Asymptotic Properties of a Probabilistic Tabu Search Algorithm

Siyeong Lee

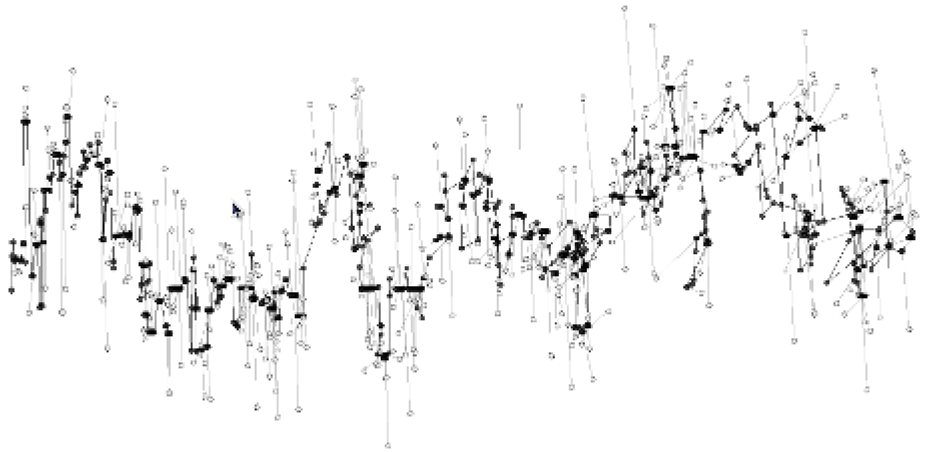
Dept. of Electronic Engineering

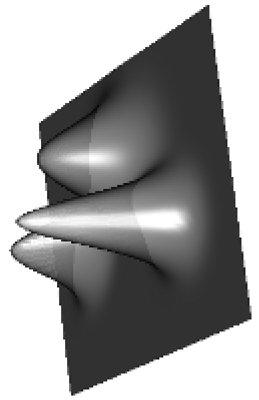
Sogang University

May 30, 2017

Introduction

- Unfortunately, we still know little about the behavior of the method, its asymptotic properties, and probability of finding an optimal solutio n, even for classical combinatorial problems.





Introduction

- Strategy
  - Memorize the feature of local optima → "Tabu list(queue)"
    - prevent repetition of the same state
  - Efficient search space exploration
    - search for an optimum solution without stagnation
- Good
  - This method can be **easy adapted** to complicated models and is simple to code

Until now

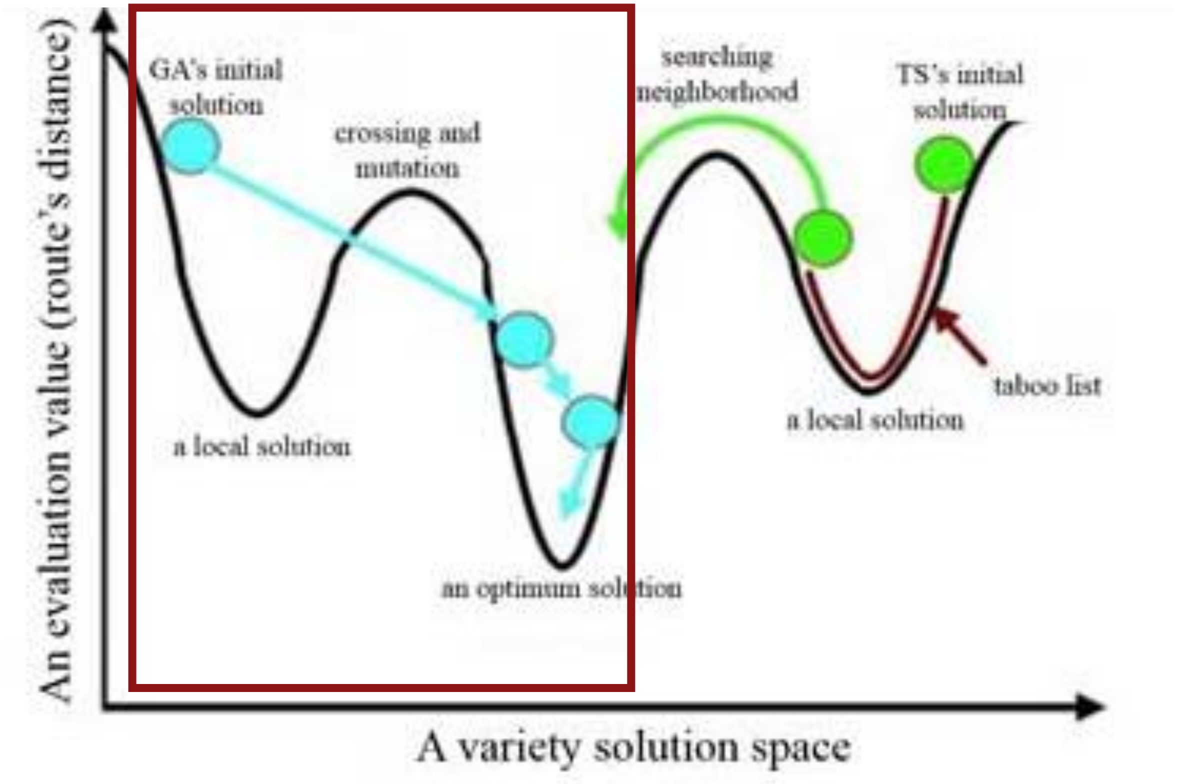


Figure 1: Difference between GA and TS

Problem setting – Optimization problem

Definition 1. Combinatory Optimization problem

$\boldsymbol{B}: the\ boolean\ cube$

$$Minimize\ f_0(x)$$

Subject to $f_i(x) \leq b_i, i = 1, \dots, m$

$f_0 : \boldsymbol{B}^n \to \boldsymbol{R}$; Objective function
$x = (x_1, \dots, x_n) \in \boldsymbol{B}^{\boldsymbol{n}}$: *Optimization variables*

To find the global optimum $\boldsymbol{f_{opt} = f(x_{opt})}$

Representation

- Teminology
  - A neighborhood of the point x: $N(x)$
    - Assume that it contains all neighboring points $y \in B^n$ with Hamming distance $\mathrm{d}(x, y) \leq 2$.
      * At most, differ from 2 bit.
  - A randomized neighborhood $N_p(x)$ with probabilistic threshold p, $0 \leq p \leq 1$,
    - Subset of $N(x)$
    - For each $y \in N(x)$, $y \in N_p(x)$ randomly with prob. p
    - Independently from other points.

Representation

- Consider a finite sequence $\{x_t\}, 1 \leq t \leq k$ with property $x_{t+1} \in N(x_t)$.
- Tabu list (or tabu queue)
  - An ordered set $\varphi = \{(i_k, j_k), (i_{k-1}, j_{k-1}), \ldots, (i_{k-l+1}, j_{k-l+1})\}$
    - if vectors x_t and x_{t+1} differ by coordinates (i_t, j_t).
    - The constant l is called the length of the tabu list.

✓

1. i_t and j_t may be equal
   * the vectors x_t and x_{t+1} are differed by exactly one coordinate.
2. $i_t = j_t = 0$ if $x_{t+1} = x_t$.

- $N_p(x_t, \varphi)$: a set of points $y \in N_p(x_t)$ not forbidden by the tabu list φ.

✓

$N_p(x_t, \varphi)$ may be empty for nonempty set $N_p(x_t)$.

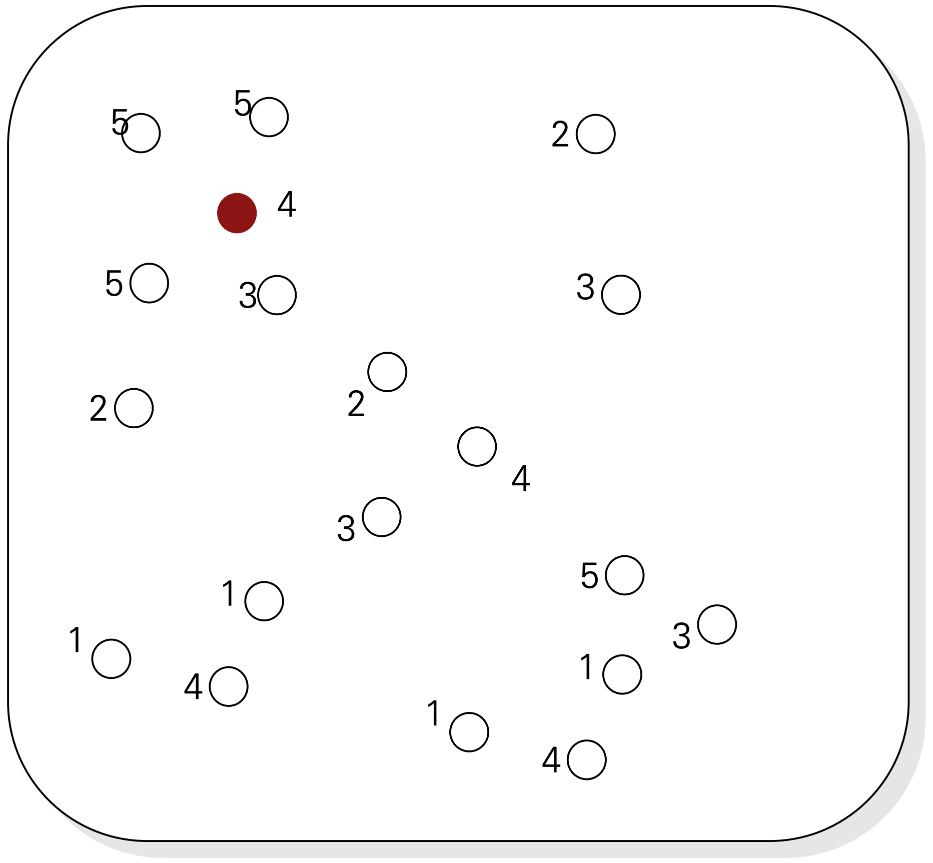
Pseudo Code

Algorithm PTS

```
1. Initialize $x_0 \in B^n$, $f^* := f(x_0)$, $\varphi := \emptyset$, $t := 0$.
2. While a stopping condition is not fulfilled do
    2.1. Generate neighborhood $N_p(x_t, \varphi)$.
    2.2. If $N_p(x_t, \varphi) = \emptyset$ then $x_{t+1} := x_t$,
         else find $x_{t+1}$ such that $f(x_{t+1}) = \min\{f(y), y \in N_p(x_t, \varphi)\}$.
    2.3. If $f(x_{t+1}) < f^*$ then $f^* := f(x_{t+1})$.
    2.4. Update the tabu list $\varphi$ and the counter $t := t + 1$.
```

$$\varphi = \emptyset$$

(Current Point) ●



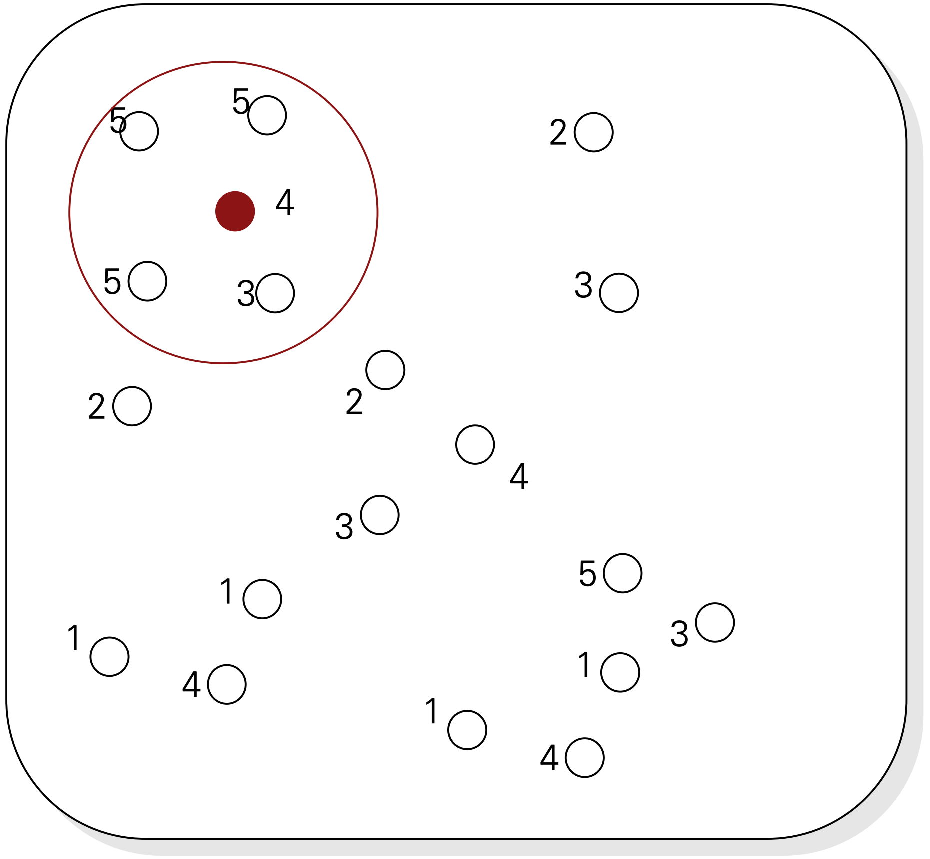
Search space S

Pseudo Code

(Current Point) ●

```
Algorithm PTS
1. Initialize $x_0 \in B^n$, $f^* := f(x_0)$, $\varphi := \emptyset$, $t := 0$.
2. While a stopping condition is not fulfilled do
    2.1. Generate neighborhood $N_p(x_t, \varphi)$.
    2.2. If $N_p(x_t, \varphi) = \emptyset$ then $x_{t+1} := x_t$,
         else find $x_{t+1}$ such that $f(x_{t+1}) = \min\{f(y), y \in N_p(x_t, \varphi)\}$.
    2.3. If $f(x_{t+1}) < f^*$ then $f^* := f(x_{t+1})$.
    2.4. Update the tabu list $\varphi$ and the counter $t := t + 1$.
```

$$\varphi = \emptyset$$



Search space S

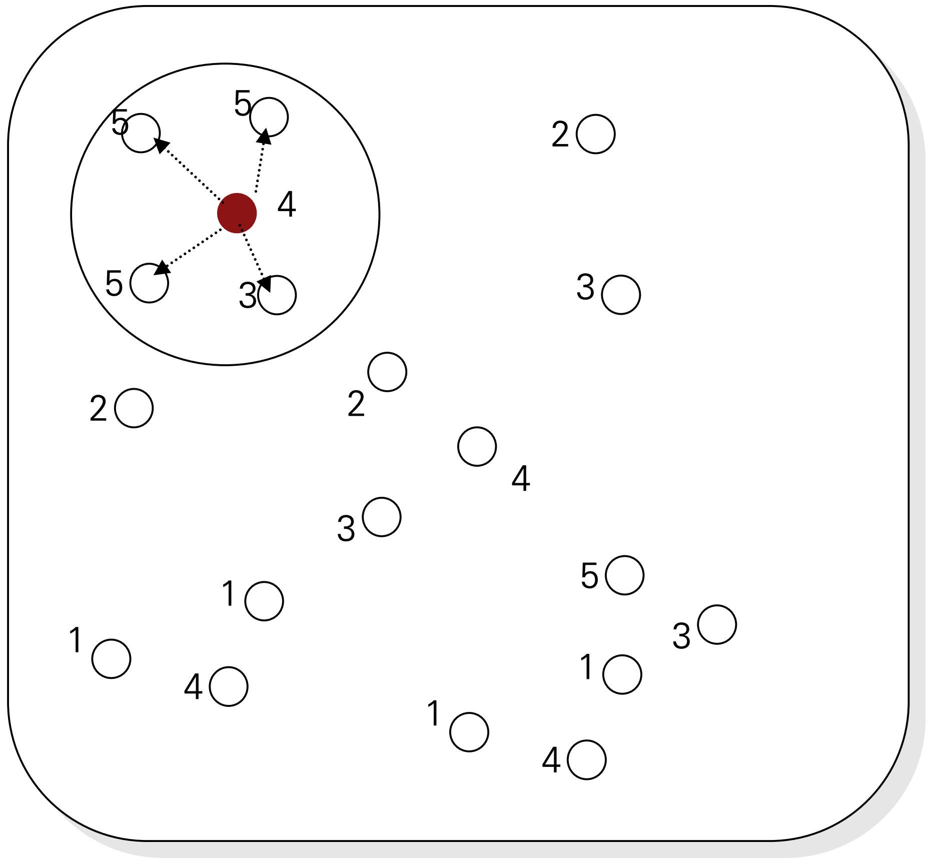
Pseudo Code

(Current Point) ●

Algorithm PTS

```
1. Initialize $x_0 \in B^n$, $f^* := f(x_0)$, $\varphi := \emptyset$, $t := 0$.
2. While a stopping condition is not fulfilled do
    2.1. Generate neighborhood $N_p(x_t, \varphi)$.
    2.2. If $N_p(x_t, \varphi) = \emptyset$ then $x_{t+1} := x_t$,
         else find $x_{t+1}$ such that $f(x_{t+1}) = \min\{f(y), y \in N_p(x_t, \varphi)\}$.
    2.3. If $f(x_{t+1}) < f^*$ then $f^* := f(x_{t+1})$.
    2.4. Update the tabu list $\varphi$ and the counter $t := t + 1$.
```

$$\boxed{\varphi = \emptyset}$$



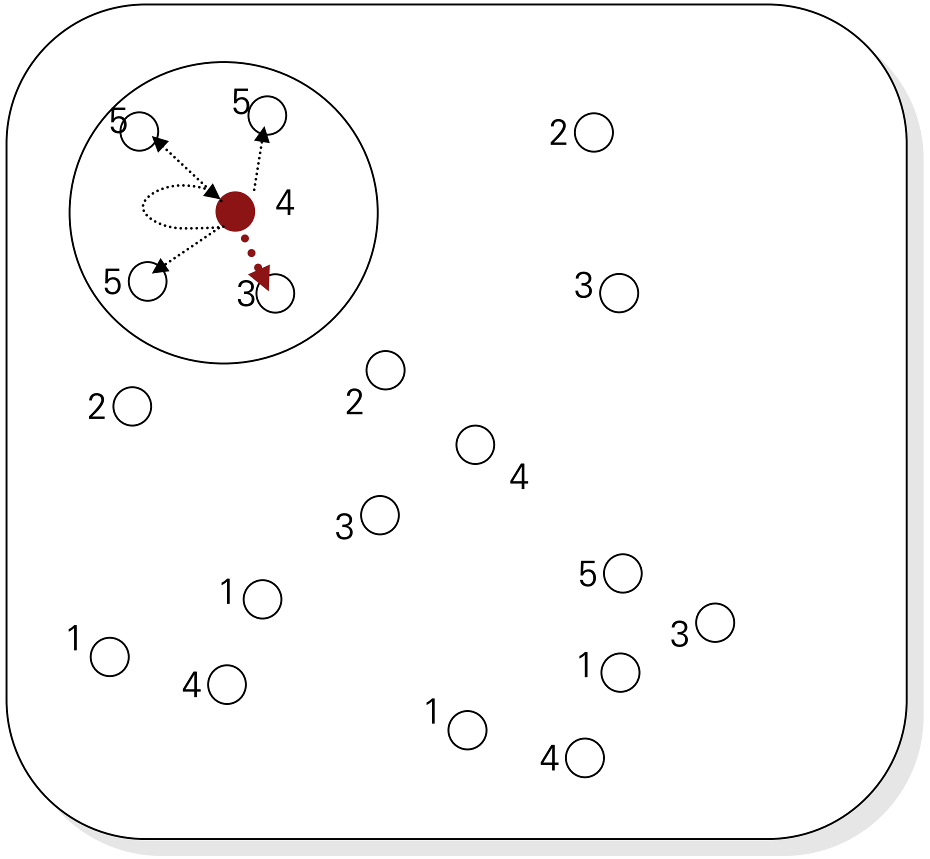
Search space S

Pseudo Code

```
Algorithm PTS
1. Initialize $x_0 \in B^n$, $f^* := f(x_0)$, $\varphi := \emptyset$, $t := 0$.
2. While a stopping condition is not fulfilled do
    2.1. Generate neighborhood $N_p(x_t, \varphi)$.
    2.2. If $N_p(x_t, \varphi) = \emptyset$ then $x_{t+1} := x_t$,
         else find $x_{t+1}$ such that $f(x_{t+1}) = \min\{f(y), y \in N_p(x_t, \varphi)\}$.
    2.3. If $f(x_{t+1}) < f^*$ then $f^* := f(x_{t+1})$.
    2.4. Update the tabu list $\varphi$ and the counter $t := t + 1$.
```

$\varphi = \emptyset$

(Current Point) ●



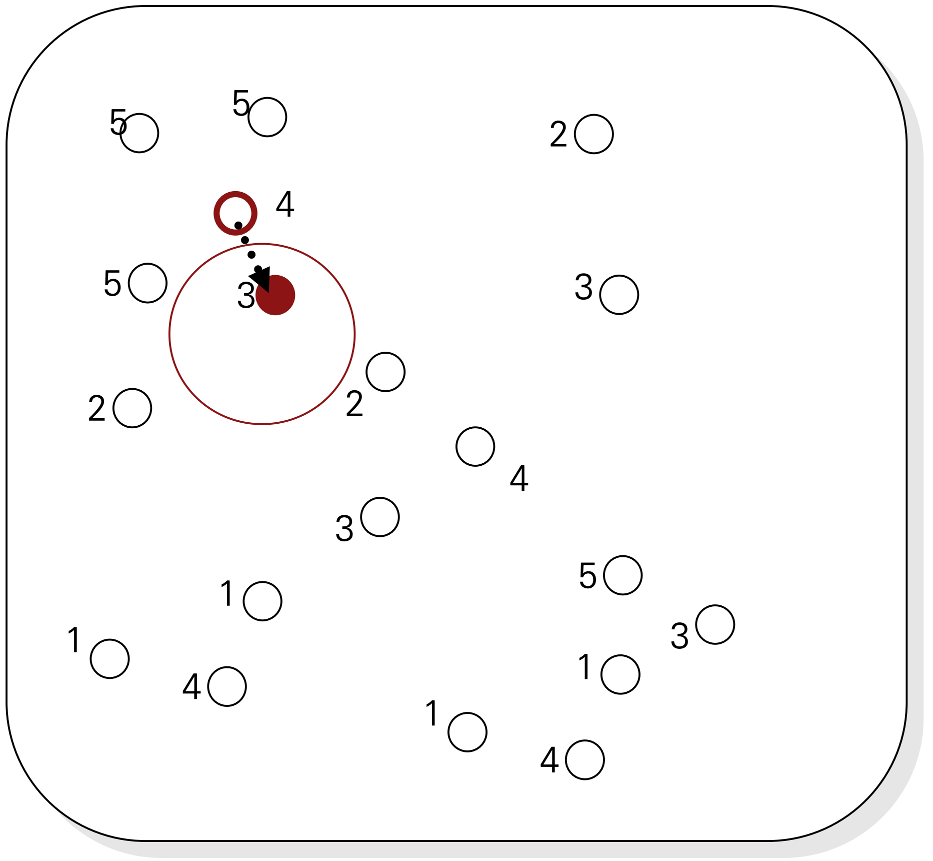
Search space S

Pseudo Code

```
Algorithm PTS
1. Initialize $x_0 \in B^n$, $f^* := f(x_0)$, $\varphi := \emptyset$, $t := 0$.
2. While a stopping condition is not fulfilled do
    2.1. Generate neighborhood $N_p(x_t, \varphi)$.
    2.2. If $N_p(x_t, \varphi) = \emptyset$ then $x_{t+1} := x_t$,
         else find $x_{t+1}$ such that $f(x_{t+1}) = \min\{f(y), y \in N_p(x_t, \varphi)\}$.
    2.3. If $f(x_{t+1}) < f^*$ then $f^* := f(x_{t+1})$.
    2.4. Update the tabu list $\varphi$ and the counter $t := t + 1$.
```

$\varphi = \{(1,1)\}$

(Current Point) ●



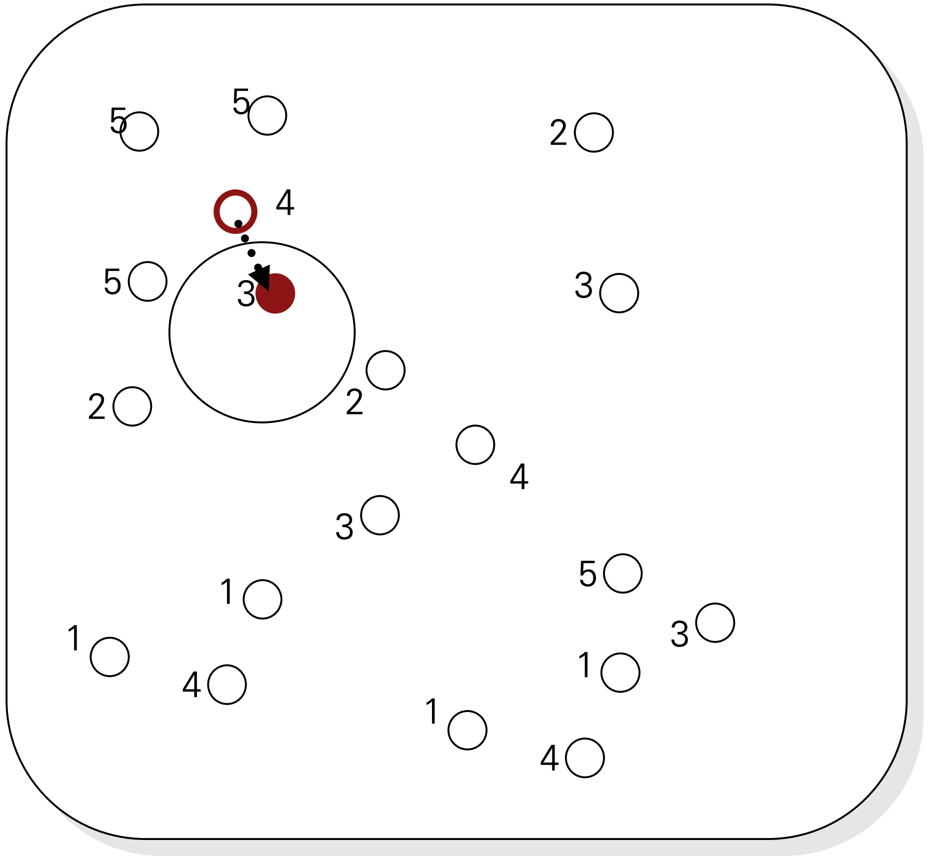
Search space S

Pseudo Code

```
Algorithm PTS
1. Initialize $x_0 \in B^n$, $f^* := f(x_0)$, $\varphi := \emptyset$, $t := 0$.
2. While a stopping condition is not fulfilled do
    2.1. Generate neighborhood $N_p(x_t, \varphi)$.
    2.2. If $N_p(x_t, \varphi) = \emptyset$ then $x_{t+1} := x_t$,
         else find $x_{t+1}$ such that $f(x_{t+1}) = \min\{f(y), y \in N_p(x_t, \varphi)\}$.
    2.3. If $f(x_{t+1}) < f^*$ then $f^* := f(x_{t+1})$.
    2.4. Update the tabu list $\varphi$ and the counter $t := t + 1$.
```

$$\varphi = \{(1,1)\}$$

(Current Point) ●



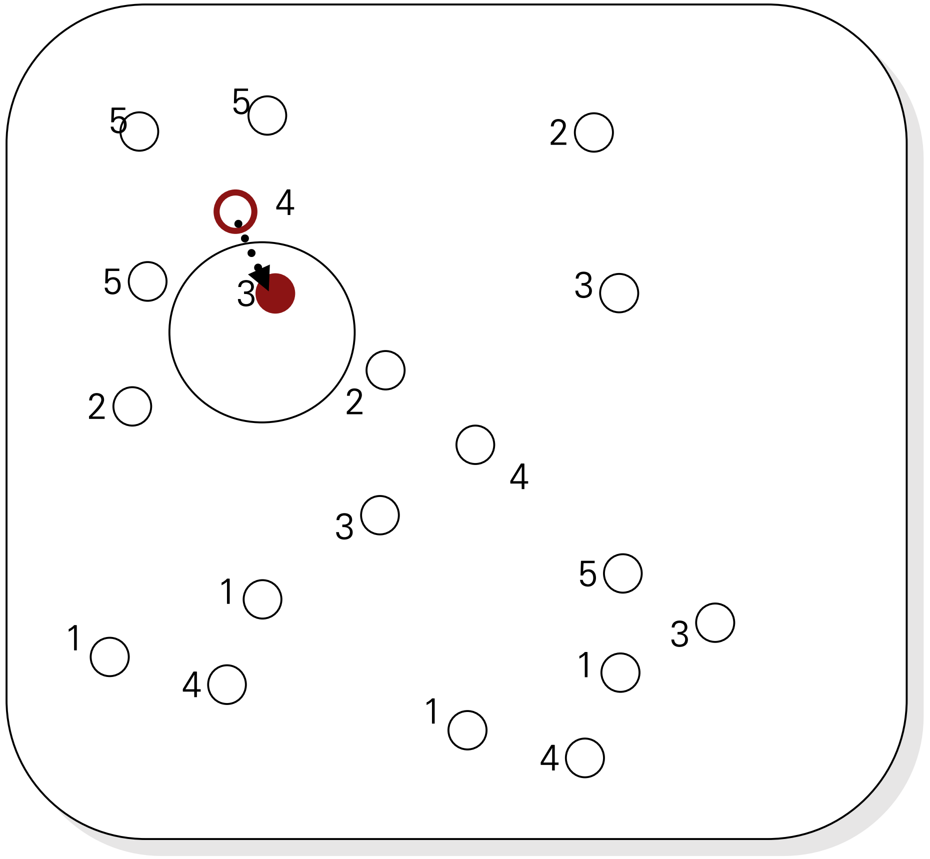
Search space S

Pseudo Code

(Current Point) ●

```
Algorithm PTS
1. Initialize $x_0 \in B^n$, $f^* := f(x_0)$, $\varphi := \emptyset$, $t := 0$.
2. While a stopping condition is not fulfilled do
    2.1. Generate neighborhood $N_p(x_t, \varphi)$.
    2.2. If $N_p(x_t, \varphi) = \emptyset$ then $x_{t+1} := x_t$,
         else find $x_{t+1}$ such that $f(x_{t+1}) = \min\{f(y), y \in N_p(x_t, \varphi)\}$.
    2.3. If $f(x_{t+1}) < f^*$ then $f^* := f(x_{t+1})$.
    2.4. Update the tabu list $\varphi$ and the counter $t := t + 1$.
```

$$\varphi = \{(0,0), (1,1)\}$$



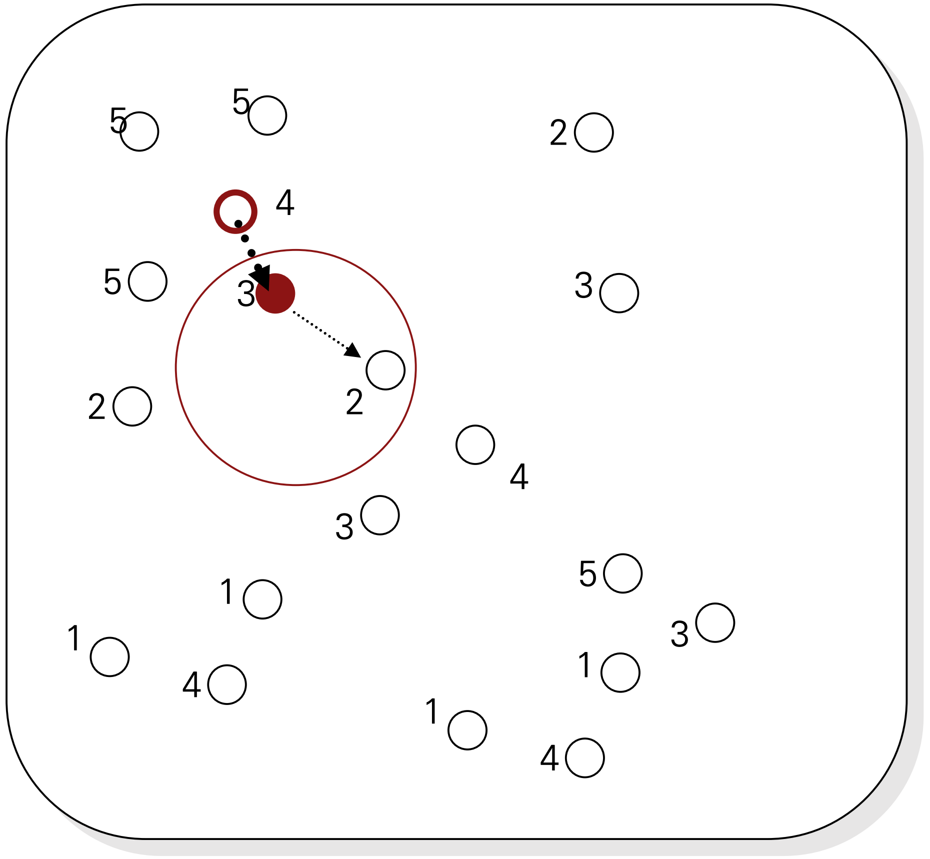
Search space S

Pseudo Code

```
Algorithm PTS
1. Initialize $x_0 \in B^n$, $f^* := f(x_0)$, $\varphi := \emptyset$, $t := 0$.
2. While a stopping condition is not fulfilled do
    2.1. Generate neighborhood $N_p(x_t, \varphi)$.
    2.2. If $N_p(x_t, \varphi) = \emptyset$ then $x_{t+1} := x_t$,
         else find $x_{t+1}$ such that $f(x_{t+1}) = \min\{f(y), y \in N_p(x_t, \varphi)\}$.
    2.3. If $f(x_{t+1}) < f^*$ then $f^* := f(x_{t+1})$.
    2.4. Update the tabu list $\varphi$ and the counter $t := t + 1$.
```

$$\varphi = \{(0,0),(1,1)\}$$

(Current Point) ●



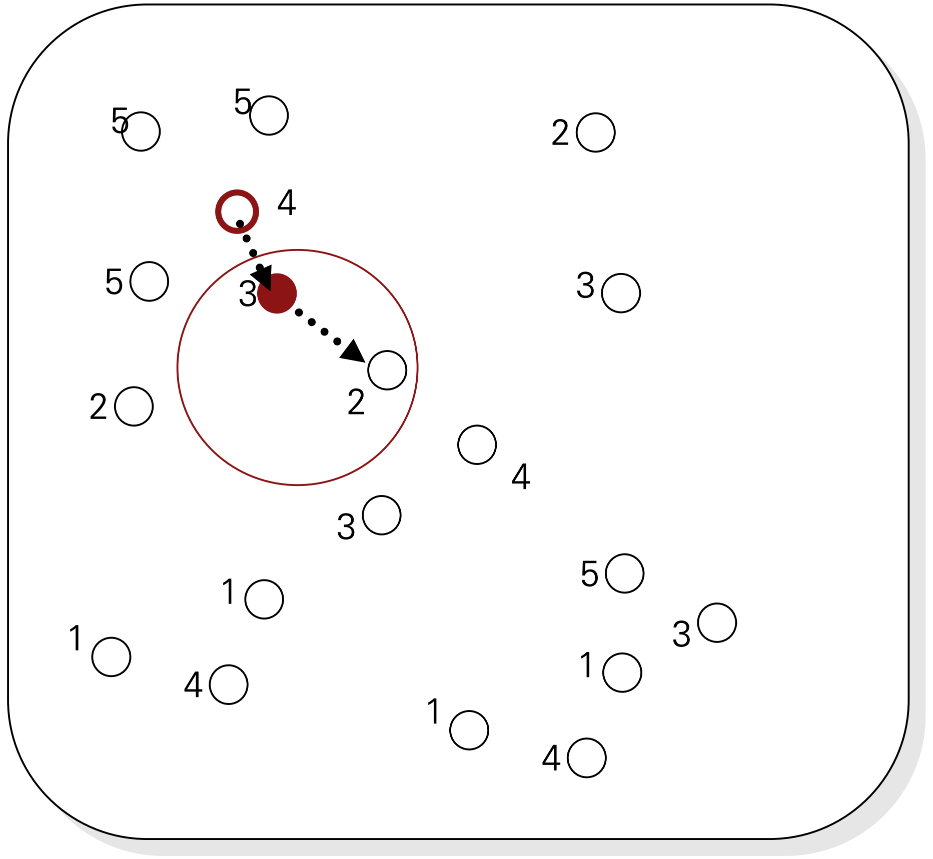
Search space S

Pseudo Code

```
Algorithm PTS
1. Initialize $x_0 \in B^n$, $f^* := f(x_0)$, $\varphi := \emptyset$, $t := 0$.
2. While a stopping condition is not fulfilled do
    2.1. Generate neighborhood $N_p(x_t, \varphi)$.
    2.2. If $N_p(x_t, \varphi) = \emptyset$ then $x_{t+1} := x_t$,
         else find $x_{t+1}$ such that $f(x_{t+1}) = \min\{f(y), y \in N_p(x_t, \varphi)\}$.
    2.3. If $f(x_{t+1}) < f^*$ then $f^* := f(x_{t+1})$.
    2.4. Update the tabu list $\varphi$ and the counter $t := t + 1$.
```

$$\boxed{\varphi = \{(0,0),(1,1)\}}$$

(Current Point) ●



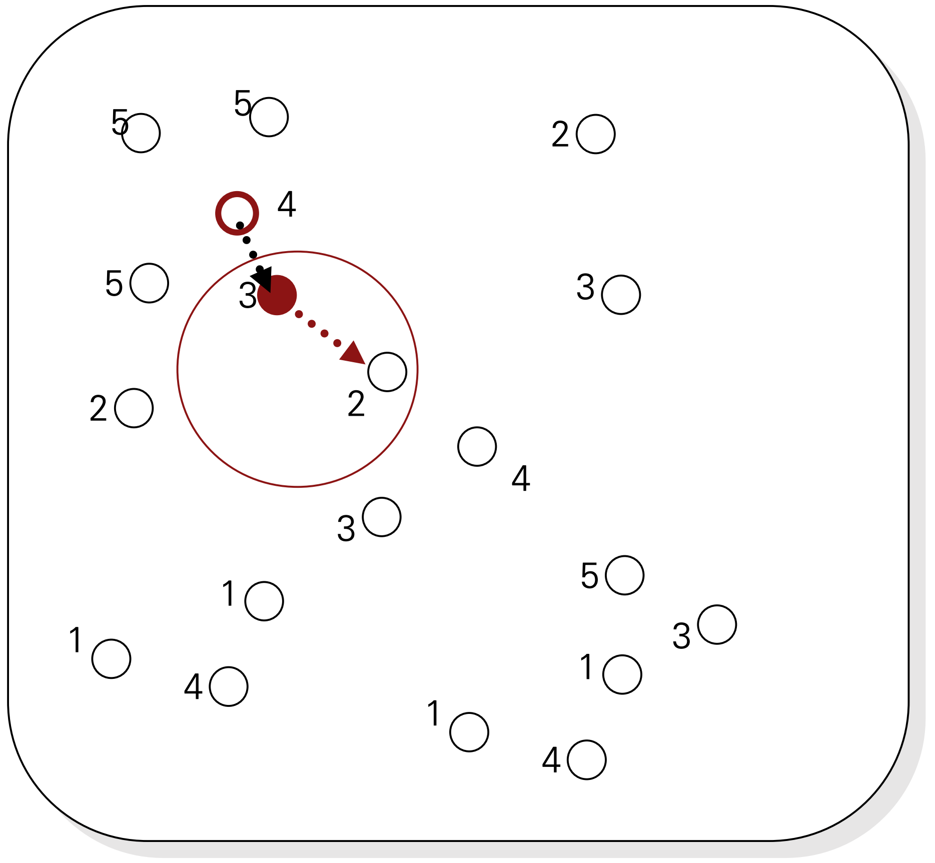
Search space S

Pseudo Code

```
Algorithm PTS
1. Initialize $x_0 \in B^n$, $f^* := f(x_0)$, $\varphi := \emptyset$, $t := 0$.
2. While a stopping condition is not fulfilled do
    2.1. Generate neighborhood $N_p(x_t, \varphi)$.
    2.2. If $N_p(x_t, \varphi) = \emptyset$ then $x_{t+1} := x_t$,
         else find $x_{t+1}$ such that $f(x_{t+1}) = \min\{f(y), y \in N_p(x_t, \varphi)\}$.
    2.3. If $f(x_{t+1}) < f^*$ then $f^* := f(x_{t+1})$.
    2.4. Update the tabu list $\varphi$ and the counter $t := t + 1$.
```

$$\varphi = \{(0,0),(1,1)\}$$

(Current Point) ●



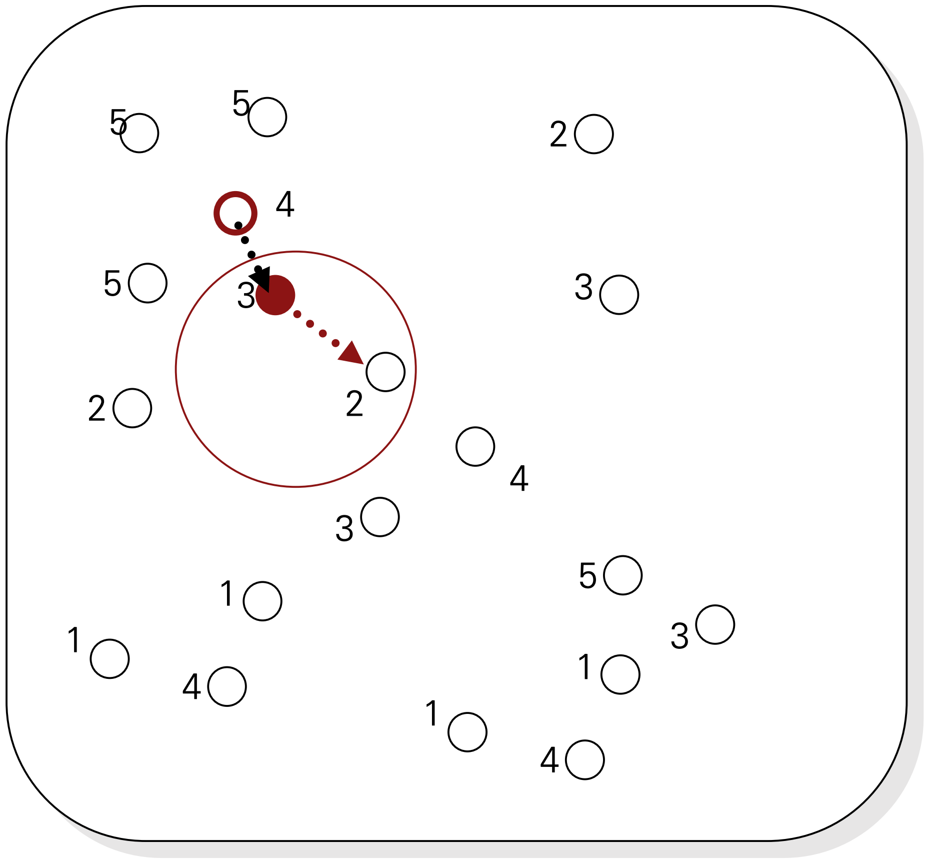
Search space S

Pseudo Code

```
Algorithm PTS
1. Initialize $x_0 \in B^n$, $f^* := f(x_0)$, $\varphi := \emptyset$, $t := 0$.
2. While a stopping condition is not fulfilled do
    2.1. Generate neighborhood $N_p(x_t, \varphi)$.
    2.2. If $N_p(x_t, \varphi) = \emptyset$ then $x_{t+1} := x_t$,
         else find $x_{t+1}$ such that $f(x_{t+1}) = \min\{f(y), y \in N_p(x_t, \varphi)\}$.
    2.3. If $f(x_{t+1}) < f^*$ then $f^* := f(x_{t+1})$.
    2.4. Update the tabu list $\varphi$ and the counter $t := t + 1$.
```

$$\varphi = \{(1,4), (0,0), (1,1)\}$$

(Current Point) ●



Search space S

Toy Problem

Search space S

- Representation of Solutions

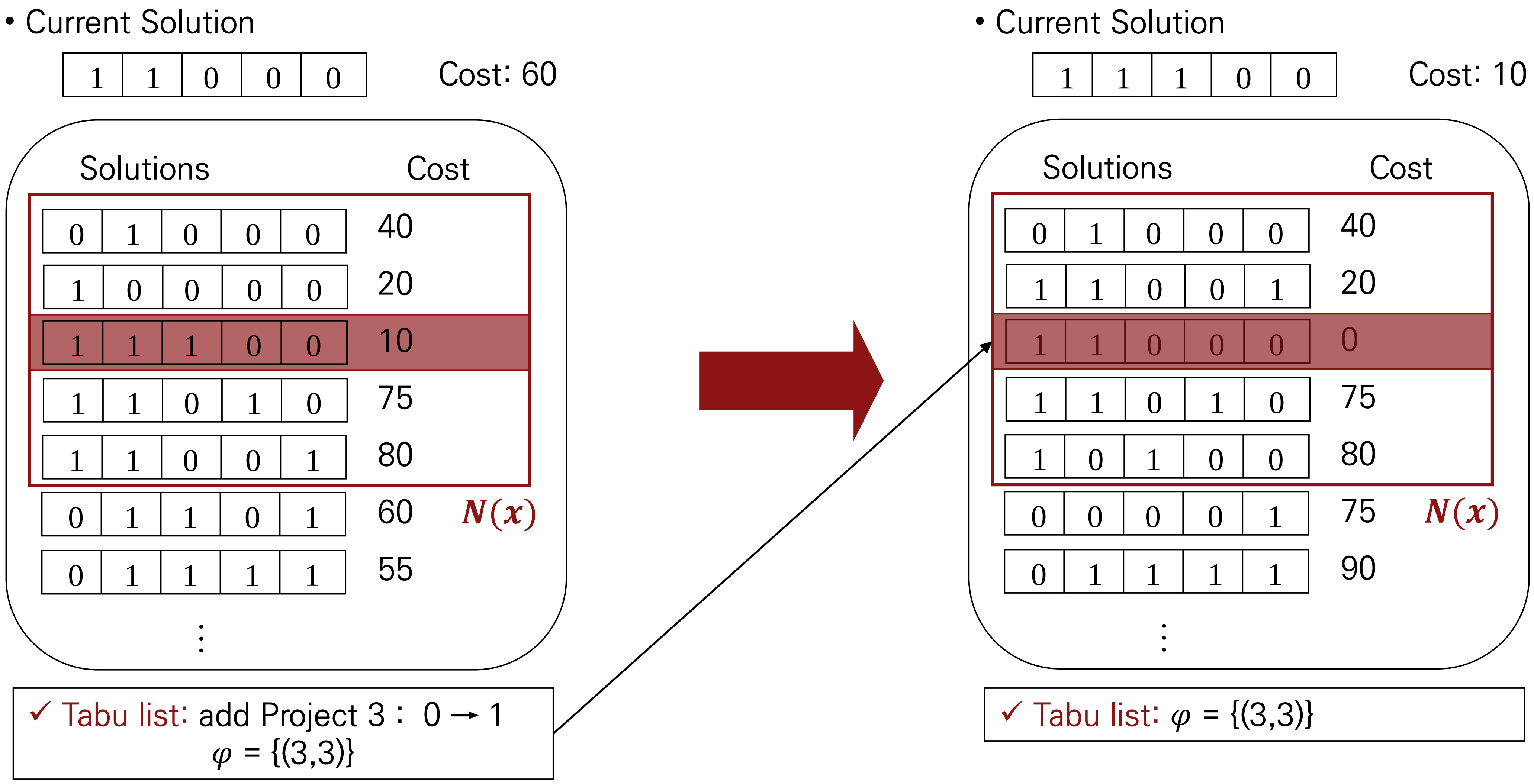
x_1	x_2	x_3	x_4	x_5

$$x_i = \begin{cases} 1 & \text{if project } i \rightarrow \text{O} \\ 0 & \text{if project } i \rightarrow \times \end{cases}$$

- Neighbor of current solution x
  - $N(x) = \{y \in S \mid \text{hamming distance } d(x, y) \leq 2\}$
  - $N_p(x, \varphi) \subset N(x)$; $For\ each\ y \in N(x), y \in N_p(x)\ randomly\ with\ prob.\, p$

- Tuba condition
  - If Project j added (removed) in this step,
    will not be removed (added) for the next k steps.

Toy Problem



- Current Solution

1	1	0	0	0	Cost: 60

Solutions					Cost
0	1	0	0	0	40
1	0	0	0	0	20
1	1	1	0	0	10
1	1	0	1	0	75
1	1	0	0	1	80
0	1	1	0	1	60
0	1	1	1	1	55
⋮					

$N(x)$

✓ Tabu list: add Project 3 : 0 → 1
$\varphi = \{(3,3)\}$

- Current Solution

1	1	1	0	0	Cost: 10

Solutions					Cost
0	1	0	0	0	40
1	1	0	0	1	20
1	1	0	0	0	0
1	1	0	1	0	75
1	0	1	0	0	80
0	0	0	0	1	75
0	1	1	1	1	90
⋮					

$N(x)$

✓ Tabu list: $\varphi = \{(3,3)\}$

Preliminary

- Properties of Markov chain
  - *finite*
    - if the set of outcomes is finite.
  - *homogeneous*
    - if the transition probabilities do not depend on the step number.
    - e.g. For $l = 0$, PTS algorithm generates a finite homogeneous Markov chain on the Boolean cube B^n.
  - *irreducible*
    - if for each pair of outcomes x, y, there is a positive probability of reaching y from x in a finite number of steps.

Convergence behavior to an optimal solution

Theorem 1.

For arbitrary $l > 0$ *and* $0 < p < 1$,
the PTS algorithm generates an irreducible Markov chain on Ω.

proof. Suppose $l = 0$

- Therefore, the selection of x_{t+1} depends on the current point x_t. and does not depend on the previous points $x_s, s < t$

✓ $\boxed{\text{a finite homogeneous Markov chain on the boolean cube } B^n}$

Convergence behavior to an optimal solution

- From the definition of $N_p(x)$, it is follows that the Markov chain is irreducible
  - So, we can obtain

$$f^* = \min_{t \leq k} f(x_t) = f_{opt} \text{ for large } k.$$

irreducible
if for each pair of outcomes x, y, there is a positive proba bility of reaching y from x in a finite number of steps.

  - Without any restrictions for the length of tabu list l?
    - If $l > |N(x)|$, then all points may be forbidden and $N_p(x, \varphi) = \emptyset$
    - For this case, we get $x_{t+1} = x_t$ on the *step 2.2* and $(i_t, j_t) = (0,0)$
  - The algorithm regulates the tabu list by itself.

Convergence behavior to an optimal solution

- From the definition of *N_p(x),*
  it is follows that the Markov chain is irreducible
  - So, we can obtain

$$f^* = \min_{t \le k} f(x_t) = \quad$$

This element of self learning allows us
to prove the theorem and get asymptotic properties of the algorithm

pair of outcomes x, y, there is a positive proba
bility of reaching y from x in a finite number of steps.

strictions for the length of tabu list l?

$> |N(x)|$, then all points may be forbidden and $N_p(x, \varphi) = \emptyset$

  - For this case, we get $x_{t+1} = x_t$ on the *step 2.2* and $(i_t, j_t) = (0,0)$
- The algorithm regulates the tabu list by itself.

Convergence behavior to an optimal solution

✓ This element of self learning allows us
to prove the theorem and get asymptotic properties of the algorithm

- Denote a randomized neighborhood of x by $N_r(x, \varphi)$ which contains e xactly $r > 1$ unforbidden points from $N(x)$
  - The algorithm PTS with $N_r(x, \varphi)$ neighborhood generates a Markov chain
  - But we can not prove the irreducibility for this case

In $r = |N_x - l|$, Deterministic Tabu Search algorithm DTS

- If the l is too small algorithm, DTS finds a local optimum and has no o pportunity to escape from it
- If not, DTS can not find the optimal solution.

Convergence behavior to an optimal solution

Corollary 2.

For an arbitrary initial point $x_0 \in B^n$

1. $lim_{t \to \infty} Pr\{f^* = f_{opt}\} = 1$
2. *there exist constants* $b > 0$ *and* $1 > c > 0$ *such that* $\Pr\{\min_{\tau \leq t} f(x_\tau) \neq f_{opt}\} \leq bc^t$
3. *the Markov chain* $\{x_t, \varphi_t\}$ *has a unique stationary distribution* $\pi > 0$*.*

Proof. The first and the second properties immediately follow from the property of irreducibility. In order to prove the last statement, it suffice to note that the Marko v chain is aperiodic

Convergence behavior to an optimal solution

- Property 1
  - obtains an optimal solution with probability 1 for sufficiently large number of steps.
  - Don’t say that
    - $lim_{t\to\infty} \Pr\{x_t \in Xopt\} = 1$, where X_{opt} is the set of all optimal solutions.

- Property 2
  - guarantees a geometrical rate of convergence with the constant $c < 1$.

- Property 3
  - generates an ergodic Markov chain with a positive limiting distribution $\pi(x, \quad)$
  - we can find a global optimum from an arbitrary initial point.

Stopping Rules

- Many stopping rules for Markov chains
  - Stopping after a prescribed number of steps
  - Stopping if the best solution f^* so far does not change during a prescribed n umber of steps

Denote by $H(x, y)$ the expected number of steps to reach y from x.
Suppose that at the t–th step, we are at the points x_t

Corollary 3.

For each $x \in B^n$, we have $\pi(x) = \sum_{\varphi} \pi(x, \varphi) = 1/H(x, x)$

The value of $\pi(x)$: the probability to be in the point x on the t for large t

Stopping Rules

- Obviously, x depends on the parameters p and l of algorithm PTS.

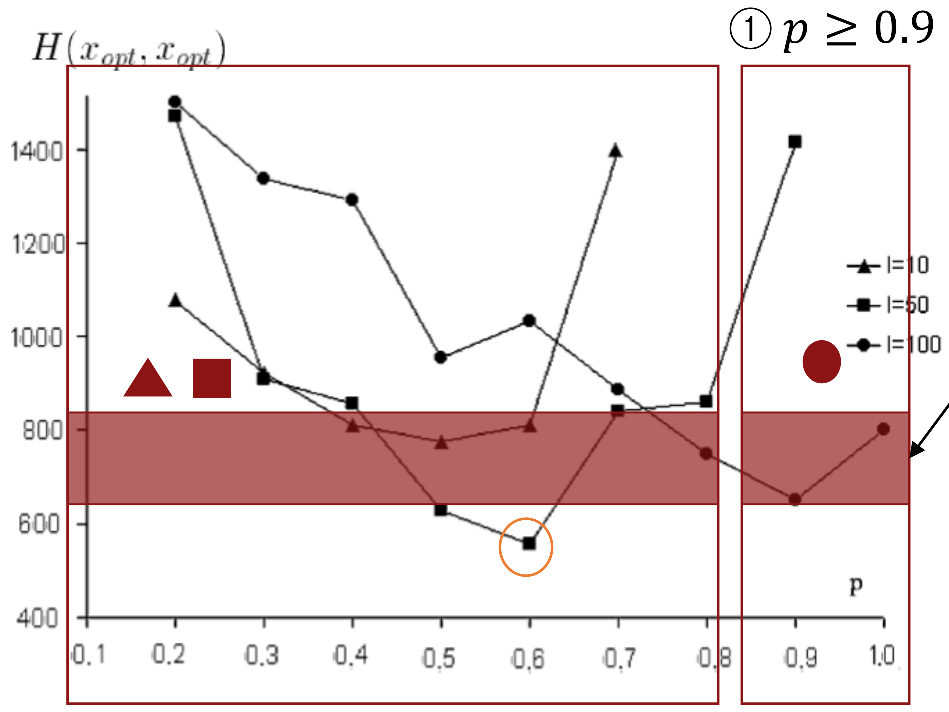


Figure 2: $H(x_{opt}, x_{opt})$ as a function of the threshold p

✓ About p

1. For large values of the threshold $p > 0.9$, the least value of H is achieved on the large tabu list $l = 100$
2. For small p, best results are obtained with small tabu lists $l = 10{,}50$

✓ About the minima

- The pair $p = 0.6, l = 50$ seems to be best values of the parameters H

The best values of the parameters $H(x_{opt}, x_{opt}) \approx 500$

➡ If PTS returns to x very often then the algorithm is stopped and restarted with a new initial point.

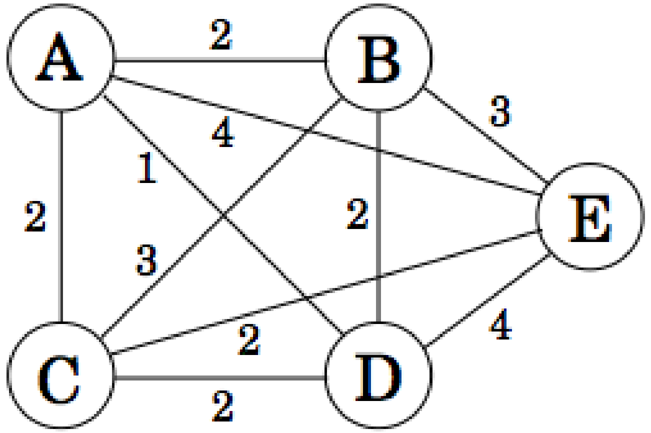
Application of the algorithm for an NP–C/NP–hard example

- Traveling Salesman Problem
  - Finding a shortest closed tour
    - Visiting each node of a given graph with given edge length exactly once.

Figure 3: Traveling Salesman Problem.

Let $G = (X, E, W)$ be a complete weight graph,

$$X = (x_1, x_2, \dots, x_n)\ (n \geq 3)$$
$$E = \{e_{ij} | x_i, x_j \in X\}$$
$$W = \{w_{ij} | w_{ij} \geq 0 \ and \ w_{ii} = 0, \quad for \ all \ i, j \in \{1, 2, \dots, n\}\}$$



Application of the algorithm for an NP–C/NP–hard example

- Solution Representation
  - represented as a sequence of nodes
    - each node appearing only once and in the order it is visited.

3	5	2	4	7	6	8	1

Figure 4: Solution Representation

- Initial solution
  - Each time find the nearest unvisited node from the current node until all the nodes are visited

Application of the algorithm for an NP−C/NP−hard example

- Neighborhood
  - given solution, any other solution that is obtained by a pair wise exchange of any two nodes in the solution.

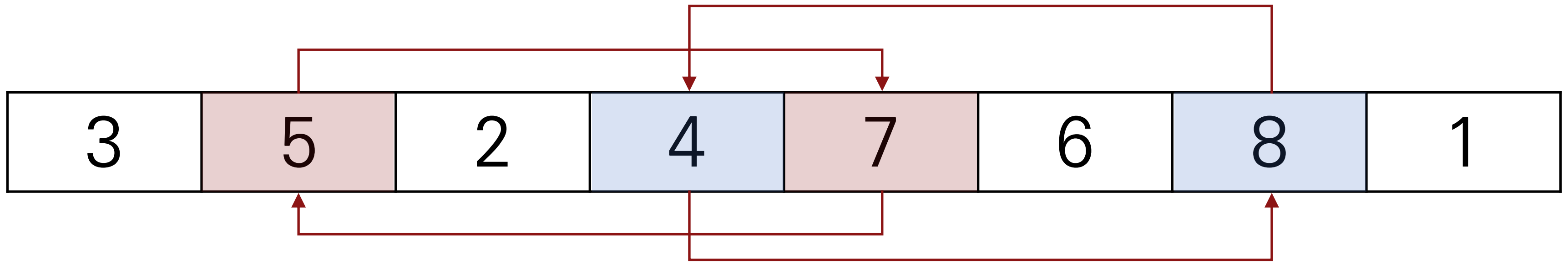
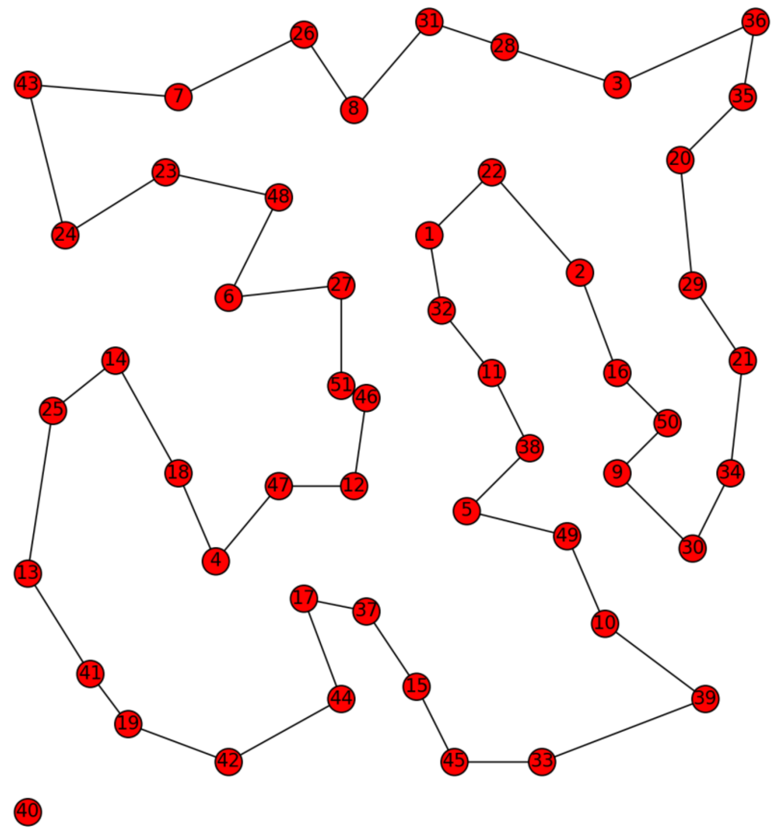


Figure 5: example of Neighborhood solution

- Tabu list
  - the attribute used is a pair of nodes that have been exchanged recently.

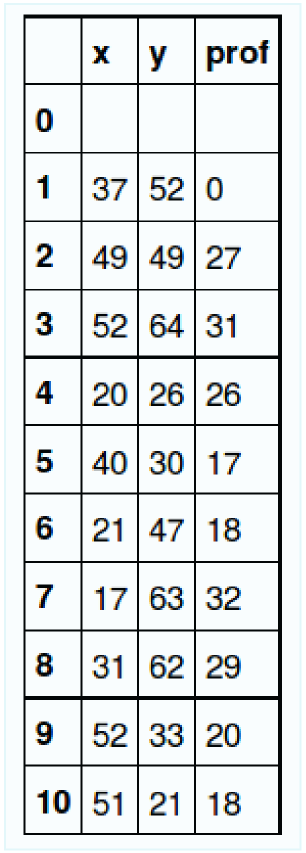
Application of the algorithm for an NP–C/NP–hard example

- Best Objective Value: 704.73
- Number of Customers Visited: 49
- Sequence of Customers Visited
  - [1, 32, 11, 38, 5, 49, 10, 39, 33, 45, 15, 37, 17, 44, 42, 19, 41, 13, 25, 14, 18, 4, 47, 12, 46, 51, 27, 6, 48, 23, 24, 43, 7, 26, 8, 31, 28, 3, 36, 35, 20, 29, 21, 34, 30, 9, 50, 16, 2, 22, 1]
- CPU Time (s): 39.19

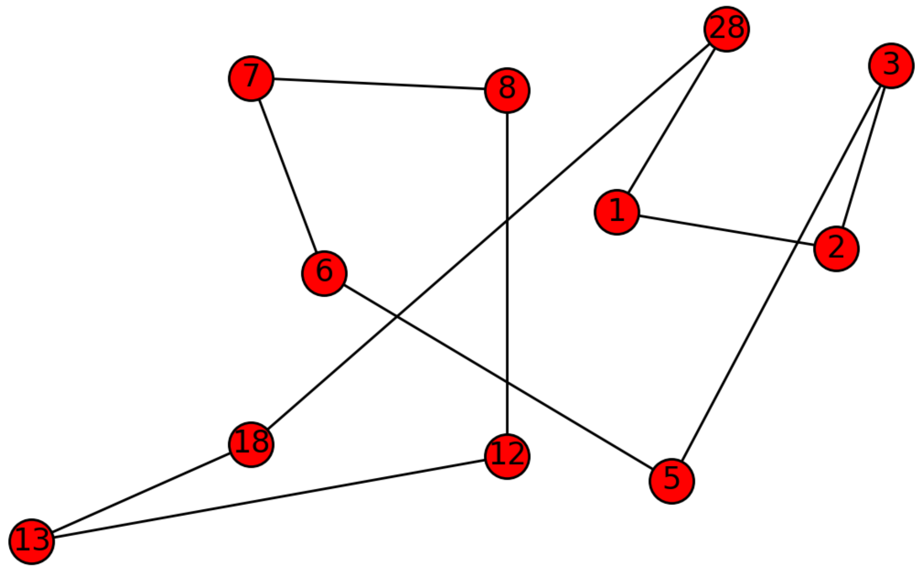


Application of the algorithm for an NP–C/NP–hard example

- Initial solution
  - path = [1,2,3,5,6,7,8,12,13,18,28,1]
  - greedy solution

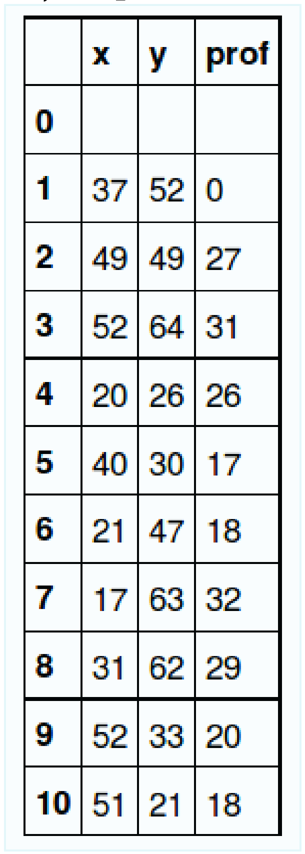


	x	y	prof
0			
1	37	52	0
2	49	49	27
3	52	64	31
4	20	26	26
5	40	30	17
6	21	47	18
7	17	63	32
8	31	62	29
9	52	33	20
10	51	21	18

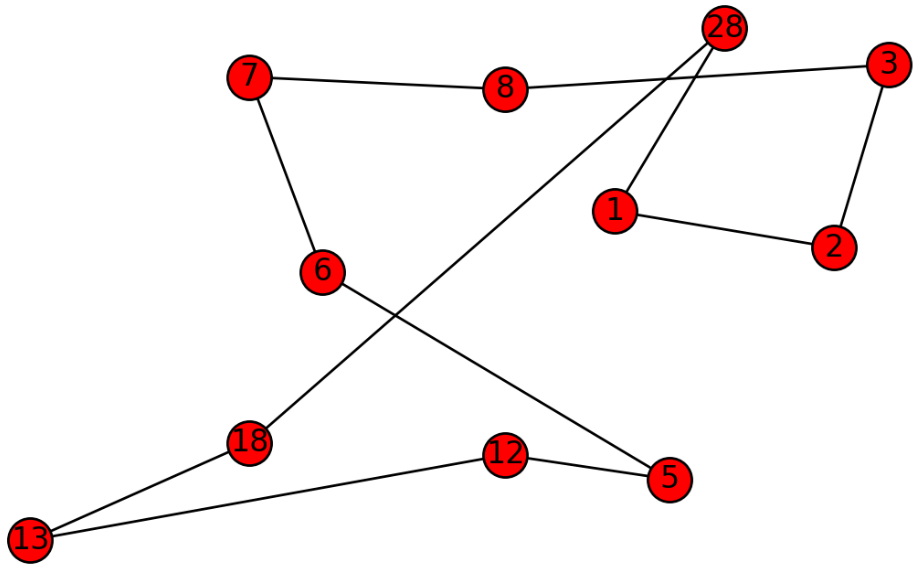


Application of the algorithm for an NP–C/NP–hard example

- Current solution
  - path = [1,2,3,8,7,6,5,12,13,18,28,1]
  - Change → [3,5 – 8,12]



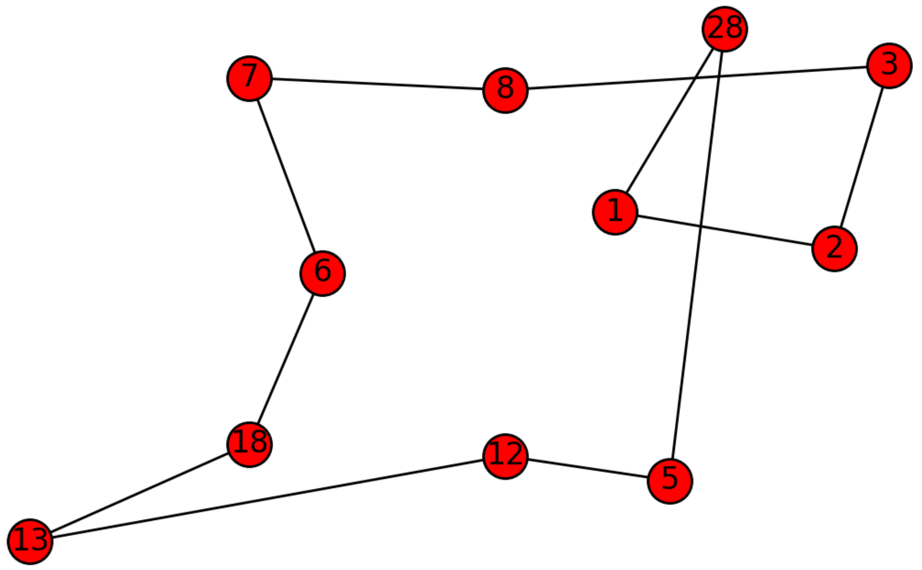
	x	y	prof
0			
1	37	52	0
2	49	49	27
3	52	64	31
4	20	26	26
5	40	30	17
6	21	47	18
7	17	63	32
8	31	62	29
9	52	33	20
10	51	21	18



Application of the algorithm for an NP–C/NP–hard example

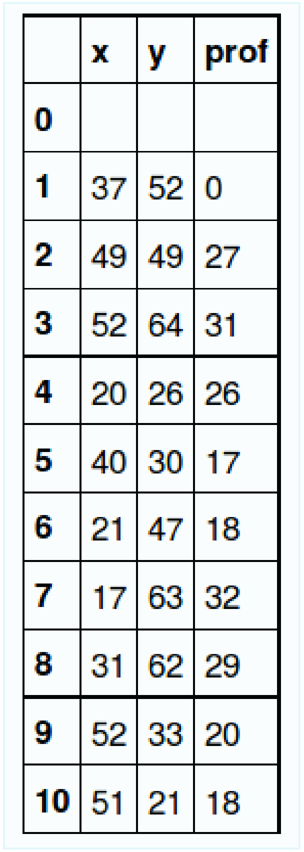
- Current solution
  - path = [1,2,3,8,7,6,18,13,12,5,28,1]
  - Change → [6,5 – 18,28]

	x	y	prof
0			
1	37	52	0
2	49	49	27
3	52	64	31
4	20	26	26
5	40	30	17
6	21	47	18
7	17	63	32
8	31	62	29
9	52	33	20
10	51	21	18

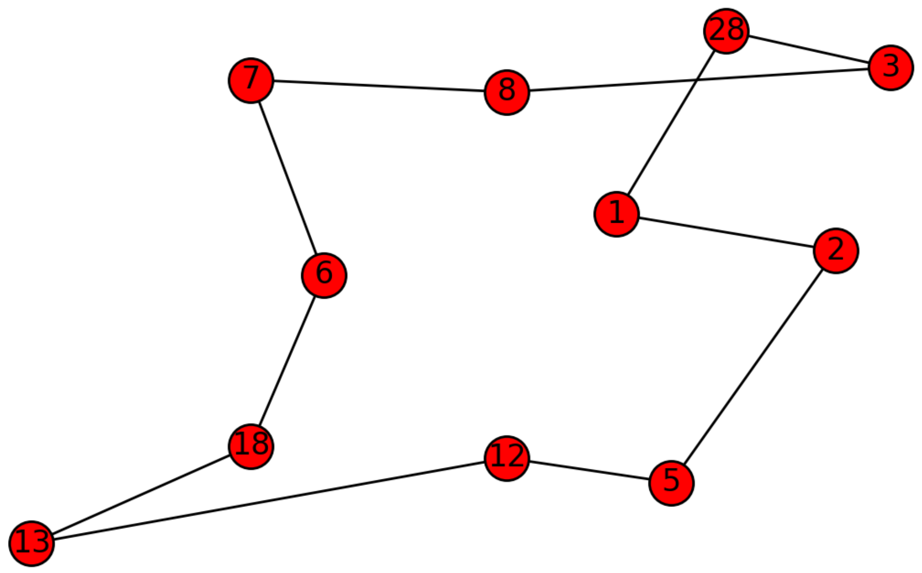


Application of the algorithm for an NP–C/NP–hard example

- Current solution
  - path = [1,2,5,12,13,18,6,7,8,3,28,1]
  - Change → [2,3 – 5,28]



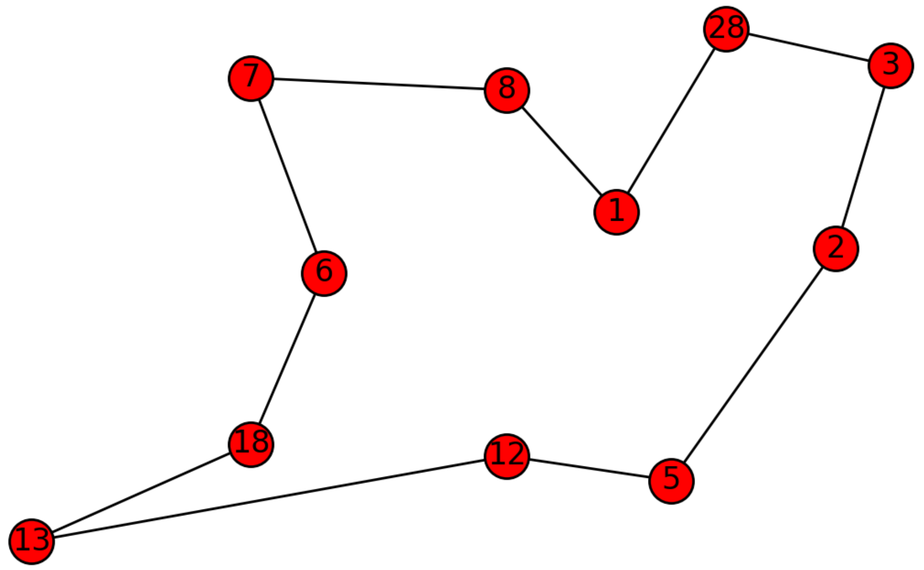
	x	y	prof
0			
1	37	52	0
2	49	49	27
3	52	64	31
4	20	26	26
5	40	30	17
6	21	47	18
7	17	63	32
8	31	62	29
9	52	33	20
10	51	21	18



Application of the algorithm for an NP−C/NP−hard example

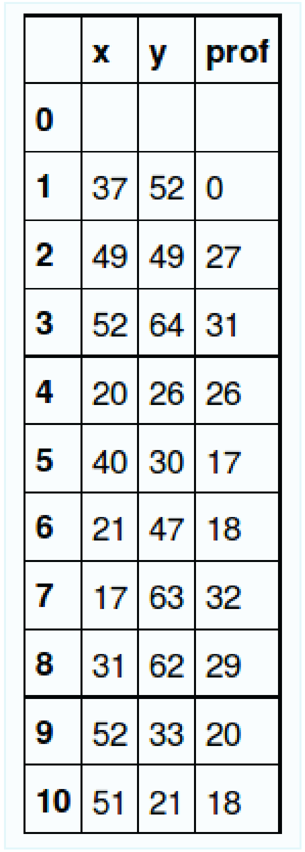
- Current solution
  - path = [1,28,3,2,5,12,13,18,6,7, 8,1]
  - Change → [1,2 – 8,3]

	x	y	prof
0			
1	37	52	0
2	49	49	27
3	52	64	31
4	20	26	26
5	40	30	17
6	21	47	18
7	17	63	32
8	31	62	29
9	52	33	20
10	51	21	18

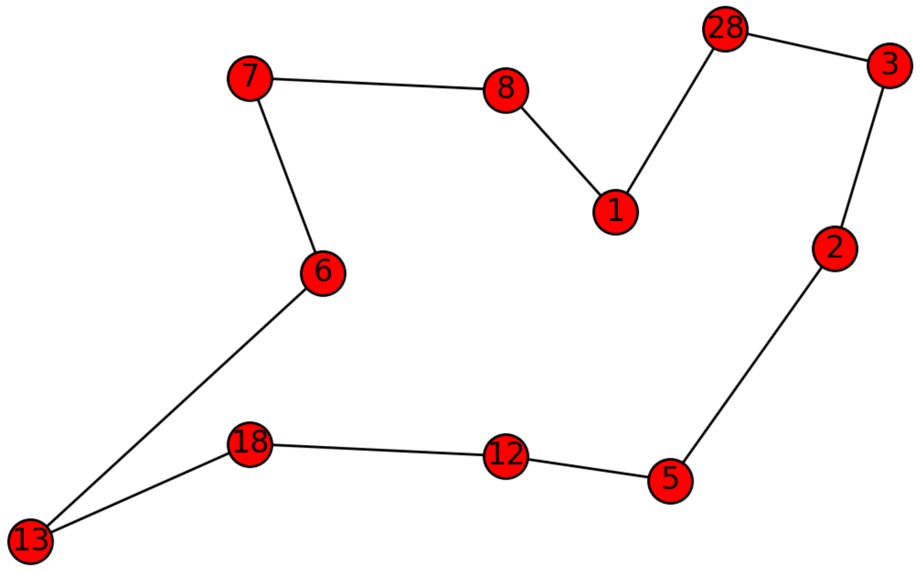


Application of the algorithm for an NP–C/NP–hard example

- Current solution
  - path = [1,28,3,2,5,12,18,13,6,7, 8,1]
  - Change → [6,18 – 13,12]



	x	y	prof
0			
1	37	52	0
2	49	49	27
3	52	64	31
4	20	26	26
5	40	30	17
6	21	47	18
7	17	63	32
8	31	62	29
9	52	33	20
10	51	21	18



References

[1] Kochetov, Yuri A., and Eugene N. Goncharov. "BEHAVIOR OF A PR OBABILISTIC TABU SEARCH ALGORITHM FOR THE MULTI STAGE UN CAPACITATED FACILITY LOCATION PROBLEM."

[2]Kochetov, Yuri A., and Eugene N. Goncharov. "Probabilistic tabu sear ch algorithm for the multi-stage uncapacitated facility location problem ." *Operations research proceedings*. Springer Berlin Heidelberg, 2001.