Outline

In this lecture, we first consider the maximum matching problem for a general graph, that is not necessarily bipartite. We cover Edmonds' blossom algorithm which fixes the issue of the augmenting path algorithm when an odd cycle is present. The second part provides some background materials for polyhedral theory and integer programming.

1 Nonbipartite matching

In this lecture, we study the matching problem for general graphs that are not necessarily bipartite. In particular, a **nonbipartite** graph contains an odd cycle. Let us consider odd cycles. Figure 7.1

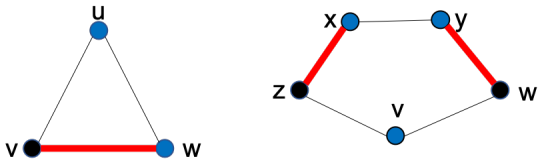

Figure 7.1: odd cycles

has a triangle and a cycle of length five. A triangle can take at most one edge for a matching while it requires at least two vertices for a vertex cover. An odd cycle of length five can take at most two edges for a matching but any vertex cover of it needs at least three vertices. In general, for an odd cycle of length $2k+1$ for $k \geq 1$, we can prove that the maximum size of a matching is k while the minimum size of a vertex cover is $k+1$. This means that König's theorem does not hold for odd cycles.

For a bipartite graph, we learned that the augmenting path algorithm with the alternating tree procedure computes a maximum matching and a minimum vertex cover. Can we just apply the augmenting path to a nonbipartite graph? In Figure 7.2, we have a bipartite graph on the left.

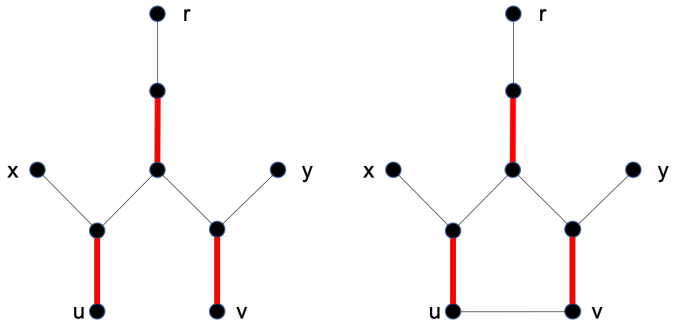

Figure 7.2: alternating trees for a bipartite graph (left) and a nonbipartite graph (right)

Adding an edge between vertices u and v, we create an odd cycle and thus a nonbipartite graph.

Nevertheless, the alternating tree procedure that starts with M-exposed vertex r would return the same alternating tree, which consists of the ru-path and the rv-path. Deleting the alternating tree, we are left with vertices x and y. Then the alternating tree procedure would conclude that there is no M-augmenting path. This is true for the bipartite graph on the left. What about the graph with the additional edge uv? In fact, uv gives rise to an M-augmenting path. Note that there is an M-augmenting path from r to x. This suggests that the alternating tree procedure is incomplete for the case of nonbipartite graphs.

What goes wrong with the alternating tree procedure when applied to a nonbipartite graph? Let us recall our analysis of the alternating tree procedure. Suppose that the alternating tree procedure returns an alternating tree from an M-exposed vertex r, without finding an M-augmenting path. As before, we collect the vertices from an even level in W and the other vertices in U. We argued

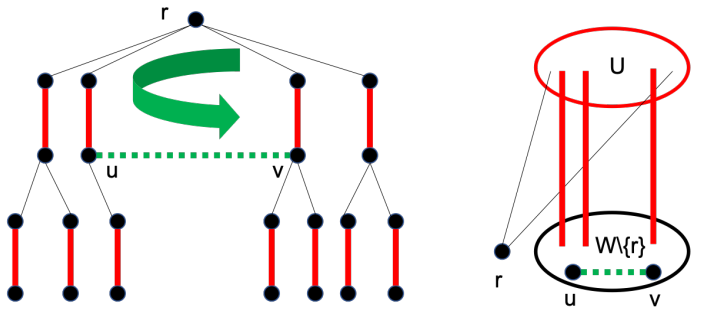

Figure 7.3: illustrating the issue with the alternating tree procedure for a nonbipartite graph

that for a bipartite graph, no two vertices in W are adjacent because an edge between two vertices in W would create an odd cycle, illustrated in Figure 7.3. For a nonbipartite graph, however, we may run into such a scenario.

2 Edmonds' blossom algorithm

Fortunately, there is a simple remedy for the alternating tree procedure due to Jack Edmonds [Edm65]. In the previous section, we saw that the analysis of the alternating tree procedure may fail when two vertices from an even level are adjacent. In such a case, as in Figure 7.4, the paths from an

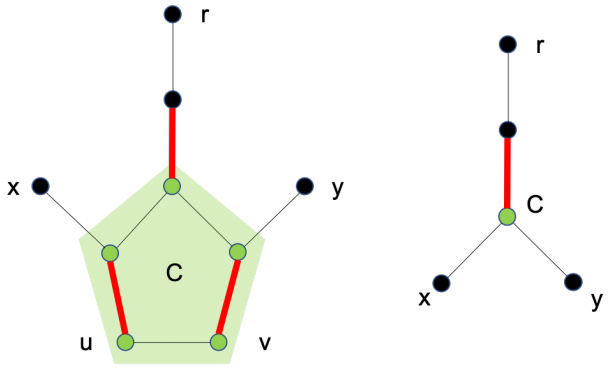

Figure 7.4: contracting a blossom

M-exposed vertex r to the two vertices and the edge between them would create an odd cycle. We

refer to the odd cycle C as a **blossom** and the path from the M-exposed vertex to the blossom as a **stem**. Just for a reference, the structure with a blossom attached to a stem is called a **flower**. we have an odd cycle C. Edmonds' idea is that every time the alternating tree procedure detects a blossom, we may just **contract** it. The operation of **contraction** is illustrated in Figure 7.4. We replace the vertices of the blossom C by a single vertex. Then connect the new vertex to the vertices that are adjacent to a vertex in the blossom. We denote by G/C the graph obtained from G after contracting the blossom C.

Lemma 7.1. *Let $G = (V, E)$ be a graph, and let M be a matching. Let C be a blossom. Then there is an M-augmenting path in G if and only if there is an $(M \setminus C)$-augmenting path in G/C.*

Proof. For the first direction, suppose that there is an $(M \setminus C)$-augmenting path in G/C. If the path does not goes through the vertex C in G/C, then the same path is an M-augmenting path in G. Thus we may assume that it passes through C in G/C. Note that the path consists of two parts, an $(M \setminus C)$-alternating path from an $(M \setminus C)$-exposed vertex to C and an $(M \setminus C)$-alternating path from C to another $(M \setminus C)$-exposed vertex that starts with a non-matching edge. Note that the first part touches the vertex u of C in G that is not covered by a matching edge of C. Let v be the vertex of C which is the starting point of the second part. By the alternating structure of the blossom C, one of the two paths from u to v on C would extend the augmenting path to an M-augmenting path in G.

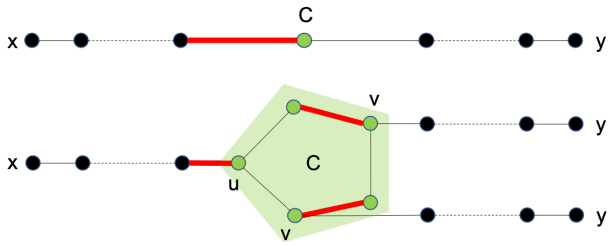

Figure 7.5: reconstructing an M-augmenting path from an $(M \setminus C)$-augmenting path

For the other direction, suppose that there is an M-augmenting path in G. This means that M is not a maximum matching. Then we take the stem of the blossom and reverse the roles of matching and non-matching on the stem. Denoting M' by the resulting matching, we know that $|M| = |M'|$ and $|M \setminus C| = |M' \setminus C|$ since the stem is an M-alternating path of an even length. This implies that M' is not a maximum matching, so there is an M'-augmenting path in G. Moreover, the vertex C in G/C becomes $(M' \setminus C)$-exposed. If the M'-augmenting path is vertex-disjoint from C in G, then the same path is an $(M' \setminus C)$-augmenting path in G/C. Thus we may assume that it touches C. Note that one end point of the M'-augmenting path must be outside C, because C has a single vertex u that is M'-exposed. Let us take its subpath from the end point outside C up to the first vertex v where it touches C. Since C is u is M'-exposed while the other vertices of C are covered by M', the subpath touches C with a non-matching edge, as in Figure 7.6. Note that the subpath can be extended from v to u, providing another M'-augmenting path. Moreover, contracting C, it gives us an $(M' \setminus C)$-augmenting path in G/C. □

Lemma Theorem 7.1 suggests the following modification of the augmenting path algorithm.

1. Given a matching M in G, take an M-exposed vertex r and start growing an M-alternating tree from r.

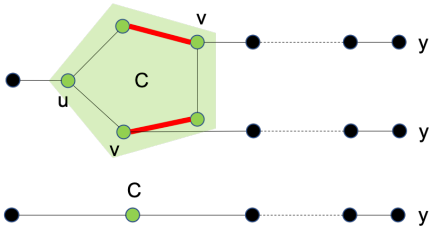

Figure 7.6: constructing an $(M' \setminus C)$-augmenting path from an M'-augmenting path

2. If the alternating tree procedure detects two adjacents vertices that have an even distance from r, we detect a blossom C.

3. Contract the blossom C and continue the algorithm with G/C.

This algorithm is referred to as the **blossom algorithm** for nonbipartite matching. Note that one application of the contraction operation reduces the number of vertices. Therefore, we may contract the graph at most $O(|V|)$ times. Recall that the alternating tree procedure takes $O(|E|)$ iterations as it is equivalent to enumerating the edges. Moreover, we know that we may augment the matching at most $O(|V|)$ times. Therefore, the time complexity of Edmonds' blossom algorithm is $O(|V|^2|E|)$.

3 Polyhedral theory basics

A set $X \subseteq \mathbb{R}^d$ is **convex** if it holds for any $u, v \in X$ and any $\lambda \in [0,1]$ that $\lambda u + (1-\lambda)v \in X$. In words, the line segment joining any two points is entirely contained the set. In Figure 7.7, we have a convex set and a non-convex set.

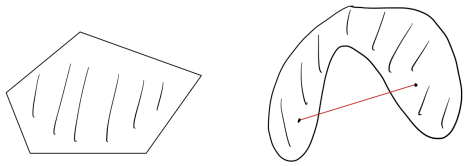

Figure 7.7: A convex set and a nonconvex set

Given $v^1, \ldots, v^k \in \mathbb{R}^d$, any linear combination $\lambda_1 v^1 + \cdots + \lambda_k v^k$ is a **convex combination** of $v^1, \ldots, v^k$ if

$$\sum_{i=1}^{k} \lambda_i = 1 \quad \text{and} \quad \lambda_i \geq 0 \quad \text{for } i = 1, \ldots, k.$$

The convex combination of two distinct points u, v is the line segment $\{\lambda u + (1-\lambda)v : \ 0 \leq \lambda \leq 1\}$ connecting them. The **convex hull** of a set $S \subseteq \mathbb{R}^d$, denoted $\text{conv}(S)$, is the set of all convex combinations of points in S. By definition,

$$\text{conv}(S) = \left\{ \sum_{i=1}^{n} \lambda_i v^i : \begin{array}{c} n \in \mathbb{N},\ v^1, \ldots, v^n \in S, \\ \sum_{i=1}^{n} \lambda_i = 1,\ \lambda_1, \ldots, \lambda_n \geq 0 \end{array} \right\}.$$

conv(S) is always convex regardless of S. Figure 7.8 shows some examples of taking the convex hull of a (nonconvex) set.

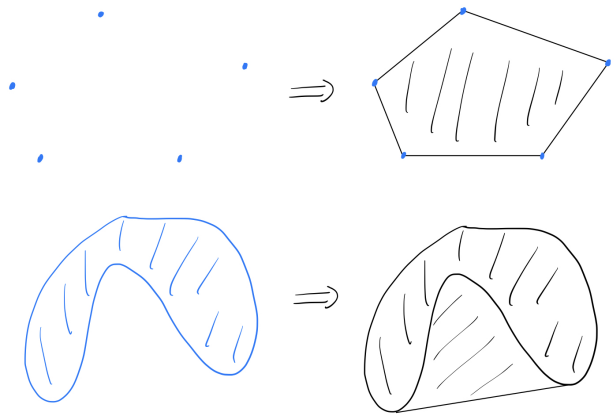

Figure 7.8: A convex set and a nonconvex set

A set $P \subseteq \mathbb{R}^d$ is a **polyhedron** if it is defined by a **finite** number of linear inequalities, i.e.

$$P = \{x \in \mathbb{R}^d : \ Ax \leq b\}.$$

Hence, a polyhedron is a finite intersection of half-spaces. A polyhedron is **rational** if it is defined by

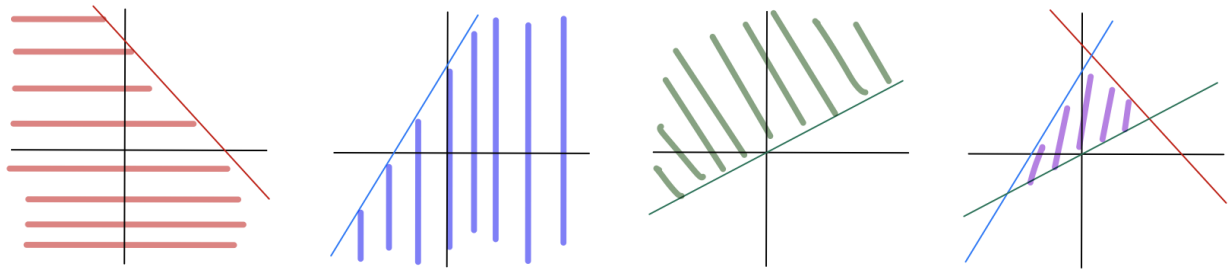

Figure 7.9: Polyhedron defined by three inequalities

a system of linear inequalities where all coefficients and right-hand sides are rational. A polyhedral cone is by definition a polyhedron.

A set $P \subseteq \mathbb{R}^d$ is a **polytope** if it is a polyhedron and bounded, i.e., $P \subseteq [-M, M]^d$ for some sufficiently large $M > 0$.

Theorem 7.2 (Minkowski-Weyl theorem for polytopes)**.** *A set $P \subseteq \mathbb{R}^d$ is a polytope if and only if*

$$P = \text{conv}(v^1, \ldots, v^p)$$

for some vectors $v^1, \ldots, v^p$.

4 Reduction to linear programming

Consider an integer linear program given by

$$\begin{aligned} \max \quad & c^\top x + h^\top y \\ \text{s.t.} \quad & Ax + Gy \le b, \\ & x \in \mathbb{Z}^d,\ y \in \mathbb{R}^p. \end{aligned} \tag{7.1}$$

Let S be the set of solutions satisfying the constraints of (7.1), whose convex hull is given by

$$\operatorname{conv}(S) = \operatorname{conv}\left(\left\{(x,y) \in \mathbb{Z}^d \times \mathbb{R}^p :\ Ax + Gy \le b\right\}\right).$$

Lemma 7.3. *The integer program (7.1) whose feasible region is given by $S \subseteq \mathbb{Z}^d \times \mathbb{R}^p$ satisfies*

$$\max\left\{c^\top x + h^\top y :\ (x,y) \in S\right\} = \max\left\{c^\top x + h^\top y :\ (x,y) \in \operatorname{conv}(S)\right\}.$$

Moreover, the supremum of $c^\top x + h^\top y$ is attained over S if and only if it is attained over $\operatorname{conv}(S)$.

Proof. Since $S \subseteq \operatorname{conv}(S)$, it is straightforward that

$$\max\left\{c^\top x + h^\top y :\ (x,y) \in S\right\} \le \max\left\{c^\top x + h^\top y :\ (x,y) \in \operatorname{conv}(S)\right\}.$$

Next we show that

$$\max\left\{c^\top x + h^\top y :\ (x,y) \in S\right\} \ge \max\left\{c^\top x + h^\top y :\ (x,y) \in \operatorname{conv}(S)\right\}$$

holds. Let $z^* = \max\left\{c^\top x + h^\top y :\ (x,y) \in S\right\}$. Then we may assume that z^* is finite. Let us consider

$$H = \left\{(x,y) \in \mathbb{R}^d \times \mathbb{R}^p :\ c^\top x + h^\top y \le z^*\right\}.$$

By definition, we have $S \subseteq H$. Moreover, as H is convex, it follows that $\operatorname{conv}(S) \subseteq H$. This implies that

$$\max\left\{c^\top x + h^\top y :\ (x,y) \in \operatorname{conv}(S)\right\} \le z^* = \max\left\{c^\top x + h^\top y :\ (x,y) \in S\right\},$$

which proves the desired inequality.

Assume that the supremum of $c^\top x + h^\top y$ is attaned at $(\bar{x}, \bar{y}) \in S$. Then

$$\max\left\{c^\top x + h^\top y :\ (x,y) \in S\right\} = c^\top \bar{x} + h^\top \bar{y}.$$

Note that $(\bar{x}, \bar{y}) \in \operatorname{conv}(S)$, and the first part implies that

$$\max\left\{c^\top x + h^\top y :\ (x,y) \in \operatorname{conv}(S)\right\} = c^\top \bar{x} + h^\top \bar{y}.$$

Now assume that the supremum of $c^\top x + h^\top y$ is attained at a point $(\bar{x}, \bar{y}) \in \operatorname{conv}(S)$. By the definition of $\operatorname{conv}(S)$, the point can be written as a convex combination of n points in S, given by $(x^1, y^1), \ldots, (x^n, y^n)$. As these n points also belong to $\operatorname{conv}(S)$, it follows that $c^\top x^i + h^\top y^i \le c^\top \bar{x} + h^\top \bar{y}$ for all i. Moreover,

$$c^\top \bar{x} + h^\top \bar{y} = \sum_{i=1}^{n} \lambda_i (c^\top x^i + h^\top y^i)$$

for some $\lambda_1, \ldots, \lambda_n \geq 0$ such that $\sum_{i \in [d]} \lambda_i = 1$. Then

$$c^\top \bar{x} + h^\top \bar{y} = \sum_{i=1}^{n} \lambda_i (c^\top x^i + h^\top y^i) \leq \sum_{i=1}^{n} \lambda_i (c^\top \bar{x} + h^\top \bar{y}) = c^\top \bar{x} + h^\top \bar{y},$$

so the equalities hold throughout. Therefore, $c^\top x^i + h^\top y^i = c^\top \bar{x} + h^\top \bar{y}$ for all $i \in [n]$. □

By Lemma 7.3, solving the integer program (7.1) is equivalent to optimizing over the convex hull $\text{conv}(S)$. By Meyer's theorem [Mey74], we know that there exists a system of rational linear inequalities $A'x + G'y \leq b'$ such that

$$\text{conv}(S) = \left\{ (x, y) \in \mathbb{R}^d \times \mathbb{R}^p : \ A'x + G'y \leq b' \right\}.$$

Consequently, the integer program $\max \left\{ c^\top x + h^\top y : \ (x, y) \in S \right\}$ is equivalent to the linear program

$$\max \left\{ c^\top x + h^\top y : \ A'x + G'y \leq b' \right\}$$

for some rational matrices A', G', b'. Therefore, we may say that integer programming reduces to linear programming. Wait, does this contradict our earlier discusstion that integer programming is NP-hard while linear programming is in class P? The answer is NO. The reason is that Meyer's theorem shows the *existence* of such a linear system, and in fact, computing the linear system that gives us the convex hull of S is in general hard.

References

[Edm65] J. Edmonds. Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965. 2

[Mey74] R. R. Meyer. On the existence of optimal solutions to integer and mixed integer programming problems, 1974. 4