Shortest paths

Jean Cousty

MorphoGraph and Imagery

2011

Outline

Network

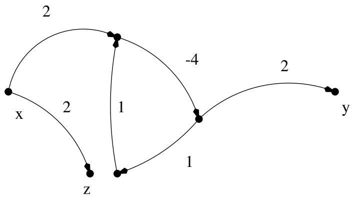
Definition

- *A* network *is a triple* $N = (E, \Gamma, \ell)$ *such that*
  - (E, Γ) *is a graph without loop; and*
  - ℓ *is a map from* $\overrightarrow{\Gamma}$ *in* $\mathbb{R}$
- *If* (E, Γ, ℓ) *is a network and if* $u \in \overrightarrow{\Gamma}$ *is an arc, the real number* $\ell(u)$ *is called the* length of u

Network

Definition

- *A* network *is a triple* $N = (E, \Gamma, \ell)$ *such that*
  - (E, Γ) *is a graph without loop; and*
  - ℓ *is a map from* $\overrightarrow{\Gamma}$ *in* $\mathbb{R}$
- *If* (E, Γ, ℓ) *is a network and if* $u \in \overrightarrow{\Gamma}$ *is an arc, the real number* $\ell(u)$ *is called the* length of u



Notations

- In the sequel of this lecture, $N = (E, \Gamma, \ell)$ denotes a network, and G denotes the graph $G = (E, \Gamma)$
- If $u = (x, y)$ is an arc of G, we write $\ell(x, y)$ instead of $\ell((x, y))$

Length of a path

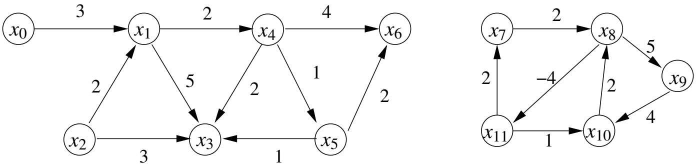
Definition

- *Let* $\pi = (x_0, \ldots, x_n)$ *be a path in* G
- *The* length of π (in N) *is the sum of the length of the arcs in* π*:*
  - $L(\pi) = \sum\{\ell(x_i, x_{i+1}) \mid 0 \leq i \leq n-1\}$

Length of a path

Definition

- *Let $\pi = (x_0, \ldots, x_n)$ be a path in G*
- *The length of π (in N) is the sum of the length of the arcs in π:*
  - $L(\pi) = \sum\{\ell(x_i, x_{i+1}) \mid 0 \leq i \leq n-1\}$



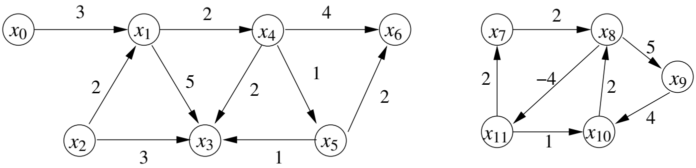
$$L((x_0, x_1, x_3)) = 8$$

Shortest path

Definition

- *Let x and y be two vertices of G*
- *A* shortest path from x to y (in N) *is a path π from x to y such that the length of π is less than or equal to the length of any other path from x to y:*
  - $\forall \pi'$ *path from x to y*, $L(\pi) \leq L(\pi')$

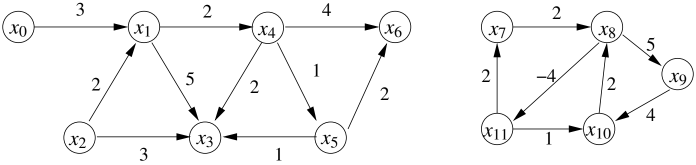
Shortest path: illustration



Example

- $\pi = (x_0, x_1, x_3)$

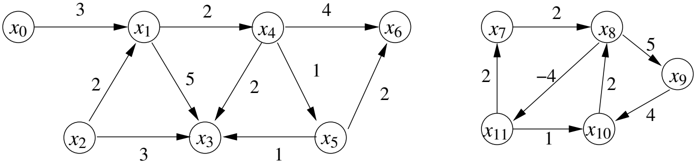
Shortest path: illustration



Example

- $\pi = (x_0, x_1, x_3)$ *is not a shortest path from* x_0 *to* x_3 *(*$L(\pi) = 8$*)*

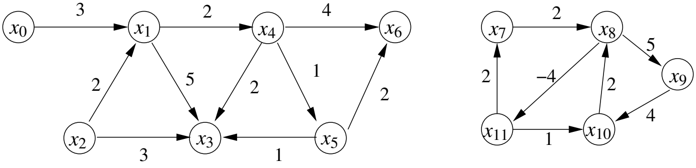
Shortest path: illustration



Example

- $\pi = (x_0, x_1, x_3)$ *is not a shortest path from* x_0 *to* x_3 *(*$L(\pi) = 8$*)*
- $\pi = (x_0, x_1, x_4, x_3)$ *is a shortest path from* x_0 *to* x_3 *(*$L(\pi) = 7$*)*

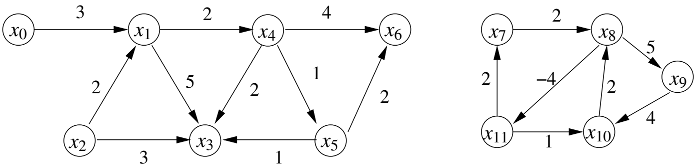
Shortest path: illustration



Example

- $\pi = (x_0, x_1, x_3)$ *is not a shortest path from* x_0 *to* x_3 *(*$L(\pi) = 8$*)*
- $\pi = (x_0, x_1, x_4, x_3)$ *is a shortest path from* x_0 *to* x_3 *(*$L(\pi) = 7$*)*
- *shortest path from* x_2 *to* x_0 *?*

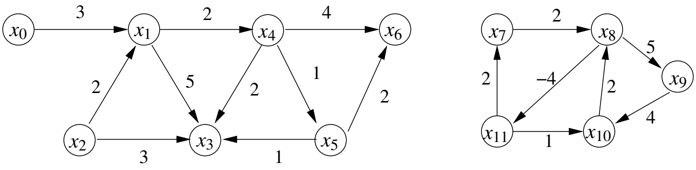
Shortest path: illustration



Example

- $\pi = (x_0, x_1, x_3)$ *is not a shortest path from* x_0 *to* x_3 *(*$L(\pi) = 8$*)*
- $\pi = (x_0, x_1, x_4, x_3)$ *is a shortest path from* x_0 *to* x_3 *(*$L(\pi) = 7$*)*
- *There is no shortest path from* x_2 *to* x_0

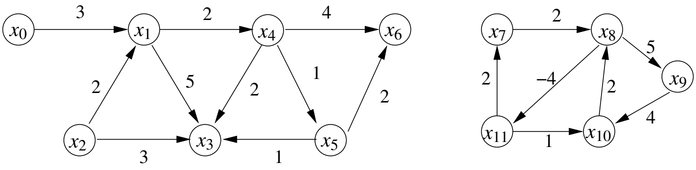
Shortest path: illustration



Example

- $\pi = (x_0, x_1, x_3)$ *is not a shortest path from* x_0 *to* x_3 *(*$L(\pi) = 8$*)*
- $\pi = (x_0, x_1, x_4, x_3)$ *is a shortest path from* x_0 *to* x_3 *(*$L(\pi) = 7$*)*
- *There is no shortest path from* x_2 *to* x_0
- *shortest path from* x_7 *to* x_9 *?*

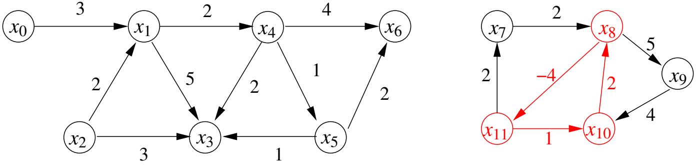
Shortest path: illustration



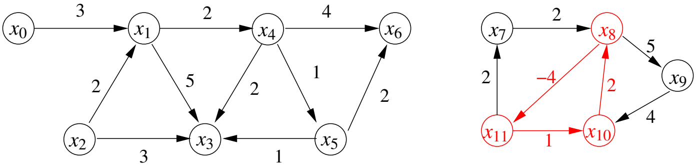
Example

- $\pi = (x_0, x_1, x_3)$ *is not a shortest path from* x_0 *to* x_3 *(*$L(\pi) = 8$*)*
- $\pi = (x_0, x_1, x_4, x_3)$ *is a shortest path from* x_0 *to* x_3 *(*$L(\pi) = 7$*)*
- *There is no shortest path from* x_2 *to* x_0
- *There is no shortest path from* x_7 *to* x_9

Negative circuit



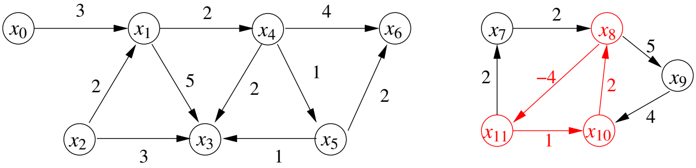
Negative circuit



Definition

- *A* negative circuit *in N is a circuit of negative length*

Negative circuit



Definition

- *A* negative circuit *in N is a circuit of negative length*

Remark. If a strongly connected component has a negative circuit, then there is no shortest path between any two arbitrary vertices of this component

Existence of a shortest path

Property

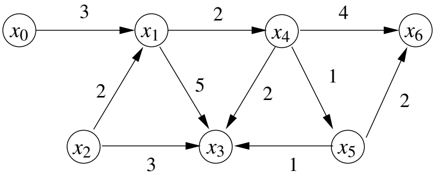
- *There exists a shortest path from x to any other vertex in E if and only if*
  - $\forall y \in E$, $\exists$ *a path from x to y*
  - *there is no negative circuit in N*

Shortest path or negative circuit?

- There exist algorithms for
  1. Finding shortest paths if they exist and
  2. Detecting if a graph has a negative circuit
- For instance, Bellman algorithm

Positive lengths network

- A *positive length network* is a network (E, Γ, ℓ) such that:
  - $\forall u \in \overrightarrow{\Gamma}$, $\ell(u) \geq 0$

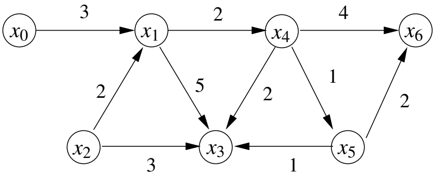


Positive lengths network

- A *positive length network* is a network (E, Γ, ℓ) such that:
  - $\forall u \in \overrightarrow{\Gamma}, \ell(u) \geq 0$

Property

- *If (E, Γ, ℓ) is a positive lengths network, then $\forall x, y \in E$*
  - *$\exists$ a path from x to y $\Leftrightarrow$ $\exists$ a shortest path from x à y*

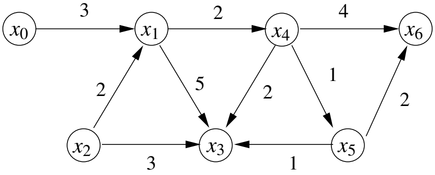


Shortest paths

- Let $N = (E, \Gamma, \ell)$ be a positive lengths network, let $x \in E$
- We define the map $L_x : E \longrightarrow \mathbb{R} \cup \{\infty\}$ by:

$$L_x(y) = \begin{cases} \text{the length of a shortest path from } x \text{ to } y, \text{ if such path exists} \\ \infty \text{ , otherwise} \end{cases}$$

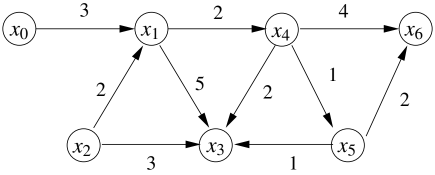
Illustration: the map L_x



Example

$y =$	x_0	x_1	x_2	x_3	x_4	x_5	x_6
$L_{x_0}(y) =$							

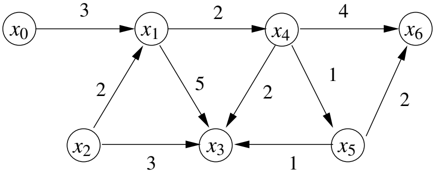
Illustration: the map L_x



Example

$y =$	x_0	x_1	x_2	x_3	x_4	x_5	x_6
$L_{x_0}(y) =$	0						

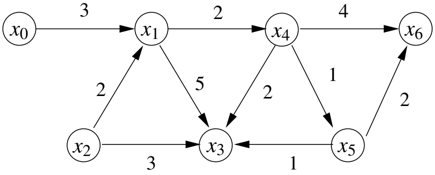
Illustration: the map L_x



Example

$y =$	x_0	x_1	x_2	x_3	x_4	x_5	x_6
$L_{x_0}(y) =$	0	3					

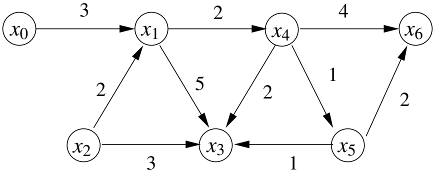
Illustration: the map L_x



Example

$y =$	x_0	x_1	x_2	x_3	x_4	x_5	x_6
$L_{x_0}(y) =$	0	3	∞				

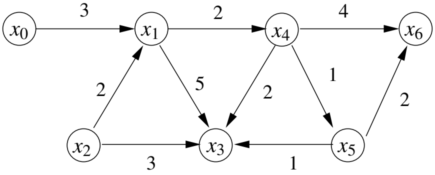
Illustration: the map L_x



Example

$y =$	x_0	x_1	x_2	x_3	x_4	x_5	x_6
$L_{x_0}(y) =$	0	3	∞	7			

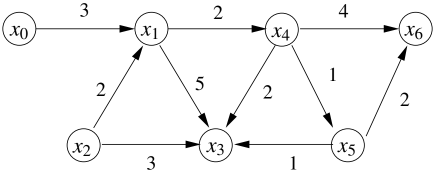
Illustration: the map L_x



Example

$y =$	x_0	x_1	x_2	x_3	x_4	x_5	x_6
$L_{x_0}(y) =$	0	3	∞	7	5		

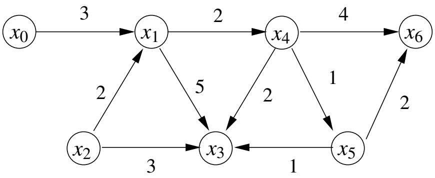
Illustration: the map L_x



Example

$y =$	x_0	x_1	x_2	x_3	x_4	x_5	x_6
$L_{x_0}(y) =$	0	3	∞	7	5	6	

Illustration: the map L_x



Example

$y =$	x_0	x_1	x_2	x_3	x_4	x_5	x_6
$L_{x_0}(y) =$	0	3	∞	7	5	6	8

Problems

1. Given a network (E, Γ, ℓ) and two vertices x and y in E
   - Find a shortest path from x to y
   - Find the length $L_x(y)$ of a shortest path from x to y
2. Given a network (E, Γ, ℓ) and a vertex x in E
   - Find for each vertex y in E the length $L_x(y)$ of a shortest path from x to z
3. Given a network (E, Γ, ℓ)
   - Find, for each pair x, y of vertices in E, the length of a shortest path from x to y
4. Having solved problem 2
   - Solve problem 1

Dijkstra algorithm

1. Given a network (E, Γ, ℓ) and two vertices x and y in E
   - Find a shortest path from x to y
   - Find the length $L_x(y)$ of a shortest path from x to y
2. Given a network (E, Γ, ℓ) and a vertex x in E
   - Find for each vertex y in E the length $L_x(y)$ of a shortest path from x to z
3. Given a network (E, Γ, ℓ)
   - Find, for each pair x, y of vertices in E, the length of a shortest path from x to y
4. Having solved problem 2
   - Solve problem 1

Computing the lengths of shortest paths

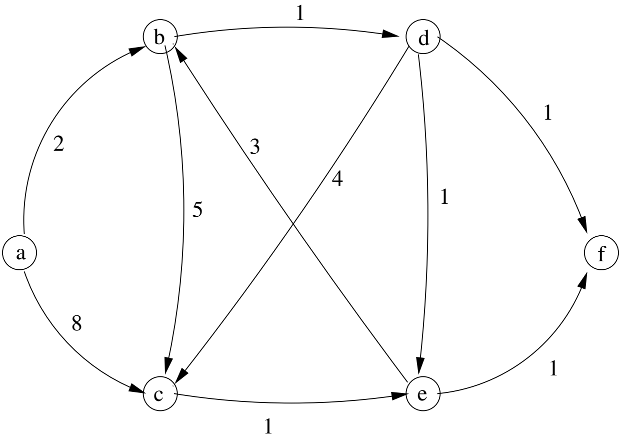
Algorithm DIJKSTRA (Data: (E, Γ, ℓ), $n = |E|$, $x \in E$;
Result: L_x)

$\overline{S} := \emptyset$;
For each $y \in E$ **Do** $L_x[y] = \infty$; $\overline{S} := \overline{S} \cup \{y\}$;
$L_x[x] := 0$; $k := 0$; $\mu := 0$;
While $k < n$ and $\mu \neq \infty$ **Do**

- Extract a vertex $y^\star \in \overline{S}$ such that $L_x[y^\star] = \min\{L_x[y], y \in \overline{S}\}$
- $k++$; $\mu := L_x[y^\star]$;
- **For each** $y \in \Gamma(y^\star) \cap \overline{S}$ **Do**
  - $L_x[y] := \min\{L_x[y], L_x[y^\star] + \ell(y^\star, y)\}$;

Computing the lengths of shortest paths

- *Exercise.* Execute "by hand" Dijsktra algorithm on the following network with $x = a$, and on any positive length network of your choice



Loop invariant of Dijkstra algorithm (# 1)

- Let $x \in E$ and $\mu \in \mathbb{R}$
- A subset S of E is called a *μ-separating (for x)* if the two following conditions hold true:

Loop invariant of Dijkstra algorithm (# 1)

- Let $x \in E$ and $\mu \in \mathbb{R}$
- A subset S of E is called a *μ-separating (for x)* if the two following conditions hold true:
  1. S contains any vertex y such that the length $L_x(y)$ of a shortest path from x to y is less than μ

Loop invariant of Dijkstra algorithm (# 1)

- Let $x \in E$ and $\mu \in \mathbb{R}$
- A subset S of E is called a *μ-separating (for x)* if the two following conditions hold true:
  1. S contains any vertex y such that the length $L_x(y)$ of a shortest path from x to y is less than μ
  2. $\overline{S} = E \setminus S$ contains any vertex y such that the length of a shortest path from x to y is greater than μ

Loop invariant of Dijkstra algorithm (# 2)

- Let $x \in E$, let $\mu \in \mathbb{R}$, and let S be a set that is μ-separating for x
- An *S-path* is a path whose intermediary vertices are all in S

Loop invariant of Dijkstra algorithm (# 2)

- Let $x \in E$, let $\mu \in \mathbb{R}$, and let S be a set that is μ-separating for x
- An *S-path* is a path whose intermediary vertices are all in S
- The length of a shortest S-path from x to y is denoted by $L_x^S(y)$

Loop invariant of Dijkstra algorithm (# 2)

- Let $x \in E$, let $\mu \in \mathbb{R}$, and let S be a set that is μ-separating for x
- An *S-path* is a path whose intermediary vertices are all in S
- The length of a shortest S-path from x to y is denoted by $L_x^S(y)$

Property (proof of Dijkstra algorithm)

- *Let* $y^\star \in \overline{S}$ *such that* $L_x^S(y^\star) = \min\{L_x^s(y) \mid y \in \overline{S}\}$

Loop invariant of Dijkstra algorithm (# 2)

- Let $x \in E$, let $\mu \in \mathbb{R}$, and let S be a set that is μ-separating for x
- An *S-path* is a path whose intermediary vertices are all in S
- The length of a shortest S-path from x to y is denoted by $L_x^S(y)$

Property (proof of Dijkstra algorithm)

- *Let $y^\star \in \overline{S}$ such that $L_x^S(y^\star) = \min\{L_x^s(y) \mid y \in \overline{S}\}$*
- *Then, $L_x^S(y^\star) = L_x(y^\star)$*

Loop invariant of Dijkstra algorithm (# 2)

- Let $x \in E$, let $\mu \in \mathbb{R}$, and let S be a set that is μ-separating for x
- An *S-path* is a path whose intermediary vertices are all in S
- The length of a shortest S-path from x to y is denoted by $L_x^S(y)$

Property (proof of Dijkstra algorithm)

- *Let* $y^\star \in \overline{S}$ *such that* $L_x^S(y^\star) = \min\{L_x^s(y) \mid y \in \overline{S}\}$
- *Then,* $L_x^S(y^\star) = L_x(y^\star)$
- *Thus,* $S \cup \{y^\star\}$ *is a set that is* μ'*-separating with* $\mu' = L_x^S(y^\star)$

Computing the lengths of shortest paths

Algorithm DIJKSTRA (Data: (E, Γ, ℓ), $n = |E|$, $x \in E$;
Result: L_x)

$\overline{S} := \emptyset$;
For each $y \in E$ **Do** $L_x[y] = \infty$; $\overline{S} := \overline{S} \cup \{y\}$;
$L_x[x] := 0$; $k := 0$; $\mu := 0$;
While $k < n$ and $\mu \neq \infty$ **Do**

- Extract a vertex $y^\star \in \overline{S}$ such that $L_x[y^\star] = \min\{L_x[y], y \in \overline{S}\}$
- $k++$; $\mu := L_x[y^\star]$;
- **For each** $y \in \Gamma(y^\star) \cap \overline{S}$ **Do**
  - $L_x[y] := \min\{L_x[y], L_x[y^\star] + \ell(y^\star, y)\}$;

Complexity

Complexity

- *Initialization:* $O(n)$
- **While** *loop (line 4):* $O(n)$
- *Extract (line 5):* $O(n^2)$
- **For each** *loop (line 7):* $O(n + m)$
- *DIJKSTRA:* $O(n^2)$

Complexity

Complexity

- *Initialization:* $O(n)$
- **While** *loop (line 4):* $O(n)$
- *Extract (line 5):* $O(n^2)$
- **For each** *loop (line 7):* $O(n+m)$
- *DIJKSTRA:* $O(n^2)$
- *can be easily reduced to* $O(n\log(n)+m)$

Exercise

- Propose an algorithm whose **data** are:
  - a positive lengths network N
  - a pair (x, y) of vertices
- and whose **result** is:
  - a shortest path from x to y if such path exists

Help. Start by computing the lengths $L_x(z)$ for all vertices $z \in E$ using Dijkstra algorithm.