Exact and approximate solutions for the Constrained Shortest Path Tour Problem

D. Ferone[1], **P. Festa**[1] and **F. Guerriero**[2]

September 7, 2017

[1]Dep. of Mathematics and Applications, University of Napoli, Federico II
[2]Dep. of Mechanical, Energetic and Management Engineering, University of Calabria

Overview

1. Introduction
2. Computational complexity
3. Mathematical formulation
4. Exact approach
5. Heuristic approach
6. Some experimental results

Introduction

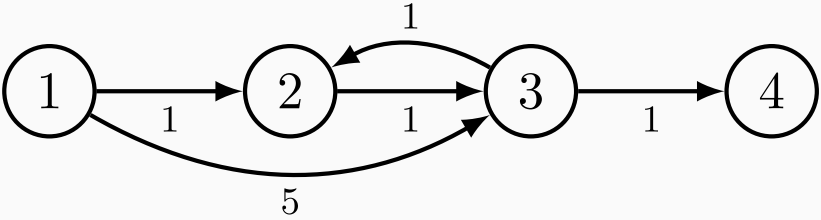
The Shortest Path Tour Problem

Let be $G = (V, A)$ be a directed graph, where

- V is a set of n nodes;
- A is a set of m arcs;
- C: $A \to \mathbb{R}^+ \cup \{0\}$

Let $T_1, \ldots, T_N$ be disjoint subsets of V. The Shortest Path Tour Problem (SPTP) consists in finding a single-origin single-destination shortest path by ensuring that at least one node of each node subset $T_1, \ldots, T_N$ is involved according to the sequence wherewith the subsets are ordered.

The Shortest Path Tour Problem

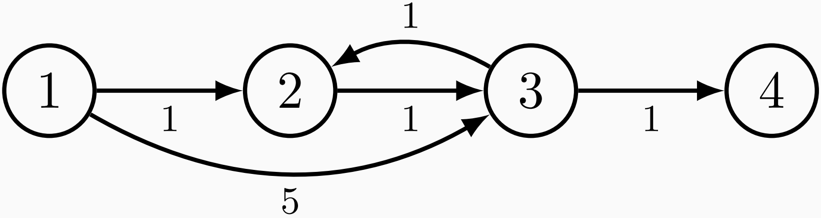


$$T_1 = \{1\}, T_2 = \{3\}, T_3 = \{2\}, T_4 = \{4\}$$

$$P_T = \{1, 2, 3, 2, 3, 4\} \quad c(P_T) = 5$$

The Constrained Shortest Path Tour Problem

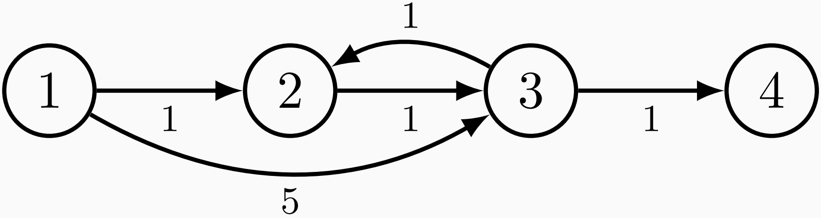
Additional constraint: the path does not include repeated arcs.



$T_1 = \{1\}, T_2 = \{3\}, T_3 = \{2\}, T_4 = \{4\}$

The Constrained Shortest Path Tour Problem

Additional constraint: the path does not include repeated arcs.



$T_1 = \{1\}, T_2 = \{3\}, T_3 = \{2\}, T_4 = \{4\}$

$P_T = \{1, 3, 2, 3, 4\} \quad c(P_T) = 8$

Complexity

Hardness

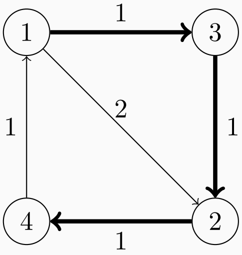
Theorem

*The CSPTP is **NP**-hard.*

Hardness

Theorem

*The CSPTP is **NP**-hard.*

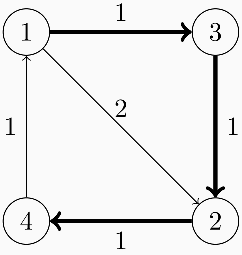


Ham-Path

Hardness

Theorem

*The CSPTP is **NP**-hard.*



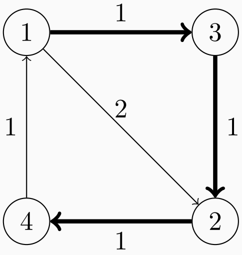
Ham-Path

- for each node $i \in V$,

Hardness

Theorem

*The CSPTP is **NP**-hard.*



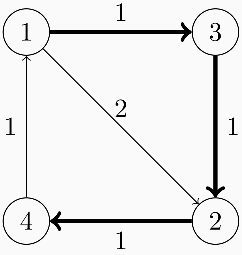
Ham-Path

- for each node $i \in V$,
    - insert in V' nodes i^- and i^+;

Hardness

Theorem

The CSPTP is ***NP****-hard.*



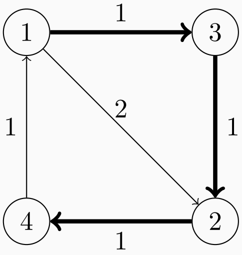
Ham-Path

- for each node $i \in V$,
  - insert in V' nodes i^- and i^+;
  - insert in A' arc (i^-, i^+) with cost 0;

Hardness

Theorem

The CSPTP is ***NP****-hard.*



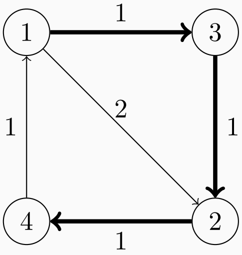
Ham-Path

- for each node $i \in V$,
  - insert in V' nodes i^- and i^+;
  - insert in A' arc (i^-, i^+) with cost 0;
- for each arc $(i, j) \in A$ and for each $k = 2, \ldots, n$,

Hardness

Theorem

*The CSPTP is **NP**-hard.*



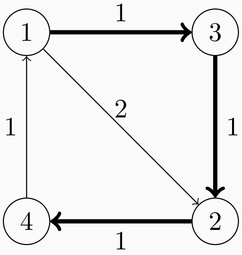
Ham-Path

- for each node $i \in V$,
  - insert in V' nodes i^- and i^+;
  - insert in A' arc (i^-, i^+) with cost 0;
- for each arc $(i, j) \in A$ and for each $k = 2, \ldots, n$,
  - insert in V' node ij^k;

Hardness

Theorem

*The CSPTP is **NP**-hard.*



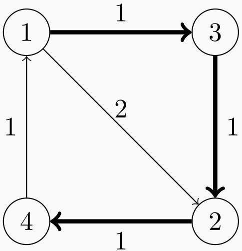
Ham-Path

- for each node $i \in V$,
  - insert in V' nodes i^- and i^+;
  - insert in A' arc (i^-, i^+) with cost 0;
- for each arc $(i, j) \in A$ and for each $k = 2, \ldots, n$,
  - insert in V' node ij^k;
  - insert in T_k node ij^k;

Hardness

Theorem

*The CSPTP is **NP**-hard.*



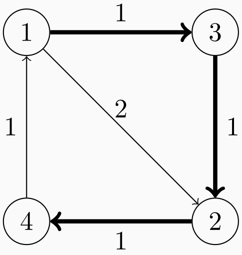
Ham-Path

- for each node $i \in V$,
  - insert in V' nodes i^- and i^+;
  - insert in A' arc (i^-, i^+) with cost 0;
- for each arc $(i, j) \in A$ and for each $k = 2, \ldots, n$,
  - insert in V' node ij^k;
  - insert in T_k node ij^k;
  - insert in A' arc (i^+, ij^k) with cost c_{ij} and arc (ij^k, j^-) with cost 0;

Hardness

Theorem

*The CSPTP is **NP**-hard.*



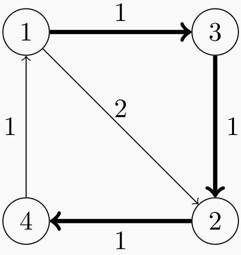
Ham-Path

- for each node $i \in V$,
  - insert in V' nodes i^- and i^+;
  - insert in A' arc (i^-, i^+) with cost 0;
- for each arc $(i, j) \in A$ and for each $k = 2, \ldots, n$,
  - insert in V' node ij^k;
  - insert in T_k node ij^k;
  - insert in A' arc (i^+, ij^k) with cost c_{ij} and arc (ij^k, j^-) with cost 0;
- set $T_1 = \{s^-\}$ and $T_{n+1} = \{d^+\}$.

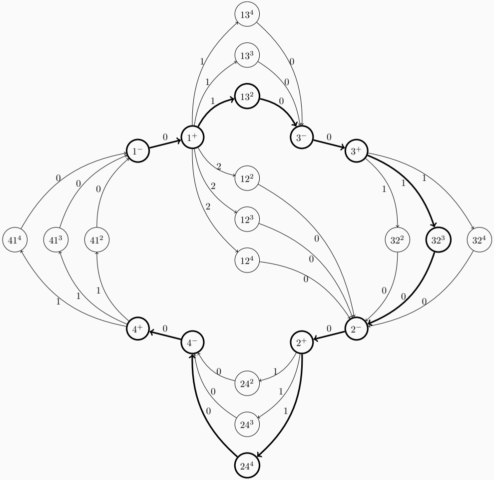
Hardness

Theorem

*The CSPTP is **NP**-hard.*



Ham-Path



Path existence lemma

Lemma

There exists a path $P = i_1, i_2, \ldots, i_k$, $k \leq n$, *in*

$$\langle G = (V, A, C), s, d \rangle ,$$

if and only if in

$$\left\langle G' = (V', A', C'), s^-, d^+, \{T_h\}_{h=1,\ldots,n+1} \right\rangle$$

there exists a path P' *from* i_1^- *to* i_k^+*, such that*

$$P' = \left\{ \bigoplus_{l=1}^{k-1} \left(i_l^-, i_l^+, {i_l i_{l+1}}^{l+1} \right), i_k^-, i_k^+ \right\}.$$

Path existence lemma (proof)

$\Rightarrow$ Exists in G a path $P = \{i_1, i_2, \ldots, i_k\}$, $k \leq n$.

Path existence lemma (proof)

$\Rightarrow$ Exists in G a path $P = \{i_1, i_2, \ldots, i_k\}$, $k \leq n$.

- by construction $(i_l^-, i_l^+) \in A'$ for each $l = 1, \ldots, k$;

Path existence lemma (proof)

$\Rightarrow$ Exists in G a path $P = \{i_1, i_2, \ldots, i_k\}$, $k \leq n$.

- by construction $(i_l^-, i_l^+) \in A'$ for each $l = 1, \ldots, k$;
- $(i_l^+, {i_l i_{l+1}}^q)$, $({i_l i_{l+1}}^q, i_{l+1}^-)$ for each $q = 2, \ldots, n$;

Path existence lemma (proof)

$\Rightarrow$ Exists in G a path $P = \{i_1, i_2, \ldots, i_k\}$, $k \leq n$.

- by construction $(i_l^-, i_l^+) \in A'$ for each $l = 1, \ldots, k$;
- $(i_l^+, {i_l i_{l+1}}^q)$, $({i_l i_{l+1}}^q, i_{l+1}^-)$ for each $q = 2, \ldots, n$;

$\Leftarrow$ exists P', P is not present.

Path existence lemma (proof)

$\Rightarrow$ Exists in G a path $P = \{i_1, i_2, \ldots, i_k\}$, $k \leq n$.

- by construction $(i_l^-, i_l^+) \in A'$ for each $l = 1, \ldots, k$;
- $(i_l^+, {i_l i_{l+1}}^q)$, $({i_l i_{l+1}}^q, i_{l+1}^-)$ for each $q = 2, \ldots, n$;

$\Leftarrow$ exists P', P is not present.

- $i_l \notin V$: i_l^- and i_l^+ would not be in V';

Path existence lemma (proof)

$\Rightarrow$ Exists in G a path $P = \{i_1, i_2, \ldots, i_k\}$, $k \leq n$.

- by construction $(i_l^-, i_l^+) \in A'$ for each $l = 1, \ldots, k$;
- $(i_l^+, {i_l i_{l+1}}^q)$, $({i_l i_{l+1}}^q, i_{l+1}^-)$ for each $q = 2, \ldots, n$;

$\Leftarrow$ exists P', P is not present.

- $i_l \notin V$: i_l^- and i_l^+ would not be in V';
- $(i_l, i_{l+1}) \notin A$, then arcs $(i_l^+, {i_l i_{l+1}}^{l+1})$ and $({i_l i_{l+1}}^{l+1}, i_{l+1}^-)$ would not be in A'.

Reduction theorem

Theorem (Ham-Path $<_m^p$ CSPTP)
There exists in G an Hamiltonian path P from s to d with length $L(P)$ if and only if there exists in G' a constrained path tour P' from s^- to d^+ with length $L(P') = L(P)$.

Reduction theorem

$\Rightarrow$ By hypothesis, there exists in G an Hamiltonian path $P = \{ i_1, i_2, \ldots, i_n \}$, where $i_1 = s$ and $i_n = d$.

Reduction theorem

$\Rightarrow$ By hypothesis, there exists in G an Hamiltonian path $P = \{\, i_1, i_2, \ldots, i_n \,\}$, where $i_1 = s$ and $i_n = d$.

- Exists $P' = \left\{ \bigoplus_{l=1}^{k-1} \left(i_l^-, i_l^+, {i_l i_{l+1}}^{l+1} \right), i_k^-, i_k^+ \right\}$;

Reduction theorem

$\Rightarrow$ By hypothesis, there exists in G an Hamiltonian path $P = \{\, i_1, i_2, \ldots, i_n \,\}$, where $i_1 = s$ and $i_n = d$.

- Exists $P' = \left\{ \bigoplus_{l=1}^{k-1} \left(i_l^-, i_l^+, i_l i_{l+1}{}^{l+1} \right), i_k^-, i_k^+ \right\}$;
- If P' crosses some arcs more than once: it must be (i^-, i^+);

Reduction theorem

$\Rightarrow$ By hypothesis, there exists in G an Hamiltonian path $P = \{\, i_1, i_2, \ldots, i_n \,\}$, where $i_1 = s$ and $i_n = d$.

- Exists $P' = \left\{ \bigoplus_{l=1}^{k-1} \left(i_l^-, i_l^+, i_l i_{l+1}{}^{l+1} \right), i_k^-, i_k^+ \right\}$;
- If P' crosses some arcs more than once: it must be (i^-, i^+);
- it involves successively and sequentially all sets T_i by construction.

Reduction theorem

$\Rightarrow$ By hypothesis, there exists in G an Hamiltonian path $P = \{ i_1, i_2, \ldots, i_n \}$, where $i_1 = s$ and $i_n = d$.

- Exists $P' = \left\{ \bigoplus_{l=1}^{k-1} \left(i_l^-, i_l^+, i_l i_{l+1}{}^{l+1} \right), i_k^-, i_k^+ \right\}$;
- If P' crosses some arcs more than once: it must be (i^-, i^+);
- it involves successively and sequentially all sets T_i by construction.

$\Leftarrow$ By construction the feasible path tour must be as P', it implies exists a path $P = \{ i_1, \ldots, i_n \}$ in G.

Reduction theorem

$\Rightarrow$ By hypothesis, there exists in G an Hamiltonian path $P = \{\ i_1, i_2, \ldots, i_n\ \}$, where $i_1 = s$ and $i_n = d$.

- Exists $P' = \left\{ \bigoplus_{l=1}^{k-1} \left(i_l^-, i_l^+, {i_l i_{l+1}}^{l+1} \right), i_k^-, i_k^+ \right\}$;
- If P' crosses some arcs more than once: it must be (i^-, i^+);
- it involves successively and sequentially all sets T_i by construction.

$\Leftarrow$ By construction the feasible path tour must be as P', it implies exists a path $P = \{\ i_1, \ldots, i_n\ \}$ in G.

- Suppose P is not Hamiltonian;

Reduction theorem

$\Rightarrow$ By hypothesis, there exists in G an Hamiltonian path $P = \{ i_1, i_2, \ldots, i_n \}$, where $i_1 = s$ and $i_n = d$.

- Exists $P' = \left\{ \bigoplus_{l=1}^{k-1} \left(i_l^-, i_l^+, {i_l i_{l+1}}^{l+1} \right), i_k^-, i_k^+ \right\}$;
- If P' crosses some arcs more than once: it must be (i^-, i^+);
- it involves successively and sequentially all sets T_i by construction.

$\Leftarrow$ By construction the feasible path tour must be as P', it implies exists a path $P = \{ i_1, \ldots, i_n \}$ in G.

- Suppose P is not Hamiltonian;
- There exist $i_k = i_j$, where $k \neq j$;

Reduction theorem

$\Rightarrow$ By hypothesis, there exists in G an Hamiltonian path $P = \{\, i_1, i_2, \ldots, i_n \,\}$, where $i_1 = s$ and $i_n = d$.

- Exists $P' = \left\{ \bigoplus_{l=1}^{k-1} \left(i_l^-, i_l^+, i_l i_{l+1}{}^{l+1} \right), i_k^-, i_k^+ \right\}$;
- If P' crosses some arcs more than once: it must be (i^-, i^+);
- it involves successively and sequentially all sets T_i by construction.

$\Leftarrow$ By construction the feasible path tour must be as P', it implies exists a path $P = \{\, i_1, \ldots, i_n \,\}$ in G.

- Suppose P is not Hamiltonian;
- There exist $i_k = i_j$, where $k \neq j$;
- P' must cross twice the arc $(i_k^-, i_k^+) = (i_j^-, i_j^+)$.

Mathematical model

Mathematical model

$$\min \sum_{(i,j)\in A} \sum_{k=1}^{N-1} x_{ij}^k c_{ij}$$

$s.a.$

$$\sum_{j\in FS(i)} x_{ij}^k - \sum_{j\in BS(i)} x_{ji}^k = \begin{cases} y_i & \text{if } i \in T_k, \\ -y_i & \text{if } i \in T_{k+1}, \\ 0, & \text{otherwise.} \end{cases} \quad \forall i \in V, k = 1, \ldots, N-1$$

$$\sum_{i\in T_k} y_i = 1 \qquad \forall k = 1, \ldots, N-1$$

$$\sum_{k=1}^{N-1} x_{ij}^k \leq 1 \qquad \forall (i,j) \in A$$

$$y_s = 1, y_d = 1$$

$$x_{ij}^k \in \{0,1\}, y_i \in \{0,1\}$$

Exact approach

Main ideas

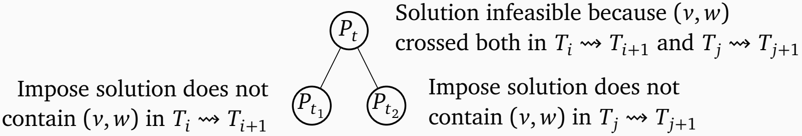
- A path tour is a concatenation of simple paths $T_i \rightsquigarrow T_{i+1}$ for $i = 1, \ldots, N-1$;

Main ideas

- A path tour is a concatenation of simple paths $T_i \rightsquigarrow T_{i+1}$ for $i = 1, \ldots, N-1$;
- an arc repetition can occur only in two different subpaths $T_i \rightsquigarrow T_{i+1}$ and $T_j \rightsquigarrow T_{j+1}$, with $i \neq j$.

Main ideas

- A path tour is a concatenation of simple paths $T_i \rightsquigarrow T_{i+1}$ for $i = 1, \ldots, N-1$;
- an arc repetition can occur only in two different subpaths $T_i \rightsquigarrow T_{i+1}$ and $T_j \rightsquigarrow T_{j+1}$, with $i \neq j$.



The B&B algorithm

Function B&B($G = \langle V, A, C \rangle, s, d, \{T_i\}_{i=1,\dots,N}$)

  ShortestPaths $\leftarrow$ $\textsc{FloydWarshall}(G)$;

  $x \leftarrow$ DP($V, A, s, \{T_i\}_{i=1,\dots,N},$) ;

  if x *is feasible* **then**

    return $(x, z(x))$

  ...;

The B&B algorithm

Function `B&B(`$G = \langle V, A, C \rangle, s, d, \{T_i\}_{i=1,\dots,N}$`)`

…;

for $i \leftarrow 1$ ***to*** $N - 1$ **do**

  foreach $v \in T_i$ **do**

    foreach $w \in T_{i+1}$ **do**

      Paths[i] $\leftarrow$ Paths[i] $\cup$ {ShortestPaths[v][w]} ;

$Q \leftarrow$ `GenerateNodes`$(x, Paths, [\emptyset]_{i=1}^{N-1})$;

$x^* \leftarrow \mathbf{Nil}; z(x^*) \leftarrow +\infty$;

…;

The B&B algorithm

Function `B&B`($G = \langle V, A, C \rangle$, $s, d, \{T_i\}_{i=1,\dots,N}$)

  …;

  while Q *is not empty* **do**

    $Node \leftarrow$ `Pop`(Q);

    $i \leftarrow Node.index$;

    $A \leftarrow A \setminus Node.costraints[i]$;

    …;

The B&B algorithm

```
Function B&B(G = ⟨V, A, C⟩, s, d, {T_i}_{i=1,...,N})
    ...;
    while Q is not empty do
        ...;
        foreach v ∈ T_i do
            foreach w ∈ T_{i+1} do
                Node.paths[i] ←
                  Node.paths[i] ∪ {Dijkstra(G, v, w)};
        ...;
```

The B&B algorithm

Function `B&B`($G = \langle V, A, C \rangle, s, d, \{T_i\}_{i=1,\dots,N}$)

    …;

    while Q *is not empty* **do**

        …;

        $x \leftarrow$ `DP`$(Node.paths)$;

        $A \leftarrow A \cup Node.costraints[i]$;

        if x *is feasible* **then**

            if $z(x) < z(x^*)$ **then**

                $x^* \leftarrow x, z(x^*) \leftarrow z(x)$;

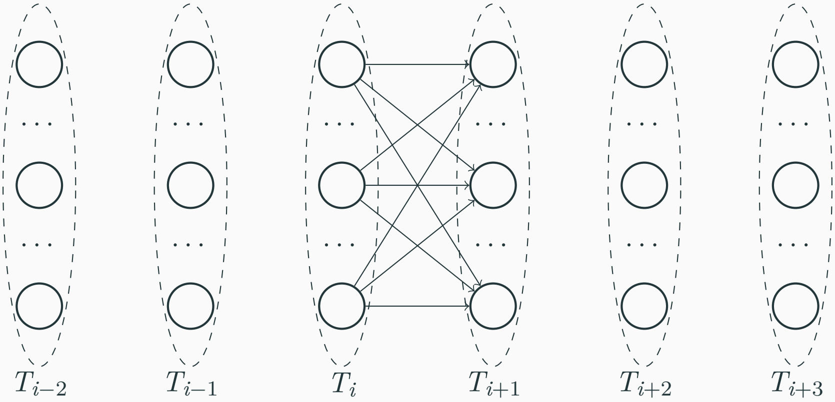
        else if $z(x) < z(x^*)$ **then**

            $Q \leftarrow$
             $Q \cup$`GenerateNodes`$(x, Node.paths, Node.constraints)$;

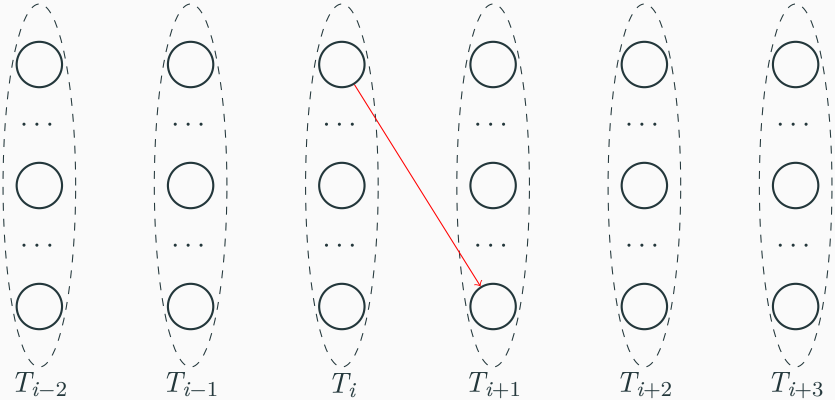
    return $(x^*, z(x^*))$;

GRASP

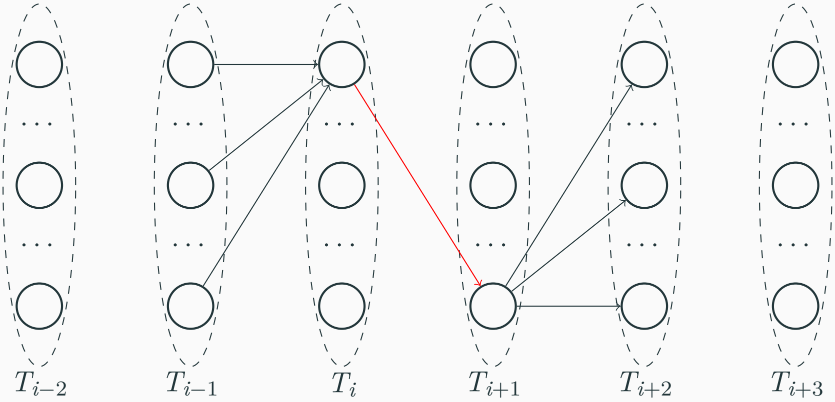
Construction phase



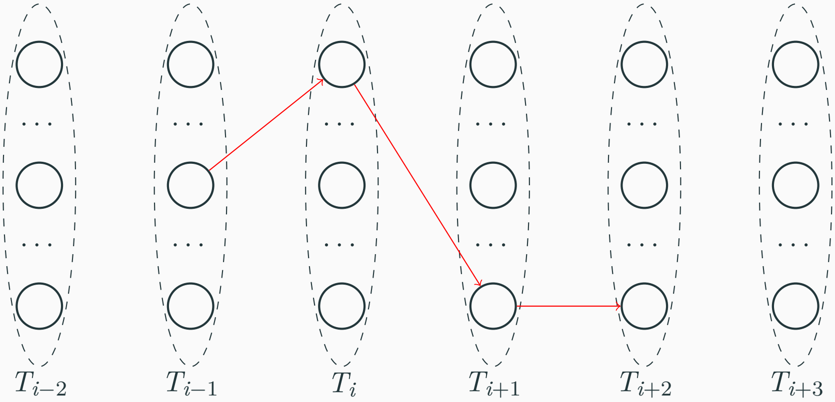
Construction phase



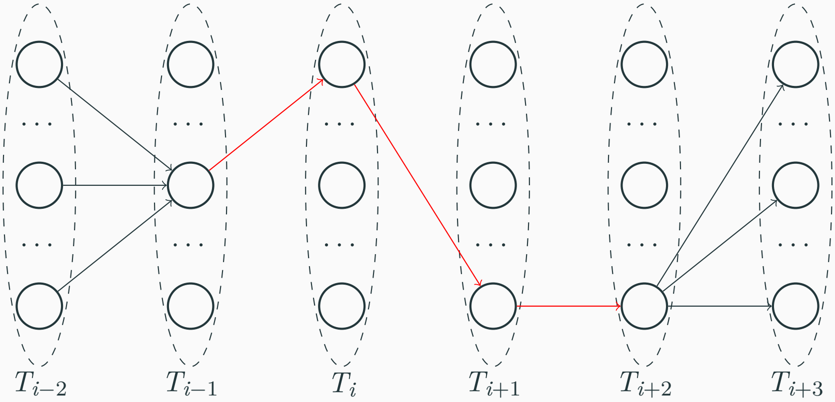
Construction phase



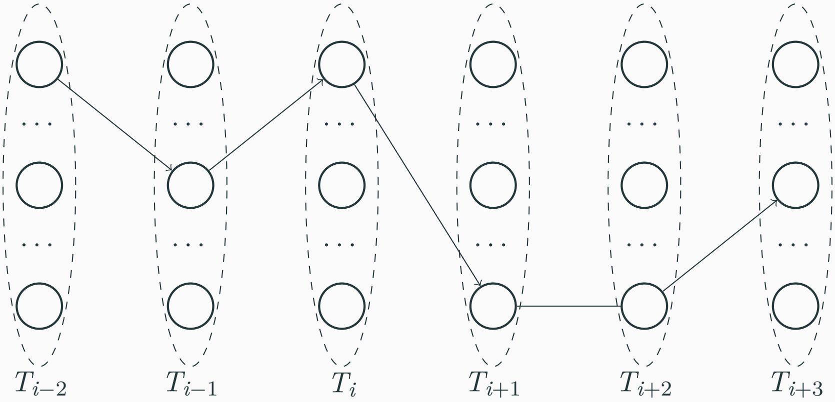
Construction phase



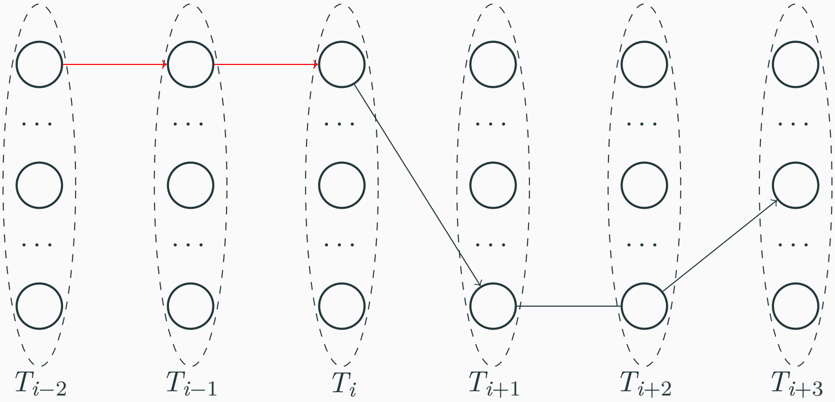
Construction phase



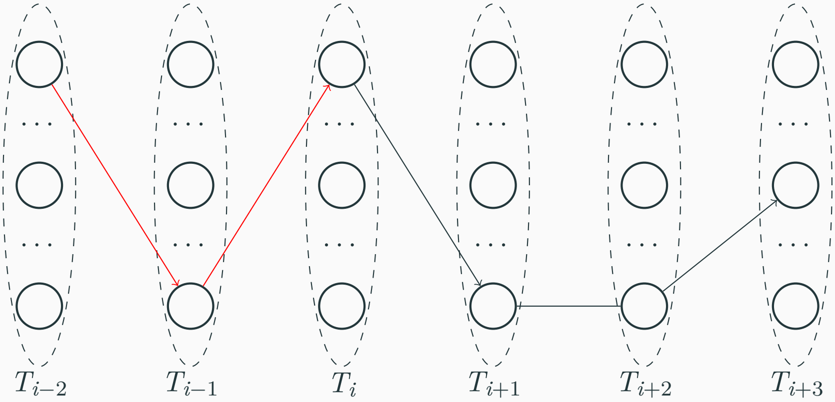
Local search



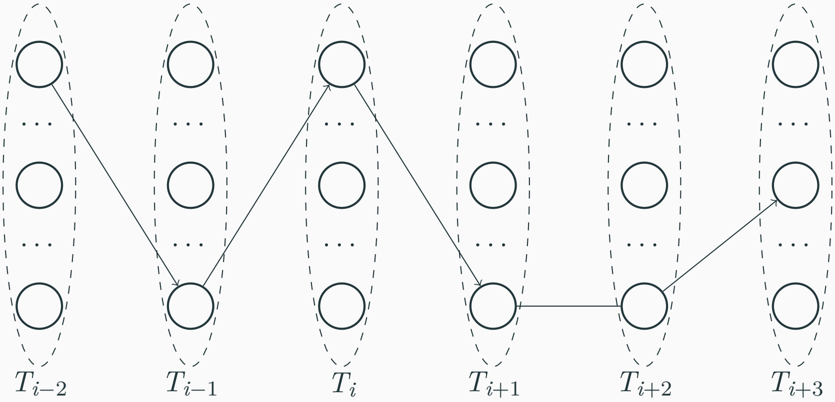
Local search



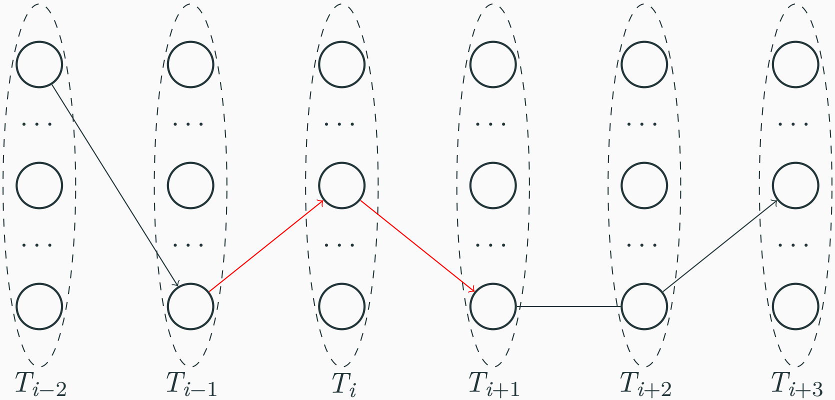
Local search



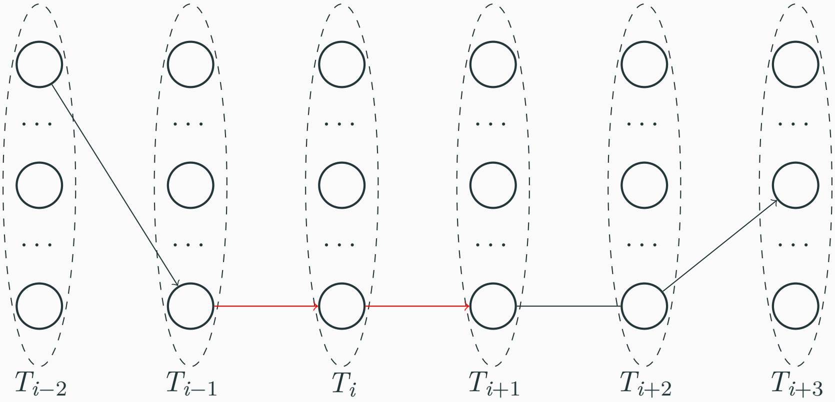
Local search



Local search



Local search



Experimental results

Experimental settings

Implemented in C++, executed on S.Co.P.E., a cluster of nodes, connected by 10 Gigabit Infiniband technology, each of them with two processors Intel Xeon E5-4610v2@2.30 Ghz.

Instances:

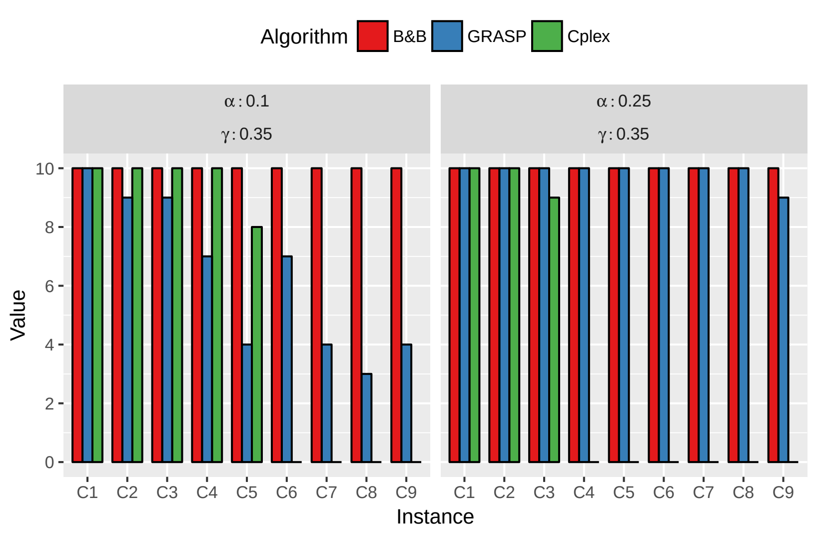
Complete Graphs The number of nodes are $n \in \{ 100, \ldots, 500 \}$ with a step of 50. ($C1, \ldots, C9$)

Random Sparse Graphs $n \in \{250, 500\}$, $m \in \{0.1, 0.2, 0.3, 0.4, 0.5\} \cdot n(n-1)$. (R1, …, R10)

Grid Graphs 5×10, 10×20, 15×30, 5×5, 10×10, 15×15. (G1, …, G6)

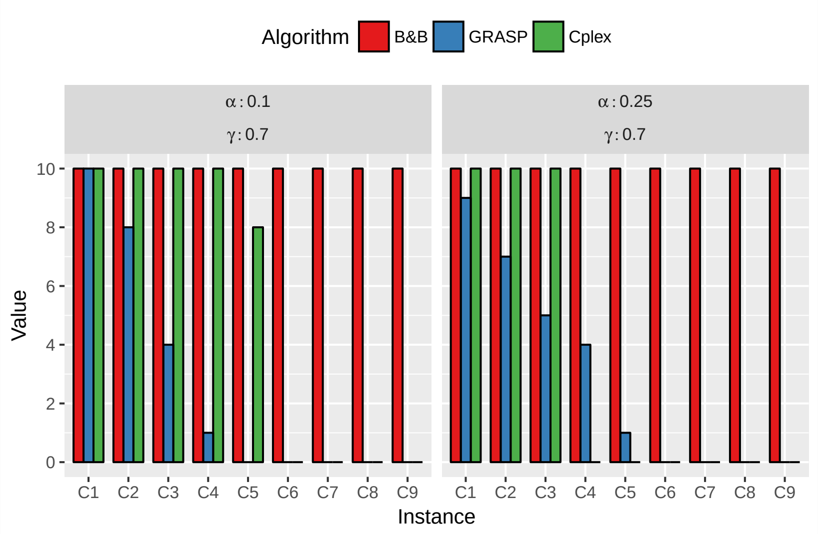
M2 vs B&B vs GRASP on complete graphs

Optimal solutions



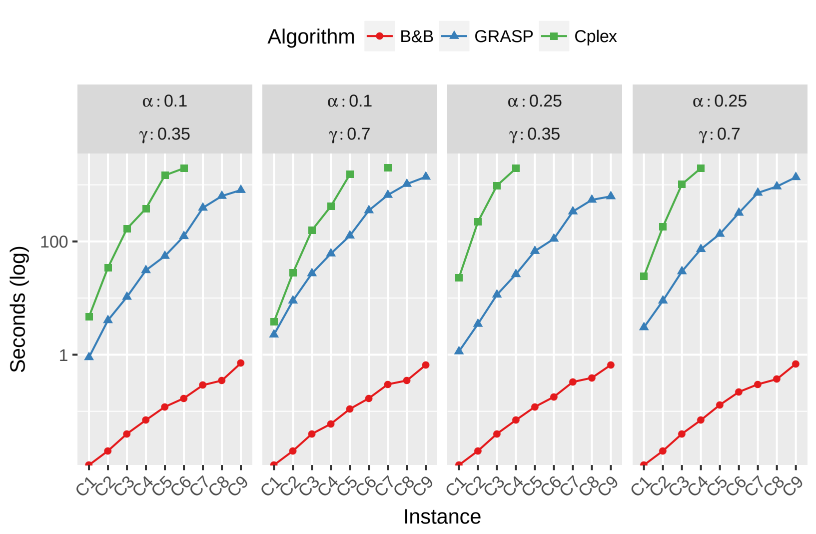
M2 vs B&B vs GRASP on complete graphs

Optimal solutions



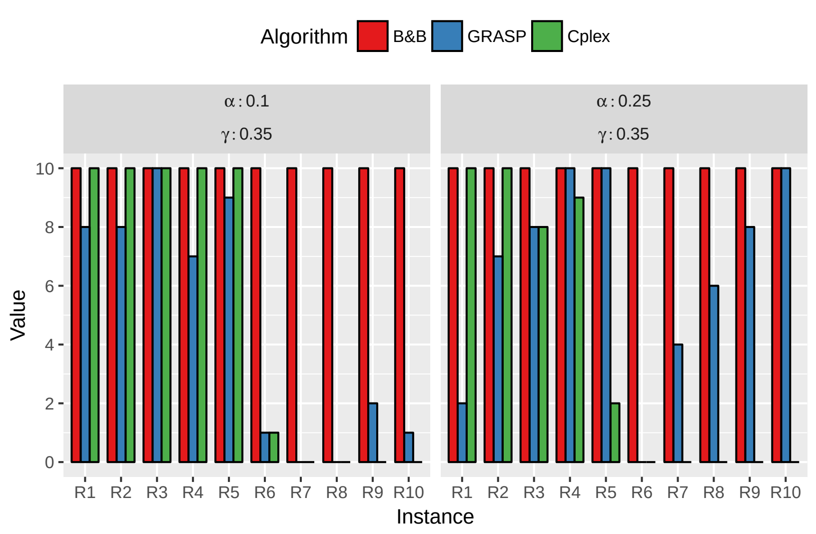
M2 vs B&B vs GRASP on complete graphs

Computational times



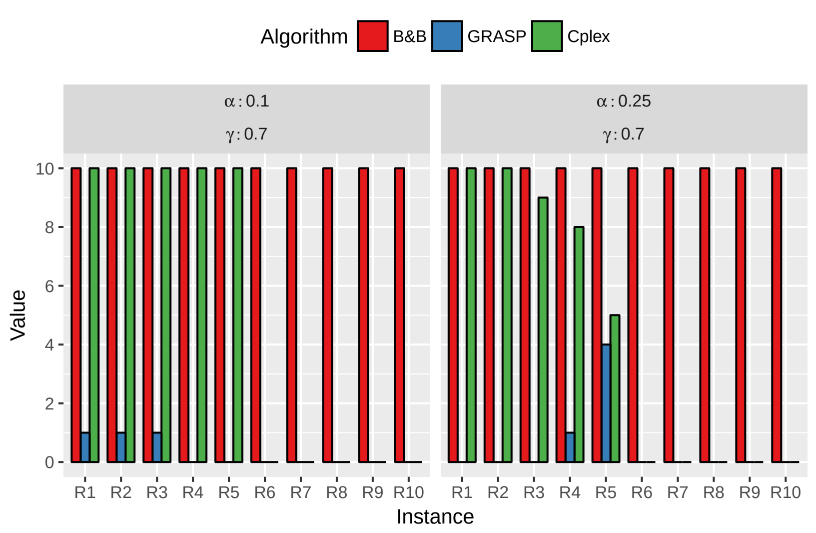
M2 vs B&B vs GRASP on random graphs

Optimal solutions



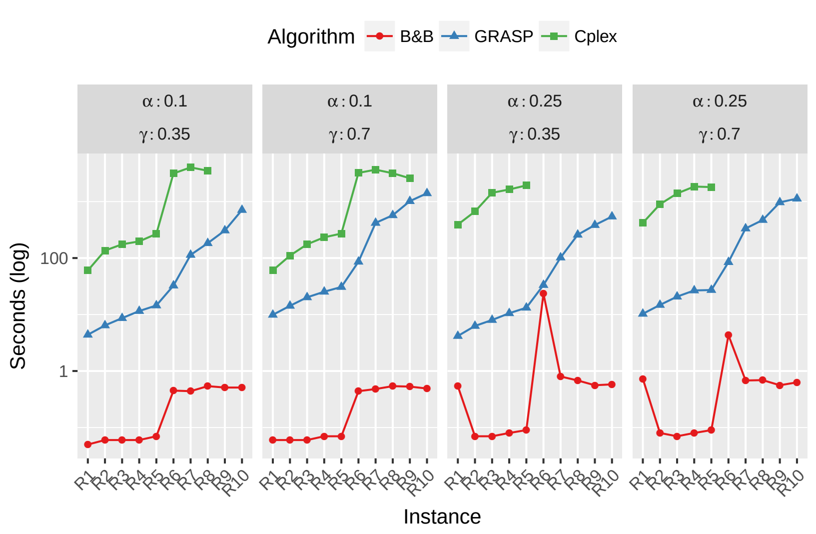
M2 vs B&B vs GRASP on random graphs

Optimal solutions



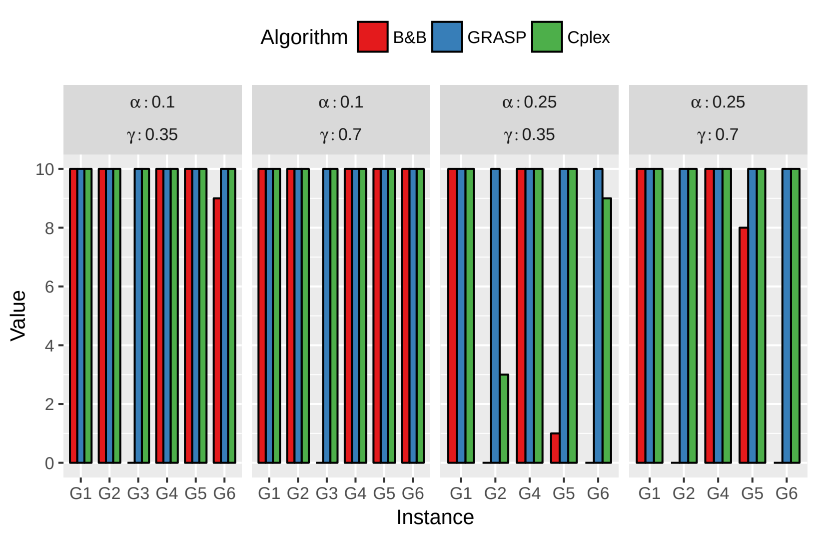
M2 vs B&B vs GRASP on random graphs

Computational times



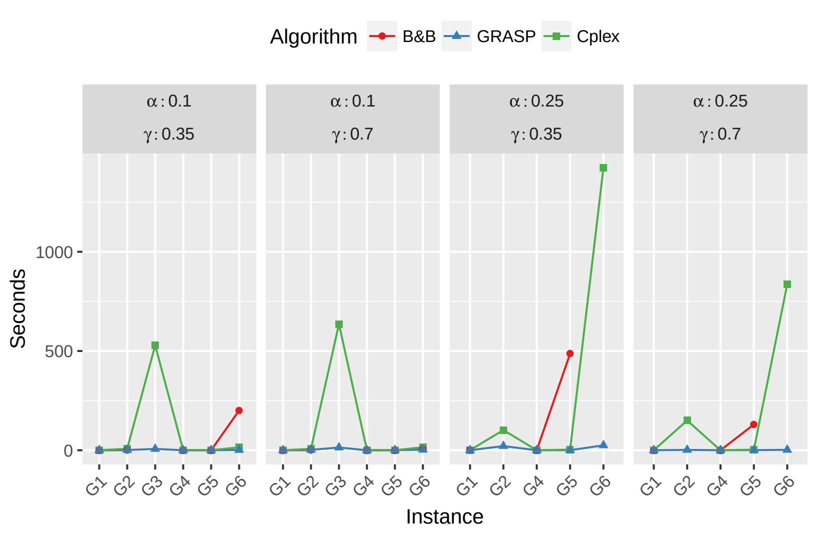
M2 vs B&B vs GRASP on grid graphs

Feasible solutions



M2 vs B&B vs GRASP on grid graphs

Computational times



Conclusions and future work

- B&B has very good performances expecially on dense graphs;

Conclusions and future work

- B&B has very good performances expecially on dense graphs;
- GRASP is useful when B&B is not able to find feasible solutions;

Conclusions and future work

- B&B has very good performances expecially on dense graphs;
- GRASP is useful when B&B is not able to find feasible solutions;
- as future work, we are investigating further variants of the problem resulting from the introduction of further constraints defined on the arcs and/or on the nodes of the graph.

01000101 01001110 01000100

(**E** **N** **D**)

Thank you.