Decision Making
Part 8: Sequential decision making

Mark Fackrell
E-mail: fackrell@unimelb.edu.au
Office: Richard Berry room 148

Topics in this part

- <u>Decision trees</u>: decision trees, incorporating risk aversion and risk seeking into decision tree analysis

- <u>Shortest path problem for acyclic directed graphs</u>: acyclic directed graphs, proper labelling, shortest path problem, forward dynamic programming (DP) algorithm

- <u>Multi-objective shortest path problem</u>: Pareto minimal elements, Pareto minimal paths, DP algorithm for finding Pareto minimal paths in acyclic directed graphs

- <u>Deterministic dynamic programming</u>: Backward DP algorithm, characteristics of dynamic programming, resource allocation problem, knapsack problem

Reference:
W. L. Winston, Operations Research, Sections 13.4 and 18.4

Introduction

In many decision making problems, decisions have to be made sequentially, and the rewards are received only after an entire sequence of decisions have been taken. A choice made at one stage may have an impact on what choices may be made at later stages and eventually on the final rewards. The well-known shortest path problem is a typical example of this type of questions. In this part we will introduce a useful methodology, namely deterministic dynamic programming, for solving the shortest path problem, single- or multi-objective, and some other sequential decision problems.

Some sequential decision problems involve randomness at some stages or all stages. We will learn a useful technique, namely decision tree analysis, to solve some of such problems. We leave discussion on more complicated sequential decision problems involving randomness to the next part.

Decision trees

Decision trees

Many decision making problems involve multi-stages, which decompose a large problem into several smaller problems.

In this situation we can often represent the decision making problem by a decision tree, which helps determine optimal decisions.

We illustrate this method by the following example.

Example 1. (decision tree analysis)

(Winston, Section 13.4) A soft drink company currently has assets of \$150,000 and wants to decide whether to market a new chocolate-flavored product, Chocola. The company has three alternatives:

(A1) Test market Chocola locally, then utilise the results of the market study to determine whether or not to market Chocola nationally;

(A2) Immediately (without testing the market) market Chocola nationally;

(A3) Immediately (without the testing market) decide not to market Chocola nationally.

Example (cont.) In the absence of a market study, the company believes that Chocola has a 55% chance of being a national success and a 45% chance of being a national failure. If Chocola is a national success, the company's asset will increase by \$300,000, and if Chocola is a national failure, the asset will decrease by \$100,000.

If a market study is performed (at a cost of \$30,000), there is a 60% chance that the study will yield favorable results (local success) and a 40% chance that the study will yield unfavorable results (local failure). If a local success is observed, there is an 85% chance that the product will be a national success. If a local failure is observed, there is only a 10% chance that the product will be a national success.

If the company is risk-neutral (wants to maximise its expected final asset position), what strategy should the company follow?

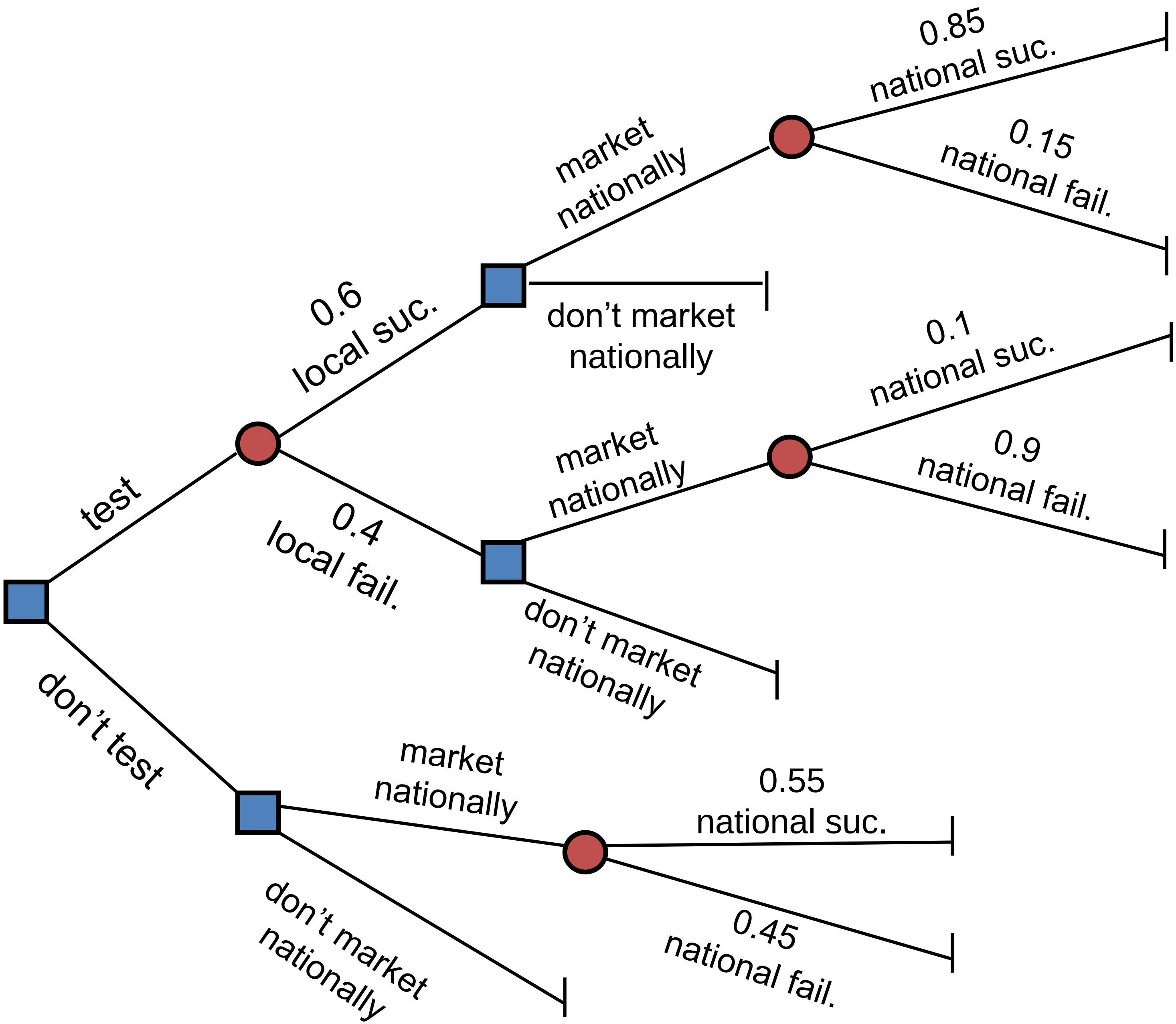
Example (cont.) Let us construct a tree to represent the problem. We need two kinds of vertices: decision vertices and event vertices (also called chance vertices).

A decision vertex represents a point in time when the company has to make a decision. Each edge incident with a decision vertex represents a possible decision.

An event vertex is drawn when one of several random events will occur. Each edge incident with an event vertex stands for a possible outcome, and the number on it represents the probability that the event will occur.

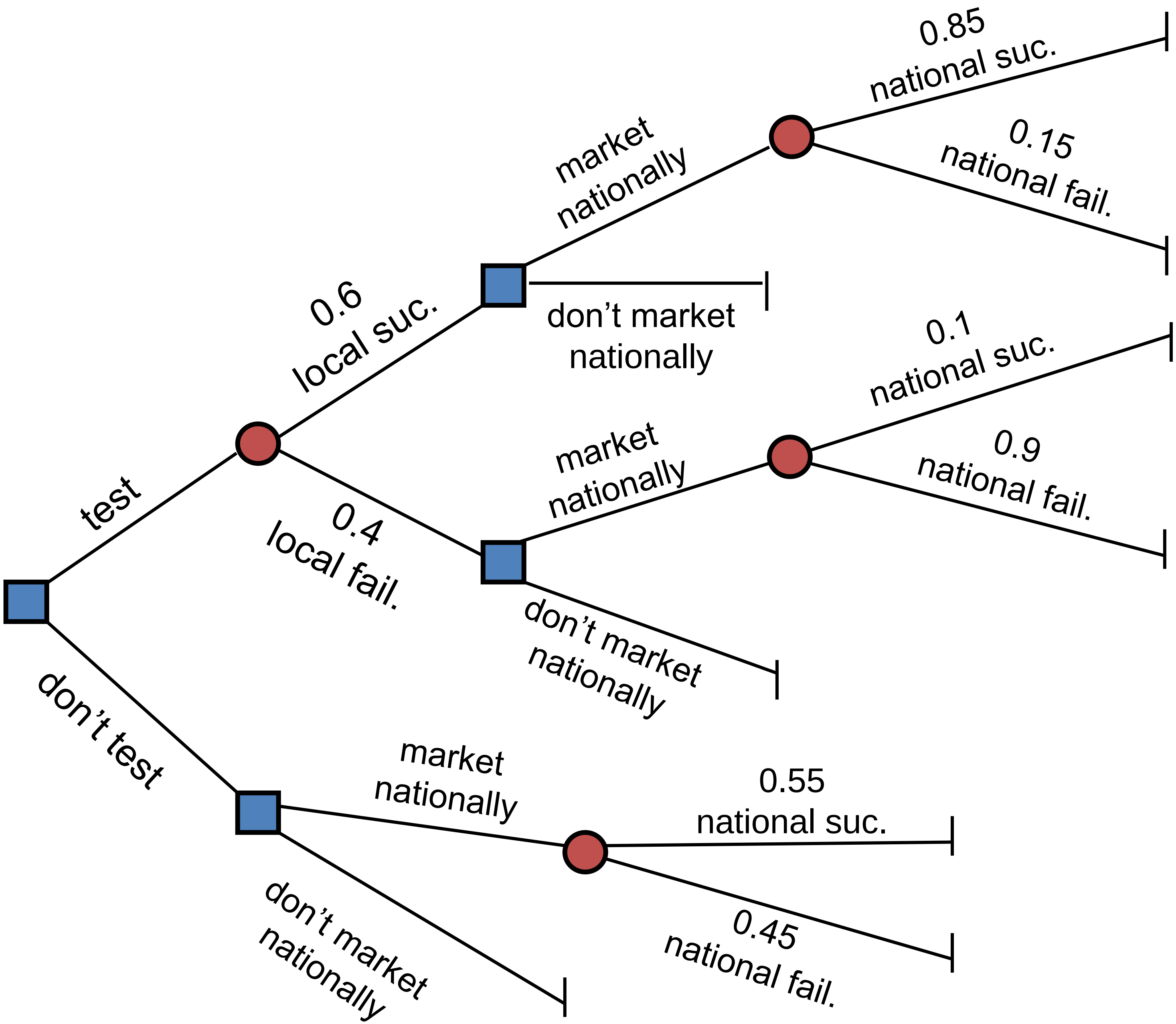
The problem now is to find an “optimal decision path” which represents an optimal strategy.

Example (cont.) Draw the decision tree that corresponds to the decision problem of the soft drink company. Indicate decision vertices by squares and event vertices by circles. Indicate on each leaving edge of a decision vertex the corresponding decision and on each leaving edge of a event vertex the corresponding event and probability.


test
don't test
0.6
local suc.
0.4
local fail.
market nationally
don't market nationally
0.85
national suc.
0.15
national fail.
market nationally
don't market nationally
0.1
national suc.
0.9
national fail.
market nationally
don't market nationally
0.55
national suc.
0.45
national fail.

Example (cont.) Now let us work backward from right to left. At each event vertex we calculate the expected final asset. At each decision vertex we mark the decision that maximises the expected final asset, and then enter this maximum expected final asset.

Continue working backward in this fashion until we reach the root (beginning) of the tree. The optimal sequence of decisions can be obtained by following the marks.


test
don't test
0.6
local suc.
0.4
local fail.
market
nationally
don't market
nationally
0.85
national suc.
0.15
national fail.
market
nationally
don't market
nationally
0.1
national suc.
0.9
national fail.
market
nationally
don't market
nationally
0.55
national suc.
0.45
national fail.

Example (cont.)

Optimal strategy:

Do not test, and market nationally. This will yield an expected final asset position of $270,000.

If the test has been performed already, then the best strategy is to market nationally after local success and do not market nationally after local failure. This will yield an expected asset position of $264,000.

Incorporating risk aversion into decision tree analysis

Let us continue our discussion on the example above.

There is a 0.45 chance that the company will end up with a small final asset position of \$50,000. On the other hand, the strategy of testing and acting optimally on the results of the test will yield only a $0.6 \times 0.15 = 0.09$ chance to have an asset position below \$100,000.

Thus, if the company is a risk-averse decision maker, then the “optimal strategy” (do not test, and market nationally) may not reflect the company’s preference.

We may incorporate risk aversion into the decision tree analysis by maximising the expected utility.

Assume that the decision maker's utility function is $u(x)$.

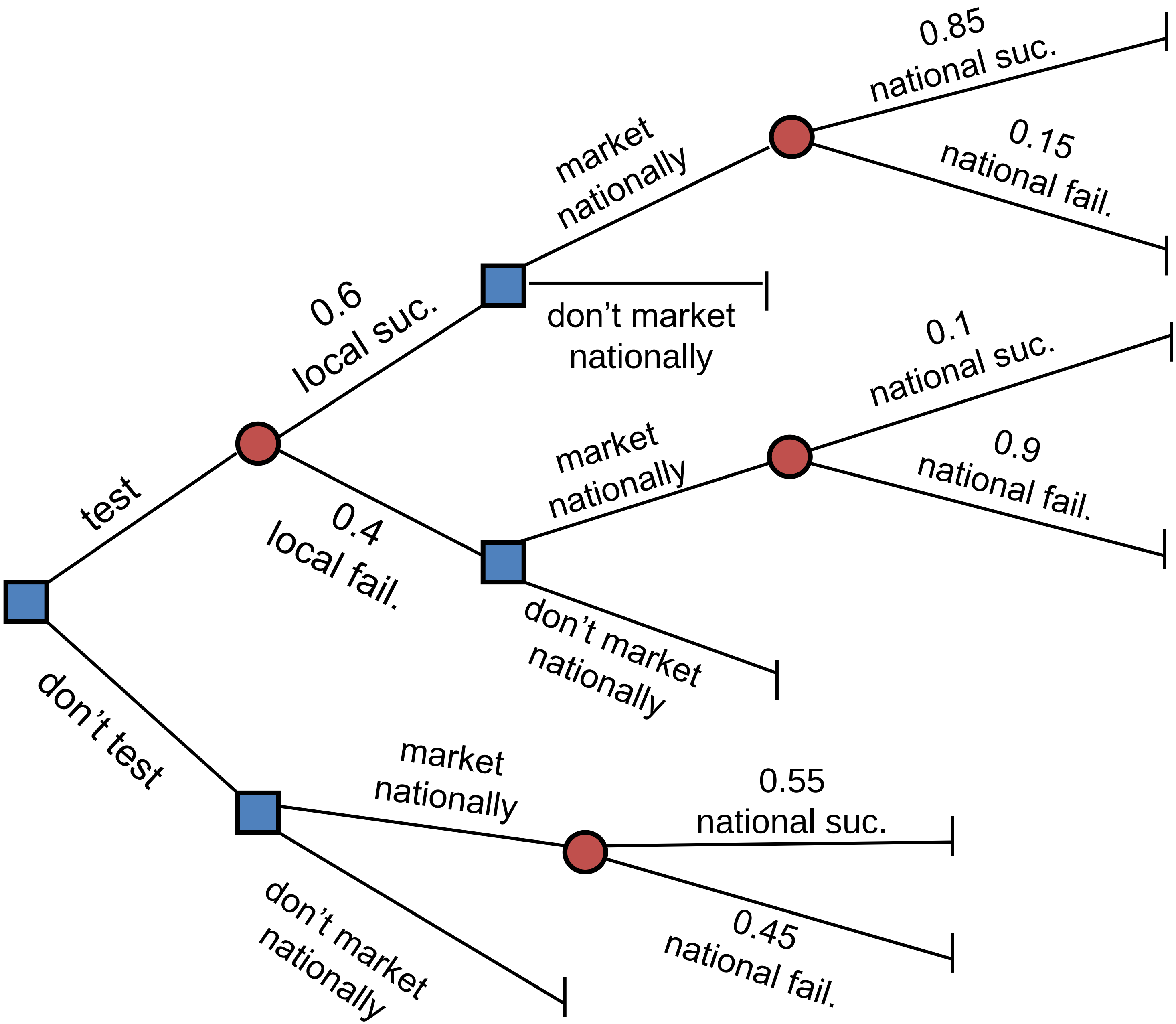
The decision tree analysis can be modified as follows.

Replace each final asset position x by its utility $u(x)$. At each event vertex compute the expected utility of the final asset position. At each decision vertex choose the edge with the maximum expected utility.

Example (cont.) In the following computation we assume that the decision maker's utility function gives

$$u(\text{20,000}) = 0,\ \ u(\text{50,000}) = 0.19,\ \ u(\text{120,000}) = 0.4,$$

$$u(\text{150,000}) = 0.48,\ \ u(\text{420,000}) = 0.99,\ \ u(\text{450,000}) = 1.$$



test
don't test
0.6
local suc.
0.4
local fail.
market
nationally
don't market
nationally
0.85
national suc.
0.15
national fail.
market
nationally
don't market
nationally
0.1
national suc.
0.9
national fail.
market
nationally
don't market
nationally
0.55
national suc.
0.45
national fail.

Example (cont.)

Optimal strategy that incorporates risk aversion:

Test market first. If a local success is achieved, market nationally; if a local failure is observed, then do not market nationally. With this strategy there is only a $0.6 \times 0.15 = 0.09$ chance of having a final position of less than $100,000.

Example (cont.)

Further discussion:

Assume that the decision maker's utility function gives $u(225,000) = 0.6649$.

Since the decision maker views the current situation as having an expected utility 0.6649, this means that the company considers the current situation as equivalent to a certain asset position of \$225,000.

Thus, if somebody offers more than \$225,000 - \$150,000 $=$ \$75,000 to buy the rights to Chocola, the company should take the offer.

Summary: analysis of a decision tree

If a decision maker is risk-neutral:

- Draw the decision tree that corresponds to the problem.
- Determine the payoff at each leaf.
- Work backwards from right to left:
  - in each event vertex determine the expected payoff;
  - in each decision vertex indicate the best decision, i.e. the decision that maximises (expected) payoff and enter this maximum payoff at the decision vertex.

  Continue this process until you reach the root.
- The optimal sequence of decisions can be read off by starting at the root and following the indicated best decisions.

If a decision maker is risk-seeking or risk-averse, we follow the same procedure, however, we replace the (expected) payoff by the corresponding (expected) utility.

Shortest path problem for acyclic digraphs

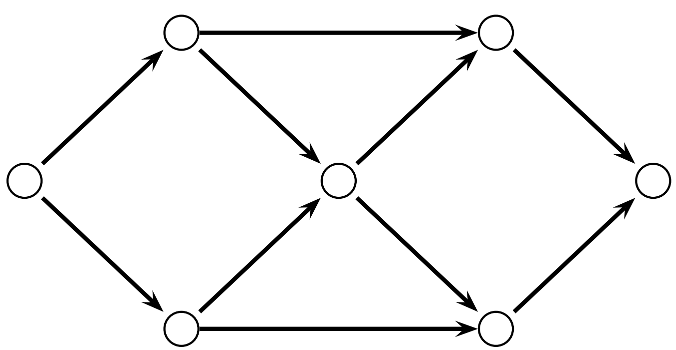
Acyclic directed graphs

Definition 1. Let $G = (V, E)$ be a directed graph. A (directed) path of G is a sequence of distinct vertices $v_1, v_2, \ldots, v_k$ such that $v_1v_2, v_2v_3, \ldots, v_{k-1}v_k$ are edges of G.

A (directed) cycle of G is a sequence of vertices $v_1, v_2, \ldots, v_k, v_1$ such that $v_1, v_2, \ldots, v_k$ is a path and v_kv_1 is an edge of G.

A directed graph which has no cycles is called acyclic.

Example 2. (acyclic graph)



Shortest path problem

Definition 2. Let $G = (V, E)$ be a directed graph such that each edge $vw \in E$ is associated with a non-negative weight $c_{vw} \geq 0$. Call G a weighted directed graph.

Let $P : v_1, v_2, \ldots, v_k$ be a path of G. The weight (or length) of P is defined as

$$w(P) := c_{v_1 v_2} + c_{v_2 v_3} + \cdots + c_{v_{k-1} v_k}.$$

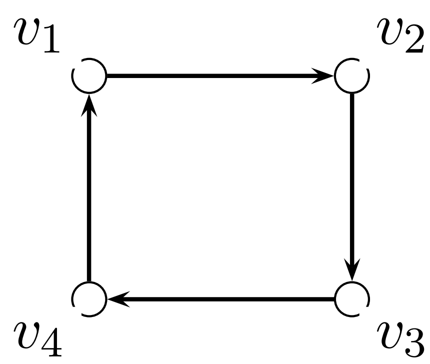
Problem. Given a weighted directed graph $G = (V, E)$ and distinct vertices s and t of G, find a path in G from s to t (i.e. with initial vertex s and terminal vertex t) with minimum length. Such a path is called a shortest path from s to t.

Proper labelling

Definition 3. A proper labelling of a directed graph $G = (V, E)$ with n vertices is an assignment of **one label** $\ell(x)$ from $\{1, 2, \ldots, n\}$ to each vertex x of G such that if there is an edge from v to w, then $\ell(v) < \ell(w)$.

Not every directed graph has a proper labelling.

Example 3. (digraph without proper labelling)



This graph does not have a proper labelling $\ell(x)$, since if $\ell(x)$ is a proper labelling, then

$$\ell(v_1) < \ell(v_2) < \ell(v_3) < \ell(v_4) < \ell(v_1),$$

which yields a contradiction.

Let G be a directed graph and v a vertex of G. Define

$$p(v) = \#\ \text{vertices } u \text{ such that there exists a path from } u \text{ to } v.$$

Lemma 1. A directed graph $G = (V, E)$ has a proper labelling iff it is acyclic. Moreover, for acyclic G of n vertices, if we order the vertices

$$v_1, v_2, \ldots, v_n$$

in such a way that

$$p(v_1) \leq p(v_2) \leq \cdots \leq p(v_n),$$

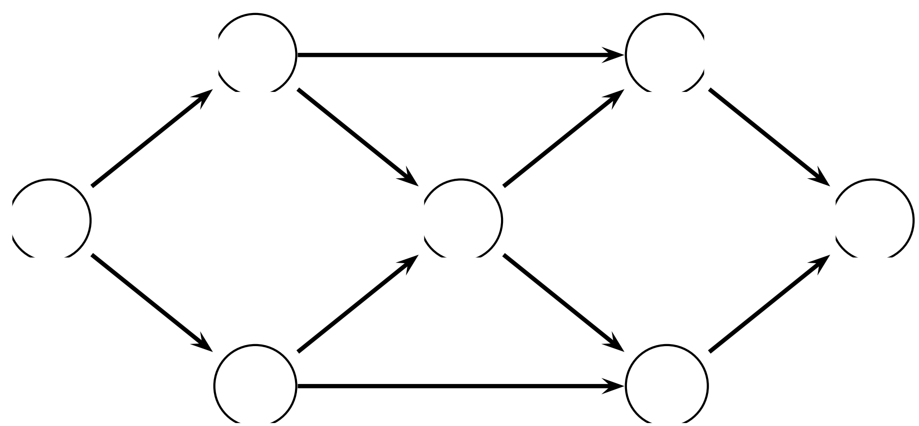
then the assignment

$$\ell(v_1) = 1, \ell(v_2) = 2, \ldots, \ell(v_n) = n$$

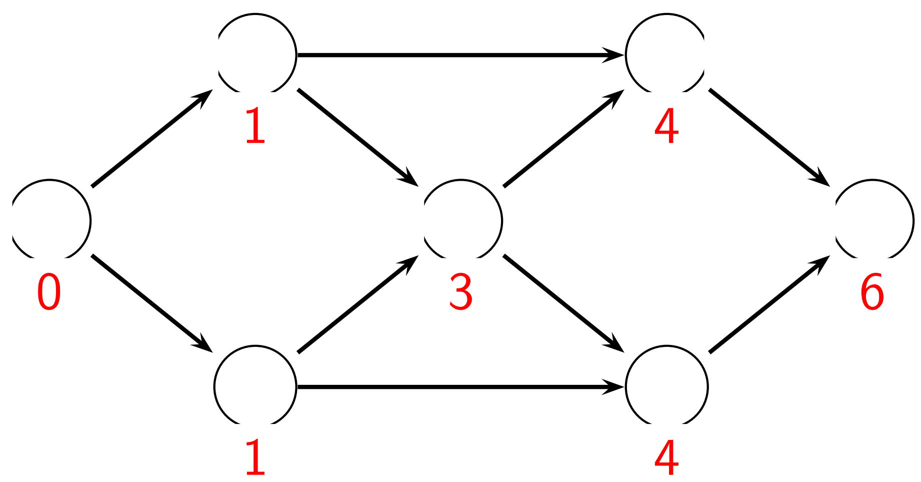
is a proper labelling of G.

Proof.

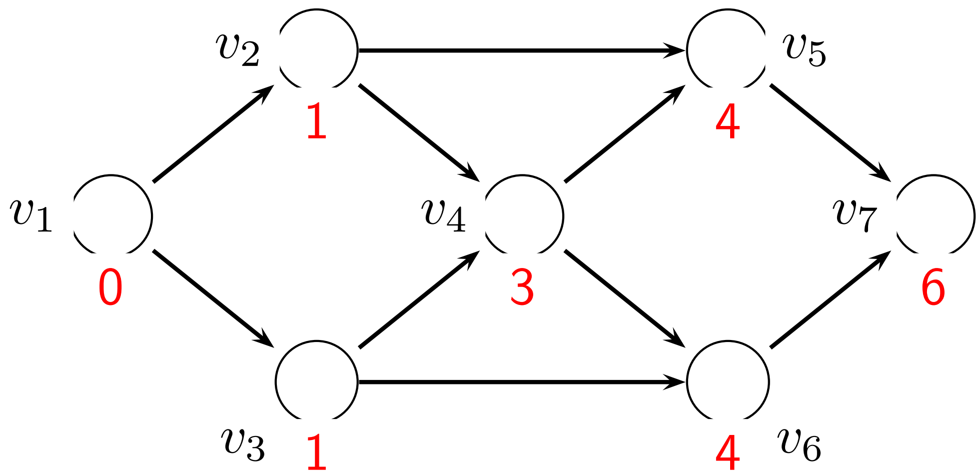
Example 4. (proper labelling)
Consider the acyclic directed graph depicted below. To determine a proper labelling, we first determine for each vertex v the number of vertices u such that there exists a path from u to v. This number is indicated in red.



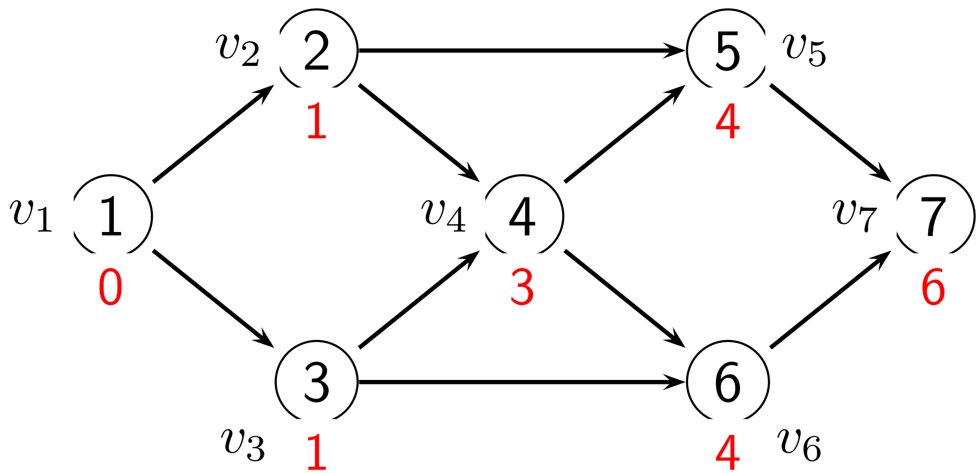
Example 4. (proper labelling)
Consider the acyclic directed graph depicted below. To determine a proper labelling, we first determine for each vertex v the number of vertices u such that there exists a path from u to v. This number is indicated in red.



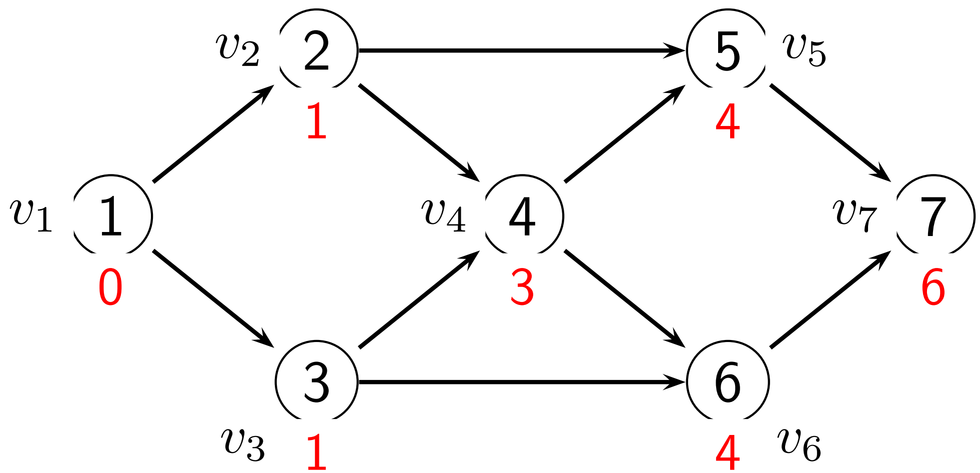
Example 4. (proper labelling)
Consider the acyclic directed graph depicted below. To determine a proper labelling, we first determine for each vertex v the number of vertices u such that there exists a path from u to v. This number is indicated in red. Second, we order the vertices $v_1, \ldots, v_7$ in such a way that $p(v_i) \leq p(v_j)$ if $i \leq j$.



Example 4. (proper labelling)
Consider the acyclic directed graph depicted below. To determine a proper labelling, we first determine for each vertex v the number of vertices u such that there exists a path from u to v. This number is indicated in red. Second, we order the vertices $v_1, \ldots, v_7$ in such a way that $p(v_i) \leq p(v_j)$ if $i \leq j$. Finally, we assign the label i to vertex v_i.



Example 4. (proper labelling)
Consider the acyclic directed graph depicted below. To determine a proper labelling, we first determine for each vertex v the number of vertices u such that there exists a path from u to v. This number is indicated in red. Second, we order the vertices $v_1, \ldots, v_7$ in such a way that $p(v_i) \leq p(v_j)$ if $i \leq j$. Finally, we assign the label i to vertex v_i.



Note that a proper labelling need not be unique. If we interchange the labels of v_2 and v_3 in the example above, then we have another proper labelling.

Shortest path problem for acyclic directed graphs

Problem. Let $G = (V, E)$ be a weighted acyclic directed graph with n vertices. Assume the vertices of G have been labelled properly. Find a shortest path from the vertex with label 1 to the vertex with label n.

For each $i = 1, 2, \ldots, n$, define

$$f(i) = \text{length of a shortest path from } 1 \text{ to } i,$$

$$P(i) = \{j : \text{there is an edge of } G \text{ from } j \text{ to } i\}$$

$$S(i) = \{k : \text{there is an edge of } G \text{ from } i \text{ to } k\}.$$

$P(i)$ is the set of immediate predecessors of i, and $S(i)$ is the set of immediate successors of i.

Since the labelling is proper, if $j > i$, then $j \notin P(i)$ and so a shortest path from 1 to i cannot go from j to i.

If $j < i$ and if $j \in P(i)$, then a shortest path from 1 to i may go from j to i. If indeed it uses edge ji, then the path from 1 to j is also a shortest path, for else we can find a shorter path from 1 to i via j, a contradiction. Thus

$$f(i) = \min\{f(j) + c_{ji} : j \in P(i)\}, \; i = 1, 2, \ldots, n.$$

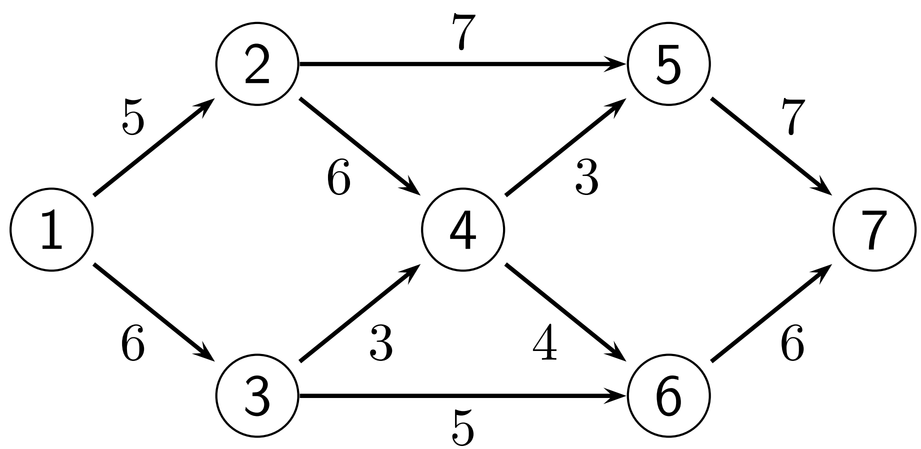
By applying this recursive equation repeatedly, we can compute $f(1), f(2), \ldots, f(n)$ sequentially in this order. This is possible since $f(1) = 0$ is known! This is a forward dynamic programming algorithm.

We keep a record of the edges achieving the minimum above.

In this way **we can find a shortest path from vertex 1 to every other vertex** in an acyclic directed graph.

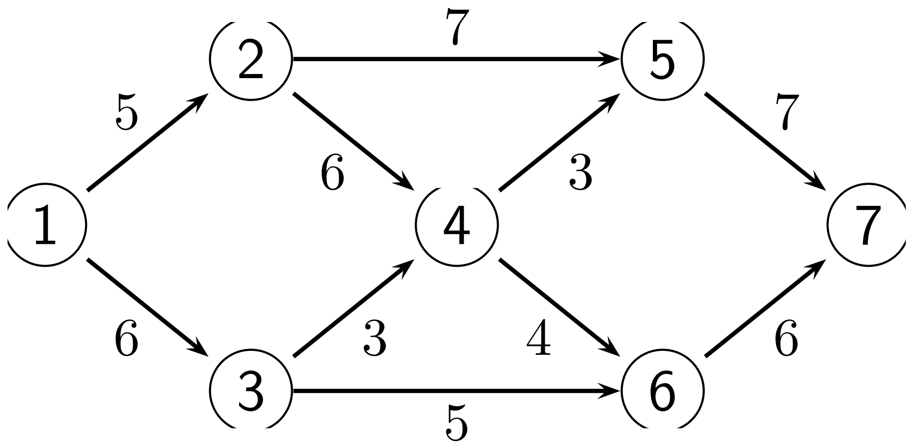
Example 5. (shortest path problem for acyclic directed graphs, single objective, forward DP)

Determine a shortest path from vertex 1 to vertex 7 in the properly labelled digraph below.

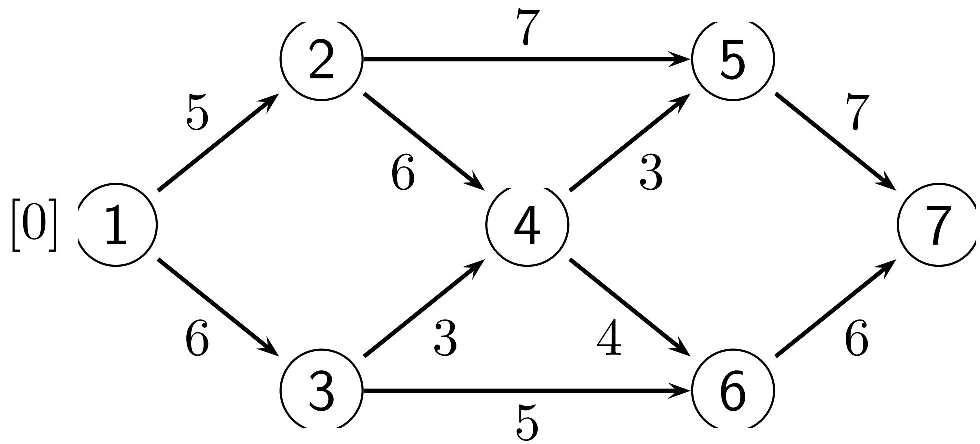


To find a shortest path, we sequentially calculate $f(1), f(2), \ldots, f(7)$. Furthermore, we clearly indicate the edges that are used in the shortest paths from vertex 1 to vertex i.

Example (cont.) The values of $f(i)$ are depicted between straight brackets.

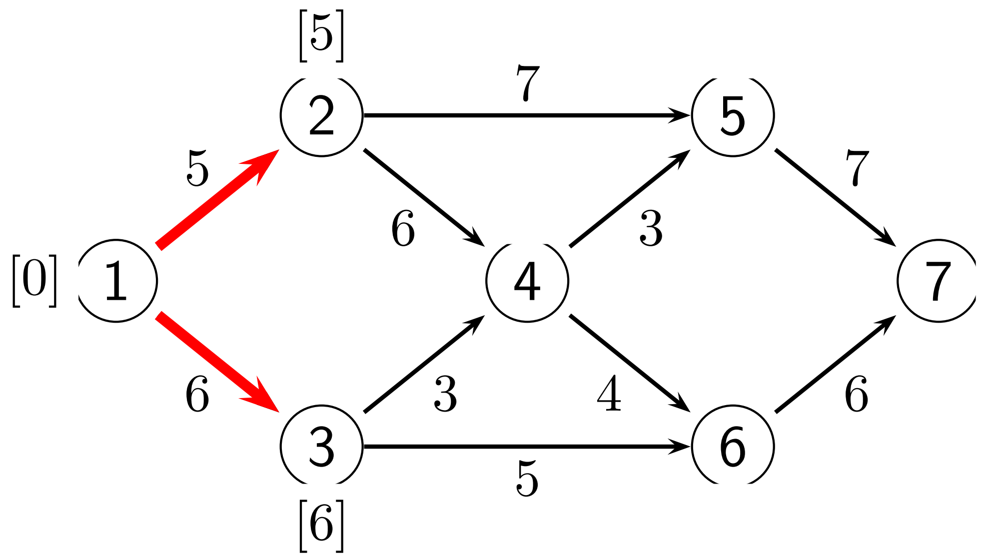


Example (cont.) The values of $f(i)$ are depicted between straight brackets.



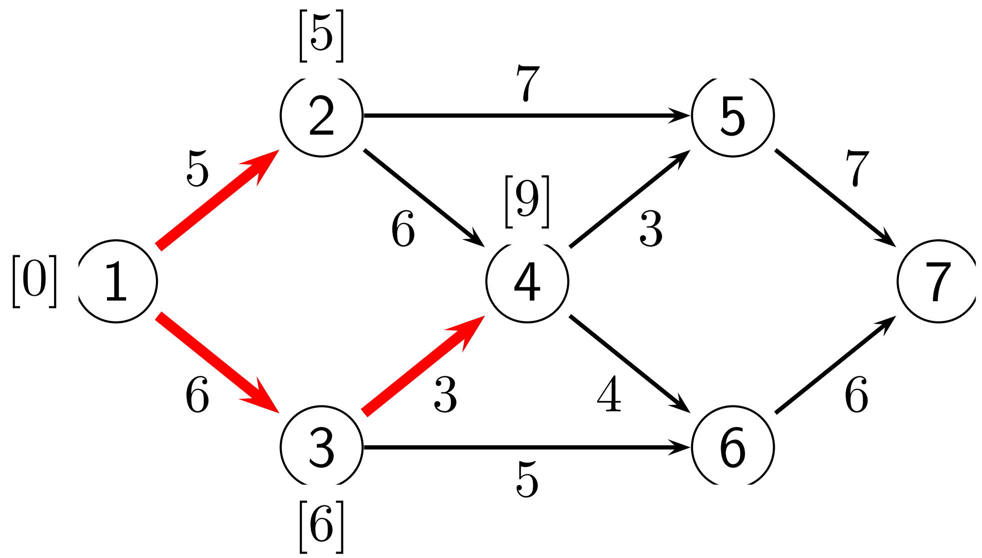
$f(1) = 0,$

Example (cont.) The values of $f(i)$ are depicted between straight brackets.



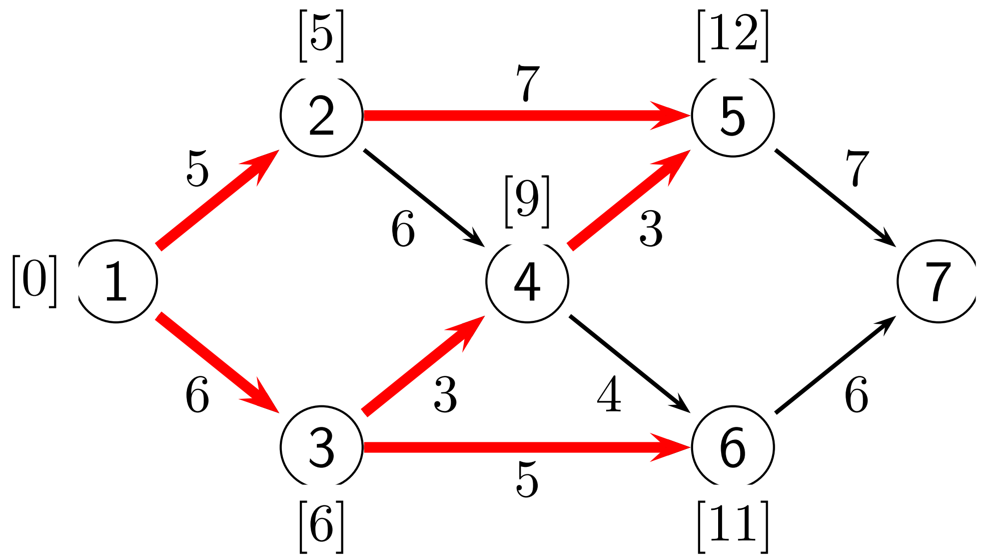
$f(1) = 0$,
$f(2) = 5$,
$f(3) = 6$,

Example (cont.) The values of $f(i)$ are depicted between straight brackets.



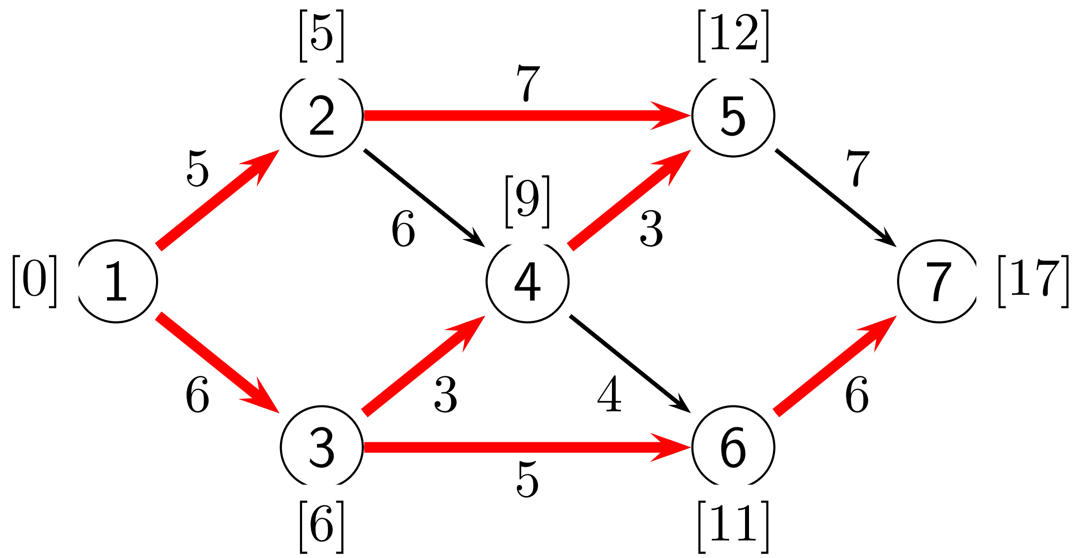
$f(1) = 0$,
$f(2) = 5$,
$f(3) = 6$,
$f(4) = \min\{f(2) + c_{24}, f(3) + c_{34}\} = \min\{11, 9\} = 9$,

Example (cont.) The values of $f(i)$ are depicted between straight brackets.



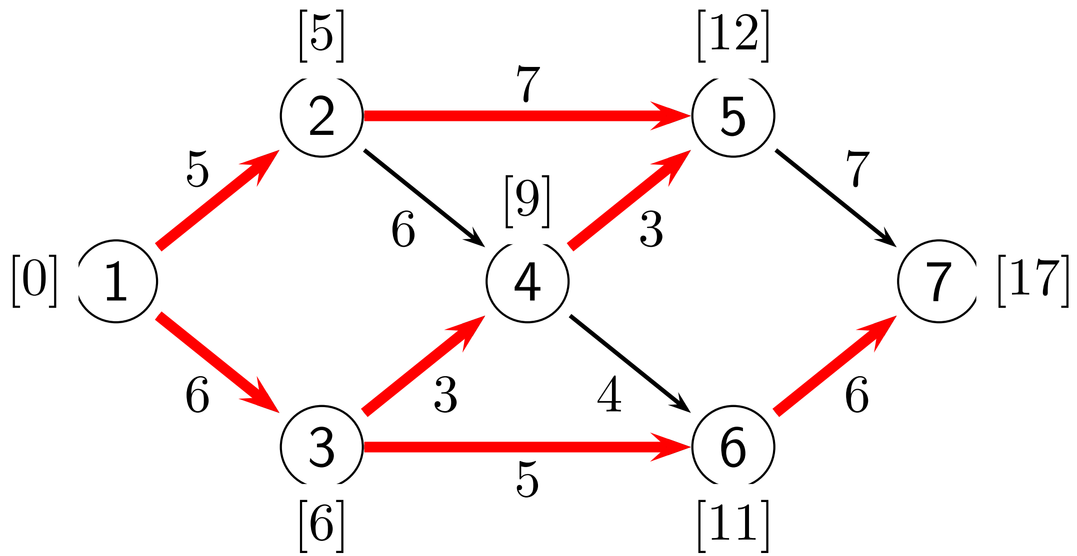
$$
\begin{aligned}
&f(1) = 0,\\
&f(2) = 5,\\
&f(3) = 6,\\
&f(4) = \min\{f(2) + c_{24}, f(3) + c_{34}\} = \min\{11, 9\} = 9,\\
&f(5) = \min\{f(2) + c_{25}, f(4) + c_{45}\} = \min\{12, 12\} = 12,\\
&f(6) = \min\{f(3) + c_{36}, f(4) + c_{46}\} = \min\{11, 13\} = 11,
\end{aligned}
$$

Example (cont.) The values of $f(i)$ are depicted between straight brackets.

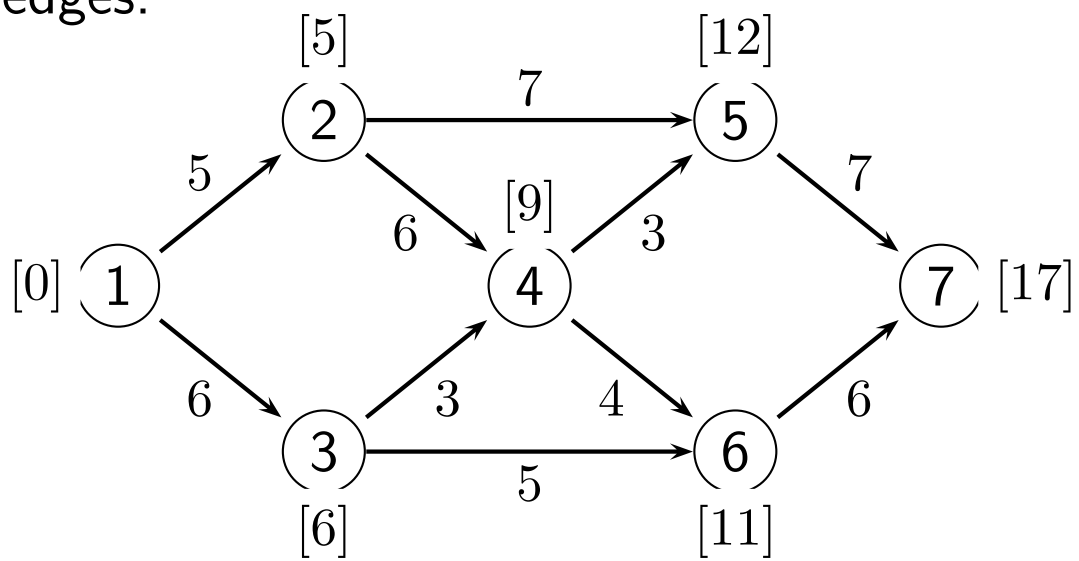


$f(1) = 0,$
$f(2) = 5,$
$f(3) = 6,$
$f(4) = \min\{f(2) + c_{24}, f(3) + c_{34}\} = \min\{11, 9\} = 9,$
$f(5) = \min\{f(2) + c_{25}, f(4) + c_{45}\} = \min\{12, 12\} = 12,$
$f(6) = \min\{f(3) + c_{36}, f(4) + c_{46}\} = \min\{11, 13\} = 11,$
$f(7) = \min\{f(5) + c_{57}, f(6) + c_{67}\} = \min\{19, 17\} = 17.$

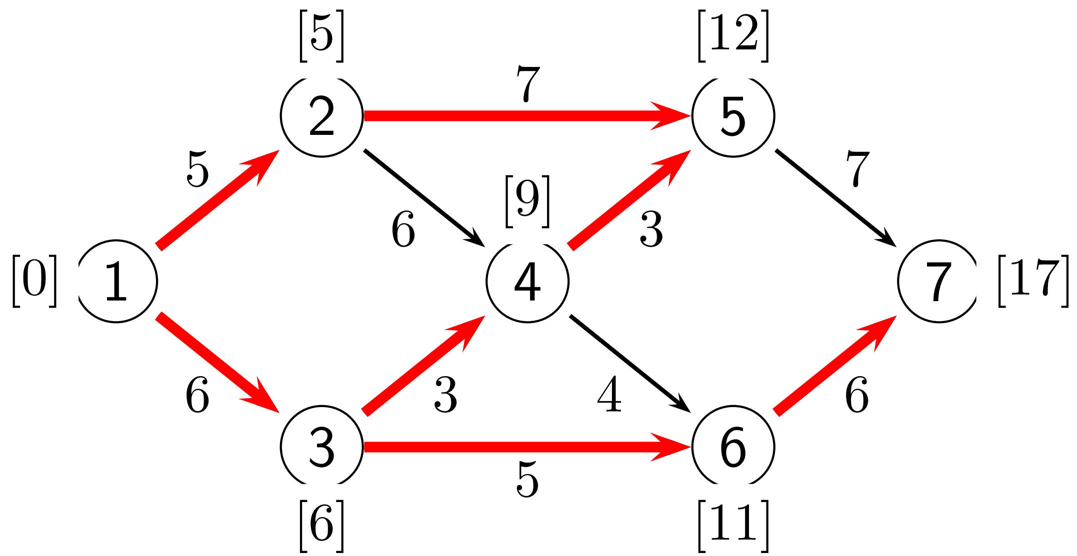
Example (cont.)



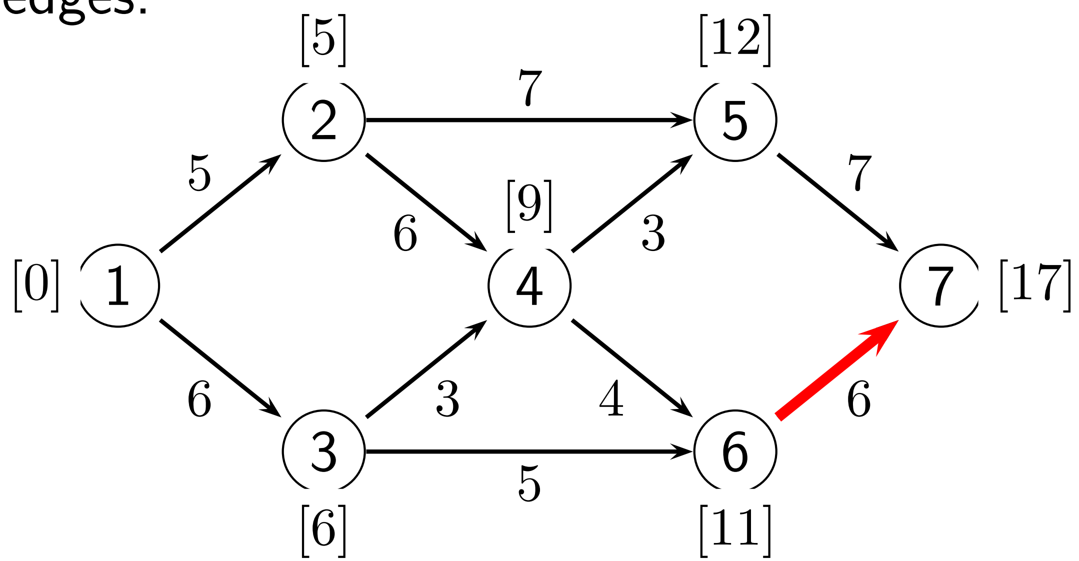
To recover the shortest path from vertex 1 to vertex 7, work back from vertex 7 and select the red edges.



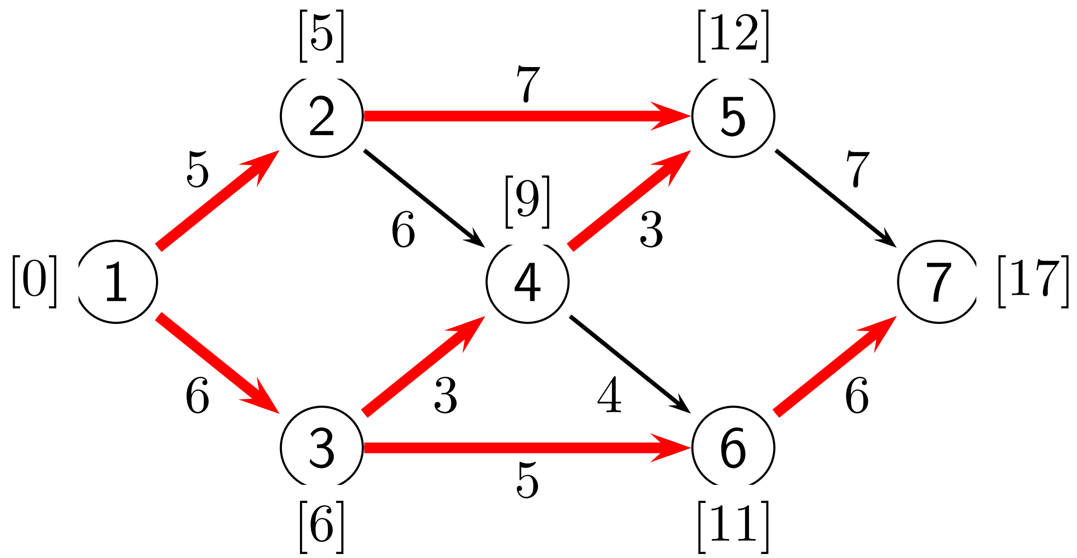
Example (cont.)



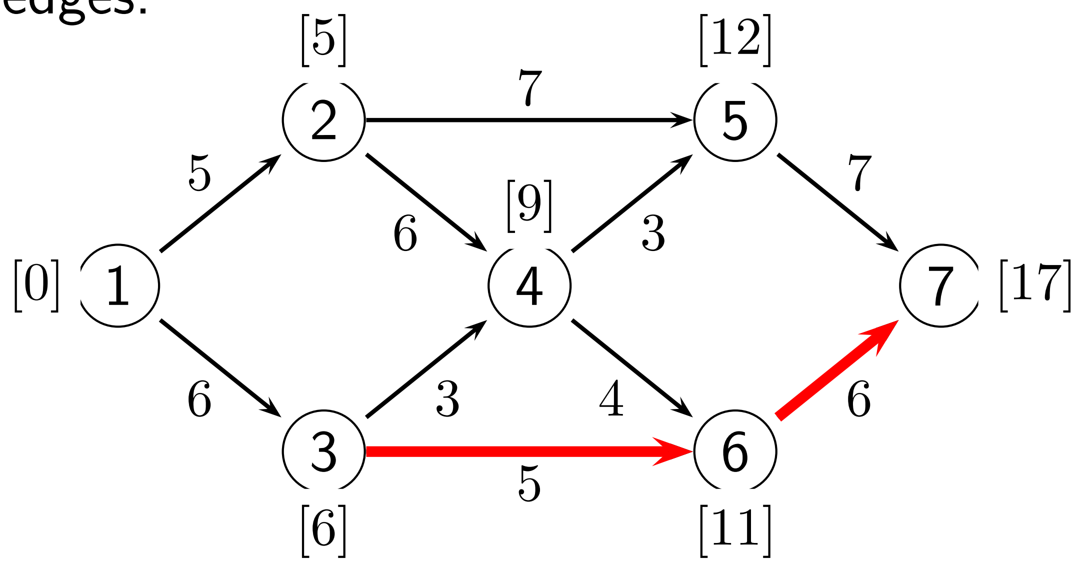
To recover the shortest path from vertex 1 to vertex 7, work back from vertex 7 and select the red edges.



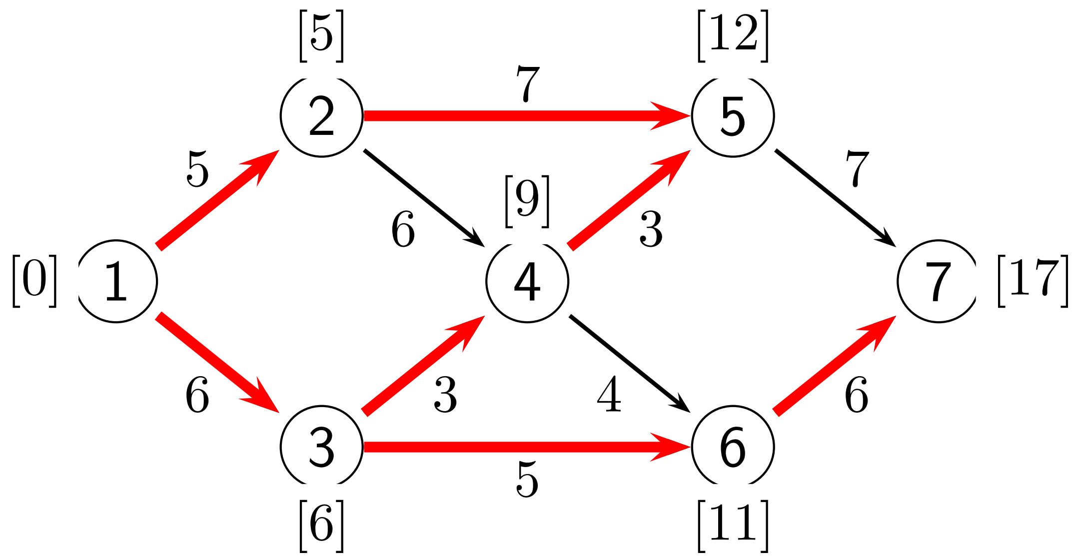
Example (cont.)



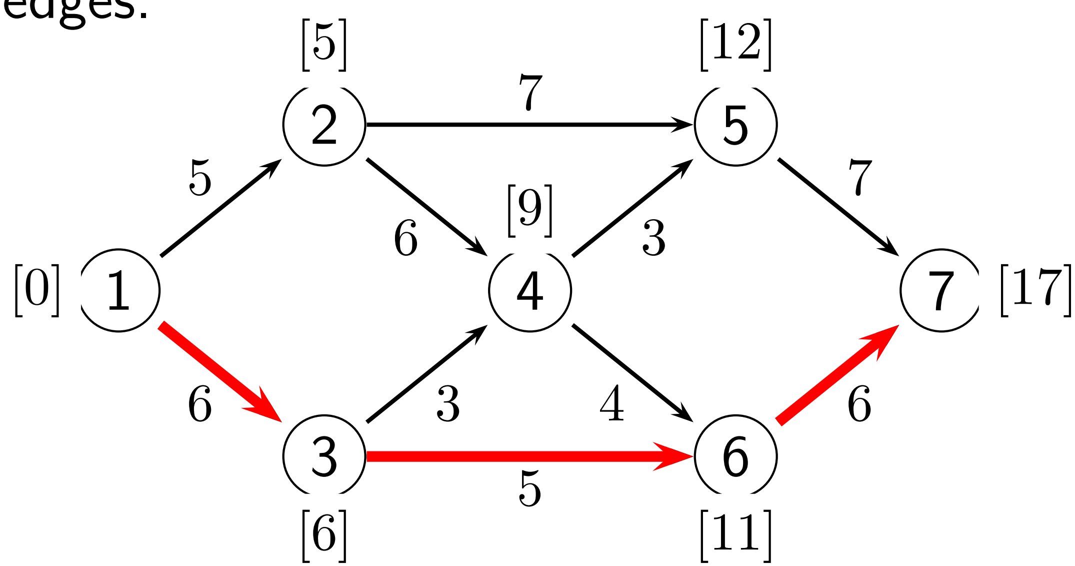
To recover the shortest path from vertex 1 to vertex 7, work back from vertex 7 and select the red edges.



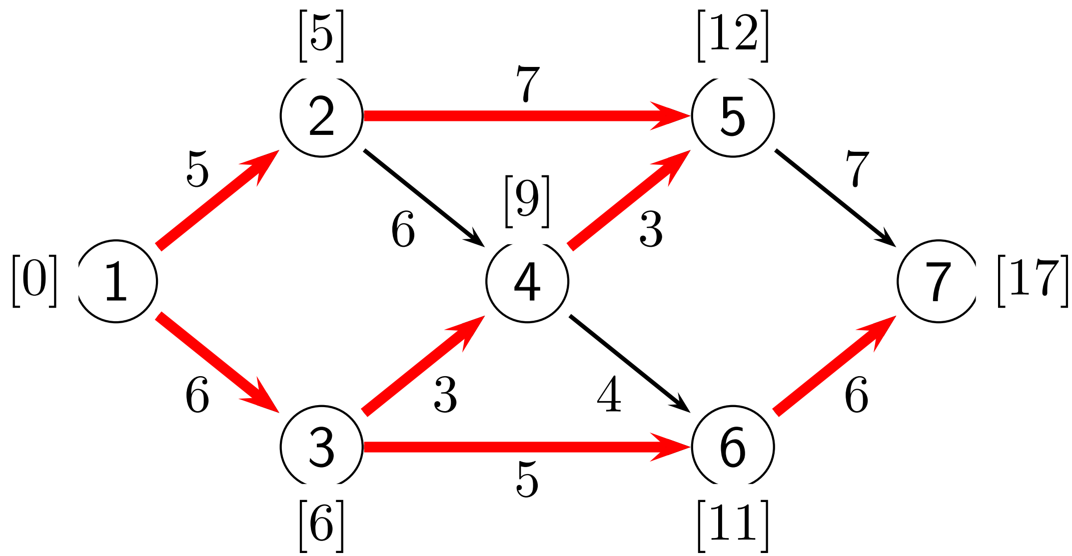
Example (cont.)



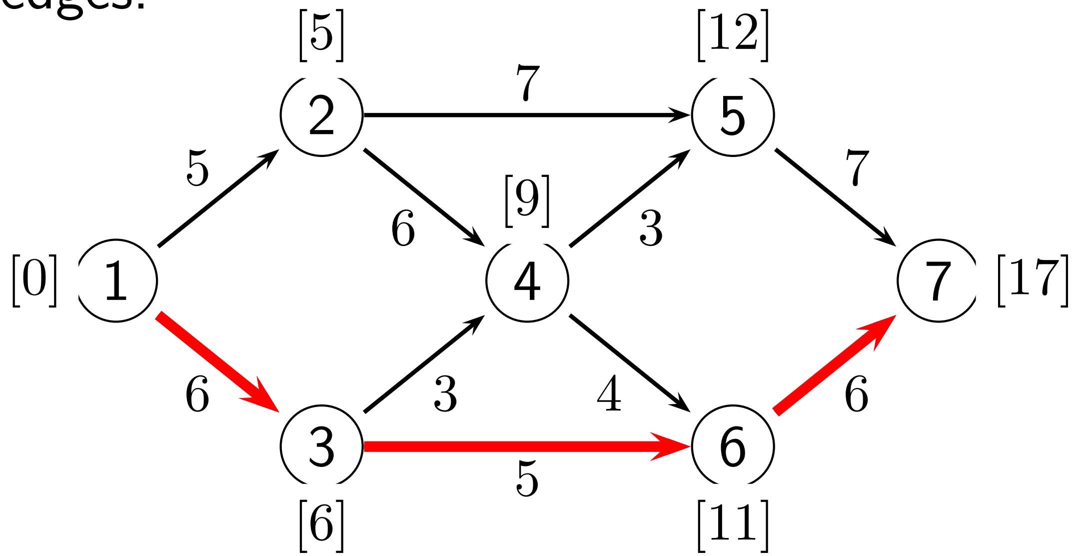
To recover the shortest path from vertex 1 to vertex 7, work back from vertex 7 and select the red edges.



Example (cont.)



To recover the shortest path from vertex 1 to vertex 7, work back from vertex 7 and select the red edges.



Optimal path: 1-3-6-7, optimal length: 17.

Multi-objective shortest path problem

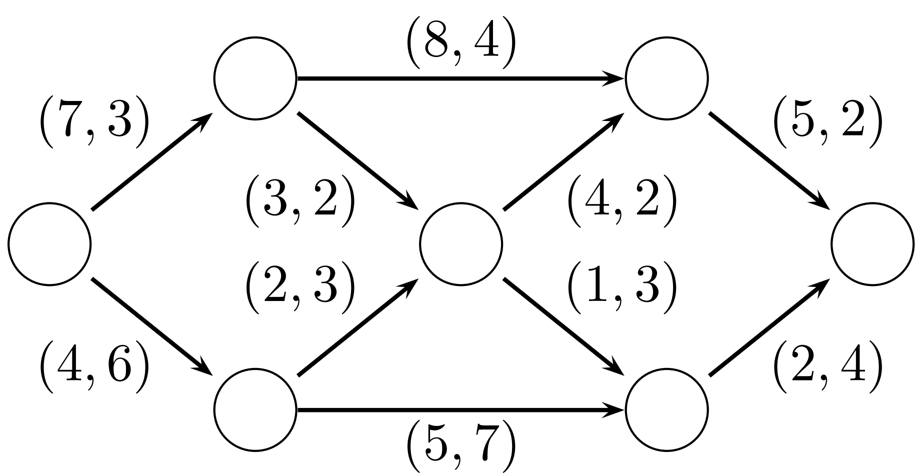
Two or more objectives

Consider a directed graph whose edges are assigned a pair of non-negative numbers as “weights”. How can we find a “shortest path” from a vertex to another vertex?

The length of a path is defined as the sum of the weights of its edges under coordinate-wise addition.

First of all, what is the meaning of a “shortest path”?

Example. What does a shortest path from vertex 1 to vertex 7 mean?



Pareto minimal elements

Recall the following definition.

Definition 4. Let A be a subset of $\mathbb{R}^n$. Call $(a_1, \dots, a_n) \in A$ a Pareto minimal element of A if there exists no $(b_1, \dots, b_n) \in A$ such that $b_1 \leq a_1, \dots, b_n \leq a_n$ and at least one inequality is strict.

$P_{\min}(A)$ denotes the set of Pareto minimal elements of A.

Pareto minimal paths

We consider the 2-dimensional case. The cases of higher dimensions are similar.

Problem. Given a directed graph $G = (V, E)$ with each edge ij assigned a pair $\mathbf{c}_{ij} = (c^1_{ij}, c^2_{ij})$ of non-negative numbers, and given distinct $s, t \in V$, find a path from s to t whose length is Pareto minimal (among all possible paths from s to t).

When G is acyclic, a method similar to the one for the single objective case works well.

This method works for higher dimensions as well (e.g. each edge is assigned a vector in $\mathbb{R}^k$, where k is the number of objectives).

Pareto minimal paths in acyclic directed graphs

Assume G is acyclic so that we can label its vertices properly by $1, 2, \ldots, n$.

Define

$$f(i) = P_{\min}(\{\text{lengths of paths from } 1 \text{ to } i\}),$$

hence $f(i) \subset \mathbb{R}^2$. Then

$$f(i) = P_{\min}\left(\cup_{j \in P(i)}(f(j) + \mathbf{c}_{ji})\right), \ \ i = 1, 2, \ldots, n,$$

where

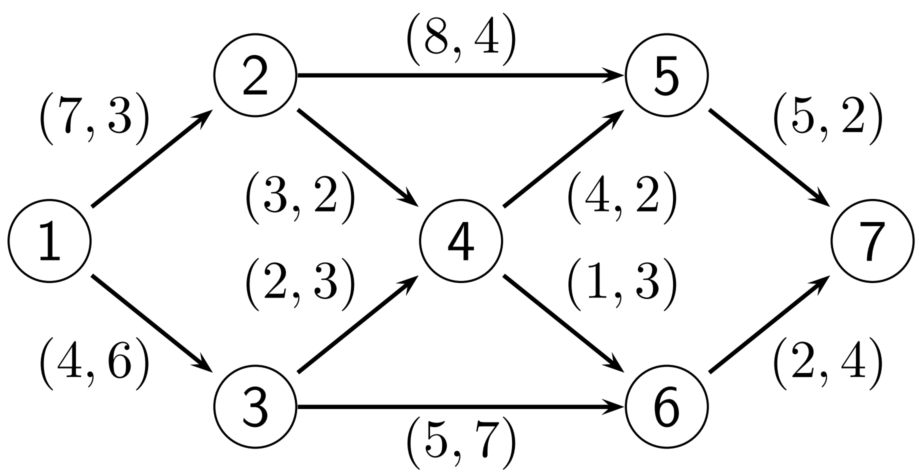
$$f(j) + \mathbf{c}_{ji} = \{(a + c_{ji}^1, b + c_{ji}^2) : (a, b) \in f(j)\} \subset \mathbb{R}^2.$$

Beginning with $f(1) = \{(0, 0)\}$ and working forward, we can compute $f(1), f(2), \ldots, f(n)$ sequentially.

The set $f(n)$ contains the lengths of all Pareto minimal paths from 1 to n. To find corresponding Pareto minimal paths, we need to recover optimal paths for each element in $f(n)$. Note that to do so, a careful registration of how these elements are obtained is necessary.

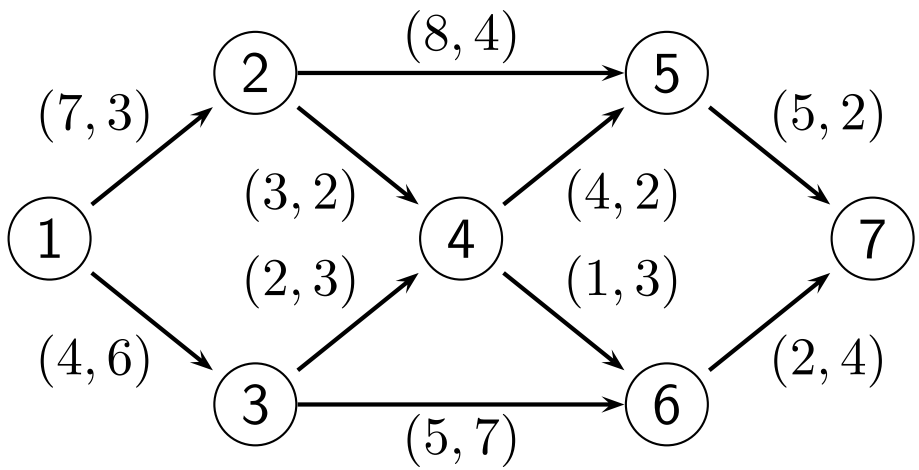
Example 6. (shortest path problem for acyclic directed graphs, two objectives, forward DP)

Find a Pareto minimal path from vertex 1 to vertex 7 in the weighted digraph depicted below.



To find Pareto minimal paths, we sequentially determine $f(1), f(2), \ldots, f(7)$. In doing so we keep track of the paths that are used to obtain the optimal elements in $f(i)$. This helps us to quickly identify the Pareto minimal paths for each element in $f(7)$.

Example (cont.)



Example (cont.)

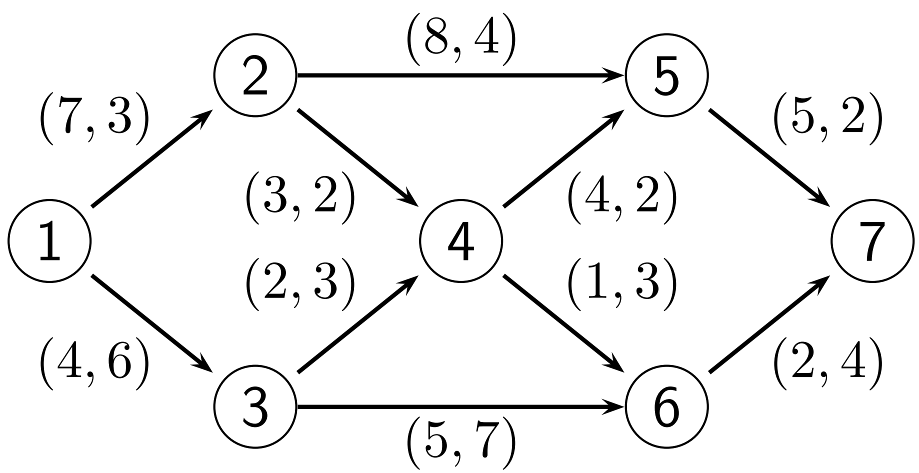
Lexicographic shortest paths

The lexicographic shortest path problem is similar to the Pareto minimal path problem.

The dynamic programming algorithm works for lexicographic case with $P_{\min}$ replaced by $L_{\min}$ in each iteration.

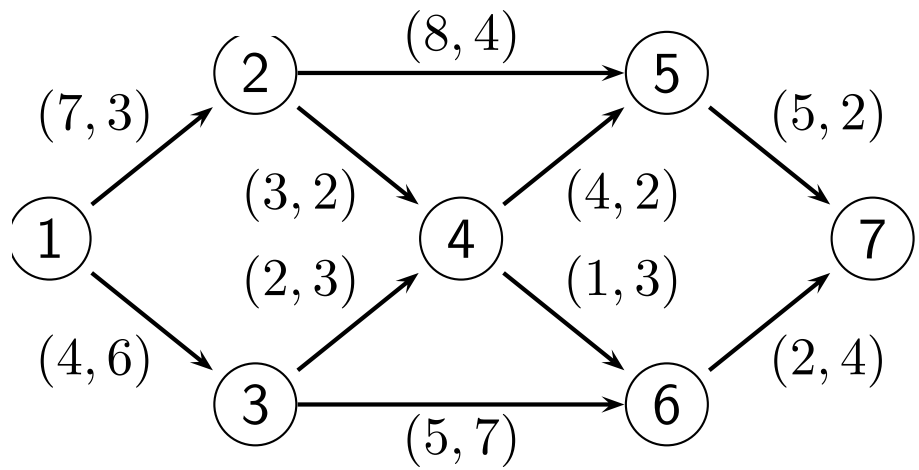
Example 7. (shortest path problem for acyclic directed graphs, two objectives, forward DP)

Find a lexicographic shortest path from vertex 1 to vertex 7 in the weighted digraph depicted below.

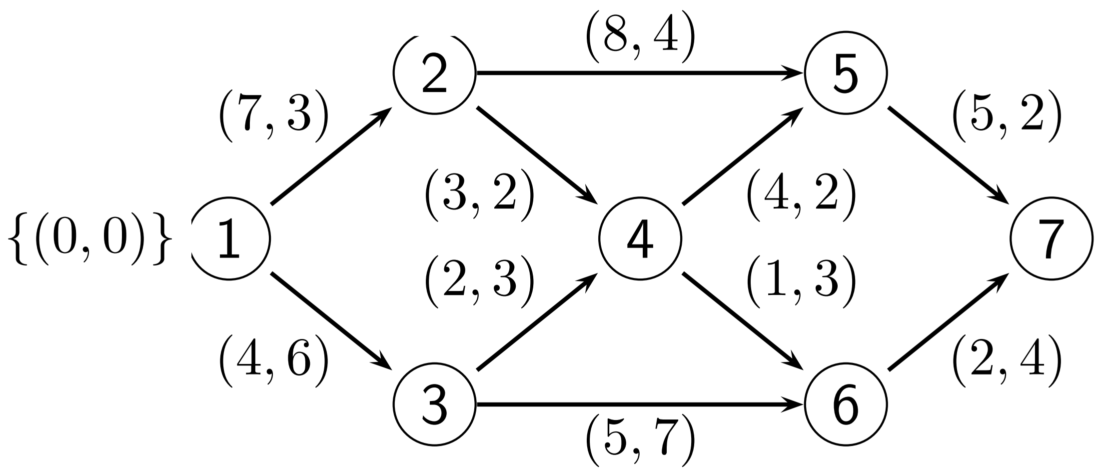


We sequentially determine $f(1)$, $f(2)$,...,$f(7)$.

Example (cont.)

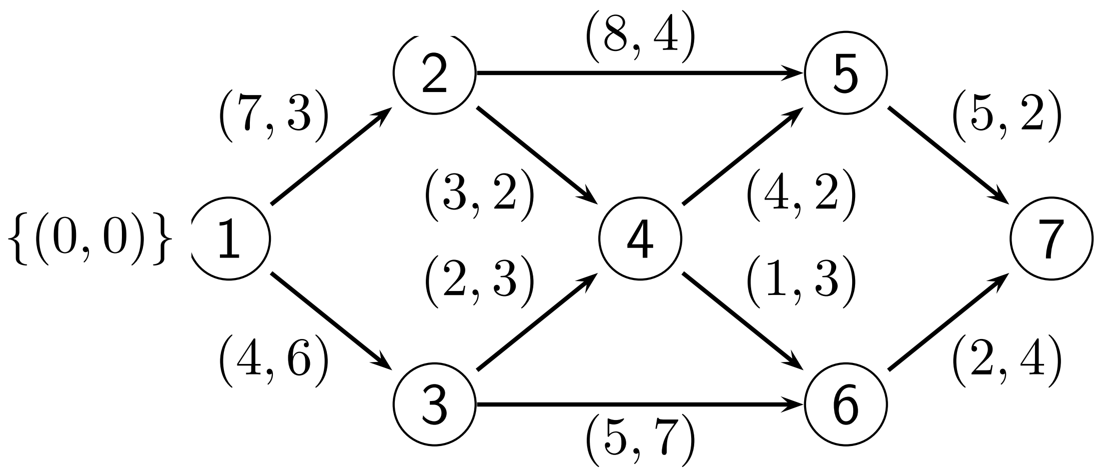


Example (cont.)



By definition $f(1) = \{(0,0)\}$.

Example (cont.)

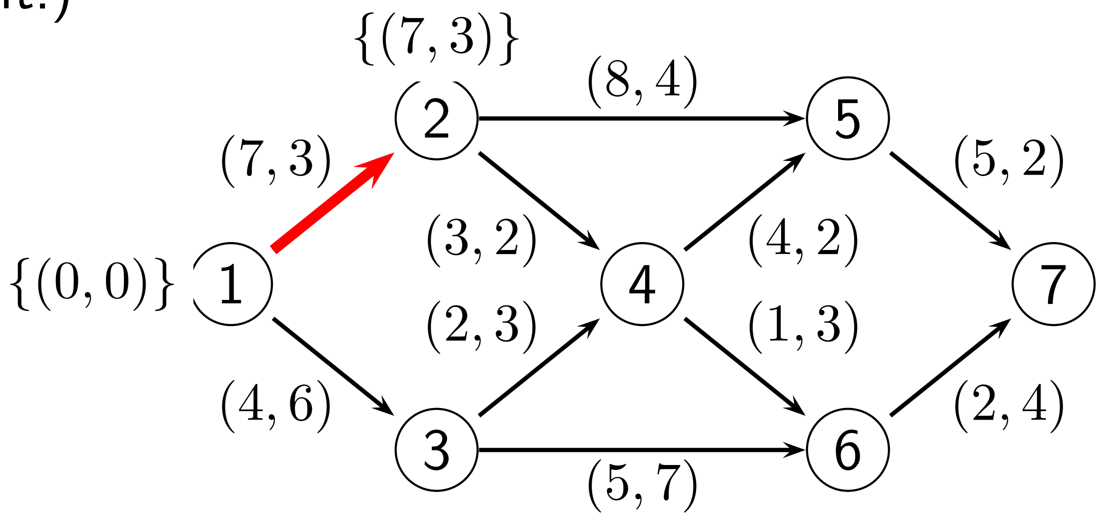


By definition $f(1) = \{(0,0)\}$.

Since $P(2) = \{1\}$, we have

$$\begin{aligned} f(2) &= L_{\min}(\{f(1) + \mathbf{c}_{12}\}) \\ &= L_{\min}(\{(0,0) + (7,3)\}) \\ &= L_{\min}(\{(7,3)\}) \\ &= \{(7,3)\}. \end{aligned}$$

Example (cont.)

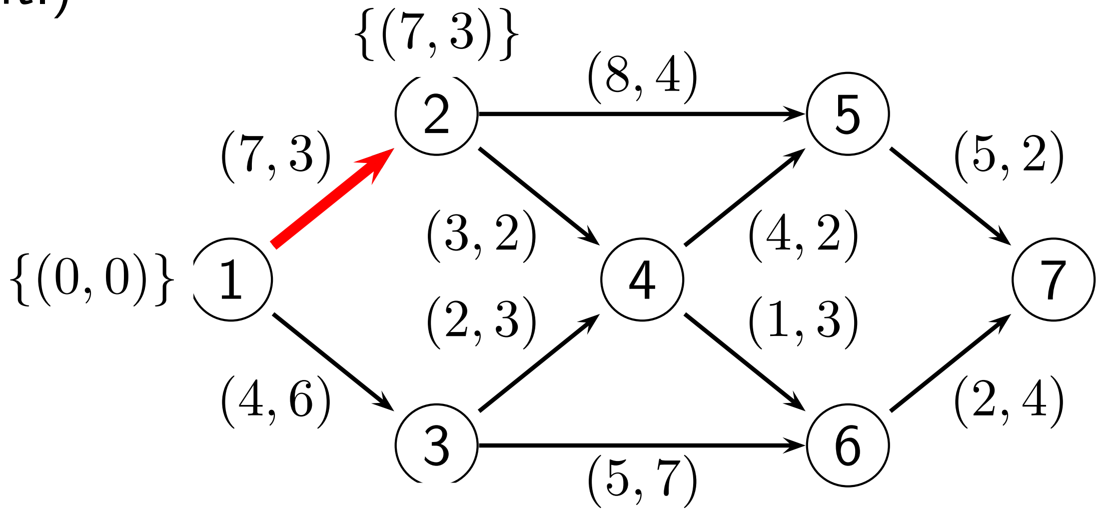


By definition $f(1) = \{(0,0)\}$.

Since $P(2) = \{1\}$, we have

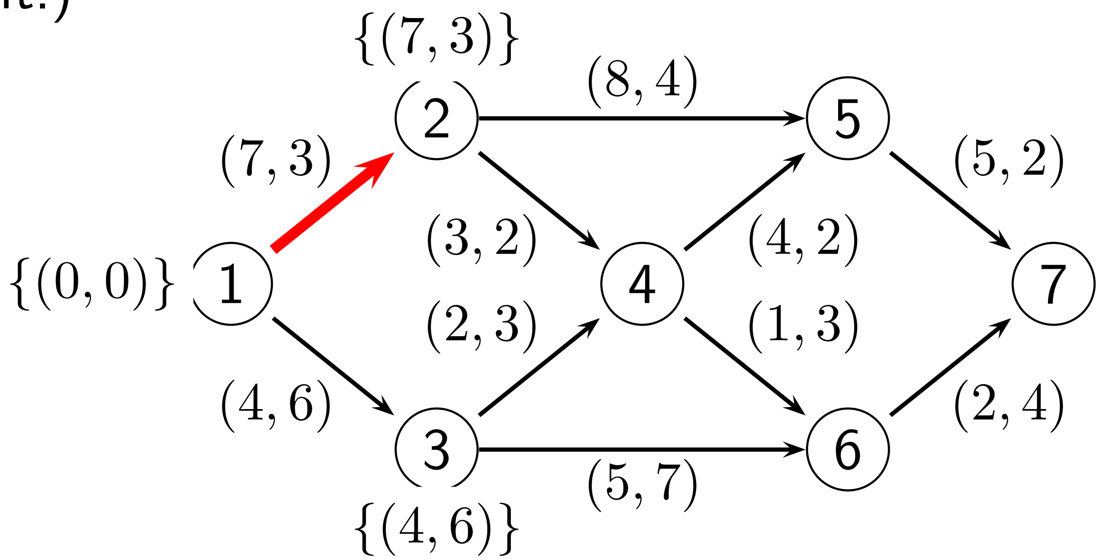
$$\begin{aligned} f(2) &= L_{\min}(\{f(1) + \mathbf{c}_{12}\}) \\ &= L_{\min}(\{(0,0) + (7,3)\}) \\ &= L_{\min}(\{(7,3)\}) \\ &= \{(7,3)\}. \end{aligned}$$

Example (cont.)



Since $P(3) = \{1\}$, we have

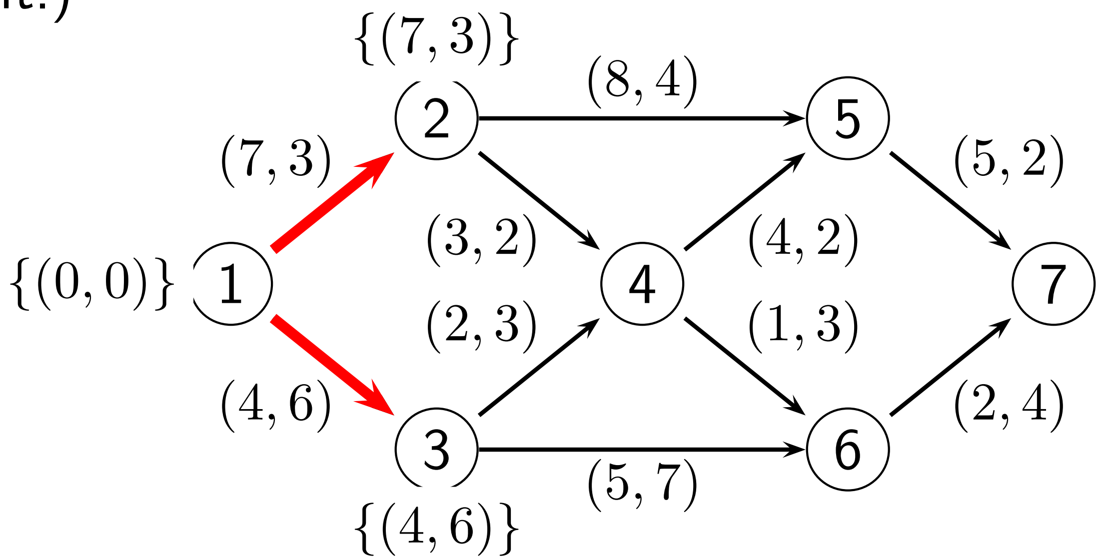
Example (cont.)



Since $P(3) = \{1\}$, we have

$$\begin{aligned} f(3) &= L_{\min}(\{f(1) + \mathbf{c}_{13}\}) \\ &= L_{\min}(\{(0,0) + (4,6)\}) \\ &= L_{\min}(\{(4,6)\}) \\ &= \{(4,6)\}. \end{aligned}$$

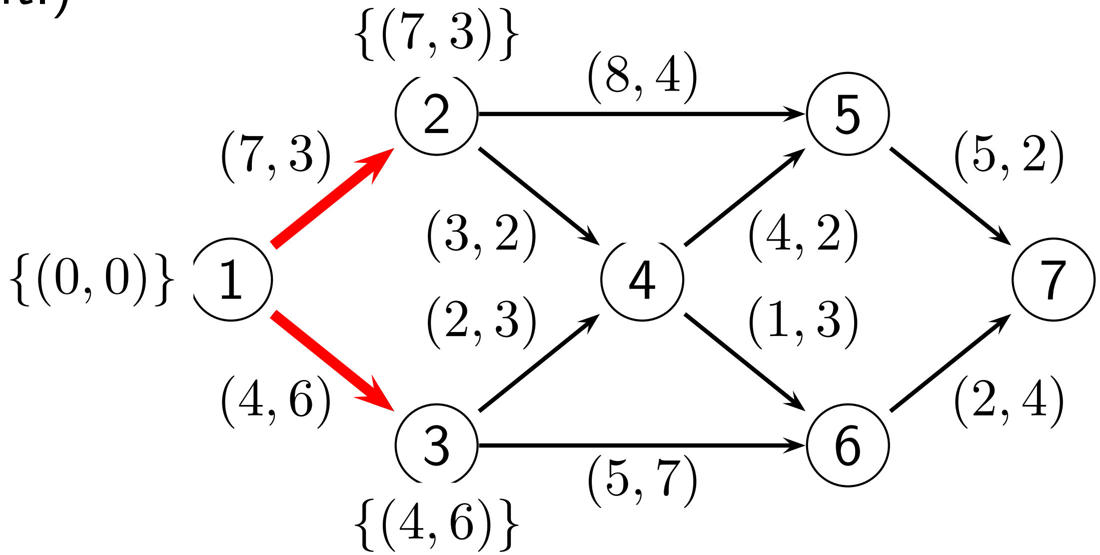
Example (cont.)



Since $P(3) = \{1\}$, we have

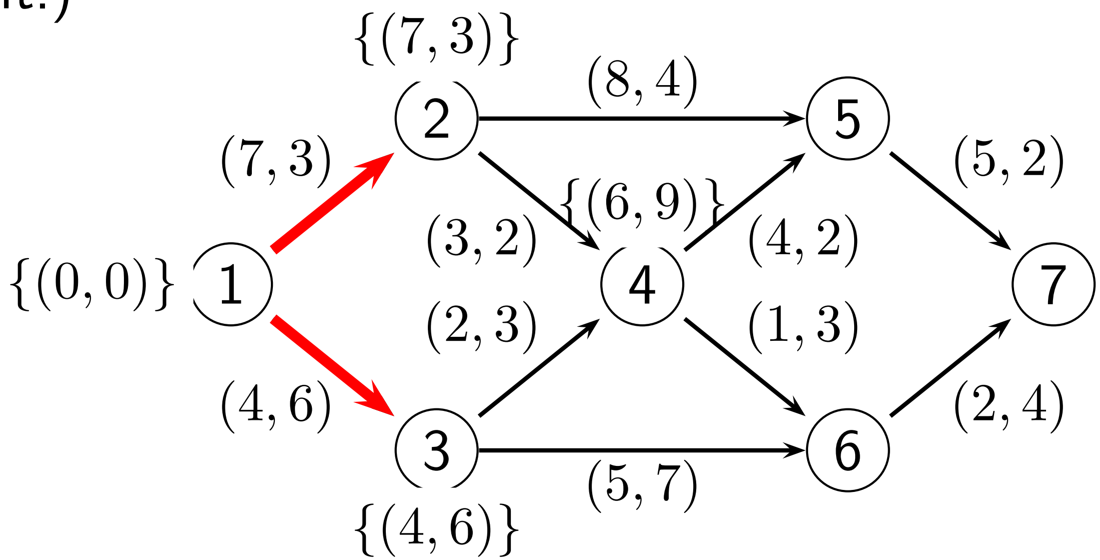
$$\begin{aligned} f(3) &= L_{\min}(\{f(1) + \mathbf{c}_{13}\}) \\ &= L_{\min}(\{(0,0) + (4,6)\}) \\ &= L_{\min}(\{(4,6)\}) \\ &= \{(4,6)\}. \end{aligned}$$

Example (cont.)



Since $P(4) = \{2, 3\}$, we have

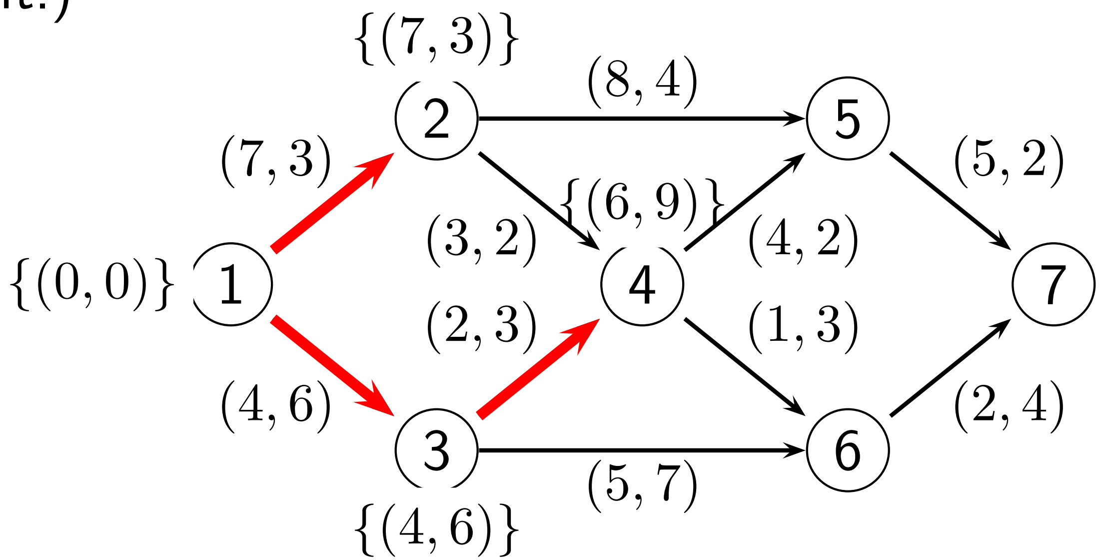
Example (cont.)



Since $P(4) = \{2, 3\}$, we have

$$\begin{aligned} f(4) &= L_{\min}(\{f(2) + \mathbf{c}_{24}, f(3) + \mathbf{c}_{34}\}) \\ &= L_{\min}(\{(7,3) + (3,2), (4,6) + (2,3)\}) \\ &= L_{\min}(\{(10,5), (6,9)\}) \\ &= \{(6,9)\}. \end{aligned}$$

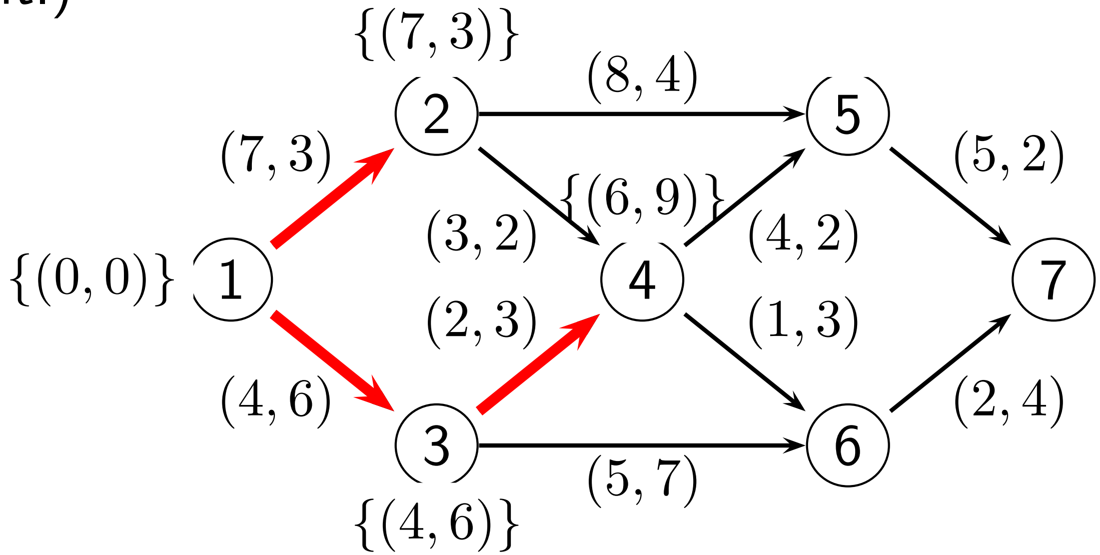
Example (cont.)



Since $P(4) = \{2, 3\}$, we have

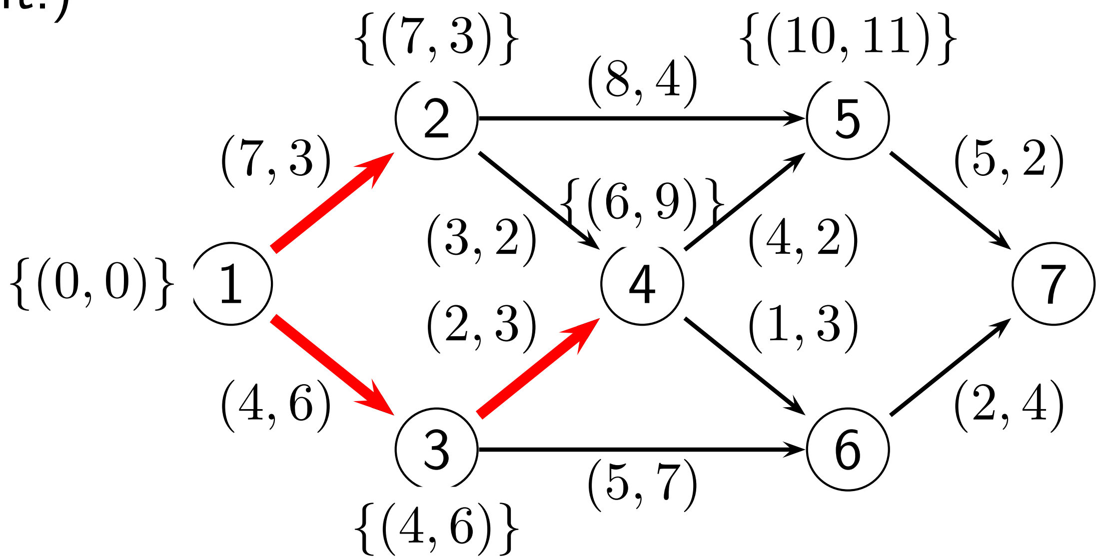
$$\begin{aligned} f(4) &= L_{\min}(\{f(2) + \mathbf{c}_{24}, f(3) + \mathbf{c}_{34}\}) \\ &= L_{\min}(\{(7,3) + (3,2), (4,6) + (2,3)\}) \\ &= L_{\min}(\{(10,5), (6,9)\}) \\ &= \{(6,9)\}. \end{aligned}$$

Example (cont.)



Since $P(5) = \{2, 4\}$, we have

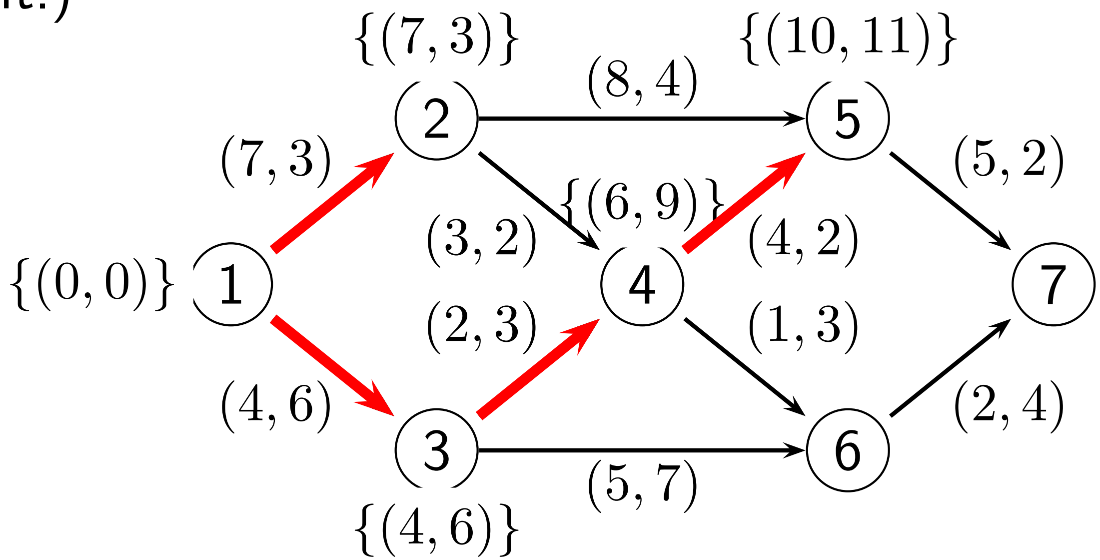
Example (cont.)



Since $P(5) = \{2, 4\}$, we have

$$\begin{aligned} f(5) &= L_{\min}(\{f(2) + \mathbf{c}_{25}, f(4) + \mathbf{c}_{45}\}) \\ &= L_{\min}(\{(7,3) + (8,4), (6,9) + (4,2)\}) \\ &= L_{\min}(\{(15,7), (10,11)\}) \\ &= \{(10,11)\}. \end{aligned}$$

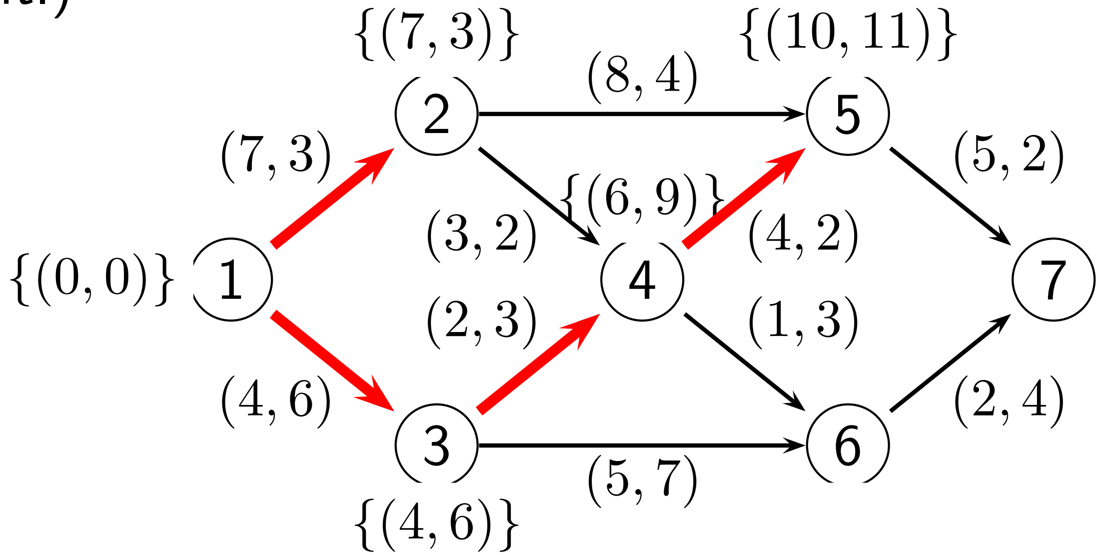
Example (cont.)



Since $P(5) = \{2,4\}$, we have

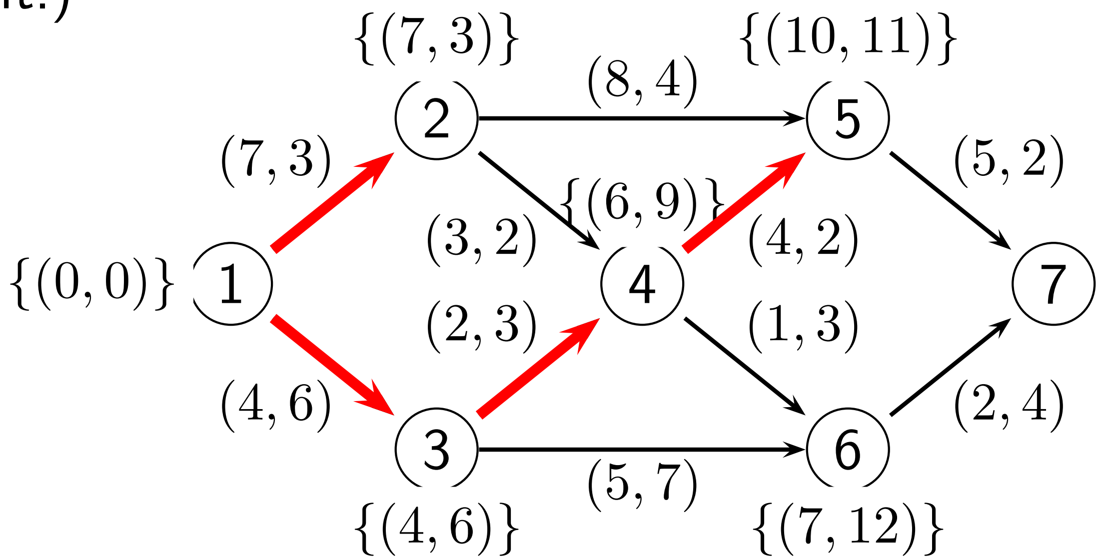
$$\begin{aligned} f(5) &= L_{\min}(\{f(2) + \mathbf{c}_{25}, f(4) + \mathbf{c}_{45}\}) \\ &= L_{\min}(\{(7,3) + (8,4), (6,9) + (4,2)\}) \\ &= L_{\min}(\{(15,7), (10,11)\}) \\ &= \{(10,11)\}. \end{aligned}$$

Example (cont.)



Since $P(6) = \{3, 4\}$, we have

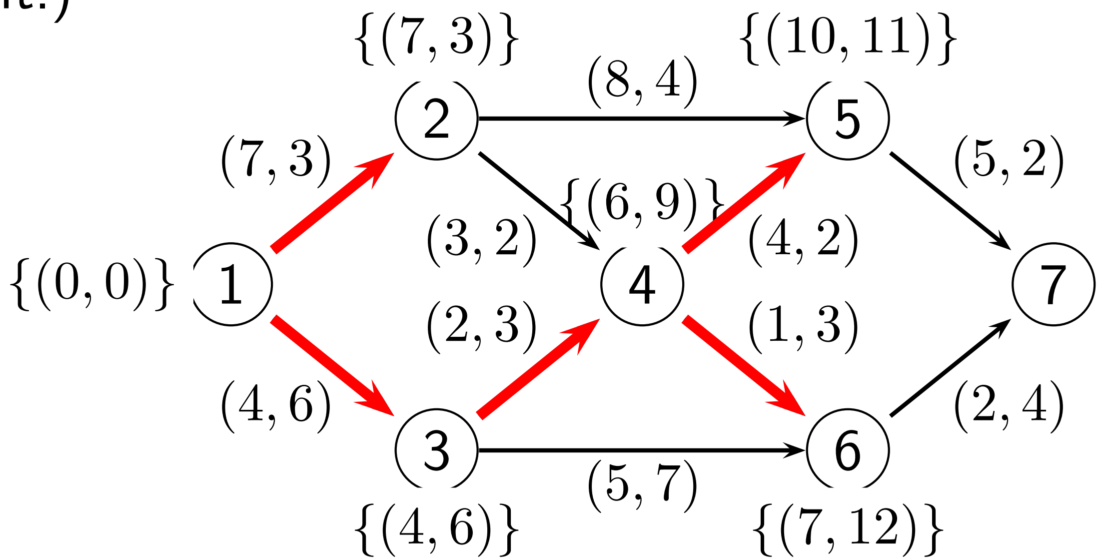
Example (cont.)



Since $P(6) = \{3, 4\}$, we have

$$\begin{aligned} f(6) &= L_{\min}(\{f(3) + \mathbf{c}_{36}, f(4) + \mathbf{c}_{46}\}) \\ &= L_{\min}(\{(4,6) + (5,7), (6,9) + (1,3)\}) \\ &= L_{\min}(\{(9,7), (7,12)\}) \\ &= \{(7,12)\}. \end{aligned}$$

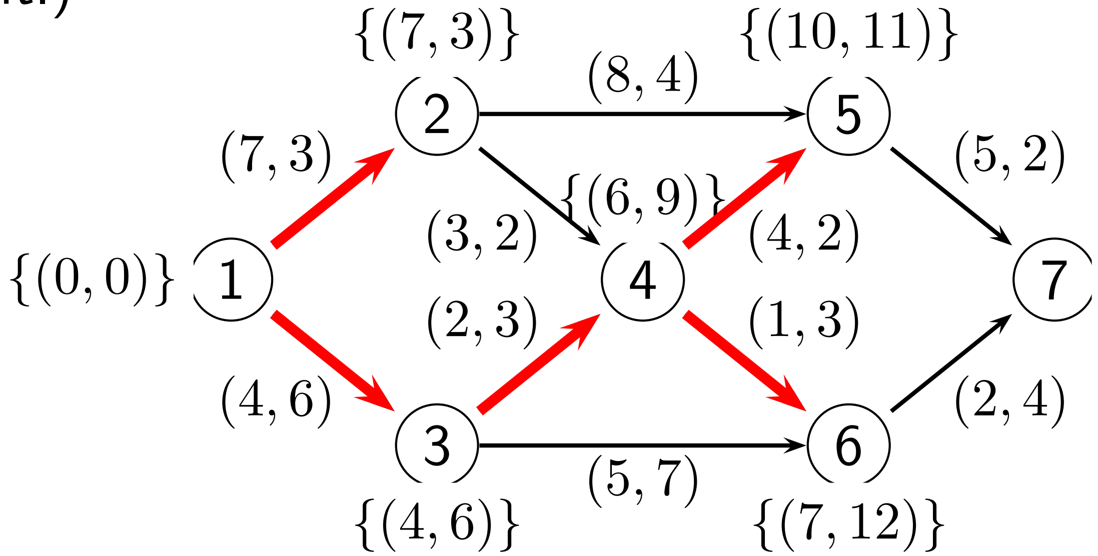
Example (cont.)



Since $P(6) = \{3,4\}$, we have

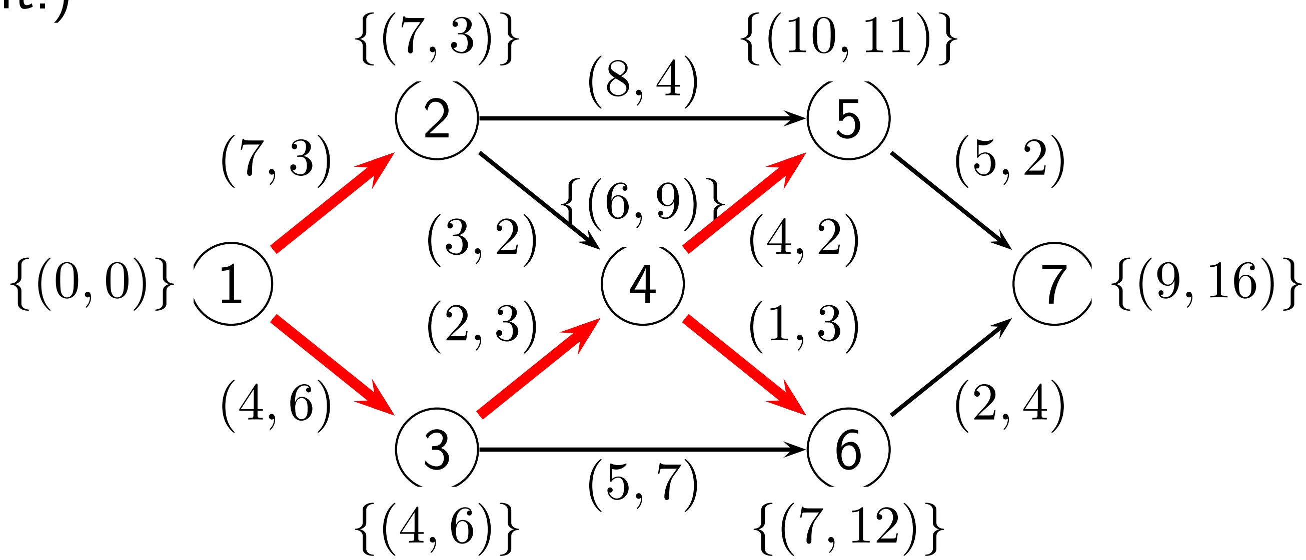
$$\begin{aligned} f(6) &= L_{\min}(\{f(3) + \mathbf{c}_{36}, f(4) + \mathbf{c}_{46}\}) \\ &= L_{\min}(\{(4,6) + (5,7), (6,9) + (1,3)\}) \\ &= L_{\min}(\{(9,7), (7,12)\}) \\ &= \{(7,12)\}. \end{aligned}$$

Example (cont.)



Since $P(7) = \{5, 6\}$, we have

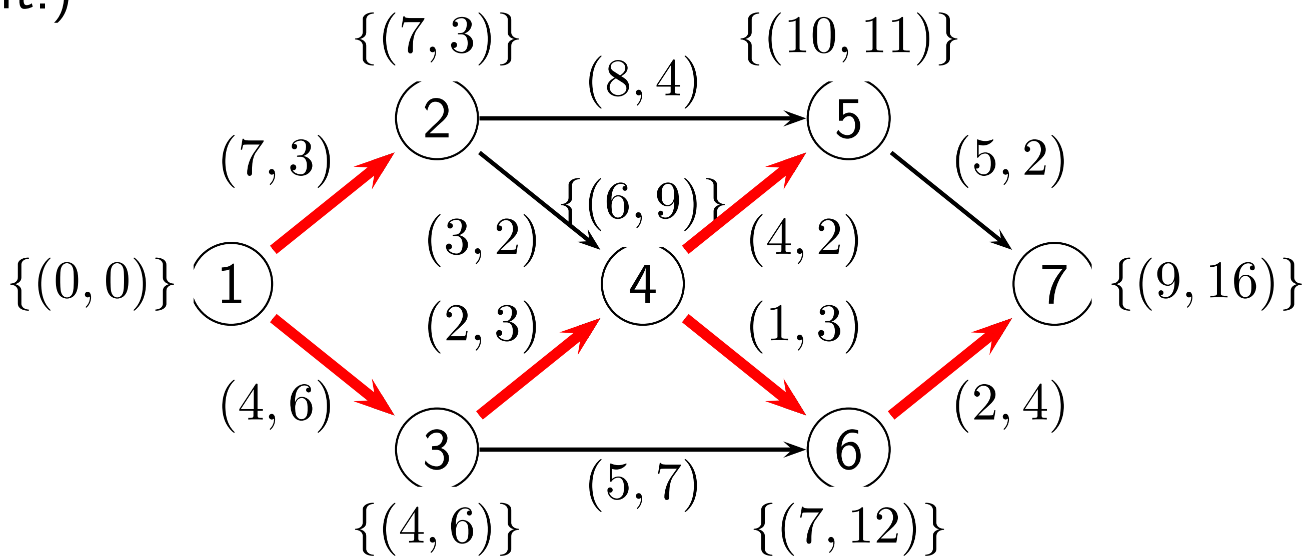
Example (cont.)



Since $P(7) = \{5, 6\}$, we have

$$\begin{aligned} f(7) &= L_{\min}(\{f(5) + \mathbf{c}_{57}, f(6) + \mathbf{c}_{67}\}) \\ &= L_{\min}(\{(10, 11) + (5, 2), (7, 12) + (2, 4)\}) \\ &= L_{\min}(\{(15, 13), (9, 16)\}) \\ &= \{(9, 16)\}. \end{aligned}$$

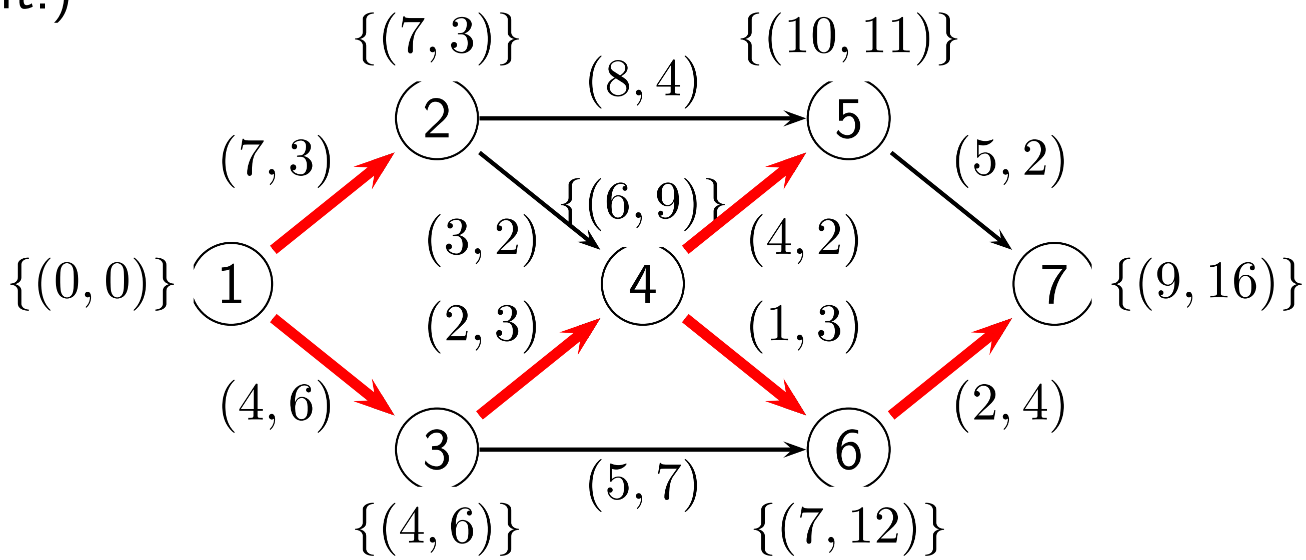
Example (cont.)



Since $P(7) = \{5, 6\}$, we have

$$\begin{aligned} f(7) &= L_{\min}(\{f(5) + \mathbf{c}_{57}, f(6) + \mathbf{c}_{67}\}) \\ &= L_{\min}(\{(10, 11) + (5, 2), (7, 12) + (2, 4)\}) \\ &= L_{\min}(\{(15, 13), (9, 16)\}) \\ &= \{(9, 16)\}. \end{aligned}$$

Example (cont.)



The lexicographic minimal distance is $(9, 16)$. To find the lexicographic minimal paths we work backwards for each element in $f(7)$ to recover the optimal paths.

Working backwards, we find the lexicographic minimal path $1 - 3 - 4 - 6 - 7$.

Deterministic dynamic programming

Characteristics of dynamic programming

- The problem can be divided into stages with a decision required at each stage.
- Each stage has a number of states associated with it.
- The decision chosen at any stage determines how the state at the current stage is transformed into the state at the next stage.
- Principle of optimality: Given the current state, the optimal decision for each of the remaining stages must not depend on previously reached states or previously chosen decisions (we are going to work backwards).
- DP recursive equation:

$$f_i(x_i) = \text{opt}\{c(x_i, x_{i+1}) + f_{i+1}(x_{i+1}) : \text{all possible } x_{i+1}\}, \quad i = 1, 2, \ldots,$$

  where opt can be min or max (depending on the given problem), x_i is the current state at stage i, x_{i+1} is the state at next stage $i+1$, and $c(x_i, x_{i+1})$ is the cost (if opt $=$ min) or reward (if opt $=$ max) during the transition from stage i to stage $i+1$.

Formulating and solving a DP problem

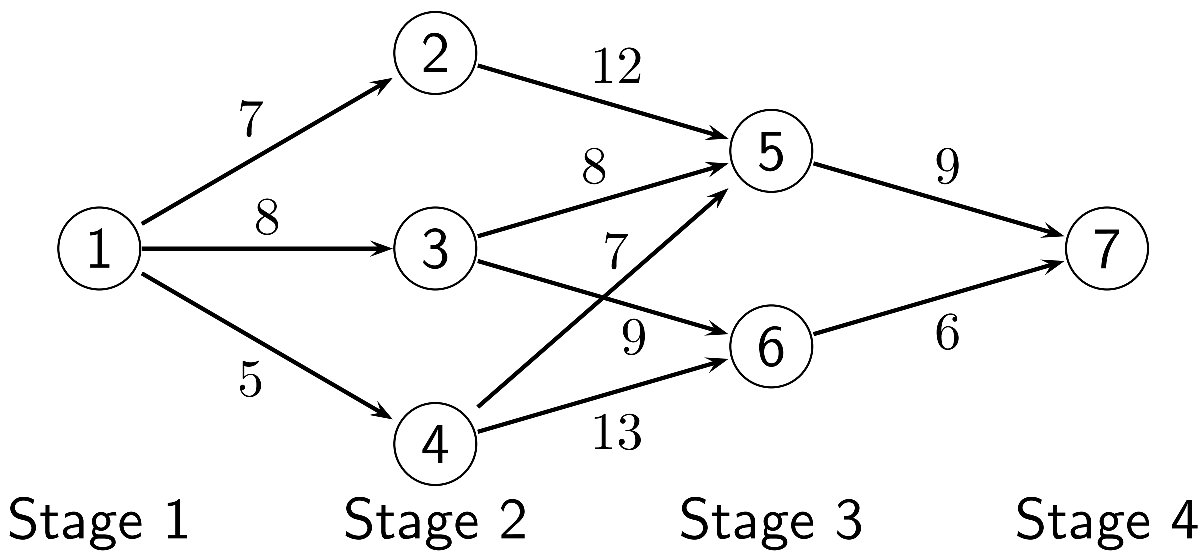
In formulating a deterministic DP problem we need to identify:

- the stages;
- the states for each stage;
- the actions (decisions) at each state;
- the cost/reward of an action at each stage;
- function $f(\cdot)$;
- relationship between $f(\cdot)$ at a typical stage and $f(\cdot)$ at next stage.

After all these are done, write down the DP equation, work *backward* beginning with the final stage until the first stage is reached.

Example 8. (A multi-stage shortest path problem, backward dynamic programming algorithm)

Determine a shortest path from vertex 1 to 7 in the graph depicted below, using backward dynamic programming. The numbers on the edges indicate the lengths between the different vertices.



In this example the objective is to minimise the length of the path from vertex 1 to vertex 7. Let $d(x_i, x_{i+1})$ denote the distance between x_i and x_{i+1}.

Example (cont.) The objective is to minimise the length of the path from vertex 1 to vertex 7. In total there are 4 stages. For $i = 3, 2, 1$ and x_i a state in stage i, let $f_i(x_i)$ denote the length of a shortest path from x_i to 7. Then

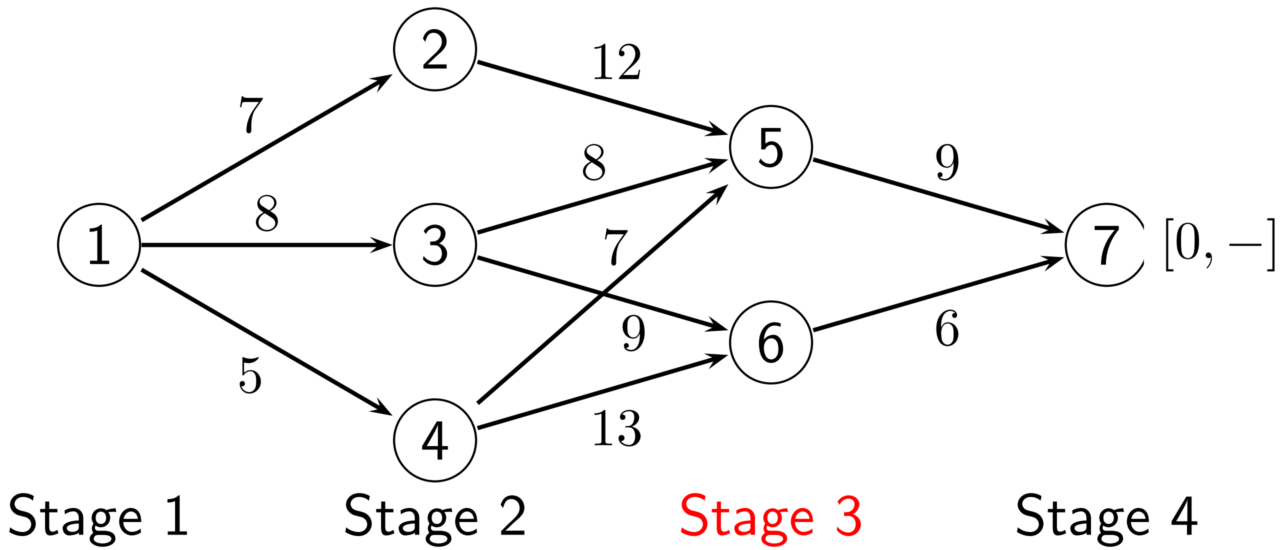
$$f_i(x_i) = \min\{d(x_i, x_{i+1}) + f_{i+1}(x_{i+1})\},\ i = 3, 2, 1,$$

where the minimum is taken over all x_{i+1} in stage $i + 1$ such that the edge $x_i x_{i+1}$ is present in the graph.

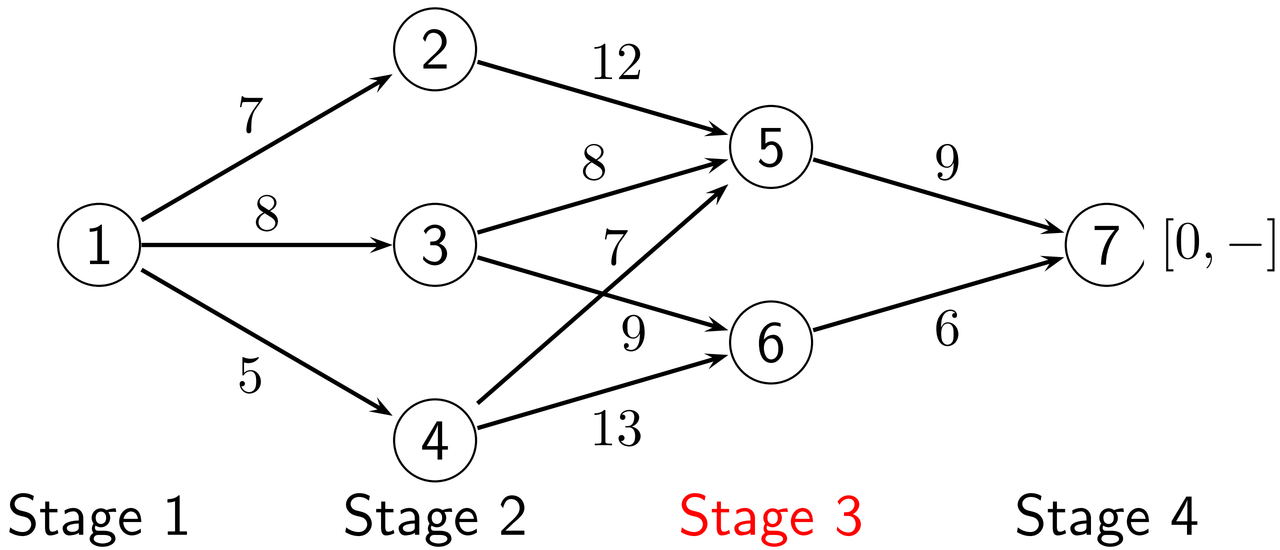
Start with state $x_4 = 7$ in stage 4. We define $f_4(x_4) = 0$, which means that the length of a shortest path from x_4 to vertex 7 is 0.

To find the optimum solution, we also sequentially indicate in the picture for each vertex $[f_i(x_i), x^*_{i+1}]$, where x^*_{i+1} attains the optimum value in $f_i(x_i)$.

Example (cont.)

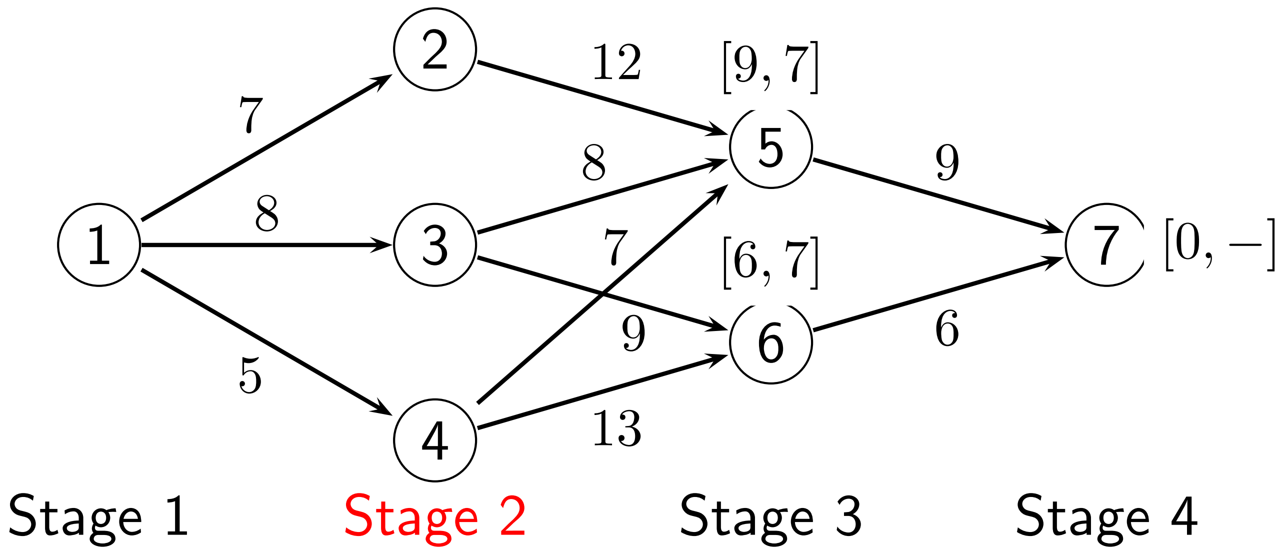


Example (cont.)

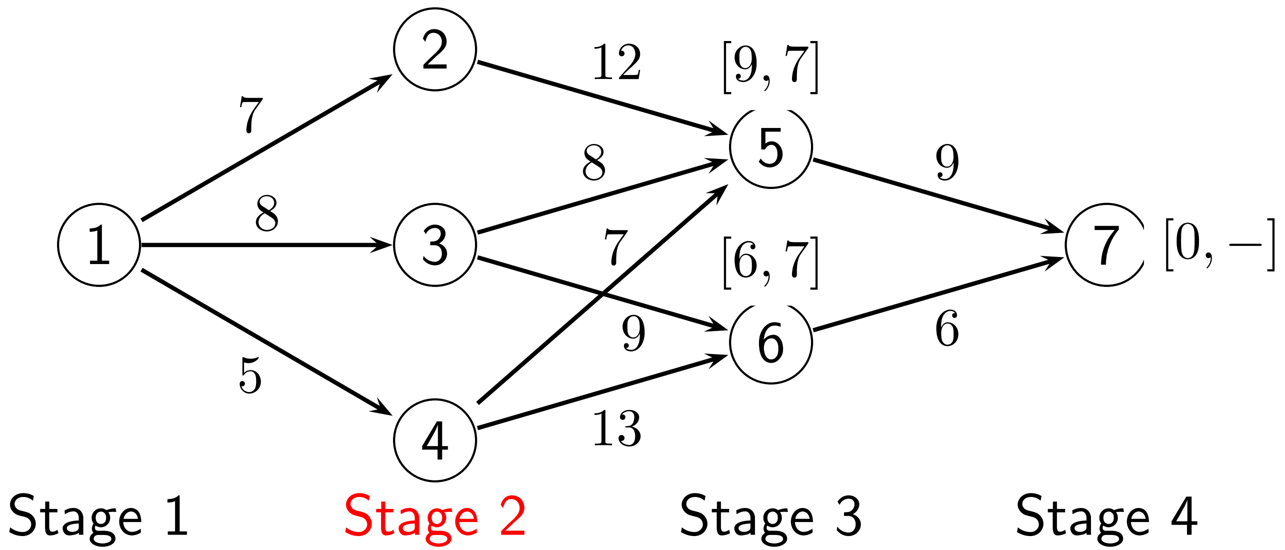


x_3	$d(x_3, x_4) + f_4(x_4)$	optimum solution	
	$x_4 = 7$	$f_3(x_3)$	x_4^*
5	9	9	7
6	6	6	7

Example (cont.)

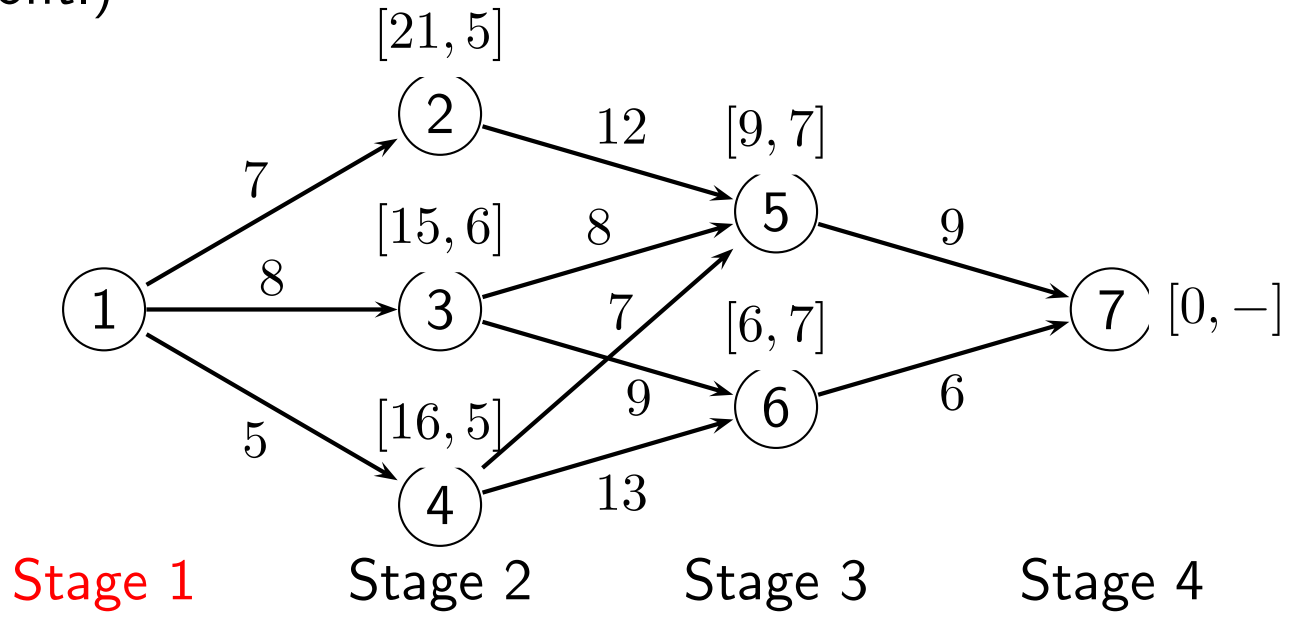


Example (cont.)

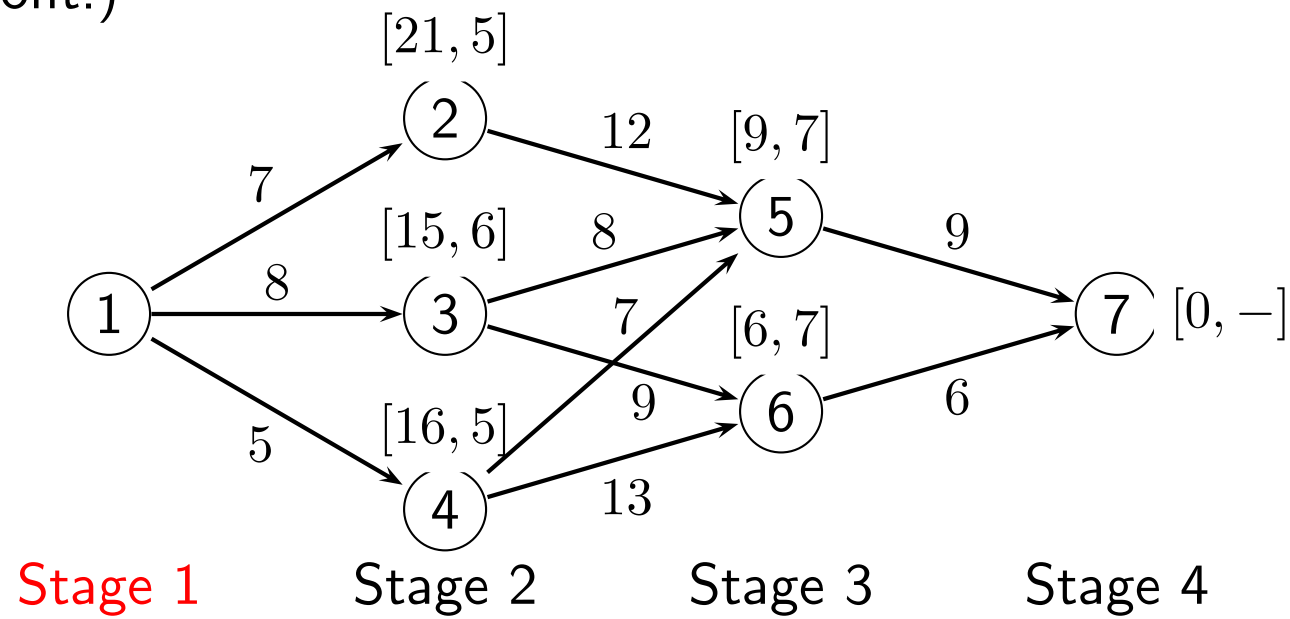


x_2	$d(x_2, x_3) + f_3(x_3)$		optimum solution	
	$x_3 = 5$	$x_3 = 6$	$f_2(x_2)$	x_3^*
2	$12 + 9 = 21$	—	21	5
3	$8 + 9 = 17$	$9 + 6 = 15$	15	6
4	$7 + 9 = 16$	$13 + 6 = 19$	16	5

Example (cont.)

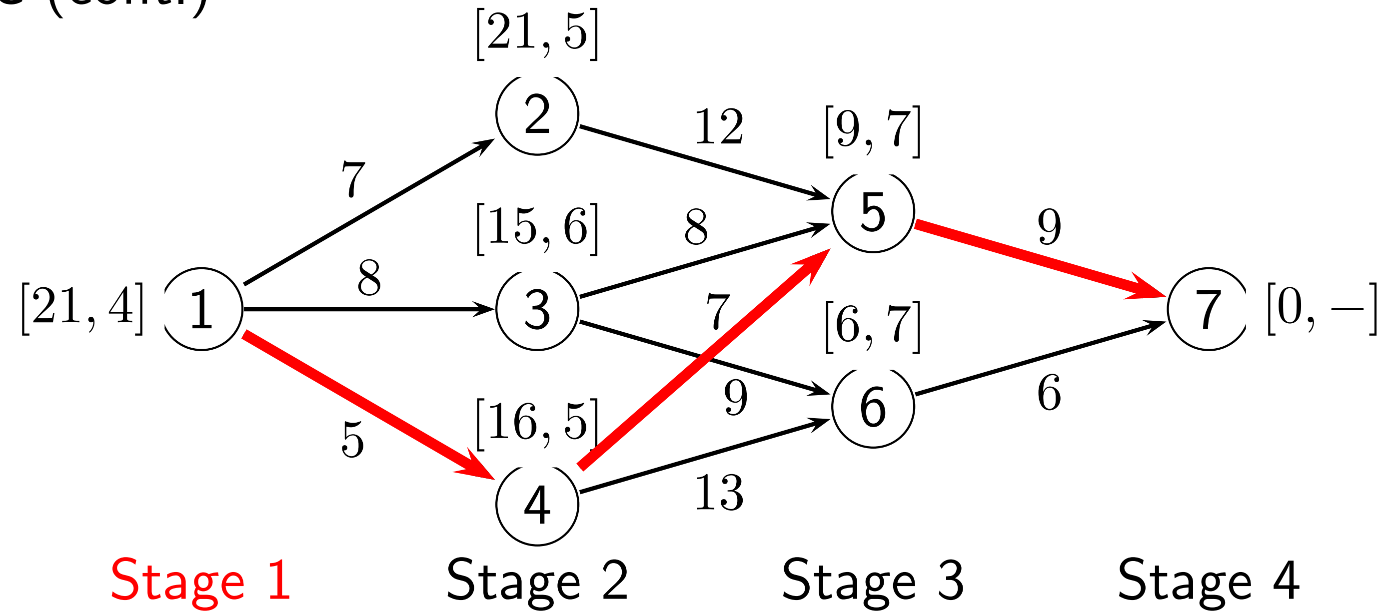


Example (cont.)



x_1	$d(x_1, x_2) + f_2(x_2)$			optimum solution	
	$x_2 = 2$	$x_2 = 3$	$x_2 = 4$	$f_1(x_1)$	x_2^*
1	$7 + 21 = 28$	8+15=23	5+16=21	21	4

Example (cont.)



Shortest path from 1 to 7: 1-4-5-7 with length 21.

Resource allocation problem

Suppose that

- there are initially w units of resource available (where w may not be an integer);
- there are T activities, each requiring resource;
- there are x_t units of resource available for activities $t, t+1, \ldots, T$;
- if activity t is implemented at level y_t (where $y_t \geq 0$ is an integer), then $g_t(y_t)$ units of resource are required;
- and $r_t(y_t)$ units of reward are obtained.

Problem (Resource allocation problem) Determine an allocation of resource to activities $1, \ldots, T$ that maximises the total reward subject to the limit of resource.

That is, find non-negative integers $y_1, \ldots, y_T$ which maximise $\sum_{t=1}^{T} r_t(y_t)$, such that $\sum_{t=1}^{T} g_t(y_t) \leq w$.

We will formulate this problem in such a way that backward deterministic dynamic programming can be used to solve it.

Let the T activities correspond to the stages of the dynamic programming problem, and let the variables x_t correspond to the states associated with stage t, for $1 \leq t \leq T$.

Define $f_t(x_t)$ to be the maximum possible reward that can be obtained from activities $t, t+1, \ldots, T$ if x_t units of resource can be used for these activities.

Then

$$f_{T+1}(x_{T+1}) = 0 \text{ for all } x_{T+1}$$

$$f_t(x_t) = \max\{r_t(y_t) + f_{t+1}(x_t - g_t(y_t)) : y_t \geq 0 \text{ an integer s.t. } g_t(y_t) \leq x_t\}.$$

Let $y_t^*(x_t)$ be any value that attains $f_t(x_t)$.

We can solve the resource allocation problem as follows.

- Begin with $f_T(\cdot)$ and $y_T^*(\cdot)$ and work backward to compute $f_{T-1}(\cdot), y_{T-1}^*(\cdot), \ldots, f_1(\cdot), y_1^*(\cdot)$ sequentially.
- The optimal solution for a resource allocation problem with w units of resource, can be read off from this sequence by starting at $f_1(w), y_1^*(w)$ and working forward to obtain $f_2(w - g_1(y_1^*(w)))$, $y_2^*(w - g_1(y_1^*(w))), \ldots$.

Example 9. (Resource allocation, Winston, Section 18.4) Mr. Gamble has \$6,000 to invest, and three investments are available. If d_j dollars (in thousands) are invested in investment j, then a net present value (in thousands) of $r_j(d_j)$ is obtained, where

$$r_1(d_1) = 7d_1 + 2 \quad (d_1 > 0)$$

$$r_2(d_2) = 3d_2 + 7 \quad (d_2 > 0)$$

$$r_3(d_3) = 4d_3 + 5 \quad (d_3 > 0)$$

$$r_1(0) = r_2(0) = r_3(0) = 0.$$

Assume that the amount placed in each investment must be an exact multiple of \$1,000. To maximise the net present value obtained from the investments, how should Mr. Gamble allocate the \$6,000?

Formulate this problem as a resource allocation problem.

Example (cont.)

Define

- stage t = investing money in investment t, $t = 1, 2, 3$;
- x_t = state at stage t = the amount of money available (in thousands of dollars) for investments $t, \ldots, 3$, where x_t is an integer smaller than or equal to 6;
- y_t = the amount of money actually invested in investment t. Observe that $g_t(y_t) = y_t$, and $y_t \leq x_t$. The net present value of investing y_t in investment t is $r_t(y_t)$.
- action at stage t: invest y_t;
- $f_t(x_t)$ = the maximum net present value that can be obtained from investments $t, \ldots, 3$, if x_t thousands dollar is available at stage t.

Example (cont.)

Then

$$f_3(x_3) = \max_{y_3 \in \{0,1,\ldots,x_3\}} \{r_3(y_3)\} = r_3(x_3), \quad \text{and } y_3^*(x_3) = x_3,$$

since $r_3(\cdot)$ is an increasing function, it is optimal to invest all money available in investment 3 at stage 3.

For $t = 1, 2$ we have

$$f_t(x_t) = \max_{y_t \in \{0,1,\ldots,x_t\}} \{r_t(y_t) + f_{t+1}(x_t - y_t)\}.$$

Example (cont.)

Stage 3 computations:

x_3	$f_3(x_3) = r_3(x_3)$	$y_3^* = x_3$
0	0	0
1	$9 = 4 \cdot 1 + 5$	1
2	$13 = 4 \cdot 2 + 5$	2
3	$17 = 4 \cdot 3 + 5$	3
4	21	4
5	25	5
6	29	6

Stage 2 computations: based on the values obtained for $f_3(x_3)$, compute

$$f_2(x_2) \quad = \quad \max_{y_2 \in \{0,1,\ldots,x_2\}} \{r_2(y_2) + f_3(x_2 - y_2)\}, \quad \text{and the optimal } y_2^*(x_2),$$

for $x_2 = 0, 1, \ldots, 6$ (where $r_2(y_2) = 0$ if $y_2 = 0$, and $r_2(y_2) = 3y_2 + 7$ if $y_2 > 0$).

Example (cont.)

Stage 1 computations (note that we know that $x_1 = 6$): based on the previous results, compute $f_1(x_1) = f_1(6)$ and the optimal $y_1^*(6)$, and read off the optimal solution

$$\begin{aligned} f_1(6) &= \max_{y_1 \in \{0,1,\ldots,6\}} \{r_1(y_1) + f_2(6 - y_1)\} \\ &= \max\{\underbrace{f_2(6)}_{y_1=0}, \underbrace{9 + f_2(5)}_{y_1=1}, \underbrace{16 + f_2(4)}_{y_1=2}, \ldots, \underbrace{44 + f_2(0)}_{y_1=6}\}. \end{aligned}$$

The knapsack problem

You have k types of items, and you want to bring some of them to an aircraft. Each item of type i has value v_i and weight w_i. The airline's weight limit is w. How should you pack your belongings such that the total value of the packed items is maximised subject to the weight constraint?

Assume you pack y_i items of type i.

Then the problem is

$$\max \sum_{i=1}^{k} y_i v_i$$

$$s.t. \sum_{i=1}^{k} y_i w_i \leq w$$

$y_1, \ldots, y_k \geq 0$ are integers.

This is a special case of the resource allocation problem for which

$$g_i(y_i) = y_i w_i, \;\; r_i(y_i) = y_i v_i, \;\; i = 1, \ldots, k.$$

Let

$$f_i(x_i) = \text{max. total value with weight limit } x_i \text{ and types } i, i+1, \ldots, k$$

Then the DP equation is

$$f_i(x_i) = \max\{y_i v_i + f_{i+1}(x_i - y_i w_i) : y_i \geq 0 \text{ an integer s.t. } y_i w_i \leq x_i\}.$$

Beginning with $f_k(\cdot)$ and $y_k(\cdot)$ and working backward, compute $f_{k-1}(\cdot), y_{k-1}(\cdot), \ldots, f_1(\cdot), y_1(\cdot)$ sequentially.

Example 10. (Winston, Section 18.4, knapsack problem)

Solve the knapsack problem with 3 types of items such that $w_1 = 4, w_2 = 3, w_3 = 5$ and $v_1 = 11, v_2 = 7, v_3 = 12$ and $w = 10$.

Example (cont.)