- 作业讲解
  - TC第34.1节练习2、3、5
  - TC第34.2节练习3、4、6、11
  - TC第34.3节练习2
  - TC第34.4节练习3、5、7
  - TC第34.5节练习6

# TC第34.1节练习3

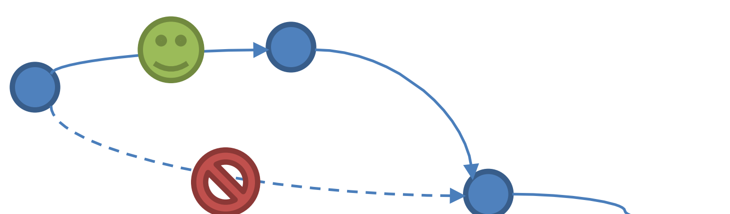
- binary string
  - 0->00
  - 1->01
  - #->11

# TC第34.1节练习5

- Also show that a polynomial number of calls to polynomial-time subroutines may result in an exponential-time algorithm.
  - subroutine: 数组自拷贝并拼接（长度翻倍）
  - *n* calls: $O(2^n)$

# TC第34.2节练习3

- 任取一点作为第一个点
- 逐个删除其邻边，并判断剩余图是否还有HAM-CYCLE
- 一旦没有，即选作第一条边，将其另一个端点作为第二个点，重复上述过程



# TC第34.2节练习11

Since any connected graph has a spanning tree, it is sufficient to consider only trees; any hamiltonian cycle on the cube of the spanning tree is also a hamiltonian cycle on the cube of the original graph. We now prove the following theorem, which obviously implies that any cube of a tree is hamiltionian.

**Theorem.** *Let $T = (V, E)$ be a tree. For any edge $e \in E$, there is a hamiltonian cycle on $T^3$ that contains edge $e$.*

*Proof.* We prove the claim by induction on $n = |T|$. The cases $n = 3$ and $n = 4$ are trivial, as the cube of the tree is always a clique.

Now assume that the claim holds for all graphs with at most $n \geq 4$ vertices and let $T = (V, E)$ be a graph with $n + 1$ vertices. Let $e = \{u, v\} \in E$ be an arbitrary edge. We now construct an hamiltonian cycle in $T^3$ that uses edge $e$. Since $T$ is a tree, removing edge $e$ will break $T$ into two connected components, both of which are tree. Denote by $T_u$ the component that contains $u$ and by $T_v$ the component that contains $v$. Without a loss of generality, we may assume that $T_u$ has at least 3 vertices. Let $u'$ be a neighbour of $u$ in $T_u$. By induction assumption, there is a hamiltonian cycle $H$ in $T_u^3$ that uses edge $\{u, u'\}$.

There are now three cases to consider.

- $T_v$ contains only one vertex. We get the desired hamiltonian cycle by taking $H$ and adding $v$ between $u$ and $u'$; this can be done since $d_T(u', v) = 1 + d_T(u, v) = 2$.
- $T_v$ contains two vertices. One of these is $v$, and we denote the other by $v'$. Obviously $v'$ is adjacent to $v$. We change the cycle $H = (\ldots, u', u, \ldots)$ to $(\ldots, u', v', v, u, \ldots)$, which yields the desired result. This modification can be done, as we have that $d_T(u', v') = 3$.
- $T_v$ contains at least three vertices. Let $v'$ be a neighbour of $v$ in $T_v$. By induction assumption, there is a hamiltonian cycle $H'$ in $T_v$ that contains edge $\{v, v'\}$. We have that $d_T(u', v') = 3$, so we can construct a hamiltonian cycle in $T$ by removing edge $\{u, u'\}$ from $H$ and edge $\{v, v'\}$ from $H'$, and joining these cycles together by adding edges $\{u', v'\}$ and $\{u, v\}$.

Thus, in any case, we can construct a hamiltonian cycle in $T^3$ that contains edge $\{u, v\}$.

- 教材讨论
  - JH第3章第4、5节

## 问题1：Lowering Worst Case Complexity of Exponential Algorithms

- 3SAT
  - 用divide-and-conquer解决这个问题的方法是什么？

$$F = (x_1 \vee \overline{x}_2 \vee x_4) \wedge (\overline{x}_2) \wedge (\overline{x}_2 \vee \overline{x}_3 \vee x_5) \wedge (x_1 \vee \overline{x}_5) \wedge (\overline{x}_1 \vee x_2 \vee x_3)$$

$$F(\overline{x}_2 = 1) = (x_1 \vee \overline{x}_5) \wedge (\overline{x}_1 \vee x_3)$$

$F$ is satisfiable $\Longleftrightarrow$ at least one of the formulae $F(l_1 = 1)$, $F(l_1 = 0, l_2 = 1), F(l_1 = 0, l_2 = 0, l_3 = 1)$ is satisfiable.

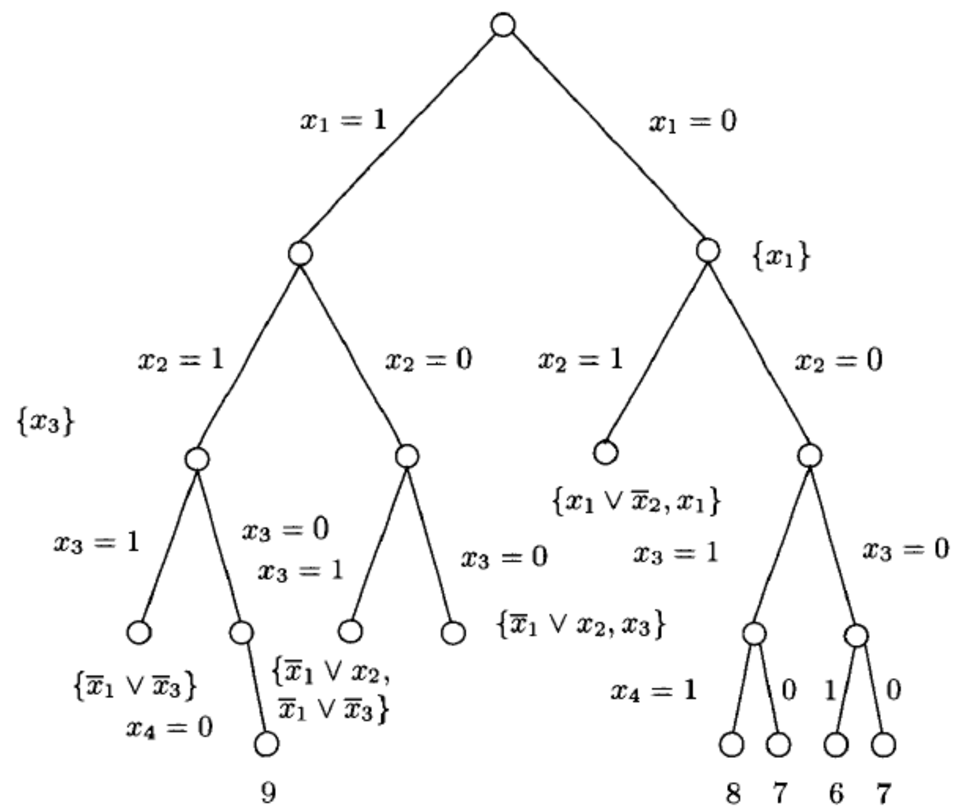
$$F(l_1 = 1) \in 3\mathrm{CNF}(n-1, r-1),$$
$$F(l_1 = 0, l_2 = 1) \in 3\mathrm{CNF}(n-2, r-1),$$
$$F(l_1 = 0, l_2 = 0, l_3 = 1) \in 3\mathrm{CNF}(n-3, r-1).$$

  - 这样做能够显著降低复杂度吗？

| Complexity | $n = 10$ | $n = 50$ | $n = 100$ | $n = 300$ |
|---|---|---|---|---|
| $2^n$ | 1024 | (16 digits) | (31 digits) | (91 digits) |
| $2^{n/2}$ | 32 | $\sim 33 \cdot 10^6$ | (16 digits) | (46 digits) |
| $(1.2)^n$ | 7 | 9100 | $\sim 29 \cdot 10^6$ | (24 digits) |
| $10 \cdot 2^{\sqrt{n}}$ | 89 | 1350 | 10240 | $\sim 1.64 \cdot 10^6$ |
| $n^2 \cdot 2^{\sqrt{n}}$ | 894 | $\sim 336000$ | $\sim 10.24 \cdot 10^6$ | $\sim 14.8 \cdot 10^9$ |

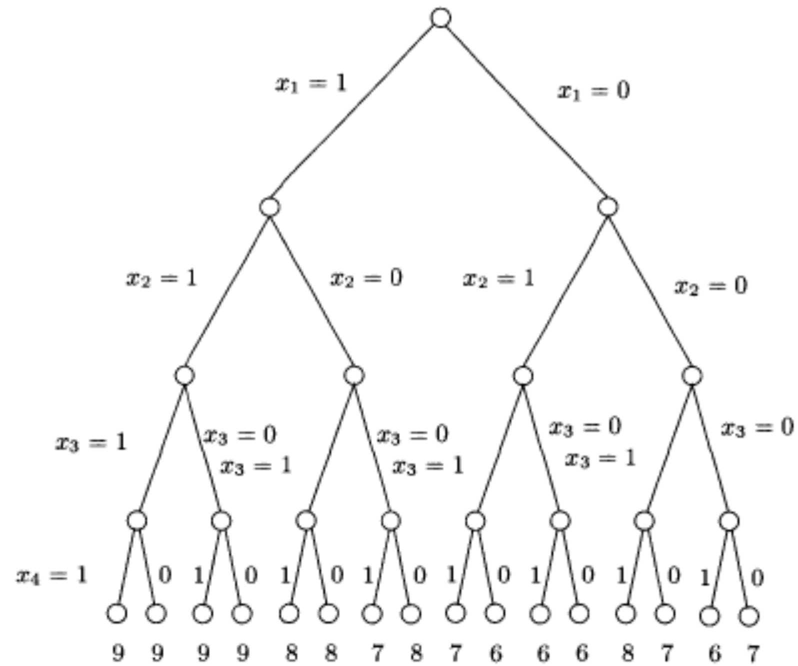
# 问题2：branch-and-bound

- branch-and-bound能够提高效率的基本原理是什么？
- branch-and-bound算法的效率与哪些因素有关？
  - 树的构造方法
  - 树的搜索策略
  - 最优解的界
  - 子树中解的范围估计



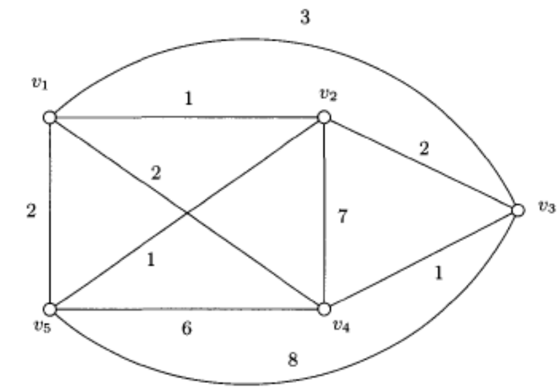
# 问题2：branch-and-bound (续)

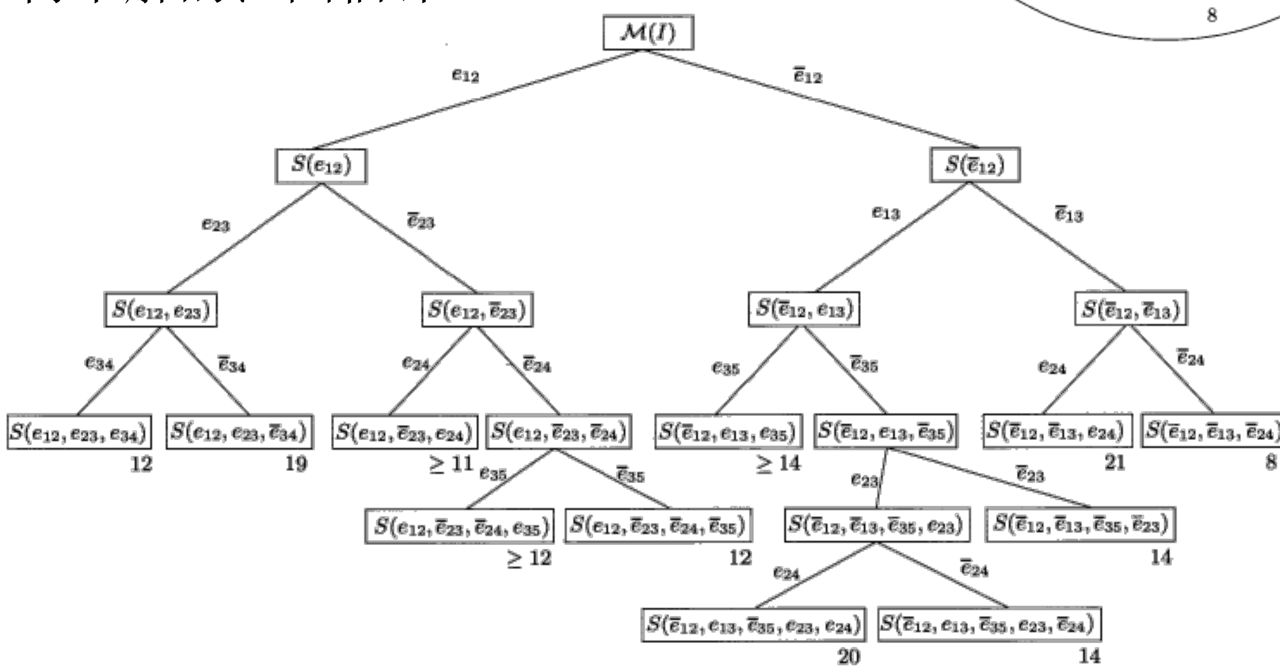
- MAX-SAT
  - 树的构造方法
  - 树的搜索策略
  - 最优解的界
  - 子树中解的范围估计
- 你能给出和书上不同的算法吗？
  - 例如：不同的搜索策略



# 问题2：branch-and-bound (续)

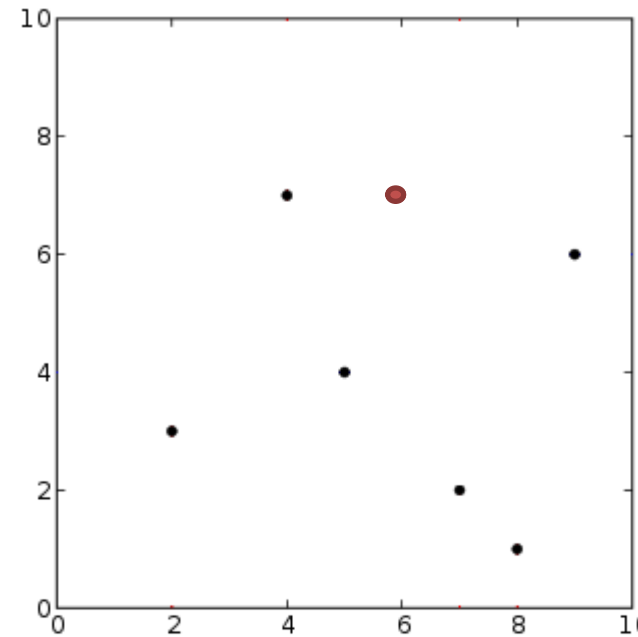
- TSP
  - 树的构造方法
  - 树的搜索策略
  - 最优解的界
  - 子树中解的范围估计



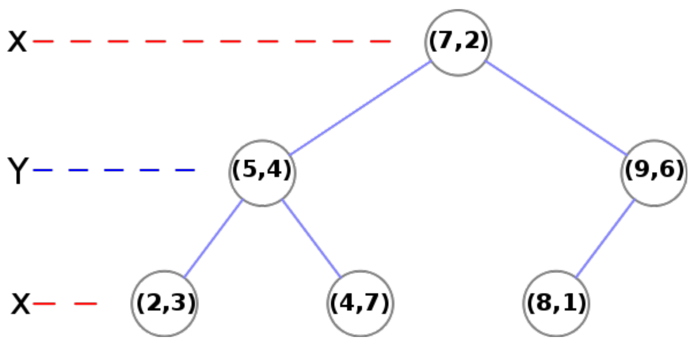
# 问题2： branch-and-bound (续)

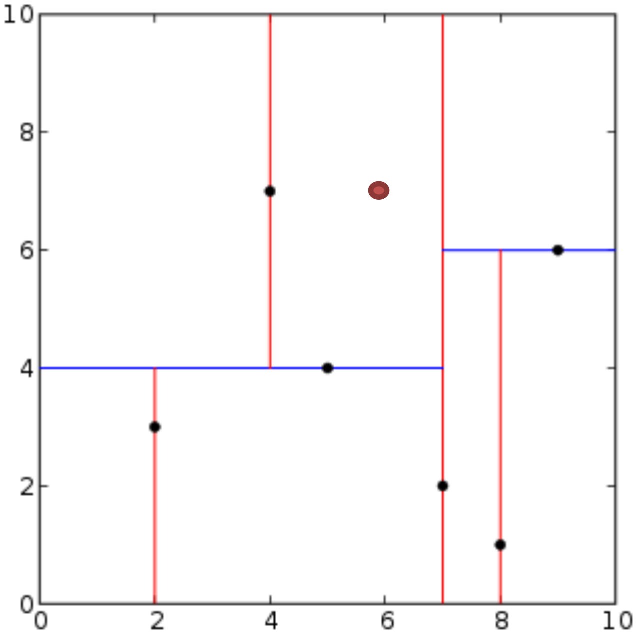
- NNS (nearest neighbor search)
  - 树的构造方法
  - 树的搜索策略
  - 最优解的界
  - 子树中解的范围估计



# 问题2： branch-and-bound (续)

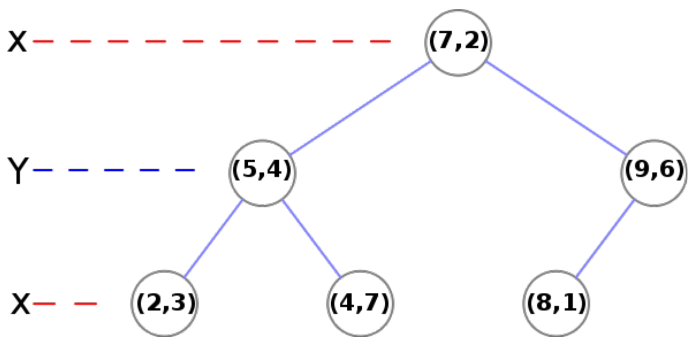
- k-d tree

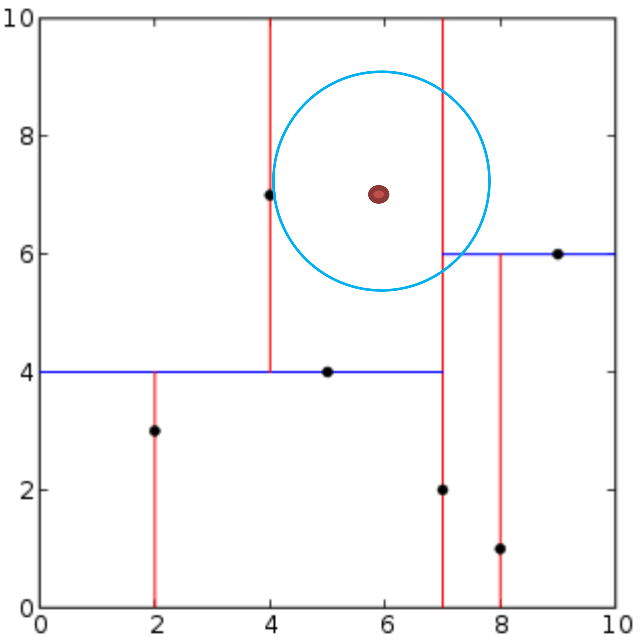


# 问题2： branch-and-bound (续)

- k-d tree



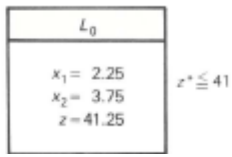



# 问题2：branch-and-bound (续)

- ILP
  - 树的构造方法
  - 树的搜索策略
  - 最优解的界
  - 子树中解的范围估计

$$
\begin{aligned}
\max z &= 5x_1 + 8x_2, \\
\text{subject to:} \quad & x_1 + x_2 \leq 6, \\
& 5x_1 + 9x_2 \leq 45, \\
& x_1, x_2 \geq 0 \quad \text{and} \quad \underline{\text{integer.}}
\end{aligned}
$$

# 问题2： branch-and-bound (续)



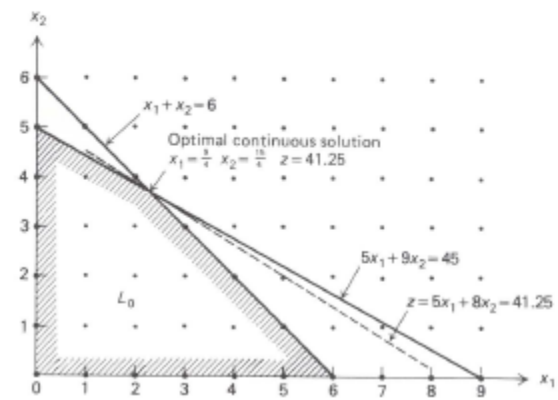

$$\max z = 5x_1 + 8x_2,$$

subject to:

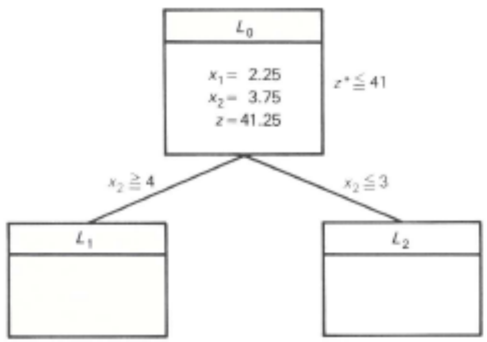
$$x_1 + x_2 \le 6,$$

$$5x_1 + 9x_2 \le 45,$$

$$x_1, x_2 \ge 0 \quad \text{and} \quad \text{integer}.$$



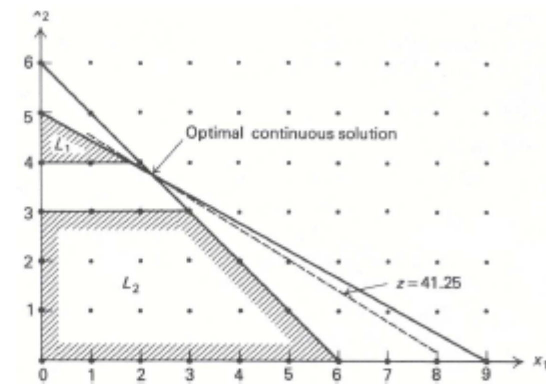
# 问题2： branch-and-bound (续)



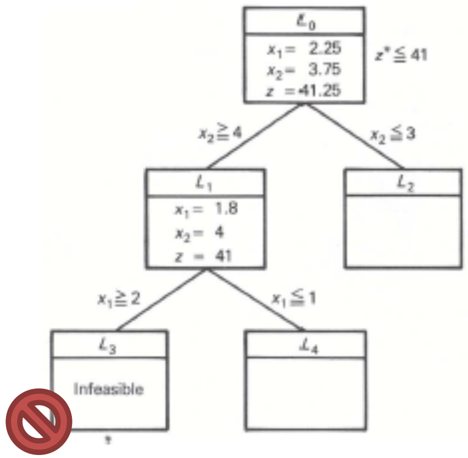

$$\max z = 5x_1 + 8x_2,$$

subject to:

$$x_1 + x_2 \leq 6,$$
$$5x_1 + 9x_2 \leq 45,$$
$$x_1, x_2 \geq 0 \quad \text{and} \quad \text{integer}.$$



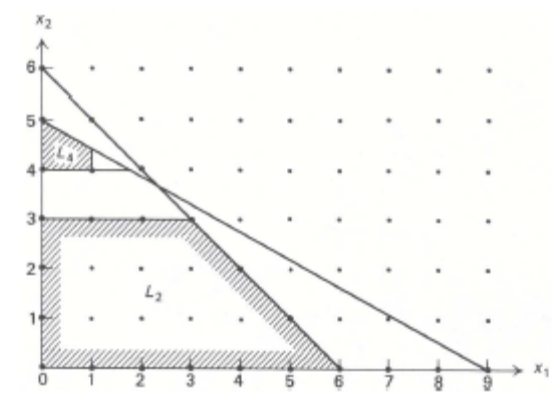
# 问题2： branch-and-bound (续)



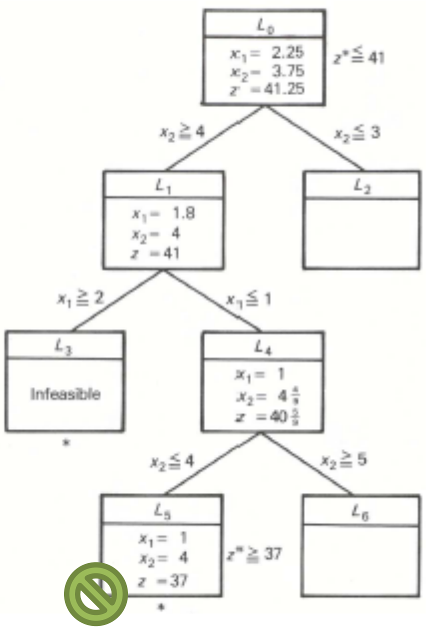

$$\max z = 5x_1 + 8x_2,$$

subject to:

$$x_1 + x_2 \leq 6,$$
$$5x_1 + 9x_2 \leq 45,$$
$$x_1, x_2 \geq 0 \quad \text{and} \quad \text{integer}.$$



# 问题2： branch-and-bound (续)



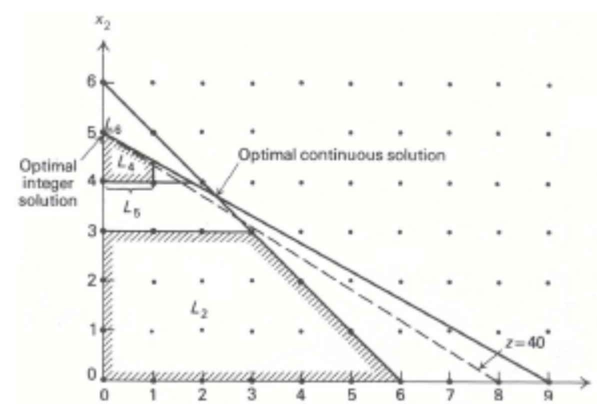

$$\max z = 5x_1 + 8x_2,$$

subject to:

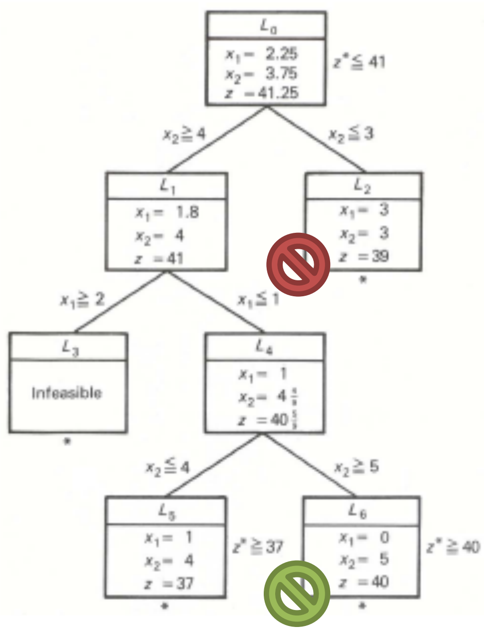
$$x_1 + x_2 \le 6,$$

$$5x_1 + 9x_2 \le 45,$$

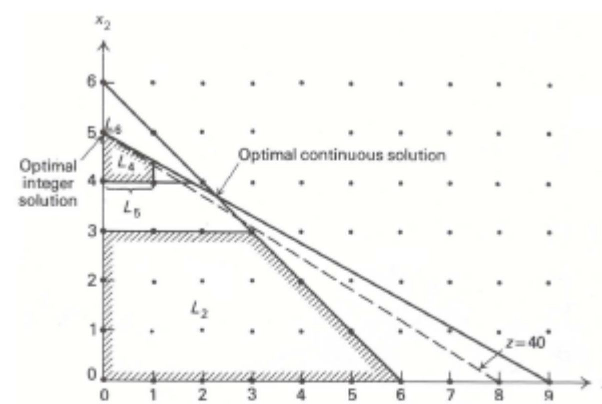
$$x_1, x_2 \ge 0 \quad \text{and} \quad \text{integer}.$$



# 问题2： branch-and-bound (续)

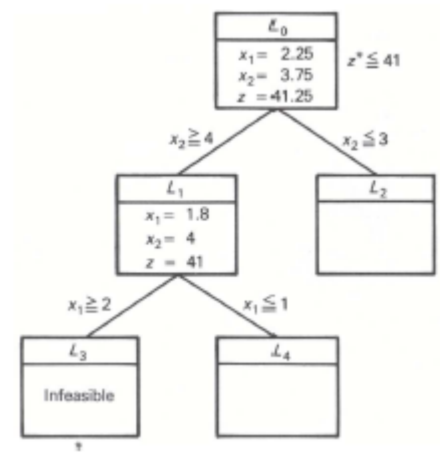



$$
\begin{aligned}
\max z &= 5x_1 + 8x_2, \\
\text{subject to:} \quad & x_1 + x_2 \leq 6, \\
& 5x_1 + 9x_2 \leq 45, \\
& x_1, x_2 \geq 0 \quad \text{and} \quad \text{integer.}
\end{aligned}
$$



# 问题2： branch-and-bound (续)

- 如何选择下一个被计算的顶点？
- 如何选择用来细分的变量？



# 问题2： branch-and-bound (续)

- QAP (quadratic assignment problem)
  - There are a set of *n* facilities and a set of *n* locations. For each pair of locations, a distance is specified and for each pair of facilities a weight or flow is specified (e.g., the amount of supplies transported between the two facilities). The problem is to assign all facilities to different locations with the goal of minimizing the sum of the distances multiplied by the corresponding flows.

Given two sets, $P$ ("facilities") and $L$ ("locations"), of equal size, together with a weight function $w : P \times P \to \mathbf{R}$ and a distance function $d : L \times L \to \mathbf{R}$. Find the bijection $f : P \to L$ ("assignment") such that the cost function:

$$\sum_{a,b \in P} w(a,b) \cdot d(f(a), f(b))$$

is minimized.