denote by T_{uv} the unique path in Remark 3.16. For instance in Figure 3.6(iii), the path T_{uv} is denoted in bold. Given a graph G we say that T is a *tree of G* if T is a tree and T is obtained from G by removing edges and removing vertices. For instance in Figure 3.7, the graph in (ii) is a tree of (i). Given a tree $T = (V, E)$, we call

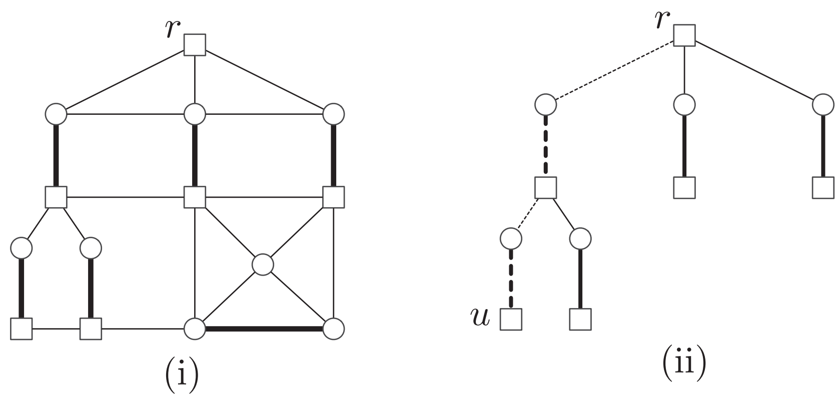


Figure 3.7: M-alternating tree (edges in M are bold).

a vertex $u \in V$ a *leaf* if it is incident to exactly one edge. For instance u is a leaf of the tree in Figure 3.7(ii). We are now ready for our key definition. Let $G = (V, E)$ be a graph with a matching M and let $r \in V$. We say that a tree T of G is an *M-alternating tree rooted at r* if,

- r is a vertex of T that is M-exposed in G,
- all vertices of T distinct from r are M-covered,
- for every vertex $u \neq r$ of T, the unique ru-path T_{ru} is M-alternating.

The graph in Figure 3.7(ii) is an M-alternating tree rooted at r. For instance for vertex u, the path T_{ru}, indicated by dashed lines, is an M-alternating path. Given an M-alternating tree T rooted at r we partition the vertices of T into sets $A(T)$ and $B(T)$ where $r \in B(T)$ and for every vertex $u \neq r$ of T, $u \in B(T)$ if and only if the path T_{ru} has an even number of edges. In Figure 3.7(ii), vertices $B(T)$ correspond to squares and vertices $A(T)$ to circles.

We can now state the algorithm that proves Theorem 3.12. We denote by $V(T)$ the set of vertices of T and by $E(T)$ the set of edges of T. Note, in this algorithm we will view paths as sets of edges. At any time during its execution, the algorithm will maintain a matching M, and an M-alternating tree T. Initially, the matching M is set to the empty set.

Algorithm 3.4 Perfect matching.

Input: Bipartite graph $H = (V,E)$ with bipartition U,W where $|U| = |W| \geq 1$.
Output: A perfect matching M, or a deficient set $B \subseteq U$.

1: $M := \emptyset$
2: $T := (\{r\},\emptyset)$ where $r \in U$ is any M-exposed vertex.
3: **loop**
4:     **if** $\exists$ edge uv where $u \in B(T)$ and $v \notin V(T)$ **then**
5:         **if** v is M-exposed **then**
6:             $P := T_{ru} \cup \{uv\}$
7:             $M := M \Delta P$
8:             **if** M is a perfect matching **then stop end if**
9:             $T := (\{r\},\emptyset)$ where $r \in U$ is any M-exposed vertex.
10:         **else**
11:             Let $w \in V$ where $vw \in M$
12:             $T := (V(T) \cup \{v,w\}, E(T) \cup \{uv,vw\})$
13:         **end if**
14:     **else**
15:         **stop** $B(T) \subseteq U$ is a deficient set.
16:     **end if**
17: **end loop**

We illustrate the algorithm in Figure 3.8. Suppose that $U = \{b,d,g\}$ and that $W = \{a,c,f\}$. Suppose that after a number of iteration we just increased the size of the matching in Step 7 to obtain the matching $M = \{ab, fd\}$ see figure (i). As M is not a perfect matching, and since $|U| = |W|$, there must exist a vertex in U that is M-exposed. In our case $g \in U$ is M-exposed and we define our tree T to consists of vertex g (with no edges), i.e. $T = (\{g\},\emptyset)$. It is trivially an M-alternating tree rooted at g. At the next iteration, we look in Step 4 for an edge incident to a vertex of $B(T) = \{g\}$ where its other endpoint is not in T. For instance gf is such an edge. We then observe that f is M-covered because of edge