بسم اللّه الرّحمن الرّحیم

دانشگاه صنعتی اصفهان ـ دانشکدهٔ مهندسی برق و کامپیوتر
(نیم‌سال تحصیلی ۴۰۲۲)

طراحی الگوریتم‌ها

حسین فلسفین

در اسلاید پیش عنوان کردیم که:

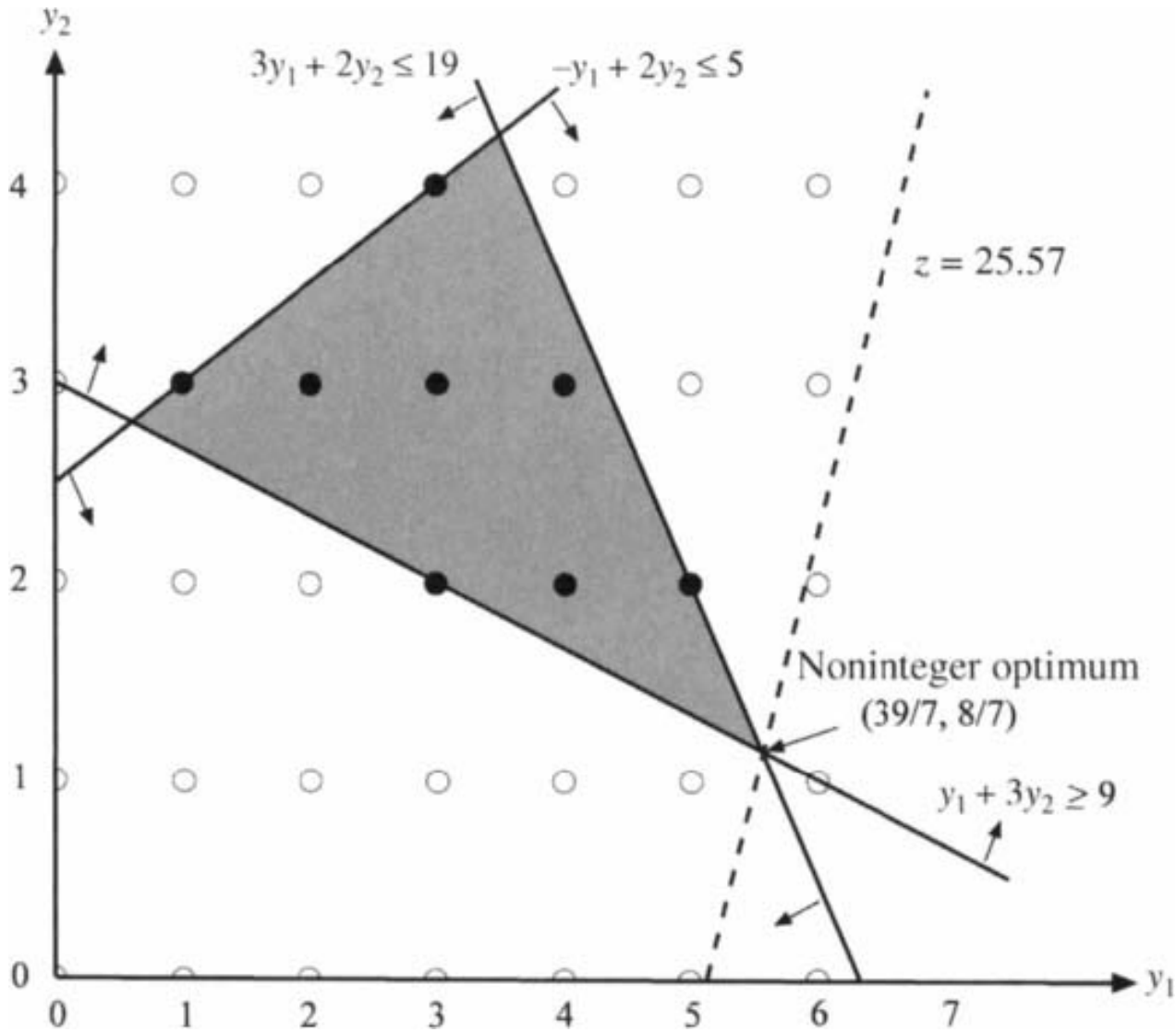
There are three cases indicating that a node can be pruned: (1) the subproblem has no feasible LP solution, (2) the sub-problem has an integer optimum solution, and (3) the bound of the subproblem is not better than the best-so-far.

These three cases are, respectively, referred to as pruned by infeasibility, pruned by optimality, and pruned by bound.

If a node is pruned by optimality, its optimum solution can be used to update the best-so-far. Whenever an integer solution to a subproblem is obtained, it is a candidate optimum to the original ILP problem. In the solution process of B&B, the best integer solution found so far is continuously updated. Such a solution is called an incumbent solution.

$$\begin{aligned}
\text{Maximize} \quad & z = 5y_1 - 2y_2 \\
\text{subject to} \quad & -y_1 + 2y_2 \leq 5 \\
& 3y_1 + 2y_2 \leq 19 \\
& y_1 + 3y_2 \geq 9 \\
& y_1, y_2 \geq 0 \text{ and integer}
\end{aligned}$$

LP and IP feasible regions

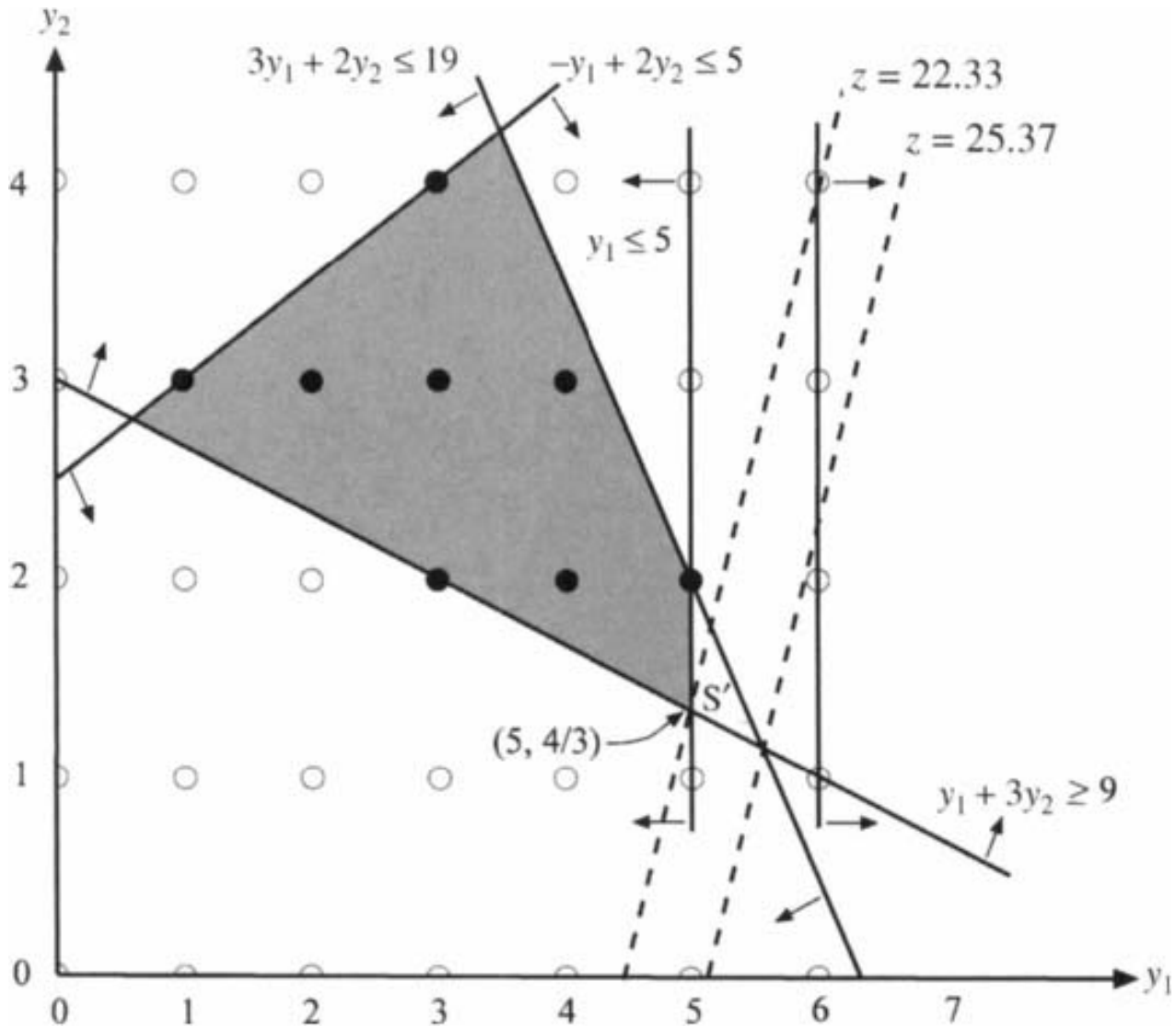


We obtain the noninteger optimum $y_1 = 39/7$, $y_2 = 8/7$, *and* $z = 25.57$. *Then the objective value* 25.57 *becomes an upper bound to the ILP problem. We set the incumbent to* $-\infty$. *Since both variables are fractional, we need to branch on them in an attempt to obtain an integer optimum.*

We arbitrarily select y_1 *as the variable to be branched. Two subproblems are generated by adding the constraint of* $y_1 \geq 6$ *and* $y_1 \leq 5$, *respectively, to the LP relaxation.*

Clearly, the branch with the added constraint $y_1 \geq 6$ *is infeasible, so it is pruned by infeasibility. The other branch with the added constraint* $y_1 \leq 5$ *is optimized at* $(y_1, y_2) = (5, 4/3)$, *with objective value* 22.33. *So the new upper bound is updated to* 22.33.

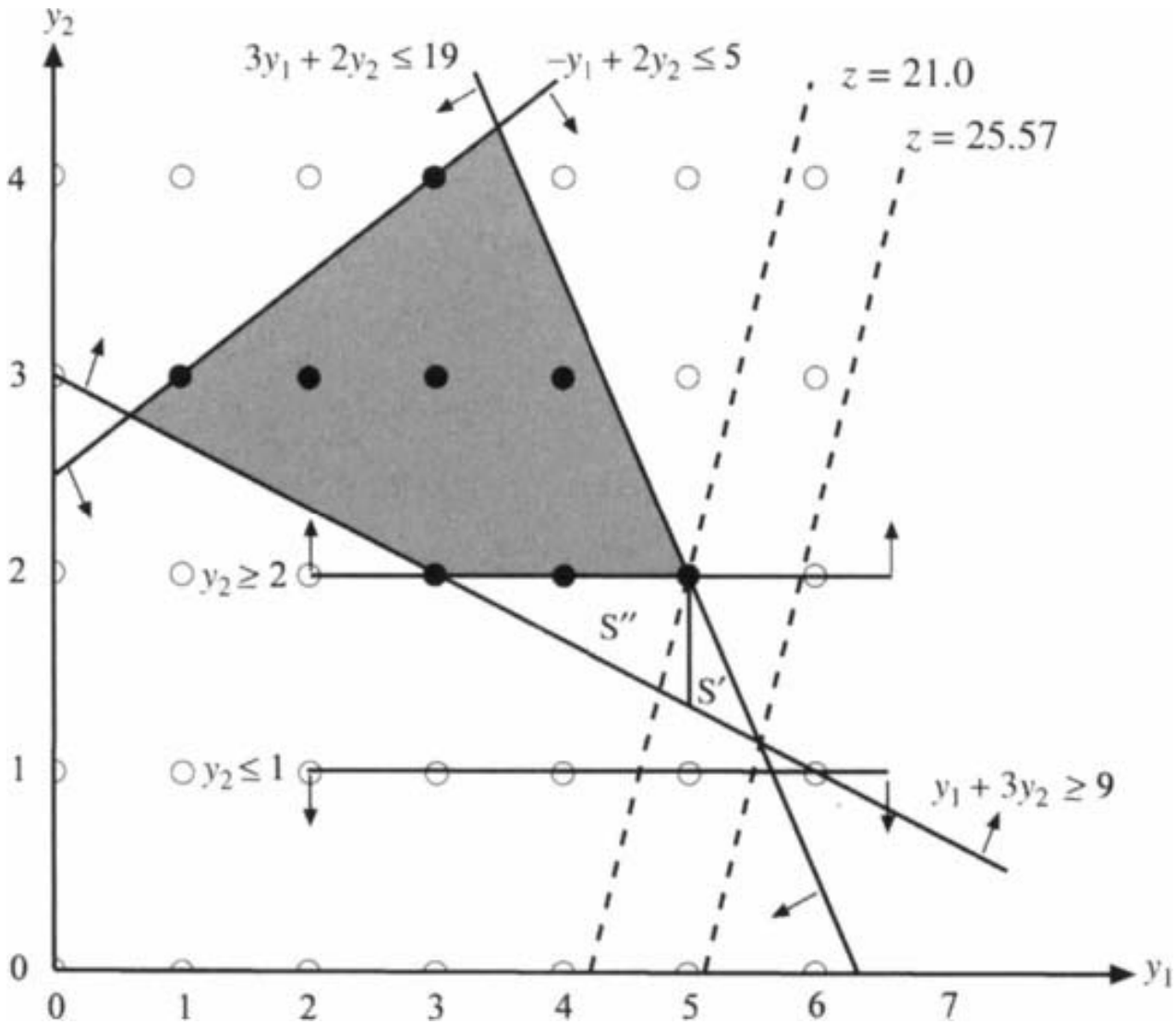
LP and IP feasible regions after the first branching



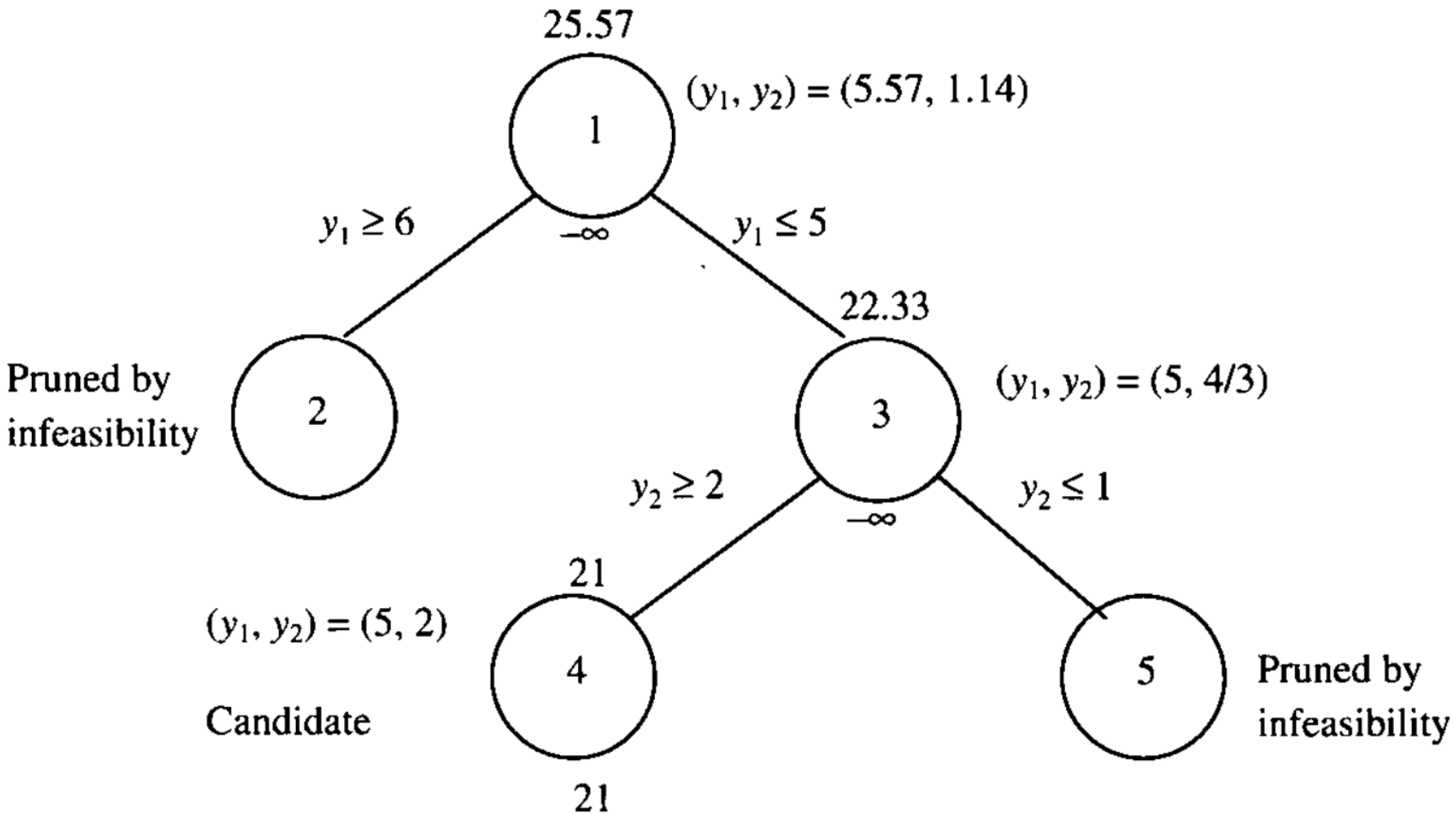
Again, the variable y_2 is fractional, so this time we branch on y_2. The two constraints $y_2 \geq 2$ and $y_2 \leq 1$ are then added.

The branch with $y_2 \leq 1$ is infeasible, and hence is pruned by infeasibility. The branch with $y_2 \geq 2$ is optimized at $(y_1, y_2) = (5, 2)$, with objective value 21. Since this is a feasible solution to the ILP problem, the value 21 becomes our new best-so-far, replacing the initial best-so-far $-\infty$, and $(5, 2)$ is a candidate solution. Checking the tree, all branches are evaluated, so $(y_1, y_2) = (5, 2)$ is the optimal solution to the ILP problem, and the optimal objective value is 21.

LP and IP solution regions after the second branching



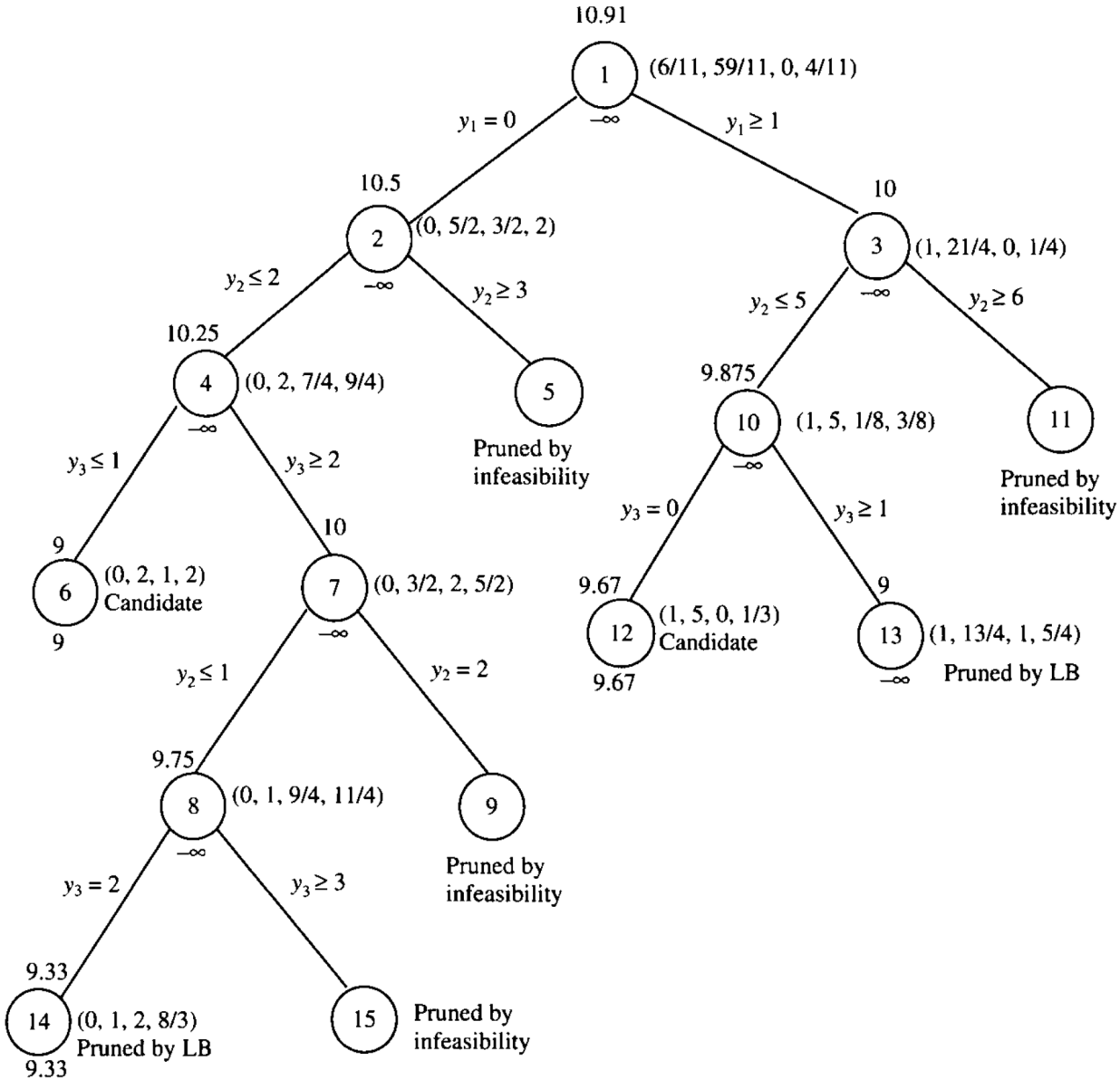
Branch-and-bound tree for our example



Another example:

$$\begin{aligned}
\text{Maximize} \quad & z = -y_1 + 2y_2 + y_3 + 2x_1 \\
\text{subject to} \quad & y_1 + y_2 - y_3 + 3x_1 \leq 7 \\
& y_2 + 3y_3 - x_1 \leq 5 \\
& 3y_1 + x_1 \geq 2 \\
& y_1, y_2, y_3 \geq 0 \text{ and integer} \\
& x_1 \geq 0
\end{aligned}$$

Branch-and-bound tree for the example using best-bound-first



جلسهٔ پیش چند مسئلهٔ بهینه‌سازی معروف را در قالب نمونه‌های ***LP/ILP*** مدلسازی کردیم

Many combinatorial problems can be expressed in the language of Integer Linear Programming (ILP). In an Integer Linear Programming instance, we are given a set of integer-valued variables, a set of linear inequalities (called constraints) and a linear cost function. The goal is to find an (integer) evaluation of the variables that satisfies all constraints, and minimizes or maximizes the value of the cost function.

In fact, it is relatively easy to express many NP-hard problems in the language of Integer Linear Programming.

مثالی دیگر از مدلسازی به ILP:
Vertex Cover Problem

Let us give an example on how to encode a Vertex Cover instance as an Integer Linear Programming instance.

- *We introduce $n = |V(G)|$ variables, one variable x_v for each vertex $v \in V(G)$. Setting variable x_v to 1 means that v is in the vertex cover, while setting x_v to 0 means that v is not in the vertex cover.*
- *To ensure that every edge is covered, we can introduce constraints $x_u + x_v \geq 1$ for every edge $uv \in E(G)$.*
- *The size of the vertex cover is given by $\sum_{v \in V(G)} x_v$.*

In the end, we obtain the following ILP formulation:

$$
\begin{array}{lll}
\text{minimize} & \sum_{v \in V(G)} x_v & \\
\text{subject to} & x_u + x_v \geq 1 & \text{for every } uv \in E(G), \\
& 0 \leq x_v \leq 1 & \text{for every } v \in V(G), \\
& x_v \in \mathbb{Z} & \text{for every } v \in V(G).
\end{array}
$$

Another Example: Maximum Matching Problem

Let $G = (V, E)$ be an unweighted undirected graph. A matching of G is any subset M of E such that if $\{x, y\}$ and $\{u, v\}$ are two different elements of M, then $\{x, y\} \cap \{u, v\} = \varnothing$.

In other words:

A matching in G is a set of edges $M \subseteq E$ with the property that, for all edges $\{u, v\}, \{x, y\} \in M$, $\{u, v\} \neq \{x, y\}$ implies $|\{u, v, x, y\}| = 4$ (i.e., no two edges of a matching share a common vertex).

The maximum matching problem is to find a matching of maximal cardinality in a given graph $G = (V, E)$.

To express the instances of this problem as instances of $0/1$*-ILP, we consider binary variables* x_e *for every* $e \in E$*, where* $x_e = 1$ *iff* $e \in M$*. Let, for every* $v \in V$*,* $E(v) = \{\{v, u\} \in E \mid u \in V\}$ *be the set of all edges incident to* v*. Now, the task is to maximize*

$$\sum_{e \in E} x_e$$

under the $|V|$ *constraints*

$$\sum_{e \in E(v)} x_e \leq 1 \text{ for every } v \in V,$$

and the following E *constraints*

$$x_e \in \{0, 1\} \text{ for every } e \in E$$

تذکر: مسئلهٔ یافتن تطابق بیشینه در یک گراف، یک مسئلهٔ ***NP-hard*** نیست.

Edmonds' Blossom algorithm is a polynomial time algorithm for finding a maximum matching in a graph. The algorithm was developed by Jack Edmonds in 1961, and published in 1965.