Contents

1 Introduction

Linear Programming deals with the optimization of linear functions subject to linear constraints. For example,

$$\begin{array}{lrcl} \text{minimize} & 3x + 4y & & \\ \text{subject to} & 2x + 3y & \leq & 7 \\ & 3x - 4y & \leq & 4 \end{array}$$

Many problems can be formulated naturally as linear programs. We will see some examples below. Linear programming gives a general method for solving problems in polynomial time.

Theorem 1 *Linear programs can be solved in time polynomial in the number m of constraints, the number n of variables, and the total size L of the coefficients. The size of a coefficient is the logarithm of* 1 *plus its absolute value.*

In Integer Linear Programming the variables are restricted to take integral values. For example,

$$\begin{array}{lrcl} \text{minimize} & 3x + 4y & & \\ \text{subject to} & 2x + 3y & \leq & 7 \\ & 3x - 4y & \leq & 4 \\ & x, y & \in & \mathbb{Z} \end{array}$$

Theorem 2 *The decision problem for integer linear programming is NP-complete. The input for the decision problem is an integer linear program and an integer k and the question is whether the program has a feasible solution with objective value at most k.*

The NP-completeness of the decision problem for integer linear programming makes it unlikely that there is a polynomial time algorithm for integer linear programming. Nevertheless, there are general methods for solving ILP exactly and approximately and these methods are among the most powerful methods for attacking NP-complete problems. The NP-completeness of ILP implies that every problem in NP can be reduced to an ILP. More important is the fact that many difficult optimization problems are formulated very naturally as ILPs.

We summarize: Linear Programming and Integer Linear Programming are extremely powerful algorithmic paradigms: many problems are easily formulated as LPs or ILPs and effective solution algorithms are available.

We next give some examples of linear and integer linear programs.

Maximum Network Flow: We have a directed graph $G = (V, E)$, a source node $s \in V$, a sink node $t \in V$, and a non-negative capacity cap_e for every edge of e. The goal is to find a maximum flow from s to t. For a node v, we use $In(v)$ and $Out(v)$ to denote the set of edges ending in and emanating from v, respectively, and for an edge e we use f_e to denote the flow along the edge.

$$\begin{array}{llcll}
\text{maximize} & \sum_{e \in Out(s)} f_e & & & \\
\text{subject to} & \sum_{e \in Out(v)} f_e & = & \sum_{e \in In(v)} f_e & \text{for every node } v \in V \setminus \{s, t\} \\
& f_e & \leq & cap_e & \text{for every edge } e \in E \\
& f_e & \geq & 0 &
\end{array}$$

The first family of constraints states that we have flow conservation at all nodes distinct from s and t and the other constraints state that flows are non-negative and bounded by the edge capacities. Subject to these constraints we are maximizing the flow out of s.

Minimum s-t-Cut: In the setting of the maximum flow problem we may also be interested in a minimum capacity cut C separating s and t. A set C of edges is called an s-t-cut if every path from s to t contains at least one edge in C. The capacity of C is the sum of the capacities of the edges in C. The minimum s-t-cut problem is easily formulated as a ILP. We have a zero-one variable for every edge e with the intended semantics that $e \in C$ iff $y_e = 1$.

$$\begin{array}{llcll}
\text{minimize} & \sum_{e \in E} cap_e y_e & & & \\
\text{subject to} & \sum_{e \in p} y_e & \geq & 1 & \text{for every } s\text{-}t\text{-path } p \\
& y_e & \in & \{0, 1\} & \text{for every edge } e \in E
\end{array}$$

The number of constraints is the ILP above is equal to the number of s-t-paths. In a graph with cycles, this number is infinite. We may restrict attention to simple s-t-paths. The number of simple paths is finite, but is in general exponential in the size of the graph.

Miracle: One can solve LPs and ILPs with an exponential or even an infinite number of constraints.

There is an alternative formulation of the minimum cut problem which uses only a linear number of constraints. We now have a zero-one variable for each node and each edge; say y_v is the variable for node v and y_e is the variable for edge e. The intended semantics is that $C = \{e \,;\, y_e = 1\}$ is an s-t-cut and that $y_v = 1$ iff v can be reached from s by a path not using an edge in C.

$$\begin{array}{llcll}
\text{minimize} & \sum_{e \in E} cap_e y_e & & & \\
\text{subject to} & y_u + y_e - y_v & \geq & 0 & \text{for every edge } e = (u, v) \in E \\
& y_e & \in & \{0, 1\} & \text{for every edge } e \in E \\
& y_v & \in & \{0, 1\} & \text{for every node } v \in V \\
& y_s & = & 0 & \\
& y_t & = & 1 &
\end{array}$$

We need to argue that the ILP captures the s-t-cut problem.

Wir first show that any feasible solution defines a cut. Consider any s-t-path p. Summing the first constraint for all edges $e \in P$ gives $y_s + \sum_{e \in p} y_e - y_t \geq 0$ and hence with the use of $y_s = 0$ and $y_t = 1$, $\sum_{e \in p} y_e \geq 1$. The set of edges with $y_e = 1$ is thus guaranteed to form an s-t-cut.

Conversely, consider any s-t-cut C. Setting $y_e = 1$ iff $e \in C$, $y_v = 0$ for all nodes reachable from s by a path not using an edge of C, and $y_v = 1$ for the remaining nodes, we obtain a feasible solution of the ILP. Observe that $y_s = 0$ and $y_t = 1$ since every path from s to t contains at least one edge in C. Consider finally an inequality $y_u + y_e \geq y_v$ where $e = (u, v)$ is an edge of G. If either $y_u = 1$ or $y_v = 0$, the inequality is certainly satisfied. So assume $y_u = 0$ and $y_v = 1$. Then there is a path from s to u using no edge in C, say p, and there is no such path from s to v. Consider the path $p \circ e$ from s to v. It must contain an edge in C and hence $e \in C$. Thus $y_e = 1$ and the inequality holds.

We conclude that an optimal solution of the ILP is a minimum capacity s-t-cut.

Miracle: We may replace the integrality constraints $y_v \in \{0,1\}$ for all $v \in V$ and $y_e \in \{0,1\}$ for all $e \in E$ by the weaker constraints $0 \leq y_v \leq 1$ for all $v \in V$ and $0 \leq y_e \leq 1$ for all $e \in E$ without changing the optimal objective value. Even more is true, the LP

$$
\begin{array}{llcll}
\text{minimize} & \sum_{e \in E} cap_e y_e & & & \\
\text{subject to} & y_u + y_e - y_v & \geq & 0 & \text{for every edge } e = (u,v) \in E \\
 & y_e & \geq & 0 & \text{for every edge } e \in E \\
 & y_e & \leq & 1 & \text{for every edge } e \in E \\
 & y_v & \geq & 0 & \text{for every node } v \in V \\
 & y_v & \leq & 1 & \text{for every node } v \in V \\
 & y_s & = & 0 & \\
 & y_t & = & 1 &
\end{array}
$$

is guaranteed to have an integral optimal solution (more generally, LPs with totally unimodular constraint matrices are guaranteed to have integral solutions); there may also be non-integral optimal solution.

Miracle: The value of the maximum flow is equal to the minimum capacity of an s-t-cut, the so-called Max-Flow-Min-Cut theorem. This is called a duality theorem. It is a consequence of the strong duality theorem of linear programming.

Maximum Weight Bipartite Matching: We have a bipartite graph $G = (A \dot{\cup} B, E)$ in which each edge e carries a real weight w_e. A matching M is a set of edges no two of which share an edge point. The goal is to find a matching of maximal weight. The problem is easily formulated as an ILP. We use a zero-one variable x_e for each edge with the intended semantics that the set of edges with $x_e = 1$ form a matching. For a node v we use $\delta(v)$ to denote the set of edges incident to v.

$$
\begin{array}{llcll}
\text{maximize} & \sum_e w_e x_e & & & \\
\text{subject to} & \sum_{e \in \delta(v)} x_e & \leq & 1 & \text{for all } v \in V \\
 & x_e & \in & \{0,1\} & \text{for all } e \in E
\end{array}
$$

Let M be the set of edges e with $x_e = 1$. The constraint $\sum_{e \in \delta(v)} x_e \leq 1$ ensures $|M \cap \delta(v)| \leq 1$ for every node v, i.e., M is a matching. We are maximizing the weight of M.

Miracle: We may replace the integrality constraints $x_e \in \{0,1\}$ for all $e \in E$ by the weaker constraints $0 \leq x_e \leq 1$ for all $e \in E$ without changing the optimal objective value. Even more is true, the LP

$$
\begin{array}{llcll}
\text{maximize} & \sum_e w_e x_e & & & \\
\text{subject to} & \sum_{e \in \delta(v)} x_e & \leq & 1 & \text{for all } v \in V \\
 & x_e & \leq & 1 & \text{for all } e \in E \\
 & x_e & \geq & 0 & \text{for all } e \in E
\end{array}
$$

is guaranteed to have an integral optimal solution; there may also be non-integral optimal solution. Again the constraint matrix is totally unimodular.

Minimum Node Covers in Bipartite Graphs: We are in the same situation as in the previous paragraph. The goal is to assign a non-negative number y_v to every node of G such that $w_e \le y_a + y_b$ for every edge $e = ab \in G$ and such that $\sum_v y_v$ is minimal.

$$\begin{array}{llcll} \text{minimize} & \sum_v y_v & & & \\ \text{subject to} & y_a + y_b & \ge & w_e & \text{for all } e = ab \in E \\ & y_v & \le & 0 & \text{for all } v \in V \end{array}$$

Miracle: The weight of a maximum weight matching in a bipartite graph is equal to the minimal weight of a node cover. This is again a duality result.

The Traveling Salesman Problem: We are given an undirected graph $G = (V, E)$ in which every edge e is assigned a cost c_e. The goal is to find a minimum cost tour passing through all nodes of G. We formulate the problem as a ILP. We have a zero-one variable x_e for every edge e. The intended semantics is that the set of edges with $x_e = 1$ forms an optimal Traveling Salesman Tour. For a set S of nodes we use $\delta(S)$ to denote the set of edges having exactly one endpoint in S.

$$\begin{array}{llcll} \text{minimize} & \sum_e c_e x_e & & & \\ \text{subject to} & \sum_{e \in \delta(v)} x_e & = & 2 & \text{for every } v \in V \\ & \sum_{e \in \delta(S)} x_e & \ge & 2 & \text{for every } S \subset V \text{ with } S \ne \emptyset \text{ and } S \ne V \\ & x_e & \in & \{0,1\} & \text{for every } e \in E \end{array}$$

We call the first set of constraints the degree constraints and the second set the cut constraints. We need to argue that the ILP above captures the Traveling Salesman Problem.

Consider any tour T and put $x_e = 1$ iff $e \in T$. A tour contains exactly two edges incident to every node and hence the degree constraints are satisfied. Also, a tour contains at least two edges in every cut and hence the cut constraints are satisfied.

Conversely, consider any solution to the ILP above and let $T = \{e\ ;\ y_e = 1\}$. The degree constraints ensure that T contains two edges incident to every vertex and hence T is a set of cycles in G. We need to show that the edges in T form a single cycle. Assume otherwise and let S be the set of nodes in one of the cycles. Then $\emptyset \ne S \ne V$ and $\sum_{e \in \delta(S)} x_e = 0$, a contradiction to the fact that the cut constraint for S is satisfied.

The number of cut constraints is $(2^n - 2)/2$; observe that the constraints for S and $V \setminus S$ are identical.

Miracle: The Traveling Salesman ILP can be solved exactly for graphs with up to thousand nodes. It can be solved within a factor of two of optimal for very large n.

The linear programming relaxation of an integer linear program is obtained by dropping the integrality constraints. For the TSP, we obtain:

$$\begin{array}{llcll} \text{minimize} & \sum_e c_e x_e & & & \\ \text{subject to} & \sum_{e \in \delta(v)} x_e & = & 2 & \text{for every } v \in V \\ & \sum_{e \in \delta(S)} x_e & \ge & 2 & \text{for every } S \subset V \text{ with } S \ne \emptyset \text{ and } S \ne V \\ & x_e & \le & 1 & \text{for every } e \in E \\ & x_e & \ge & 0 & \text{for every } e \in E \end{array}$$

Miracle: The linear program above can be solved in polynomial time in n (= the number of nodes of G) and $L = \sum_e \log(1 + |(|c_e))$. This assumes that the edge costs are integral. Observe that the LP has an exponential number of constraints. The miracle is a consequence of the Ellipsoid method for linear programming. The ellipsoid method does not need an explicit description of a linear program. The method works in rounds. In each round it produces a candidate solution x^* and asks the question whether x^* is feasible, i.e., satisfies all constraints. If x^* is not feasible, the environment must generate a constraint violated by x^*. The miracle is that the ellipsoid method works in a polynomial number of rounds.

The cut constraints can also be formulated in so-called inner form. Use $\gamma(S)$ to denote the set of edges having both endpoints in S. In the context of the degree constraints, we may also use

$$\sum_{e\in\gamma(S)} x_e \le |S|-1 \quad \text{for every } S \subset V \text{ with } S \neq \emptyset \text{ and } S \neq V$$

This leads to the following formulation of the Traveling Salesman Problem where we also add the redundant constraint that a tour contains exactly n edges.

$$\begin{array}{lrcll} \text{minimize} & \sum_e c_e x_e & & & \\ \text{subject to} & \sum_{e\in\delta(v)} x_e & = & 2 & \text{for every } v \in V \\ & \sum_{e\in\gamma(S)} x_e & \le & |S|-1 & \text{for every } S \subset V \text{ with } S \neq \emptyset \text{ and } S \neq V \\ & \sum_{e\in E} x_e & = & n & \\ & x_e & \in & \{0,1\} & \text{for every } e \in E \end{array}$$

TODO: rewrite the paragraph and use only the inner form of the cut constraint.

The Held-Karp Lower Bound for the Traveling Salesman Problem: We do something crazy. For every node v we choose an arbitrary real value $p_v \in I\!R$ and define a modified cost function c_p as follows. For an edge $e = uv$ let $c_p(e) = c(e) + p(v) + p(u)$. The values p_v are usually called node potentials. Clearly, for any tour T

$$\sum_{e\in T} c_p(e) - 2\cdot\sum_{v\in V} p(v) = \sum_{e\in T} c(e) \ .$$

and hence (v_0 is a fixed but arbitrary vertex of G) the following linear program

$$\begin{array}{lrcll} \text{minimize} & \sum_e c_p(e) x_e & & & \\ \text{subject to} & \sum_{e\in\delta(v)} x_e & = & 2 & \text{for } v = v_0 \\ & \sum_{e\in\gamma(S)} x_e & \le & |S|-1 & \text{for every } S \subset V \text{ with } S \neq \emptyset \text{ and } S \neq V \\ & \sum_{e\in E} x_e & = & n & \\ & x_e & \le & 1 & \text{for every } e \in E \\ & x_e & \ge & 0 & \text{for every } e \in E \end{array}$$

provides us with a lower bound LB_p for the cost of an optimal Traveling Salesman tour. The lower bound depends on the potential function p.

Miracle: For every fixed p, the value LB_p can be computed in time $O(m \log m)$ by (essentially) a minimum spanning tree computation; m is the number of edges of G.

LB_p is a lower bound on the cost of the optimal tour for every potential function and hence we may consider

$$LB = \max_p LB_p \ .$$

Of course, LB is a lower bound on the cost of an optimal tour.

Miracle: LB is equal to the cost of an optimal tour. This is an example of the so-called Lagrange duality.

Bibliography