Operations research meets data science: principles, models and algorithms

Immanuel Bomze, University of Vienna

Dip. Matematica@Univ. Padova 07 November 2025

Connectedness and spanning trees

Given: **un**directed connected graph $\mathcal{G} = (V, E)$
with weight/distance $w_e = w_{ij}$ on edge $e = \{i, j\}$.

A *tree* $\mathcal{T}$ is a connected subgraph $\mathcal{T} = (V_T, E_T)$ of $\mathcal{G}$ (hence $V_T \subseteq V$ and $E_T \subseteq E$) which has no cycles.

Weight of T is

$$W(\mathcal{T}) = \sum_{\{i,j\} \in E_T} w_{ij} \,.$$

Search for tree $\mathcal{T}$ with minimal $W(\mathcal{T})$ which *spans* $\mathcal{G}$, i.e. $V_T = V$.
Spanning tree is formed by dropping cycle-generating edges while maintaining connectivity.

If $\mathcal{G}$ is disconnected, repeat with components – spanning forests.

Greedy algorithm (Kruskal)

Initialize: $E_1 = \{e_1\}$ where w_{e_1} is minimal among all $e \in E$.

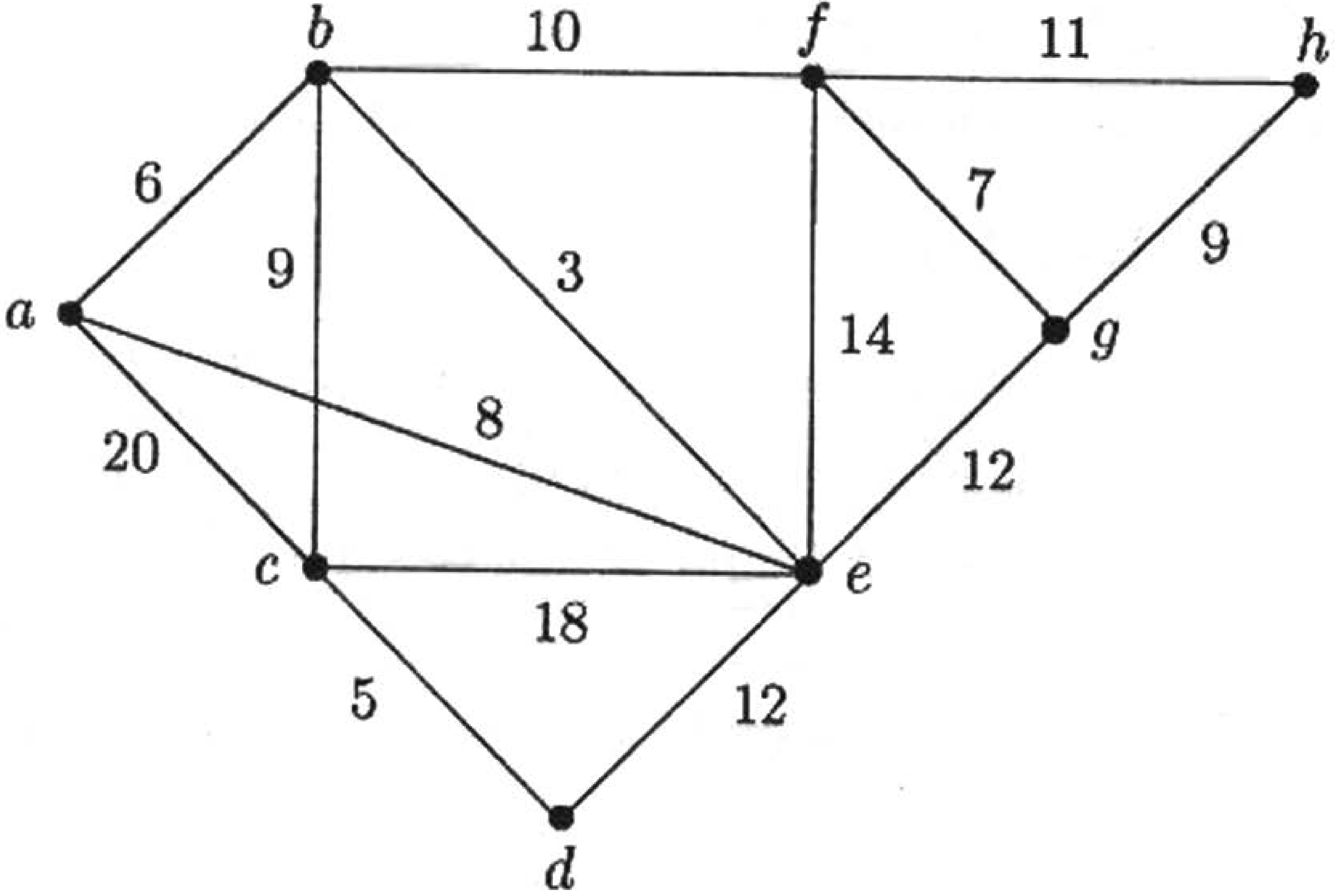
Iterate:

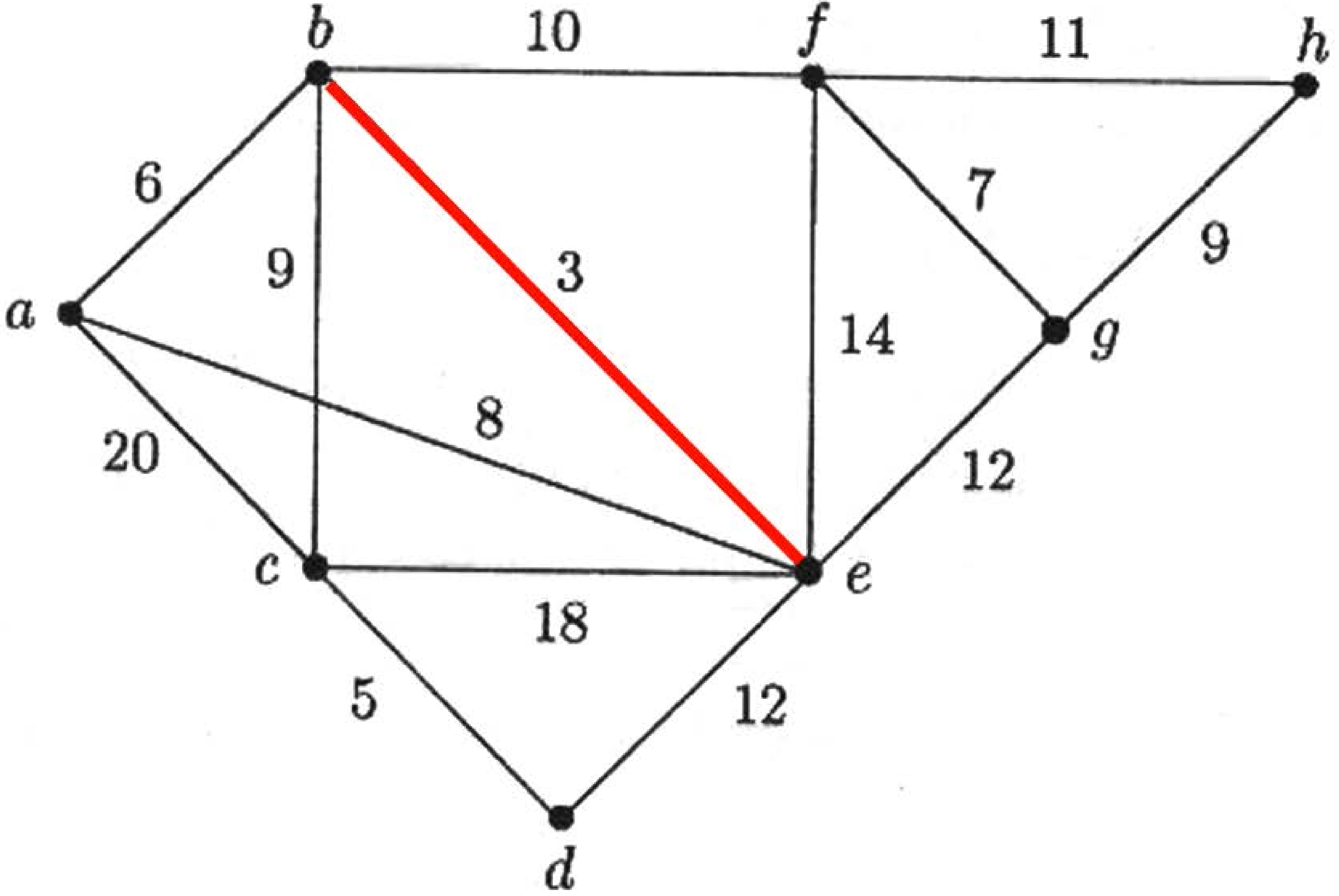
$$E_{k+1} = E_k \cup \{e_k\} \text{ if } e_k \text{ has smallest } w_e \text{ across all } e \in E \setminus E_k\,,$$

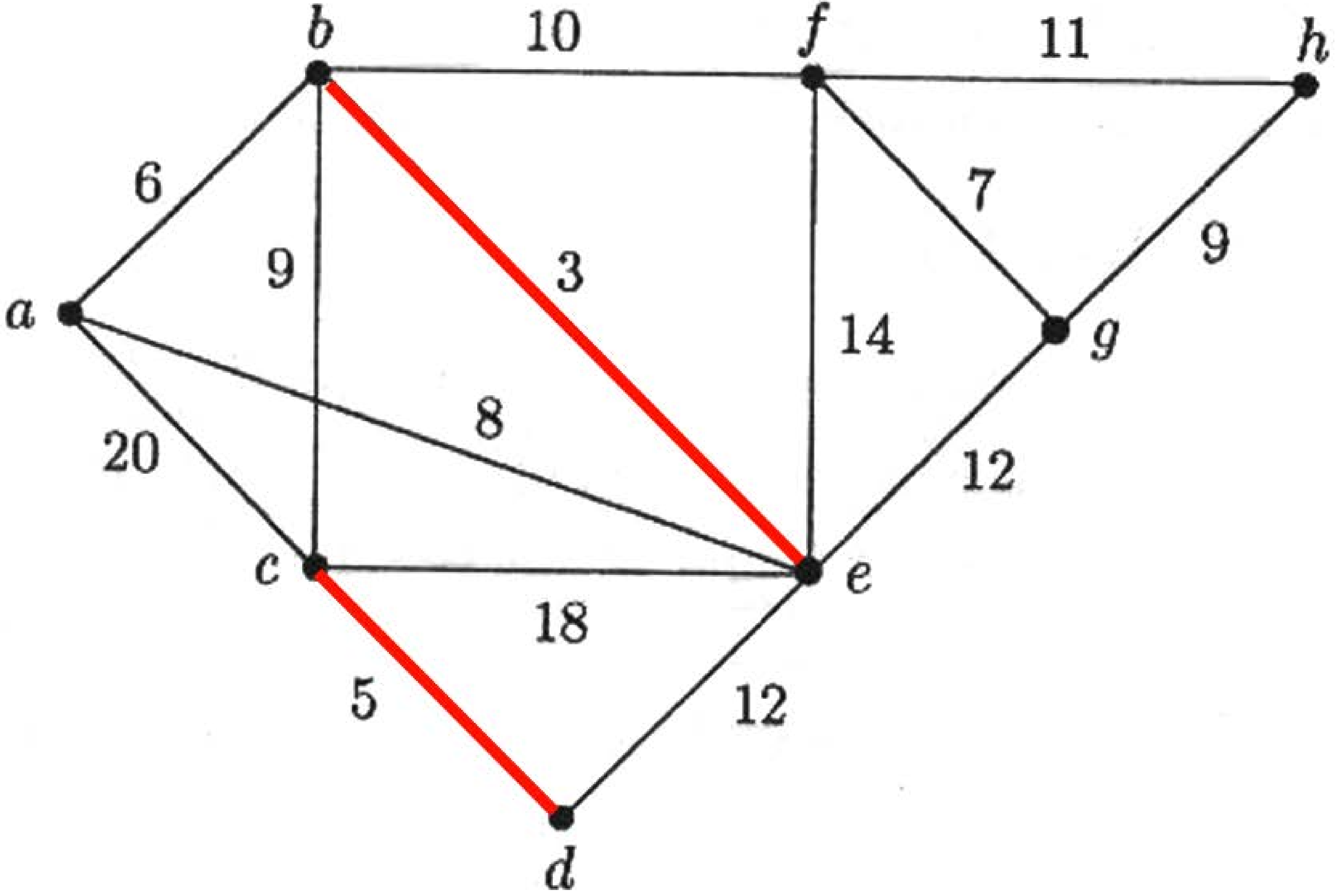
if no cycle is generated; otherwise, retry with next smallest w_e.

Stop if only cycle-generating edges are left.

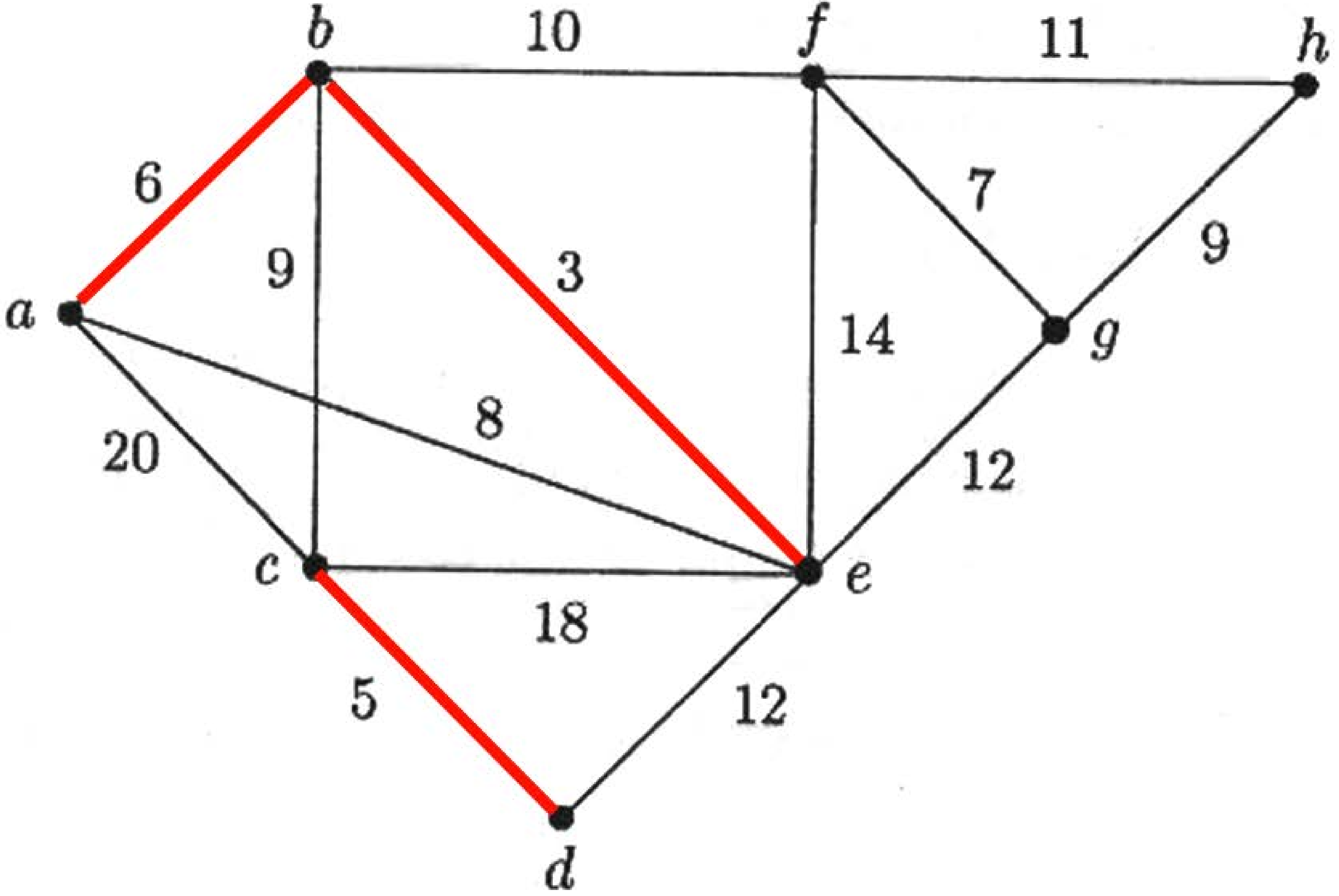
Efficient in sparse graphs ($m = |E|$ small): $\mathcal{O}(m \log m)$.


b
10
f
11
h
6
9
3
7
9
a
14
g
8
20
12
c
e
18
5
12
d

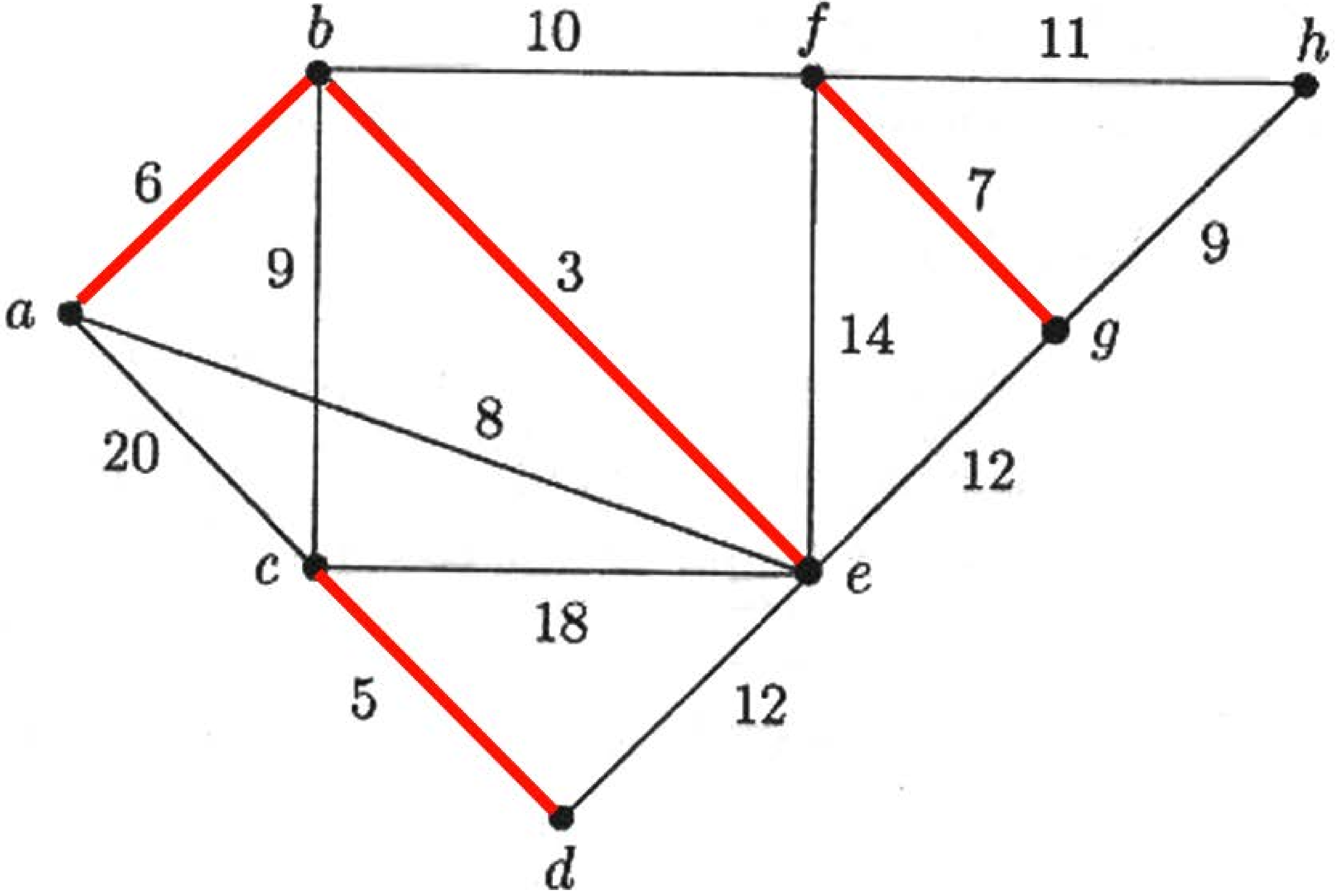

b
10
f
11
h
6
7
9
3
9
a
14
g
8
20
12
c
e
18
5
12
d



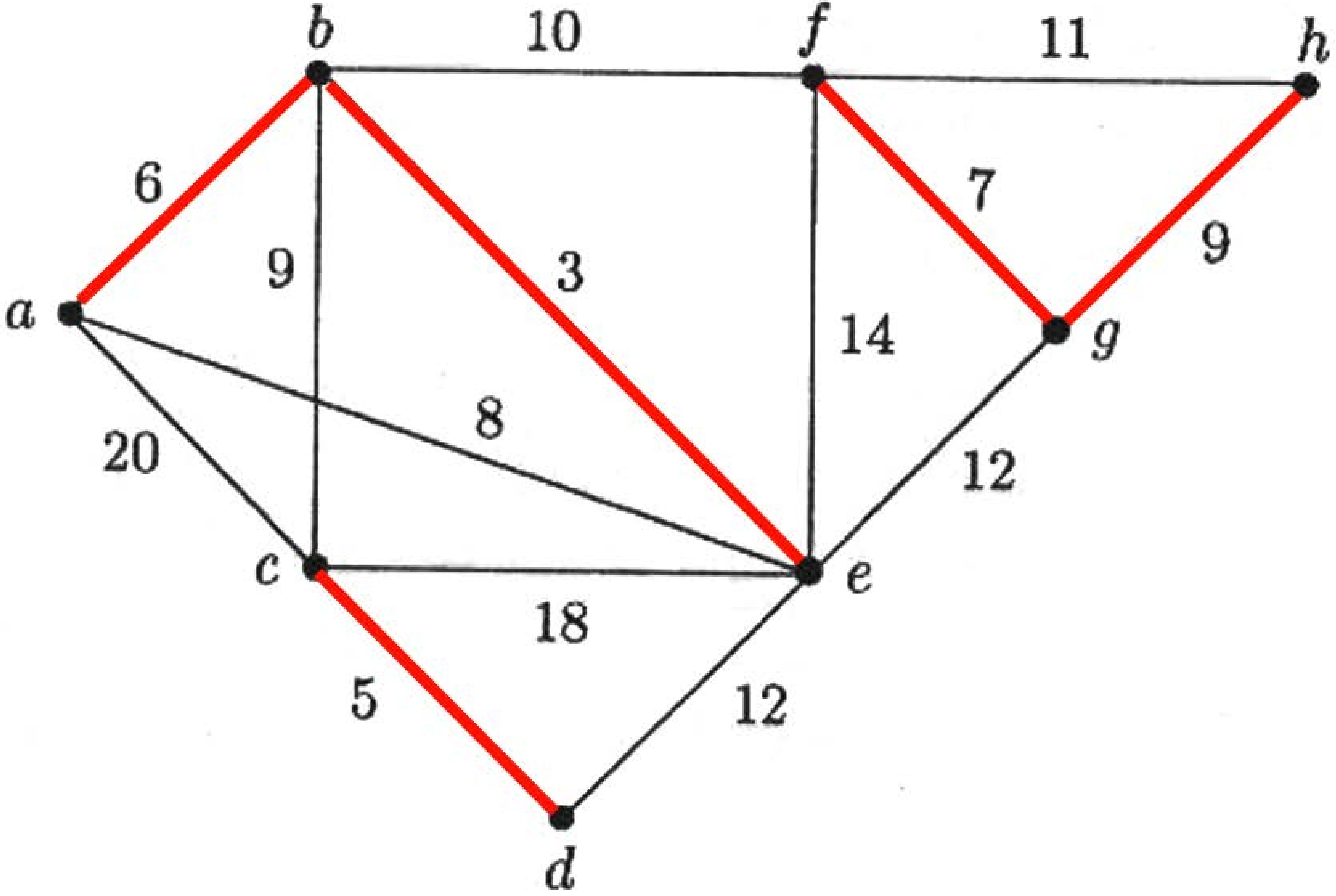
b
10
f
11
h
6
7
9
3
9
a
14
g
8
20
12
c
e
18
5
12
d



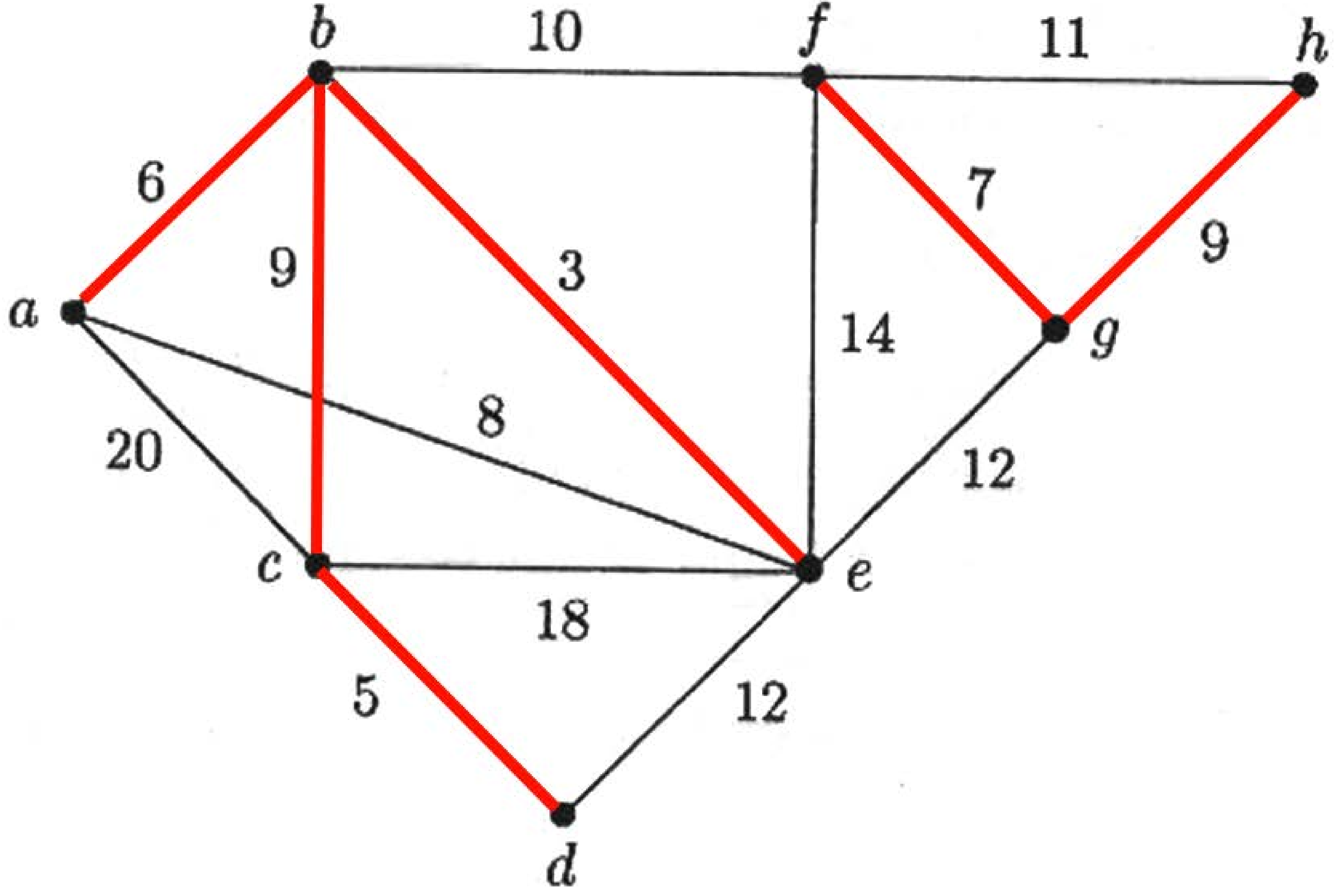
b
10
f
11
h
6
7
9
3
9
a
14
g
8
20
12
c
e
18
5
12
d

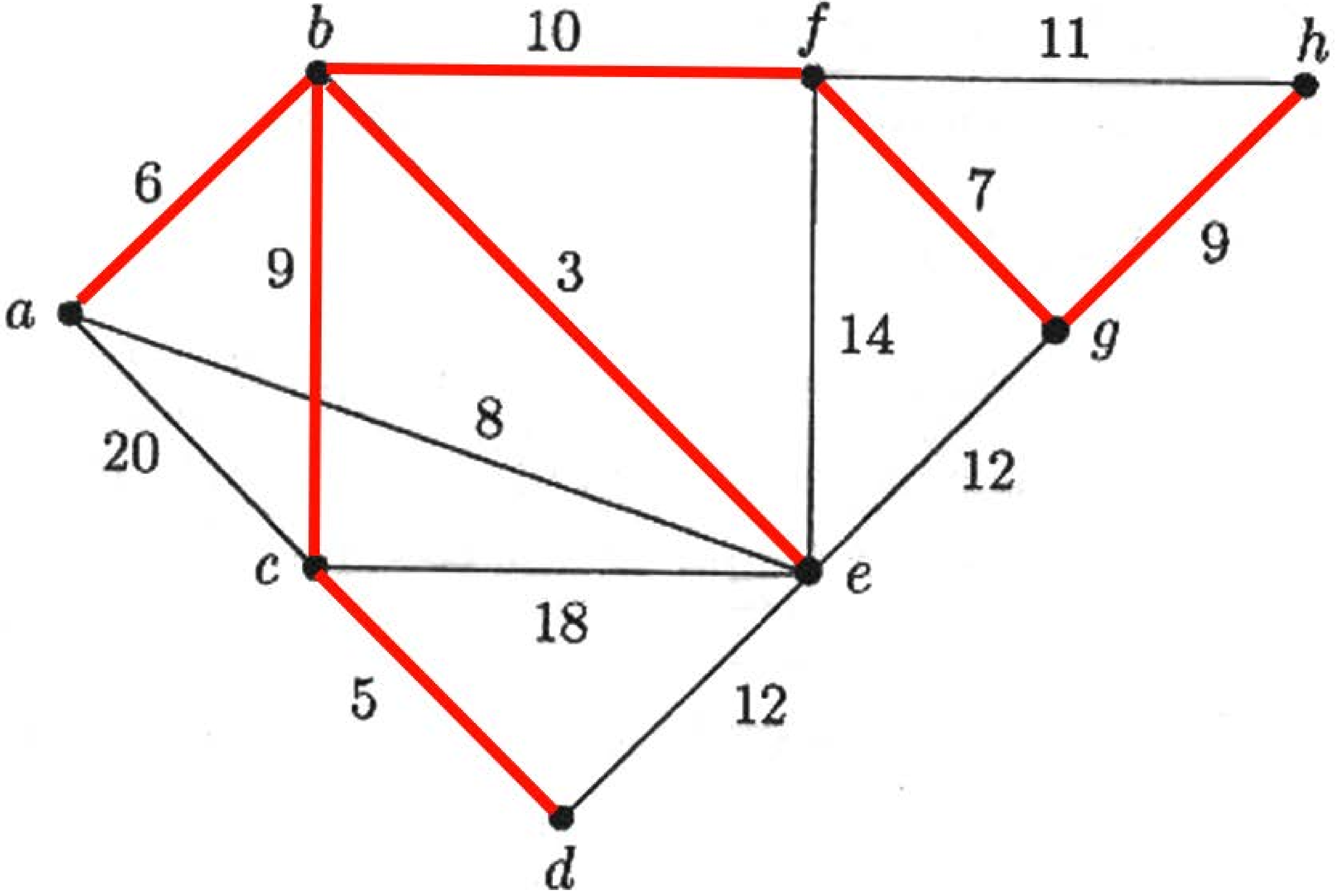


b
10
f
11
h
6
7
9
3
9
a
14
g
8
20
12
c
e
18
5
12
d



a
b
c
d
e
f
g
h
6
9
3
10
11
7
9
14
8
20
12
18
5
12


b
10
f
11
h
6
7
9
3
9
a
14
g
8
20
12
c
e
18
5
12
d



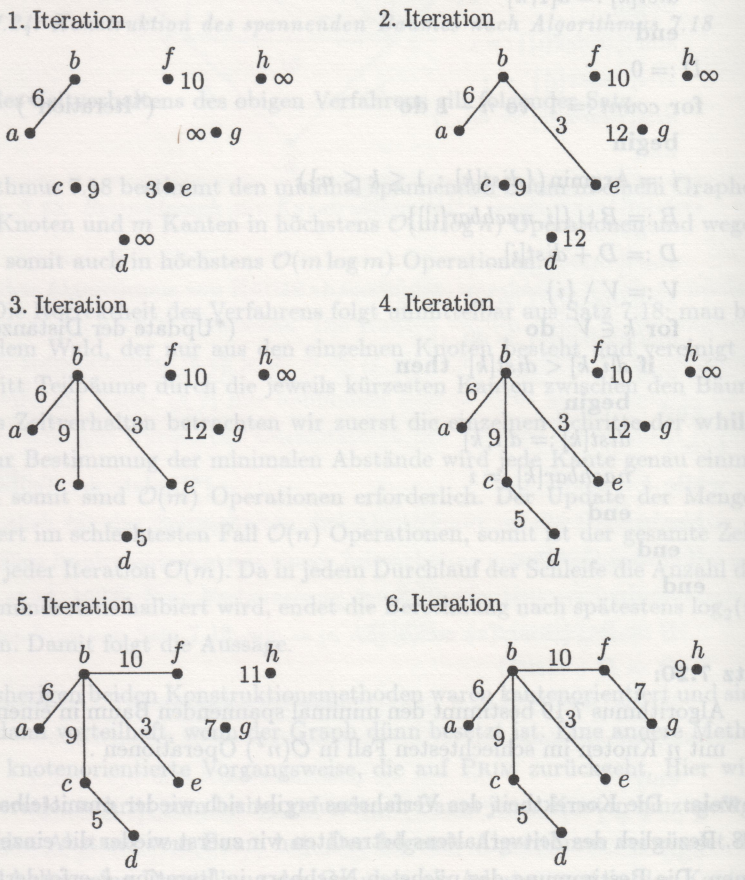
b
10
f
11
h
6
9
3
7
9
a
14
g
8
20
12
c
e
18
5
12
d

Prim's MST-algorithm

Basic idea: add nearest neighbor to current partial tree;
use shortest edge to connect it to current partial tree;

Label all remaining vertices v by distance to current subtree;
update labels after updating partial tree.

If m large ($\approx n^2/2$), then $\mathcal{O}(n^2)$, better than above.


1. Iteration
2. Iteration
3. Iteration
4. Iteration
5. Iteration
6. Iteration

Discrete Problems – iconic examples 1

General form: $\min \{f(\mathbf{x}) : \mathbf{x} \in M\}$ with M finite/discrete.

Problem here: $|M|$ is huge.

The knapsack problem (KP)

$$\max \left\{\mathbf{p}^\top \mathbf{x} : \mathbf{w}^\top \mathbf{x} \leq W \,,\, x_i \in \{0, 1\}\right\} .$$

Is an LP with additional constraints that x_j are binary, not fractional, an **Integer LP (ILP)**.

For $n = 10$ have $2^{10} = 1024 \approx 10^3$ choices.

For $n = 100$ have $\geq 10^{30}$ ...

A SERIES OF BOOKS IN THE MATHEMATICAL SCIENCES
Victor Klee, Editor

COMPUTERS AND INTRACTABILITY

A Guide to the Theory of NP-Completeness

Michael R. Garey / David S. Johnson

BELL LABORATORIES
MURRAY HILL, NEW JERSEY

1979

W. H. FREEMAN AND COMPANY
New York

Size of Largest Problem Instance
Solvable in 1 Hour

Time complexity function	With present computer	With computer 100 times faster	With computer 1000 times faster
n	N_1	100 N_1	1000 N_1
n^2	N_2	10 N_2	31.6 N_2
n^3	N_3	4.64 N_3	10 N_3
n^5	N_4	2.5 N_4	3.98 N_4
2^n	N_5	$N_5 + 6.64$	$N_5 + 9.97$
3^n	N_6	$N_6 + 4.19$	$N_6 + 6.29$

Figure 1.3 Effect of improved technology on several polynomial and exponential time algorithms.

Discrete Problems – iconic examples 2

Optimal partitions/covers of sets:

Given $Z = \{1, \ldots, m\}$ and system $\mathcal{M} = \{M_1, \ldots, M_n\}$ of subsets $M_i \subseteq Z$ with cost c_i, find $\left\{M_{i_1}, \ldots, M_{i_k}\right\} \subset \mathcal{M}$, with $Z = \bigcup_{j=1}^{k} M_{i_j}$ such that

$$\sum_{j=1}^{k} c_{i_j} \quad \text{is minimal.}$$

Partition: request in addition $M_{i_j} \cap M_{i_\ell} = \emptyset$ if $j \neq \ell$.

Incidence $m \times n$-matrix

$$a_{ij} = \begin{cases} 1\,, & \text{if } i \in M_j\,, \\ 0\,, & \text{if } i \in Z \setminus M_j\,. \end{cases}$$

ILP form of partition and set covering problems

Let $\mathbf{e} := [1, \ldots, 1]^\top \in \mathbb{R}^m$ and A the $m \times n$ incidence matrix.

Partition problem:

$$\begin{aligned} \mathbf{c}^\top \mathbf{x} &\to \text{min !} \\ \text{s.t.} \quad A\mathbf{x} &= \mathbf{e} \\ x_j &\in \{0, 1\}, \text{ all } j. \end{aligned}$$

Coverage problem:

$$\begin{aligned} \mathbf{c}^\top \mathbf{x} &\to \text{min !} \\ \text{s.t.} \quad A\mathbf{x} &\geq \mathbf{e} \\ x_j &\in \{0, 1\}, \text{ all } j. \end{aligned}$$

$x_j = 1$ means M_j is chosen. Again 2^n choices. But $n \lessapprox 2^m$ ...

Iconic example 3 – the TSP

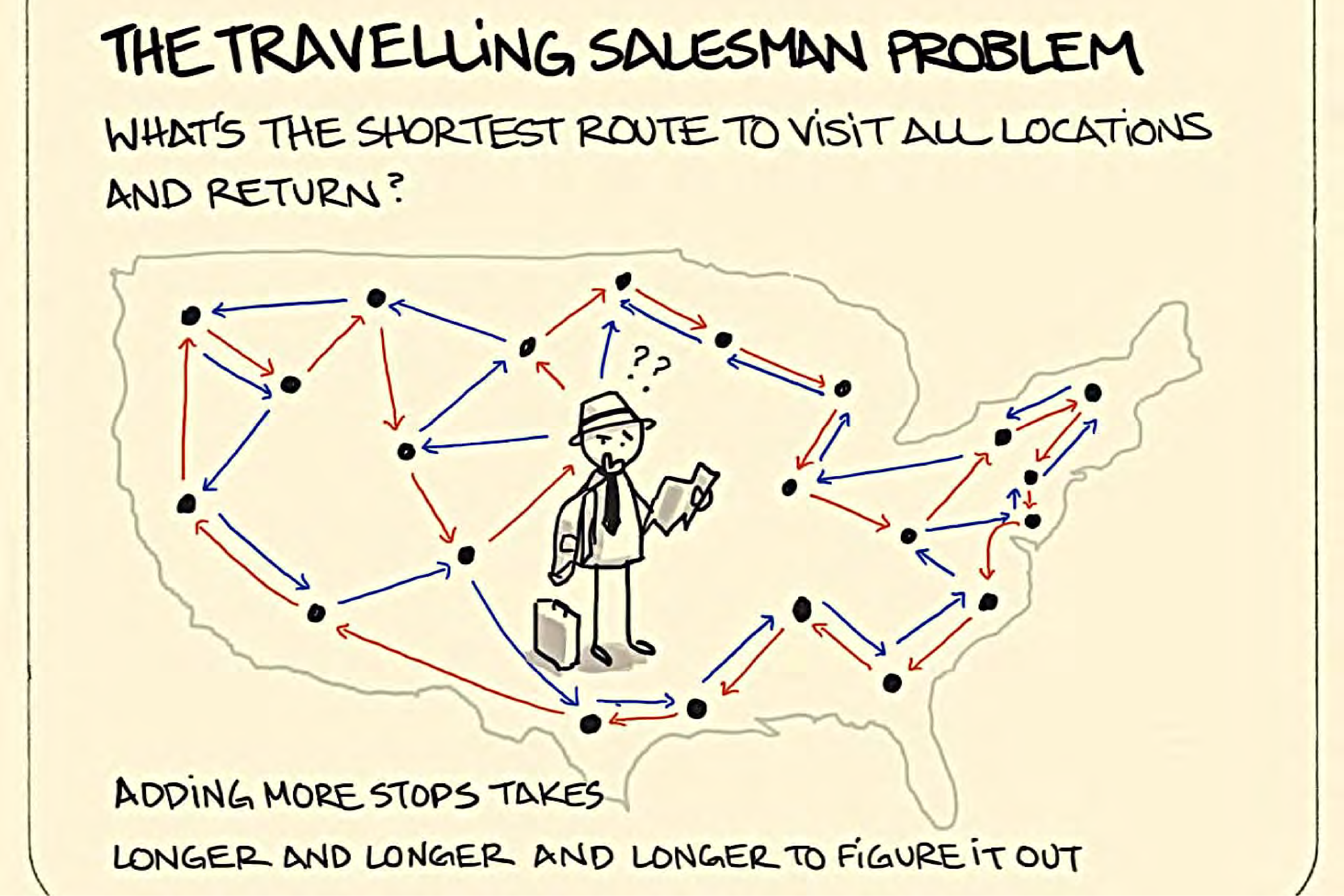
Travelling Salesman Problem searches for optimal permutation of $n+1$ stations in round trip: how many choices ?

$$n! = n(n-1)\cdots 2\cdot 1 \approx n^n e^{-n}\sqrt{2\pi n}\,.$$

Graph theory ($n+1 \hookrightarrow n$): search for *Hamiltonian cycle* $[e_1,\ldots,e_n]$ with minimal weight

$$\sum_k c(e_k) \qquad \text{where } c(e_k) = c_{ij} \text{ if } e_k = \{i,j\}\ .$$

https://co-enzyme.fr/blog/traveling-salesman-problem-tsp-in-cplex-opl-with-miller-tucker-zemlin-mtz-formulation/



Home

> Optimal Tours

VLSI 85,900

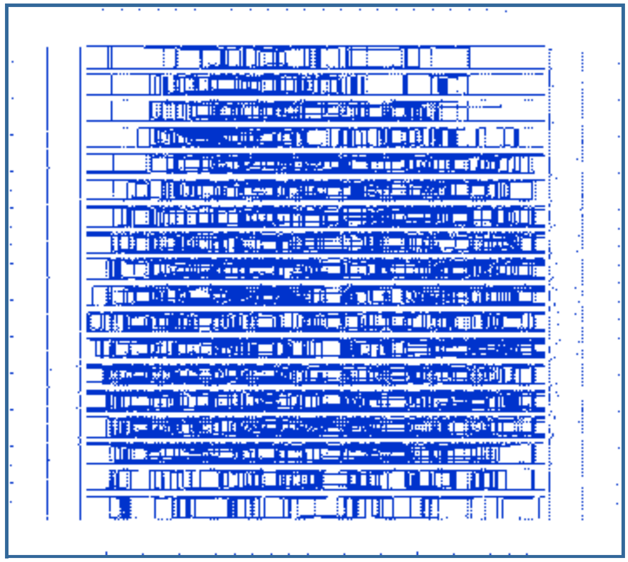
Sweden 24,978

Germany 15,112

USA 13,509

Optimal Tours

Information on the largest TSP instances solved to date can be found by following the links given below.



85,900 Locations in a VLSI Application
Solved in 2006

24,978 Cities in Sweden
Solved in 2004

15,112 Cities in Germany
Solved in 2001

ILP form of TSP

$$
\begin{aligned}
\sum_{i,j=1}^{n} c_{ij} x_{ij} &\rightarrow \text{min!} \\
\sum_{i=1}^{n} x_{ij} &= 1\,, \quad 1 \leq j \leq n\,, \\
\sum_{j=1}^{n} x_{ij} &= 1\,, \quad 1 \leq i \leq n\,, \\
\sum_{i \in S, j \notin S} x_{ij} &\geq 1 \quad \text{for all} \quad S \subset \{1, 2, \ldots, n\}\,, \\
x_{ij} &\in \{0, 1\}, \quad 1 \leq i, j \leq n\,.
\end{aligned}
$$

$x_{ij} = 1$ means i succeeds j in trip.

A disillusioning observation

A simple knapsack problem:

$$\begin{aligned} 5x_1 + 4x_2 + x_3 + x_4 &\to \text{max!} \\ 4x_1 + 5x_2 + 2x_3 + x_4 &\leq 7 \\ x_i &\in \{0,1\}, \quad 1 \leq i \leq 4. \end{aligned}$$

Idea: *relax* constraint $x_i \in \{0,1\}$ to $x_i \in [0,1]$, i.e. $0 \leq x_i \leq 1$.
Gives LP with optimal solution

$$\mathbf{x}^*_{\text{LP}} = [1, \tfrac{2}{5}, 0, 1]^\top .$$

Of course infeasible to ILP, so round down fractional value $\frac{2}{5} \approx 0$.
Value is 6. Rounding $\frac{2}{5}$ to 1 is infeasible !

Is there a better integer solution ? Far away from $\mathbf{x}^*_{\text{LP}}$:

$$\mathbf{x}^*_{\text{ILP}} = [1, 0, 1, 1]^\top \quad \text{is feasible with value } 7 .$$

Discrete Problems as decision trees

Master problem denoted by P_0 : $z^*(P_0) = \min\{f(\mathbf{x}); \mathbf{x} \in M\}$.
Decompose feasible set $M = \bigcup_i M_i$, consider *subproblem*
P_i : $z^*(P_i) = \min\{f(\mathbf{x}) : \mathbf{x} \in M_i\}$. Try to solve P_i.

If $M_i = \{\bar{\mathbf{x}}\}$, no choice, $z^*(P_i) = f(\bar{\mathbf{x}})$, solved.

If $|M_i| > 1$, then iterate above, replacing M with all M_i's.

Results in a *branching structure*, a hierarchy of subproblems, organized in a tree.

Optimal values are ordered: $z^*(P_0) \leq z^*(P_i)$ as $M_i \subset M$ (why ?).

Furthermore granularity consistence:

$$z^*(P_0) = \min_i z^*(P_i) \quad \text{as} \quad M = \bigcup_i M_i \,.$$

ECONOMETRICA

VOLUME 28 July, 1960 NUMBER 3

AN AUTOMATIC METHOD OF SOLVING DISCRETE PROGRAMMING PROBLEMS

BY A. H. LAND AND A. G. DOIG

In the classical linear programming problem the behaviour of continuous, nonnegative variables subject to a system of linear inequalities is investigated. One possible generalization of this problem is to relax the continuity condition on the variables. This paper presents a simple numerical algorithm for the solution of programming problems in which some or all of the variables can take only discrete values. The algorithm requires no special techniques beyond those used in ordinary linear programming, and lends itself to automatic computing. Its use is illustrated on two numerical examples.

1. INTRODUCTION

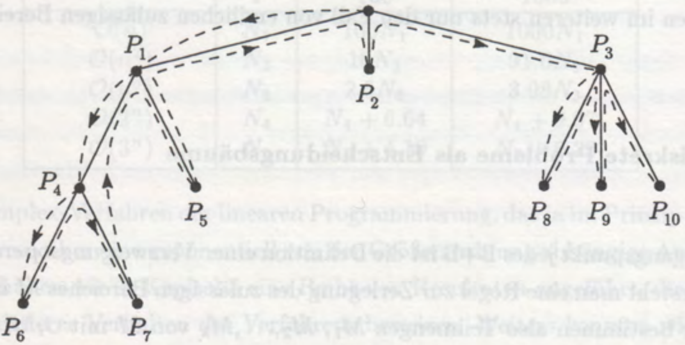
THERE IS A growing literature [**1**, **3**, **5**, **6**] about optimization problems which could be formulated as linear programming problems with additional constraints that some or all of the variables may take only integral values. This form of linear programming arises whenever there are indivisibilities. It is not meaningful, for instance, to schedule 3 – 7/10 flights between two cities, or to undertake only 1/4 of the necessary setting up operation for running a job through a machine shop. Yet it is basic to linear programming that the variables are free to take on any positive value,[1] and this sort of answer is very likely to turn up.

In some cases, notably those which can be expressed as transport problems, the linear programming solution will itself yield discrete values of the variables. In other cases the percentage change in the maximand[2] from

Ailsa Land (1927-2021)
EURO Gold medal 2021

Alison Harcourt
(née Doig, *1929)

he als mögliche Lösungen ausschließen. Für den speziellen Fall von
len bezeichnet man derartige Verfahren auch als implizite Enumerat
ten im weiteren stets nur den endlichen zulässigen Bereich



Divide and conquer: branching rules

How to decompose ? *Branching rules*, e.g. fixing variables.
If $x_1 \in \{0,1\}$, results in two M_i:

$$\begin{aligned} M_1 &= \{\mathbf{x} \in M : x_1 = 1\} \\ M_2 &= \{\mathbf{x} \in M : x_1 = 0\} \end{aligned}$$

Binary case $x_i \in \{0,1\}$ for all $i \leq n$ gives binary tree, depth n and potential width 2^n.

Many other branching rules (includes selection of variables to be fixed): strong branching, pseudocost branching, reliability branching, most infeasible branching ...

Fig. 5. Comparison of node trees resulting from *full strong branching* for **vpm2** and **neos3**.

Branching in the knapsack problem

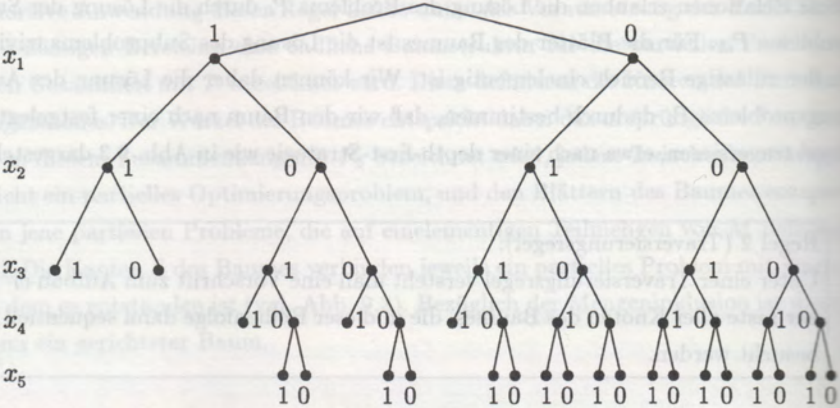
$$\begin{aligned} 5x_1 + 4x_2 + 3x_3 + 2x_4 + x_5 &\to \text{max!} \\ 4x_1 + 5x_2 + x_3 + 3x_4 + 3x_5 &\leq 9 \\ x_i &\in \{0,1\}, \quad 1 \leq i \leq 5\,. \end{aligned}$$

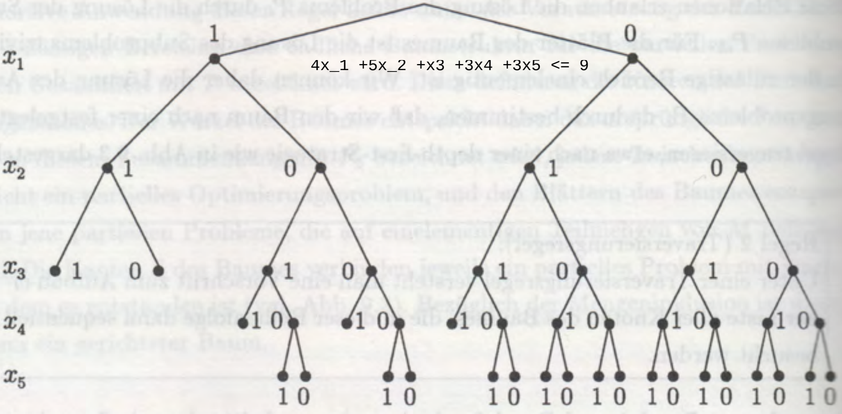
First fix x_1, resulting in two M_i:

$$\begin{aligned} M_1 &= \{\mathbf{x} \in M : x_1 = 1\} \\ M_2 &= \{\mathbf{x} \in M : x_1 = 0\} \end{aligned}$$

Then decompose both M_i by fixing x_2 etc.

Stop branching (prune tree) if all remaining x_j are forced to zero by constraint.


x_1
x_2
x_3
x_4
x_5



4x_1 +5x_2 +x3 +3x4 +3x5 <= 9
x_1
x_2
x_3
x_4
x_5

Wait – just reorganizing feasible set ?

No – *branch and bound* is not explicit enumeration !

Suppose have bounds

$$\ell_i \leq z^*(P_i) \leq u_i$$

for a subproblem P_i (node in problem tree), and a current upper bound

$$z^*(P_0) \leq u_{\text{current}} .$$

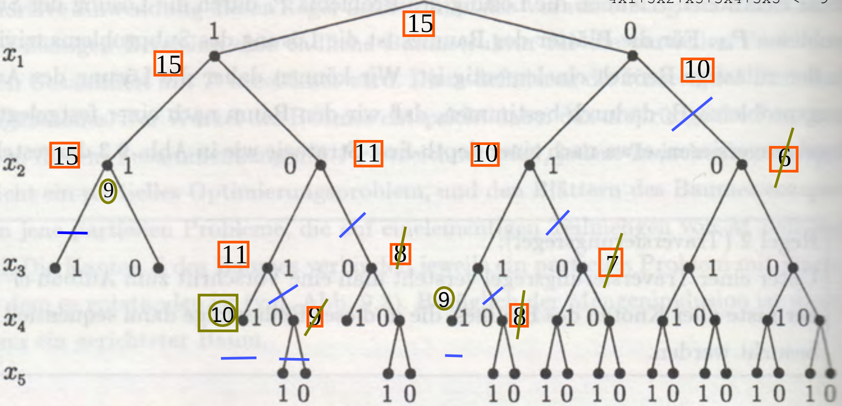
Observation: if $u_{\text{current}} < \ell_i$, no optimal solution $\mathbf{x}^*$ can lie in M_i:

$$f(\mathbf{x}^*) = z^*(P_0) \leq u_{\text{current}} < \ell_i \leq z^*(P_i) = \min_{\mathbf{x} \in M_i} f(\mathbf{x}) .$$

Hence discard P_i and **all sucessor nodes in problem tree**.

5x1+4x2+3x3+2x4+x5 -> max !

4x1+5x2+x3+3x4+3x5 <= 9



Branch-and-bound: updating and tightening

Further, if some local upper bounds $u_i \geq z^*(P_i)$ are generated, update

$$u'_{\text{current}} = \min\{u_i, u_{\text{current}}\}$$

if possible.

Any (good) feasible value may serve as local upper bound:

$$u_i = f(\mathbf{x}_i) \quad \text{with any } \mathbf{x}_i \in M_i\,,$$

the tighter the "better" P_i is (nearly) solved.

Crucial: tight lower bounds ℓ_i to discard many P_i early.

How to get these ?

LP relaxation bounds

Had them in knapsack problem already:

relax $x_i \in \{0, 1\}$ to $0 \leq x_i \leq 1$.

More generally: write latter as $\mathsf{E}\mathbf{x} \leq \mathbf{e}$ and consider

$$M = \{\mathbf{x} \in \mathbb{R}^n : \mathsf{A}\mathbf{x} \leq \mathbf{b}\,,\ \mathsf{E}\mathbf{x} \leq \mathbf{e}\,,\ x_i \text{ integer for all } i\}$$

by dropping integrality condition, to

$$F = \{\mathbf{x} \in \mathbb{R}^n : \mathsf{A}\mathbf{x} \leq \mathbf{b}\,,\ \mathsf{E}\mathbf{x} \leq \mathbf{e}\}\ .$$

In case of ILP $z^*(P) = \min\left\{\mathbf{c}^\top\mathbf{x} : \mathbf{x} \in M\right\}$ yields

an LP $z^*_{\mathsf{LP}} = \min\left\{\mathbf{c}^\top\mathbf{x} : \mathbf{x} \in F\right\}$ with $M \subseteq F$, so

$$z^*_{\mathsf{LP}} \leq z^*(P)$$

gives a (relatively cheap) lower bound for P.

Same for subproblems $P_i \hookrightarrow \ell_i$.

Lagrangian dual bounds are tighter than LP bounds

For ILP, define the ground set

$$X = \{\mathbf{x} \in \mathbb{R}^n : \mathsf{E}\mathbf{x} \leq \mathbf{e}\,,\, x_i \text{ integer for all } i\}$$

so that $M = \{\mathbf{x} \in X : \mathsf{A}\mathbf{x} \leq \mathbf{b}\}$.

Catch: $\mathsf{E}\mathbf{x} \leq \mathbf{e}$ easy to handle (e.g., variable bounds).

Lagrange function

$$L(\mathbf{x};\mathbf{y}) = \mathbf{c}^\top \mathbf{x} + \mathbf{y}^\top(\mathsf{A}\mathbf{x} - \mathbf{b}) = (\mathbf{c} + \mathsf{A}^\top \mathbf{y})^\top \mathbf{x} - \mathbf{y}^\top \mathbf{b}$$

is easier minimized over $\mathbf{x} \in X$ for any $\mathbf{y} \in \mathbb{R}^m_+$, giving $\Theta(\mathbf{y})$.

$$d^* = \sup_{\mathbf{y} \in \mathbb{R}^m_+} \Theta(\mathbf{y}) \leq z^*(P) \quad \text{by weak duality.}$$

In ILPs, Lagrangian dual bound always dominates LP bound:

$$z^*_{\mathsf{LP}} \leq d^* \leq z^*(P)\,, \text{ strict inequalities possible.}$$