Choosing best shortcuts for a path

Martin Pečar

Jozef Stefan Institute, Ljubljana, Slovenia

Koper, 17. 6. 2015

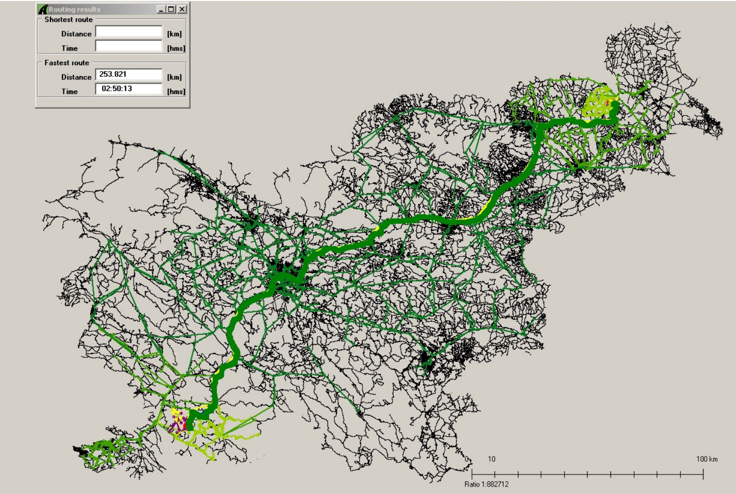
Motivation

- We want to improve the network, but are constrained (budget)
- Our goal is to improve DAGs like Contraction Hierarchies

Motivation

- We want to improve the network, but are constrained (budget)
- Our goal is to improve DAGs like Contraction Hierarchies

Motivation - example



Related problems

1. The Shortcut Problem: asks what is the best set of k additional edges (shortcuts) so that shortest paths will be preserved and the average hop length of paths will be minimal;
2. Multi-Constrained Shortest Path: find a path from s to t with lowest cost, subject to constraints;
3. Constrained Network Optimization

Related problems

1. The Shortcut Problem: asks what is the best set of k additional edges (shortcuts) so that shortest paths will be preserved and the average hop length of paths will be minimal;
2. Multi-Constrained Shortest Path: find a path from s to t with lowest cost, subject to constraints;
3. Constrained Network Optimization

Related problems

1. The Shortcut Problem: asks what is the best set of k additional edges (shortcuts) so that shortest paths will be preserved and the average hop length of paths will be minimal;
2. Multi-Constrained Shortest Path: find a path from s to t with lowest cost, subject to constraints;
3. Constrained Network Optimization

General model

Definition

Weighted multiobjective network design problem

$\mathbf{B} \in \mathbb{R}_0^{+d2}$	a vector of constraints ("budget")
$G = (V, E)$	a directed (multi)graph
$\mathbf{c} : E \mapsto \mathbb{R}_0^{+d1}$	costs
$\mathbf{b} : E \mapsto \mathbb{R}_0^{+d2}$	"building" costs
$\mathbf{w} : V \times V \mapsto \mathbb{R}_0^{+d1}$	weights

Determine $E_0 \subset E$ such that it satisfies the constrains $\mathbf{b}(E_0) \leq \mathbf{B}$ and achieves

$$\min_{E^S \subset E} \Big(\sum_{i,j} \mathbf{w}_{ij} \mathbf{c}(P_{G(V,E^S)}(i,j))\Big)$$

General model

Definition

Weighted multiobjective network design problem

$\mathbf{B} \in \mathbb{R}_0^{+d2}$	a vector of constraints ("budget")
$G = (V, E)$	a directed (multi)graph
$\mathbf{c} : E \mapsto \mathbb{R}_0^{+d1}$	costs
$\mathbf{b} : E \mapsto \mathbb{R}_0^{+d2}$	"building" costs
$\mathbf{w} : V \times V \mapsto \mathbb{R}_0^{+d1}$	weights

Determine $E_0 \subset E$ such that it satisfies the constrains $\mathbf{b}(E_0) \leq \mathbf{B}$ and achieves

$$\min_{E^S \subset E} \left(\sum_{i,j} \mathbf{w}_{ij} \mathbf{c}(P_{G(V,E^S)}(i,j)) \right)$$

General model

Definition

Weighted multiobjective network design problem

$\mathbf{B} \in \mathbb{R}_0^{+d2}$	a vector of constraints ("budget")
$G = (V, E)$	a directed (multi)graph
$\mathbf{c} : E \mapsto \mathbb{R}_0^{+d1}$	costs
$\mathbf{b} : E \mapsto \mathbb{R}_0^{+d2}$	"building" costs
$\mathbf{w} : V \times V \mapsto \mathbb{R}_0^{+d1}$	weights

Determine $E_0 \subset E$ such that it satisfies the constrains $\mathbf{b}(E_0) \leq \mathbf{B}$ and achieves

$$\min_{E^S \subset E} \Big(\sum_{i,j} \mathbf{w}_{ij} \mathbf{c}(P_{G(V,E^S)}(i,j)) \Big)$$

General model

Definition

Weighted multiobjective network design problem

$\mathbf{B} \in \mathbb{R}_0^{+d2}$	a vector of constraints (”budget”)
$G = (V, E)$	a directed (multi)graph
$\mathbf{c} : E \mapsto \mathbb{R}_0^{+d1}$	costs
$\mathbf{b} : E \mapsto \mathbb{R}_0^{+d2}$	”building” costs
$\mathbf{w} : V \times V \mapsto \mathbb{R}_0^{+d1}$	weights

Determine $E_0 \subset E$ such that it satisfies the constrains $\mathbf{b}(E_0) \leq \mathbf{B}$ and achieves

$$\min_{E^S \subset E} \left(\sum_{i,j} \mathbf{w}_{ij}\mathbf{c}(P_{G(V,E^S)}(i,j))\right)$$

General model

Definition

Weighted multiobjective network design problem

$\mathbf{B} \in \mathbb{R}_0^{+d2}$	a vector of constraints ("budget")
$G = (V, E)$	a directed (multi)graph
$\mathbf{c} : E \mapsto \mathbb{R}_0^{+d1}$	costs
$\mathbf{b} : E \mapsto \mathbb{R}_0^{+d2}$	"building" costs
$\mathbf{w} : V \times V \mapsto \mathbb{R}_0^{+d1}$	weights

Determine $E_0 \subset E$ such that it satisfies the constrains $\mathbf{b}(E_0) \leq \mathbf{B}$ and achieves

$$\min_{E^S \subset E} \left(\sum_{i,j} \mathbf{w}_{ij} \mathbf{c}(P_{G(V,E^S)}(i,j)) \right)$$

General model

Definition

Weighted multiobjective network design problem

$\mathbf{B} \in \mathbb{R}_0^{+d2}$	a vector of constraints (”budget”)
$G = (V, E)$	a directed (multi)graph
$\mathbf{c} : E \mapsto \mathbb{R}_0^{+d1}$	costs
$\mathbf{b} : E \mapsto \mathbb{R}_0^{+d2}$	”building” costs
$\mathbf{w} : V \times V \mapsto \mathbb{R}_0^{+d1}$	weights

Determine $E_0 \subset E$ such that it satisfies the constrains $\mathbf{b}(E_0) \leq \mathbf{B}$ and achieves

$$\min_{E^S \subset E} (\sum_{i,j} \mathbf{w}_{ij} \mathbf{c}(P_{G(V,E^S)}(i,j)))$$

Simple problem

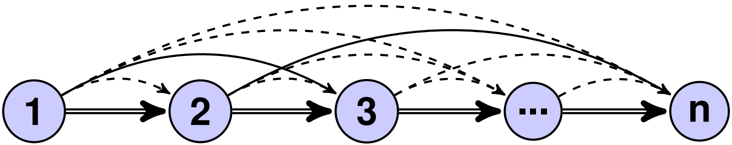
Consider a path connecting some cities. We have a budget which allows construction of a few new road sections.

Dashed shortcuts are not allowed ($b(e_s) = Inf$)

All arcs point to a vertex with higher index

Simple problem

Consider a path connecting some cities. We have a budget which allows construction of a few new road sections.

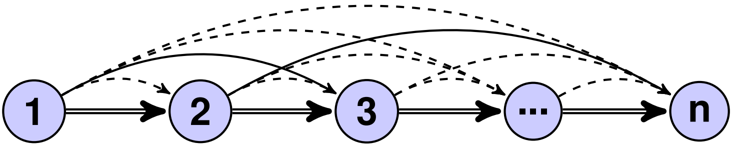


Dashed shortcuts are not allowed ($b(e_s) = Inf$)

All arcs point to a vertex with higher index

Simple problem

Consider a path connecting some cities. We have a budget which allows construction of a few new road sections.

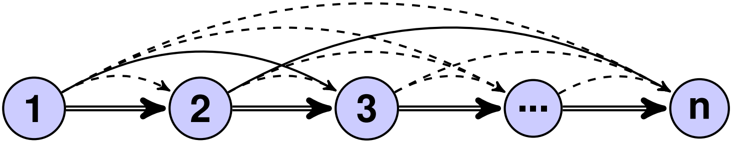


Dashed shortcuts are not allowed ($b(e_s) = Inf$)

All arcs point to a vertex with higher index

Simple problem

Consider a path connecting some cities. We have a budget which allows construction of a few new road sections.



Dashed shortcuts are not allowed ($b(e_s) = \mathit{Inf}$)

All arcs point to a vertex with higher index

Variants of the problem

Cost of using a shortcut ($b(e_s)$):

- $b(e_s) = 1$ for all allowed shortcuts,
- $b(e_s)$ can be different.

Weights (w_{ij}):

- $w_{1n} = 1$, other $w_{ij} = 0 \Rightarrow$ constrained shortest path;
- several non-zero weights $\Rightarrow$ constrained network optimization.

Variants of the problem

Cost of using a shortcut ($b(e_s)$):

- $b(e_s) = 1$ for all allowed shortcuts,
- $b(e_s)$ can be different.

Weights (w_{ij}):

- $w_{1n} = 1$, other $w_{ij} = 0 \Rightarrow$ constrained shortest path;
- several non-zero weights $\Rightarrow$ constrained network optimization.

Variants of the problem

Cost of using a shortcut ($b(e_s)$):

- $b(e_s) = 1$ for all allowed shortcuts,
- $b(e_s)$ can be different.

Weights (w_{ij}):

- $w_{1n} = 1$, other $w_{ij} = 0 \Rightarrow$ constrained shortest path;
- several non-zero weights $\Rightarrow$ constrained network optimization.

Variants of the problem

Cost of using a shortcut ($b(e_s)$):

- $b(e_s) = 1$ for all allowed shortcuts,
- $b(e_s)$ can be different.

Weights (w_{ij}):

- $w_{1n} = 1$, other $w_{ij} = 0 \Rightarrow$ constrained shortest path;
- several non-zero weights $\Rightarrow$ constrained network optimization.

Variants of the problem

Cost of using a shortcut ($b(e_s)$):

- $b(e_s) = 1$ for all allowed shortcuts,
- $b(e_s)$ can be different.

Weights (w_{ij}):

- $w_{1n} = 1$, other $w_{ij} = 0 \Rightarrow$ constrained shortest path;
- several non-zero weights $\Rightarrow$ constrained network optimization.

Variants of the problem

Cost of using a shortcut ($b(e_s)$):

- $b(e_s) = 1$ for all allowed shortcuts,
- $b(e_s)$ can be different.

Weights (w_{ij}):

- $w_{1n} = 1$, other $w_{ij} = 0 \Rightarrow$ constrained shortest path;
- several non-zero weights $\Rightarrow$ constrained network optimization.

Variants of the problem

Cost of using a shortcut ($b(e_s)$):

- $b(e_s) = 1$ for all allowed shortcuts,
- $b(e_s)$ can be different.

Weights (w_{ij}):

- $w_{1n} = 1$, other $w_{ij} = 0 \Rightarrow$ constrained shortest path;
- several non-zero weights $\Rightarrow$ constrained network optimization.

All subpaths distances

Subpath travelling costs

$$\begin{array}{cccc} c_{45} & & & \\ c_{34} & c_{35} & & \\ c_{23} & c_{24} & c_{25} & \\ c_{12} & c_{13} & c_{14} & c_{15} \end{array}$$

$$c_{ij} = c_{ij-1} + c_{j-1j}$$

All subpaths distances

Subpath travelling costs

$$
\begin{array}{llll}
c_{45} & & & \\
c_{34} & c_{35} & & \\
c_{23} & c_{24} & c_{25} & \\
c_{12} & c_{13} & c_{14} & c_{15}
\end{array}
$$

$$c_{ij} = c_{ij-1} + c_{j-1j}$$

$b = 1$ Constrained shortest path

$$c(1, n, B) = \min_{i,k}(c(1, i, k) + c(i, n, B - k))$$

$k = \lfloor B \rfloor$

Algorithm 1 Dynamic Programming CSP

```
for l = 0 to k do
    for i = 1 to n do
        for j = i to n do
            c(i, j, l) = min_{v,s}(c(i, v, s) + c(v, j, l - s))
            c(i, j, l) = min(c(i, j, l), c_{ij}^s)
        end for
    end for
end for
```

Complexity is $O(k^2 n^3)$

$b = 1$ Constrained shortest path

$$c(1, n, B) = \min_{i,k}(c(1, i, k) + c(i, n, B - k))$$

$k = \lfloor B \rfloor$

Algorithm 2 Dynamic Programming CSP

for $l = 0$ to k **do**
    for $i = 1$ to n **do**
        for $j = i$ to n **do**
            $c(i, j, l) = \min_{v,s}(c(i, v, s) + c(v, j, l - s))$
            $c(i, j, l) = \min(c(i, j, l), c_{ij}^{s})$
        end for
    end for
end for

Complexity is $O(k^2 n^3)$

$b = 1$ Constrained shortest path

$$c(1, n, B) = \min_{i,k}(c(1, i, k) + c(i, n, B - k))$$

$k = \lfloor B \rfloor$

Algorithm 3 Dynamic Programming CSP

```
for l = 0 to k do
    for i = 1 to n do
        for j = i to n do
            c(i, j, l) = min_{v,s}(c(i, v, s) + c(v, j, l − s))
            c(i, j, l) = min(c(i, j, l), c^s_{ij})
        end for
    end for
end for
```

Complexity is $O(k^2 n^3)$

$b > 0$ Constrained shortest path

Even the simplest form (each arc replaced by a shortcut) is NP-complete, can be translated to knapsack problem

$M = \max_i c_{ii+1}$

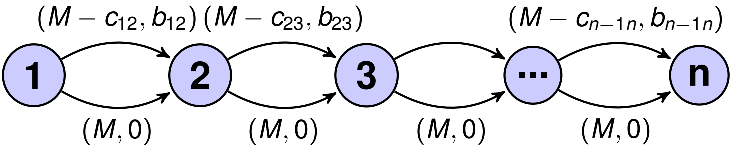
$b > 0$ Constrained shortest path

Even the simplest form (each arc replaced by a shortcut) is NP-complete, can be translated to knapsack problem

$M = \max_i c_{ii+1}$

$b > 0$ Constrained shortest path

Even the simplest form (each arc replaced by a shortcut) is NP-complete, can be translated to knapsack problem



$M = \max_i c_{ii+1}$

Observations

The set of used shortcuts uniquely defines the resulting shortest path.

Two shortcuts are *compatible*, if (interiors of) the intervals of the indexes of the end vertices are not overlapping. The shortest path can only contain compatible edges.

Example: $(5, 9)$ and $(7, 12)$ are not compatible, while $(5, 9)$ and $(9, 12)$ are.

Observations

The set of used shortcuts uniquely defines the resulting shortest path.

Two shortcuts are *compatible*, if (interiors of) the intervals of the indexes of the end vertices are not overlapping. The shortest path can only contain compatible edges.

Example: $(5, 9)$ and $(7, 12)$ are not compatible, while $(5, 9)$ and $(9, 12)$ are.

Shortcut gain

Gain of using a shortcut is

$$g(e^s_{ij}) = (\sum_{l=i}^{j-1} c_{ll+1}) - c_{ij}$$

Gain of a compatible set of shortcuts is S_C is

$$g(S_C) = \sum_{e_s \in S_C} g(e_s)$$

Shortcut gain

Gain of using a shortcut is

$$g(e_{ij}^{s}) = (\sum_{l=i}^{j-1} c_{ll+1}) - c_{ij}$$

Gain of a compatible set of shortcuts is S_C is

$$g(S_C) = \sum_{e_s \in S_C} g(e_s)$$

Optimal gain

Lemma

If the sets are compatible, then
$g(S_O) \geq g(S_S) \Rightarrow c(P_{S_o}) \leq c(P_{S_s})$.

$\overline{P_{S_s}}$ are edges from P which are skipped in P_{S_s}.
$c(P_{S_s}) = c(P) + c(S_S) - c(\overline{P_{S_s}}) = c(P) - g(S_S)$
$c(P_{S_o}) = c(P) + c(S_O) - c(\overline{P_{S_o}}) = c(P) - g(S_O)$
$g(S_O) \geq g(S_S) \Rightarrow c(P_{S_o}) \leq c(P_{S_s})$

We are looking for the compatible set of shortcuts with the biggest gain.

Optimal gain

Lemma

If the sets are compatible, then
$g(S_O) \geq g(S_S) \Rightarrow c(P_{S_o}) \leq c(P_{S_s})$.

$\overline{P_{S_s}}$ are edges from P which are skipped in P_{S_s}.
$c(P_{S_s}) = c(P) + c(S_S) - c(\overline{P_{S_s}}) = c(P) - g(S_S)$
$c(P_{S_o}) = c(P) + c(S_O) - c(\overline{P_{S_o}}) = c(P) - g(S_O)$
$g(S_O) \geq g(S_S) \Rightarrow c(P_{S_o}) \leq c(P_{S_s})$

We are looking for the compatible set of shortcuts with the biggest gain.

Optimal gain

Lemma

If the sets are compatible, then
$g(S_O) \geq g(S_S) \Rightarrow c(P_{S_o}) \leq c(P_{S_s})$.

$\overline{P_{S_s}}$ are edges from P which are skipped in P_{S_s}.

$c(P_{S_s}) = c(P) + c(S_S) - c(\overline{P_{S_s}}) = c(P) - g(S_S)$

$c(P_{S_o}) = c(P) + c(S_O) - c(\overline{P_{S_o}}) = c(P) - g(S_O)$

$g(S_O) \geq g(S_S) \Rightarrow c(P_{S_o}) \leq c(P_{S_s})$

We are looking for the compatible set of shortcuts with the biggest gain.

Optimal gain

Lemma

If the sets are compatible, then
$g(S_O) \geq g(S_S) \Rightarrow c(P_{S_o}) \leq c(P_{S_s})$.

$\overline{P_{S_s}}$ are edges from P which are skipped in P_{S_s}.
$c(P_{S_s}) = c(P) + c(S_S) - c(\overline{P_{S_s}}) = c(P) - g(S_S)$
$c(P_{S_o}) = c(P) + c(S_O) - c(\overline{P_{S_o}}) = c(P) - g(S_O)$
$g(S_O) \geq g(S_S) \Rightarrow c(P_{S_o}) \leq c(P_{S_s})$

We are looking for the compatible set of shortcuts with the biggest gain.

Optimal gain

Lemma

If the sets are compatible, then
$g(S_O) \geq g(S_S) \Rightarrow c(P_{S_O}) \leq c(P_{S_S})$.

$\overline{P_{S_S}}$ are edges from P which are skipped in P_{S_S}.
$c(P_{S_S}) = c(P) + c(S_S) - c(\overline{P_{S_S}}) = c(P) - g(S_S)$
$c(P_{S_O}) = c(P) + c(S_O) - c(\overline{P_{S_O}}) = c(P) - g(S_O)$
$g(S_O) \geq g(S_S) \Rightarrow c(P_{S_O}) \leq c(P_{S_S})$

We are looking for the compatible set of shortcuts with the biggest gain.

Optimal gain

Lemma

If the sets are compatible, then
$g(S_O) \geq g(S_S) \Rightarrow c(P_{S_o}) \leq c(P_{S_s})$.

$\overline{P_{S_s}}$ are edges from P which are skipped in P_{S_s}.
$c(P_{S_s}) = c(P) + c(S_S) - c(\overline{P_{S_s}}) = c(P) - g(S_S)$
$c(P_{S_o}) = c(P) + c(S_O) - c(\overline{P_{S_o}}) = c(P) - g(S_O)$
$g(S_O) \geq g(S_S) \Rightarrow c(P_{S_o}) \leq c(P_{S_s})$

We are looking for the compatible set of shortcuts with the biggest gain.

$b > 0$ CSP - Greedy approach

Algorithm 4 Greedy CSP

```
Assign each shortcut e_S its normalized gain g_N(e_s)
Sort shortcuts S by their normalized gain (descending)
S_O = {}, i = 1
while b(S_O) < B do
    if S(i) compatible with S_O then
        S_O = S_O ∪ S(i)
    end if
    i++
end while
for j = 1 to m do
    if g(S(j)) > g(S_O) then
        S_O = S(j)
    end if
end for
```

$b > 0$ CSP - Greedy approach

Algorithm 5 Greedy CSP

Assign each shortcut e_S its normalized gain $g_N(e_s)$
Sort shortcuts S by their normalized gain (descending)
$S_O = \{\}, i = 1$
while $b(S_O) < B$ **do**
    if $S(i)$ compatible with S_O **then**
        $S_O = S_O \cup S(i)$
    end if
    i++
end while
for $j = 1$ to m **do**
    if $g(S(j)) > g(S_O)$ **then**
        $S_O = S(j)$
    end if
end for

$b > 0$ CSP - Greedy approach

Algorithm 6 Greedy CSP

Assign each shortcut e_S its normalized gain $g_N(e_s)$
Sort shortcuts S by their normalized gain (descending)
$S_O = \{\}, i = 1$
while $b(S_O) < B$ **do**
  if $S(i)$ compatible with S_O **then**
    $S_O = S_O \cup S(i)$
  end if
  i++
end while
for $j = 1$ to m **do**
  if $g(S(j)) > g(S_O)$ **then**
    $S_O = S(j)$
  end if
end for

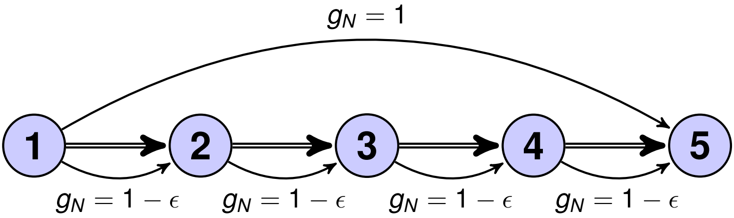
Counterexample

2-approximation?

$2\frac{B}{b_{max}}$-approximation

Counterexample

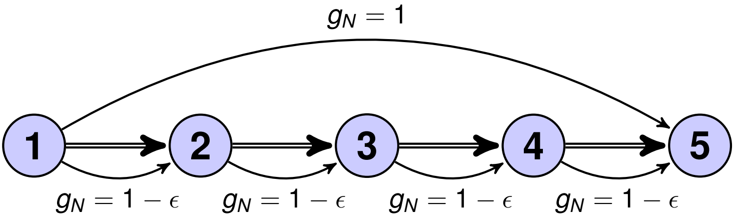
2-approximation?



$2\frac{B}{b_{max}}$-approximation

Counterexample

2-approximation?



$2\frac{B}{b_{max}}$-approximation

Conclusions

- Weighted multiobjective CNO was defined
- Special cases of CSP were discussed

Further work - find effective solutions for wider classes of graphs

Conclusions

- Weighted multiobjective CNO was defined
- Special cases of CSP were discussed

Further work - find effective solutions for wider classes of graphs

Composition of problems

CSP is composed of 2 problems:

- find shortest path
- find path, cheaper than B

Both are in P, but the composed problem is NP-hard

Open problem

What can we tell about the complexity of the composed problem, based on complexity of the subproblems and the type of composition?

Questions

Questions, comments, ideas, $\cdots$?