SỬ DỤNG THUẬT TOÁN DI TRUYỀN GIẢI BÀI TOÁN ĐỊNH TUYẾN XE VỚI RÀNG BUỘC THỜI GIAN

,

Hybrid Genetic Search for the Vehicle Routing Problem with Time Windows: a High-Performance Implementation

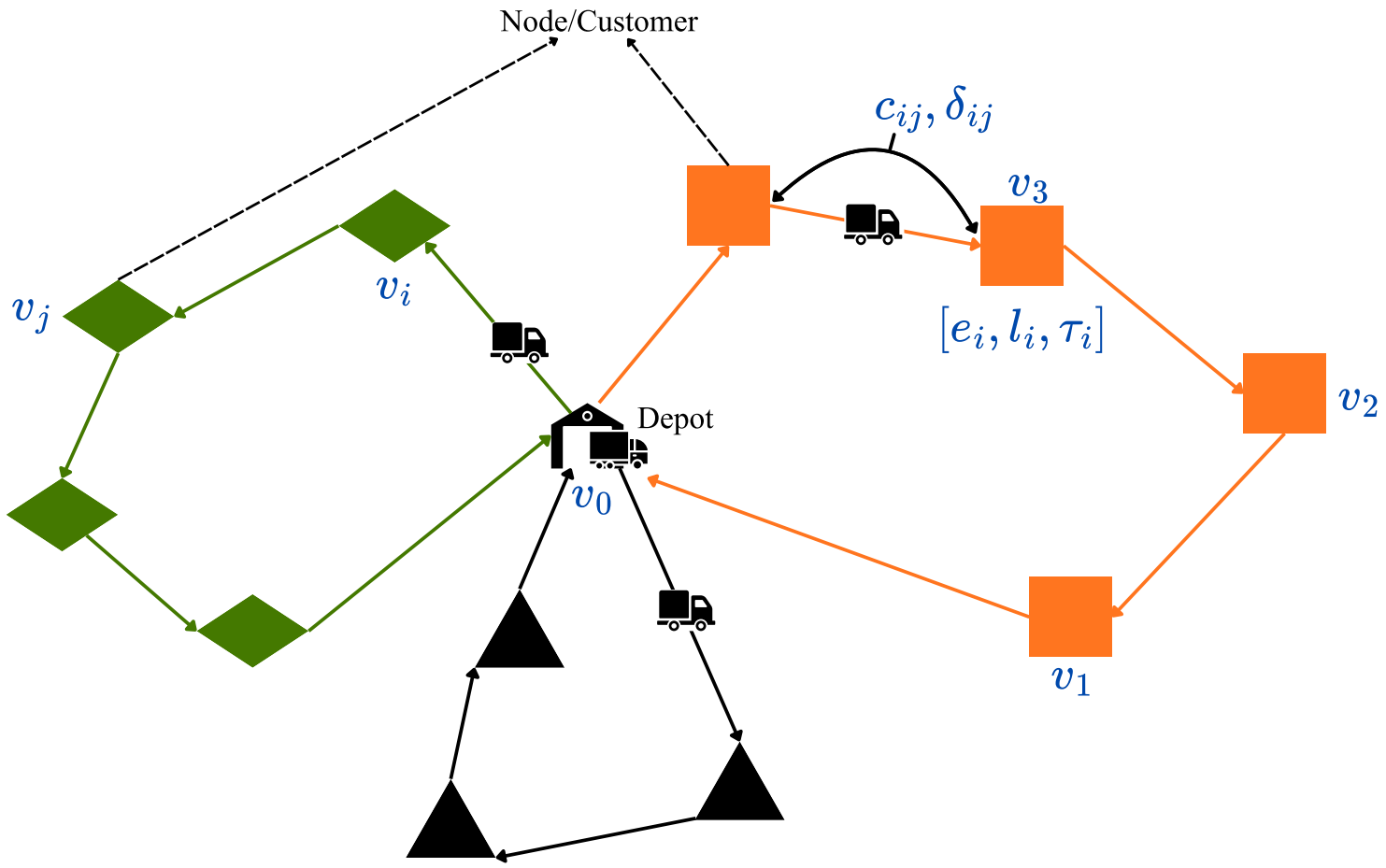
,

Wouter Kool

Mục lục

Mục lục

Bài toán



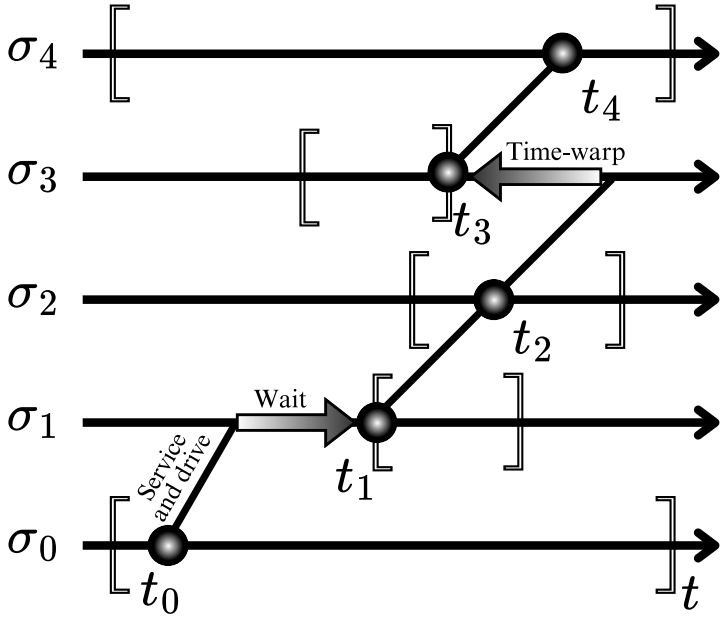
Các nghiên cứu liên quan

Table 1. VRPs of previous studies (source: current study)

Study	Objective				Problem description						Mathematical model and algorithm
					network configuration (N – nodes)	Constraint			Data		
	cost	time	no of vehicle	dis-tance		time window	vehicle capacity	kind of products	experi-ence	reality	
Qi, Hu (2020)	✓		✓		13 N				✓		MIP, heuristic
Kantawong, Pravesjit (2020)	✓			✓	100 N	✓			✓		MIP, artificial bee colony
Londoño *et al.* (2021)	✓			✓	51 N				✓		MILP, local search
Aggarwal, Kumar (2019)	✓		✓		60 N	✓			✓		MIP
Pérez-Rodríguez, Hernández-Aguirre (2019)				✓	6 N	✓				✓	an estimation of distribution
Tasar *et al.* (2019)	✓		✓		100 N				✓		MIP, heuristic
Zhu, Hu (2019)	✓				200 N		✓		✓		MILP, response surface method
Ruiz *et al.* (2019)	✓				14 N		✓			✓	MIP, biased random – key genetic
Zhang *et al.* (2017)	✓				27 N	✓			✓		tabu search, the artificial bee colony
Birim (2016)	✓				10 N		✓		✓		MILP, simulated annealing
Afifi *et al.* (2016)		✓			30 N	✓			✓		MIP, simulated annealing
Spliet, Desaulniers (2015)	✓				60 N	✓	✓		✓		MIP, exact branch – price – cut algorithm
Current study	✓		✓		39 N	✓	✓	✓		✓	MIP, clustering algorithm

Mục lục

Hàm mục tiêu

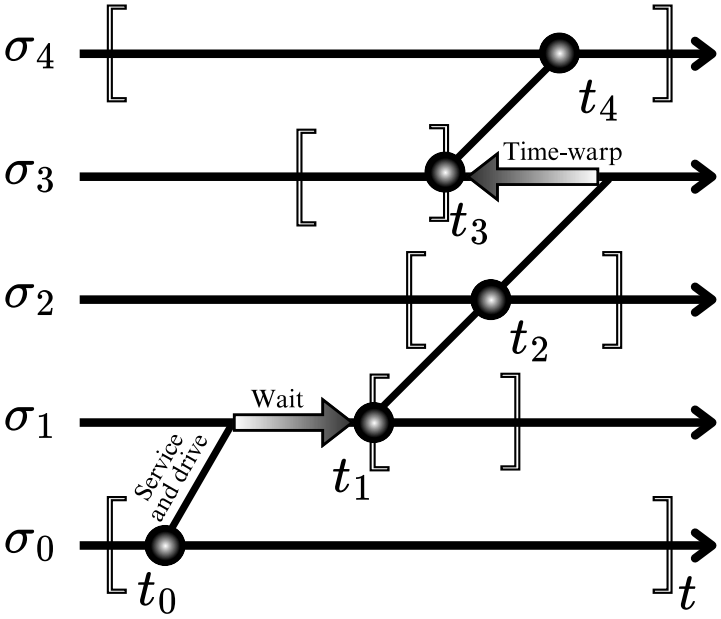


$$r = (\sigma_0^r, \sigma_1^r, ..., \sigma_n^r, \sigma_{n+1}^r)$$

$$\boldsymbol{t}^r = \left(t_0^r, ..., t_{n_r+1}^r\right)$$

$$\mathrm{tw}_{i,i+1} = \max\left(t_i^r + \tau_{\sigma_i^r} + \delta_{\sigma_i^r \sigma_{i+1}^r} - t_{i+1}^r, 0\right)$$

Hàm mục tiêu



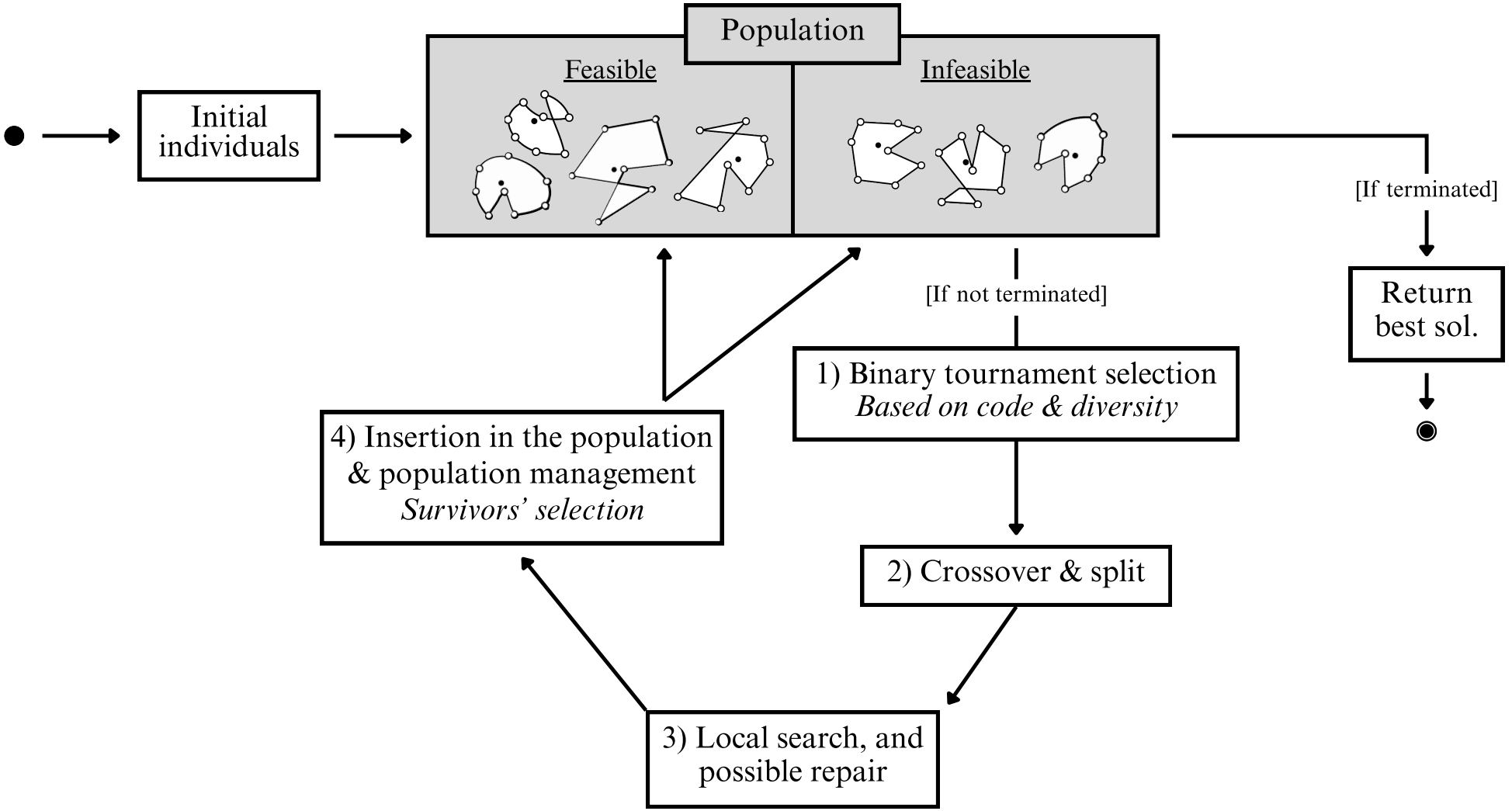
$$q(r) = \sum_{i=1}^{n_r} q_{\sigma_i^r}$$

$$c(r) = \sum_{i=0}^{n_r} c_{\sigma_i^r \sigma_{i+1}^r}$$

$$\mathrm{tw}(r) = \sum_{i=0}^{n_r} \mathrm{tw}_{i,i+1}$$

$$\varphi(r) = c_r + \omega^Q \max(0, q(r) - Q) + \omega^{\mathrm{TW}}\ \mathrm{tw}(r)$$

Tổng quan


Initial individuals
Population
Feasible
Infeasible
[If terminated]
Return best sol.
[If not terminated]
1) Binary tournament selection
Based on code & diversity
2) Crossover & split
3) Local search, and possible repair
4) Insertion in the population & population management
Survivors' selection

Heuristic kiến trúc

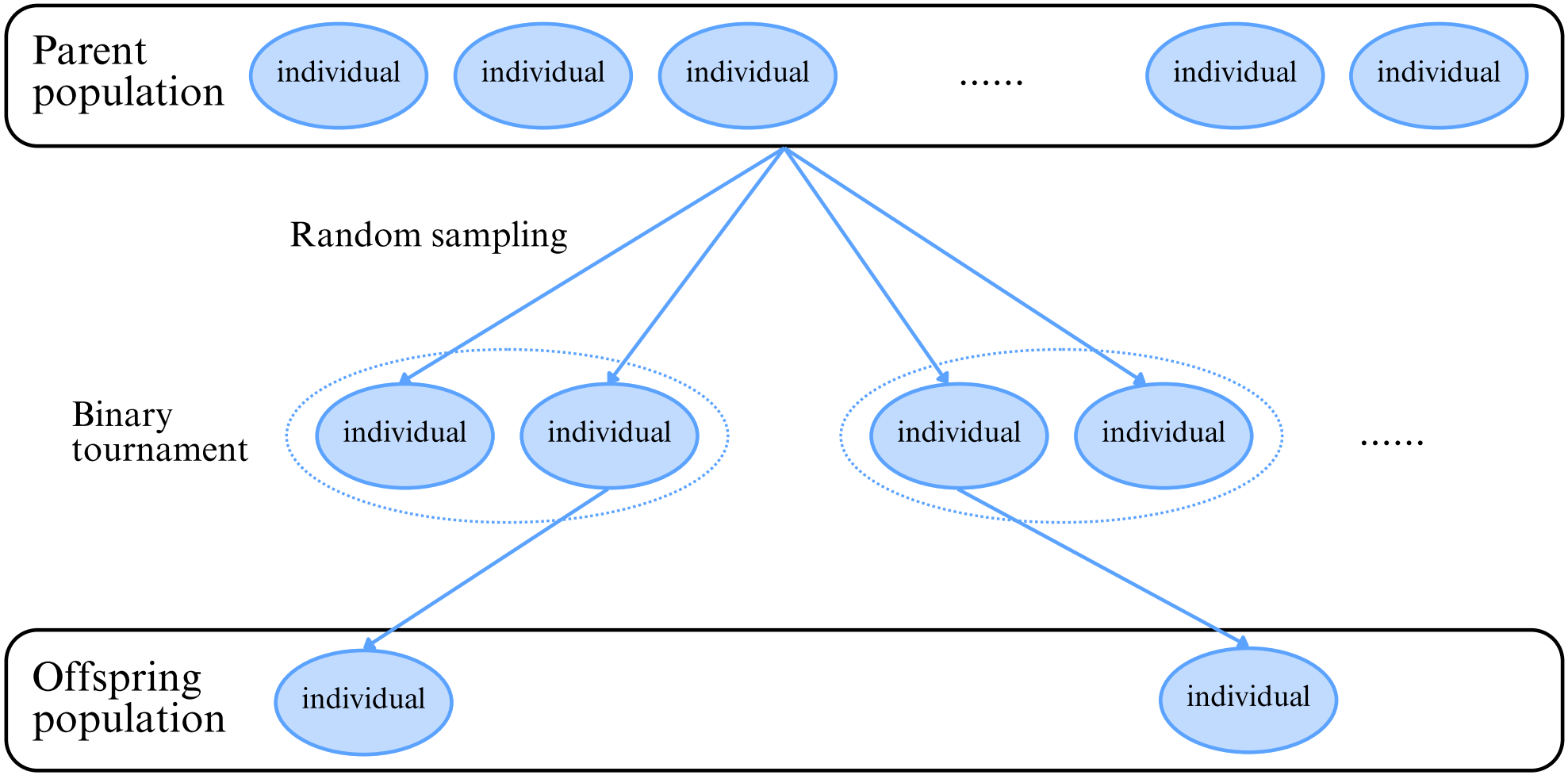
Algorithm 1: Nearest/farthest algorithm

```
1  S = {1, 2, 3, ..., n}
2  solutions ← []
3  for t = 1 to m:
4      customers ← []
5      i ← nearest/farthest customers from depot ∈ S
6      while S is not empty:
7          j ← a customer ∈ S
                 that can be add to `customers` and causes least detour distance
8          if j not exists:
9              break
10         add j to customers, remove j from S
11     add customers to solutions
```

Sweep

- Sắp xếp các khách hàng theo góc giữa nó và điểm xuất phát.
- Thêm các khách hàng cho đến khi quá trọng tải.
- Với mỗi chuyến đi:
  - Sắp xếp các khách hàng có time window ngắn ($< 50\%$) tăng dần theo l_i
  - Chèn lại các khách hàng có time window dài sao cho khoảng cách tăng là ít nhất.

Lai tạo


Parent population
individual
individual
individual
.....
individual
individual
Random sampling
Binary tournament
individual
individual
individual
individual
.....
Offspring population
individual
individual

Đánh giá độ phù hợp

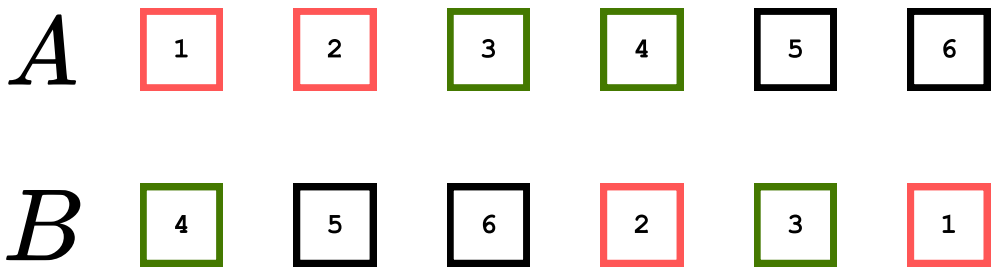
$$f_P(S) = f_P^{\varphi}(S) + \left(1 - \frac{n^{\text{ELITE}}}{|P|}\right) f_P^{\text{DIV}}(S)$$

- $f_P^{\varphi}(S)$ là hạng của lời giải S, sắp xếp theo chất lượng lời giải.

Đánh giá độ phù hợp

$$f_P(S) = f_P^{\varphi}(S) + \left(1 - \frac{n^{\text{ELITE}}}{|P|}\right) f_P^{\text{DIV}}(S)$$

- $f_P^{\text{DIV}}(S)$ là hạng của lời giải S, khi xét khả năng mở rộng
- $\Delta(S) = \frac{1}{n^{\text{CLOSEST}}} \sum_{S_2} d(S, S_2)$



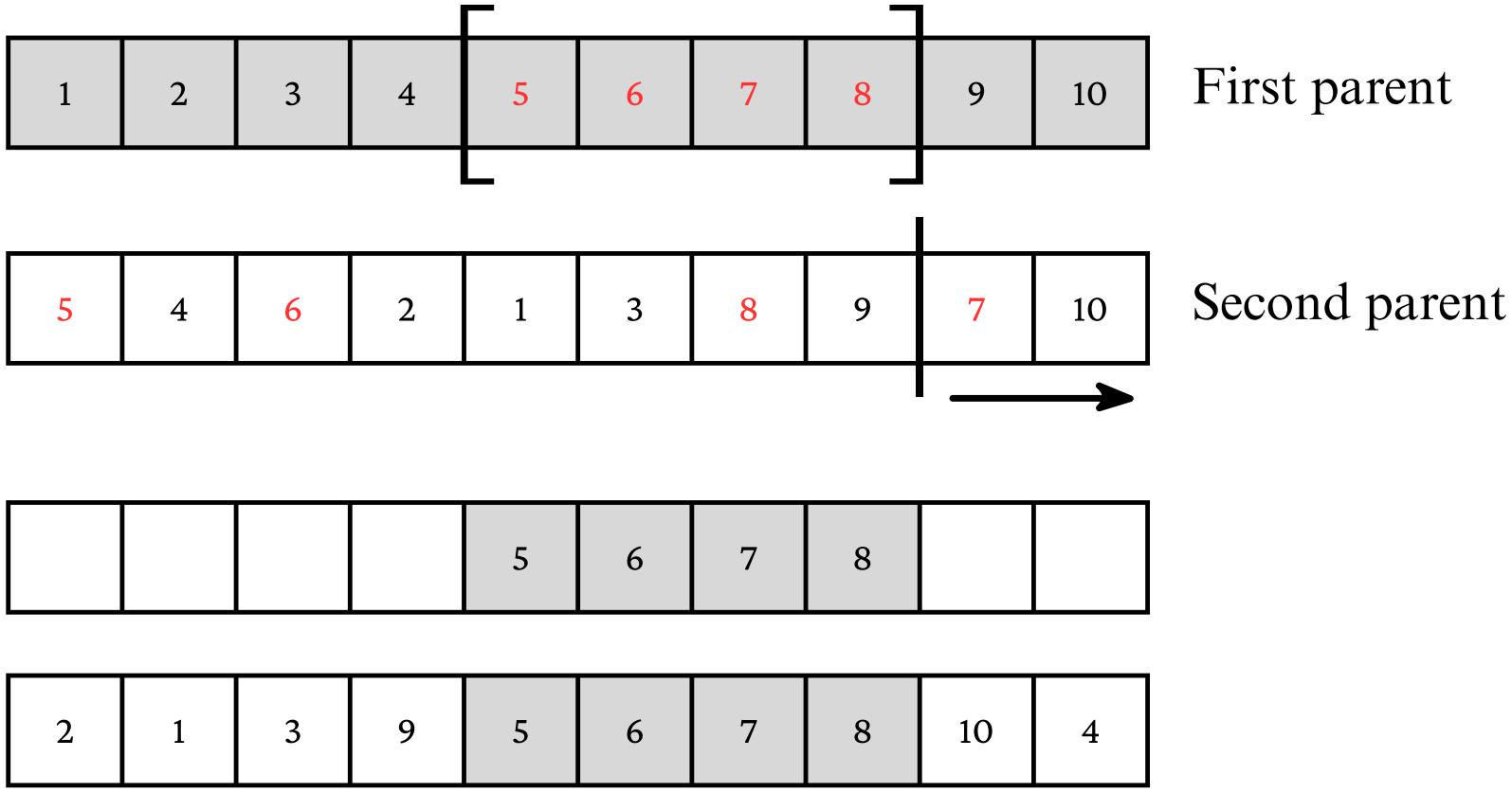
$$d(A, B) = 2$$

Đánh giá độ phù hợp

$$f_P(S) = f_P^{\varphi}(S) + \left(1 - \frac{n^{\text{ELITE}}}{|P|}\right) f_P^{\text{DIV}}(S)$$

- Hệ số $\left(1 - \frac{n^{\text{ELITE}}}{|P|}\right)$ được sử dụng để đảm bảo ta vẫn giữ lại được n^{ELITE} lời giải chất lượng tốt nhất trong suốt quá trình tìm kiếm

Crossover operator



Thuật toán SPLIT

Algorithm 2: Classical Split Algorithm: Fleet limited to m vehicles

```
1  for k ← 1 to m do
2      for t ← 1 to n do
3          f[k][t] ← ∞;
4  f[0][0] ← 0
5  for k ← 1 to m do
6      for i ← 0 to n do
7          j ← i + 1;
8          while j ≤ n and canAdd(j) do
9              if f[k][j] > f[k − 1][i] + cost(i + 1, j) then
10                 f[k][j] ← f[k − 1][i] + cost(i + 1, j);
11                 pred[k][j] ← i
12             j ← j + 1;
```

Thuật toán SPLIT

$$dominates(i,j) \equiv \begin{cases} p[i] + d_{0,i+1} - D[i+1] + \alpha \times (Q[j] - Q[i]) \leq p[j] + d_{0,j+1} - D[j+1] & \text{if } i < j \\ p[i] + d_{0,i+1} - D[i+1] \leq p[j] + d_{0,j+1} - D[j+1] & \text{if } i > j \end{cases}$$

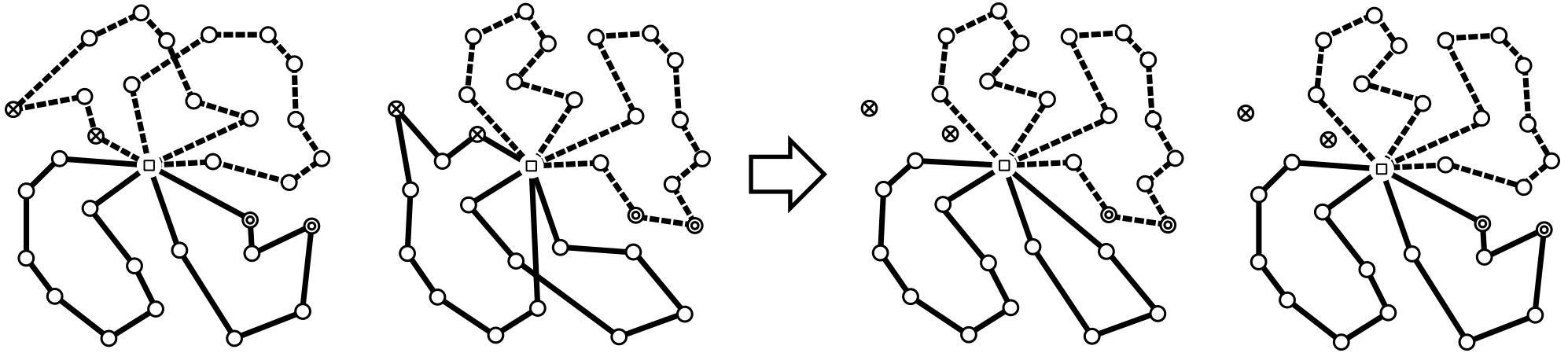
Trong đó:

- $c(i,j) = d_{0,i+1} + D[j] - D[i+1] + d_{j,0} + \alpha \times \max\{Q[j] - Q[i] - Q, 0\}$
- $D[i] = \sum_{k=1}^{i-1} d_{k,k+1}$
- $Q[i] = \sum_{k=1}^{i} q_k$

Algorithm 3: Linear Split

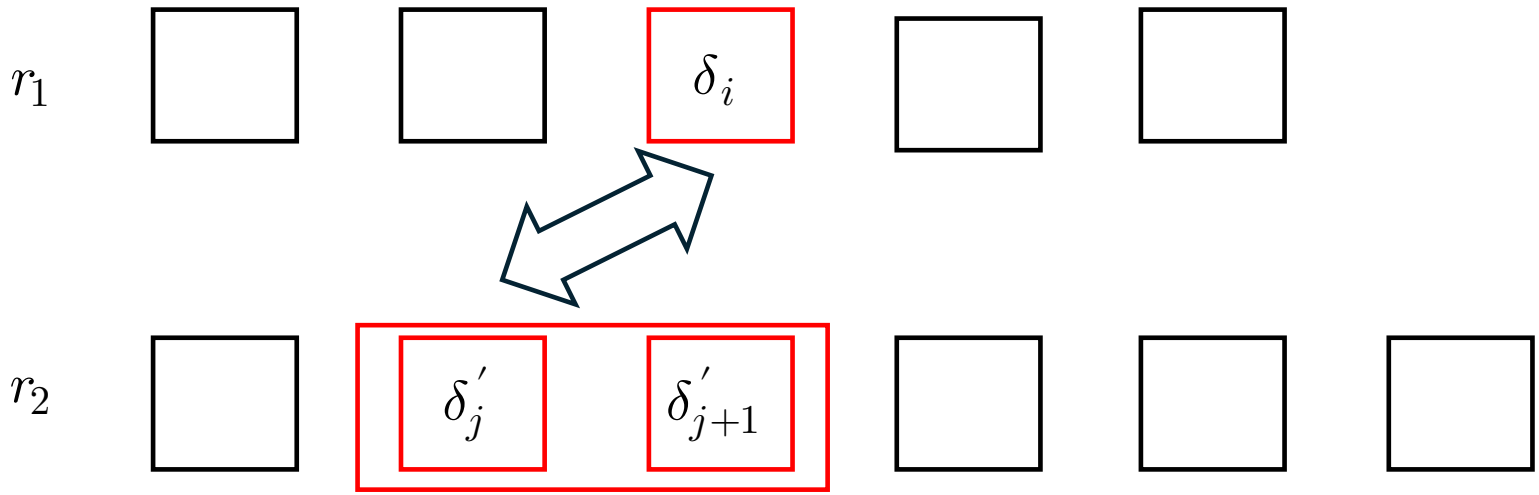
```
1  p[0] ← 0
2  S ← (0)
3  for t ← 0 to n do
4      p[t] ← p[front] + f(front, t)
5      pred[t] ← front
6      if t < n then
7          if not dominates(back, t) then
8              while |S| > 0 and dominates(back, t) do
9                  popBack()
10             pushBack(t)
11         while |S| > 1 and p[front] + f(front, t + 1) ≥ p[front2] + f(front2, t + 1)
12             popFront()
```

Selective Route Exchange (SREX)



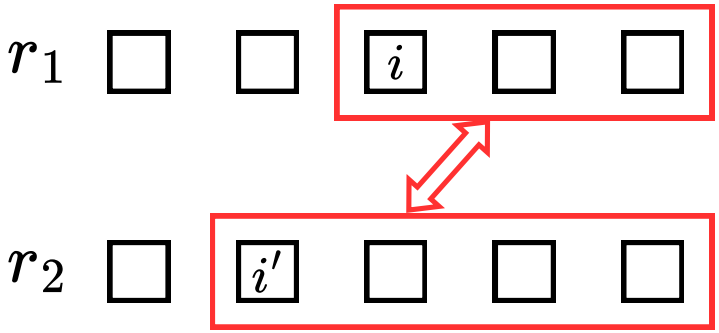
σ^p_A σ^p_B σ^c_I σ^c_{II}

Local Search

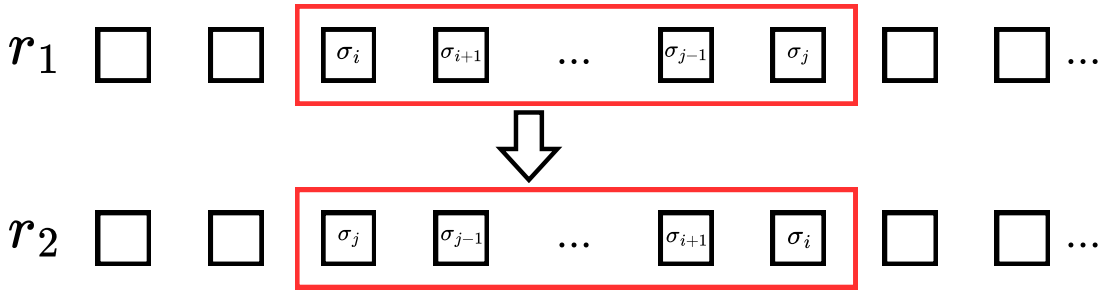


Swap and relocate

Local Search

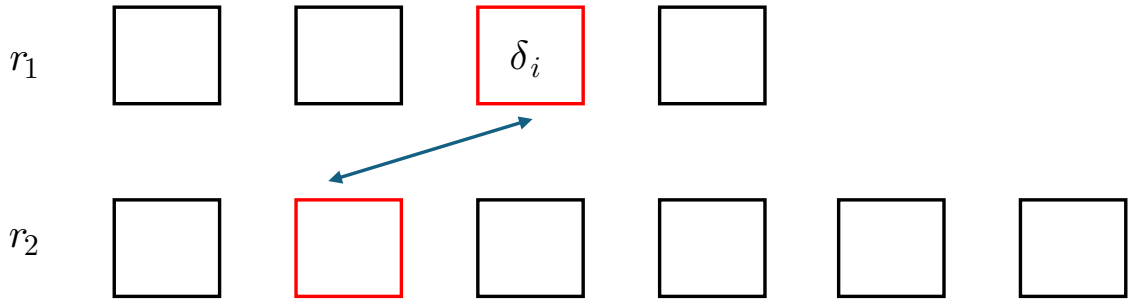


$2 - opt^*$

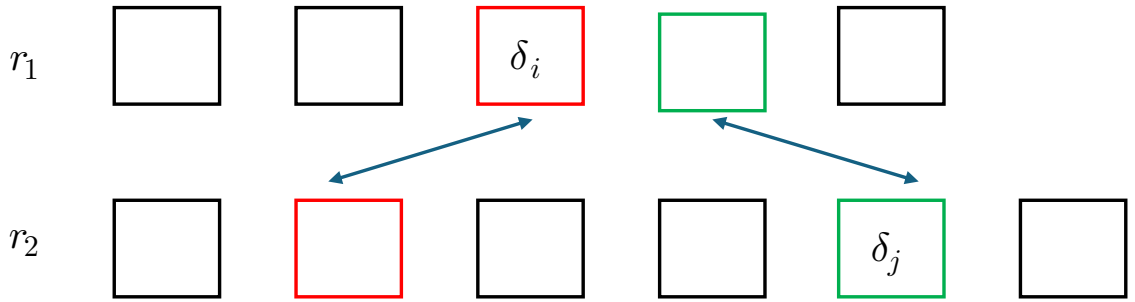


$2 - opt$

Local Search



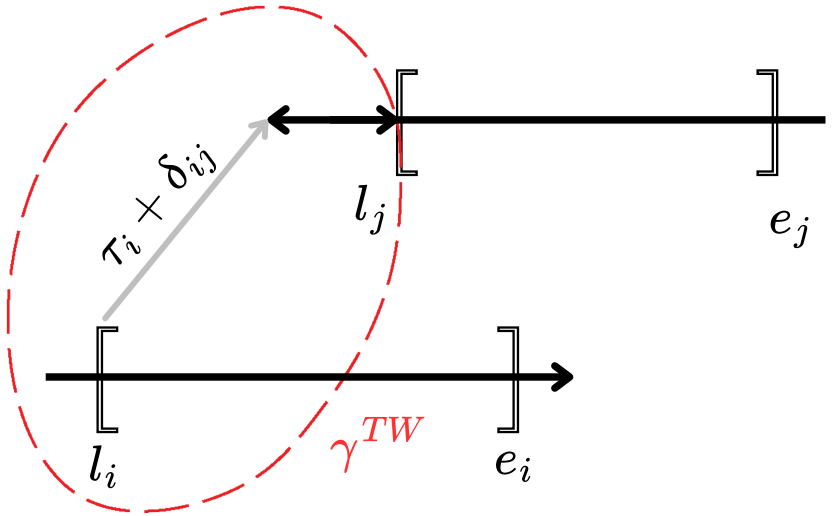
Relocate*

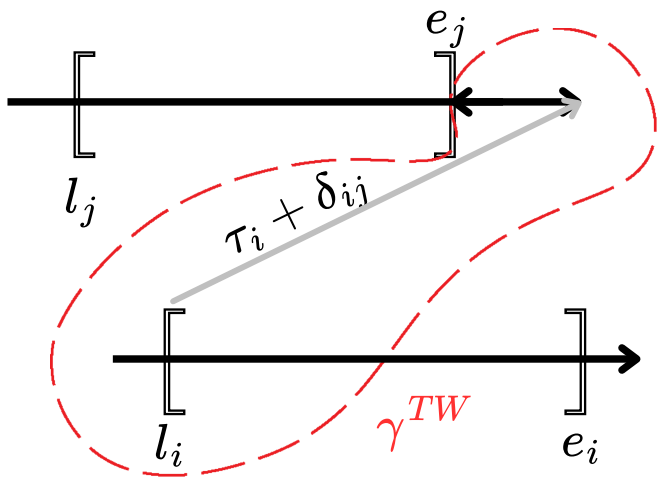


Swap*

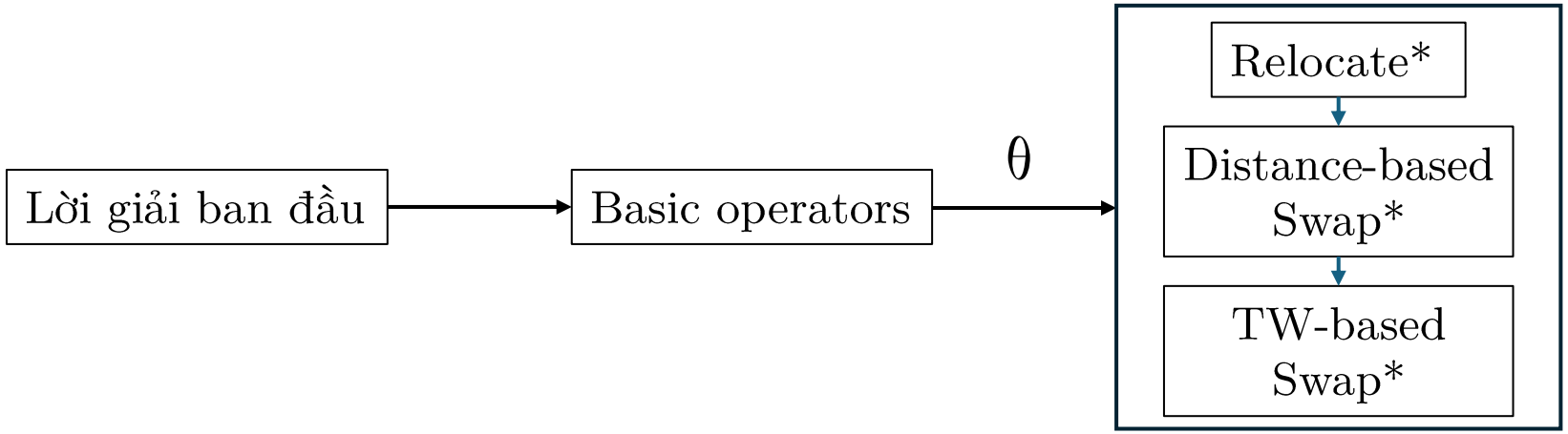
Với mỗi khách hàng i, ta định nghĩa một tập $\Gamma(i)$ gồm Γ khách hàng gần i nhất theo độ đo tương quan dưới đây:

$$\gamma(i,j) = c_{ij} + \gamma^{\mathrm{WT}} \max\left(e_j - \tau_i - \delta_{ij} - l_i, 0\right) + \gamma^{\mathrm{TW}} \max\left(e_i + \tau_i + \delta_{ij} - l_j, 0\right)$$





Local Search



Lựa chọn tham số

$$\textit{Ban đầu:} \begin{cases} \omega^{Q} & \leftarrow 1 \\ \omega^{\mathrm{TW}} & \leftarrow 1 \end{cases}$$

$$\textit{Sau 100 lần lặp: } \omega^{Q}, \omega^{\mathrm{TW}} \text{ sẽ } \begin{cases} \text{tăng 20\% nếu } < 15\% \text{ lời giải hợp lệ} \\ \text{giảm 15\% nếu } > 25\% \text{ lời giải hợp lệ} \\ \text{tăng lên 100\% nếu không tìm được lời giải hợp lệ} \end{cases}$$

Lựa chọn tham số

Các hằng số cố định: $n^{\text{ELITE}} = 4$, $n^{\text{CLOSEST}} = 5$, $\lambda = 40$.

Có hành trình dài (> 25 khách)	Có khách hàng với ràng buộc thời gian rộng ($> 70\%$)	Hành động
✓		$\theta = 15\%, \mu = 25, \Gamma = 40,$ tăng μ và Γ lên 5 sau 10000 lần lặp
✗	✗	$\theta = 100\%, \mu = 25, \Gamma = 40,$ tăng μ và Γ lên 5 sau 10000 lần lặp
✗	✓	$\theta = 100\%, \mu = 25, \Gamma = 20$,tăng μ và Γ lên 5 sau 20000 lần lặp

Sau 10000 lần lặp mà không có cải tiến nào, ta reset lại quần thể và vẫn giữ nguyên tham số cũ.

Lược đồ thuật toán

```
1  Initialize population;
2  while number of iterations without improvement < It_NI and time < T_MAX do
3      Select parent solutions P_1 and P_2;
4      Apply crossover operators on P_1 and P_2 to generate an offspring C;
5      Educate offspring C by local search;
6      Insert C into respective subpopulation;
7      if C is infeasible then
8          With 50% probability, repair C (local search) and insert it into respective
           subpopulation;
9      if maximum subpopulation size reached then
10         Select survivors;
11     Adjust penalty coefficients for infeasibility;
12 Return best feasible solution;
```

Mục lục

Kết quả

Thuật toán được tác giả chạy thử nghiệm trên bộ dữ liệu của Solomon và Homberger[1], và cho ra kết quả như sau:

Dataset	C1	C2	R1	R2	RC1	RC2	Mean
Solomon	0,000%	0,000%	-0,003%	0,000%	0,000%	0,000%	0,000%
GH200	0,000%	0,004%	0,001%	0,009%	0,016%	0,026%	0,009%
GH400	0,000%	0,000%	-0,009%	0,028%	-0,030%	-0,050%	-0,010%
GH600	-0,014%	0,022%	0,047%	-0,022%	-0,012%	-0,123%	-0,017%
GH800	0,030%	-0,018%	0,147%	0,090%	0,112%	-0,222%	0,023%
GH1000	0,123%	-0,013%	0,174%	-0,090%	0,094%	-0,158%	0,022%
Mean	0,023%	-0,001%	0,060%	0,002%	0,030%	-0,088%	0,004%

(a) Gap to reference solution

Dataset	C1	C2	R1	R2	RC1	RC2	Mean
Solomon	0,000%	0,000%	-0,001%	0,002%	0,002%	0,001%	0,001%
GH200	0,001%	0,006%	0,014%	0,016%	0,025%	0,033%	0,016%
GH400	0,011%	0,014%	0,051%	0,065%	0,026%	-0,017%	0,025%
GH600	0,037%	0,060%	0,252%	0,128%	0,181%	0,010%	0,111%
GH800	0,082%	0,028%	0,424%	0,286%	0,312%	0,037%	0,195%
GH1000	0,207%	0,020%	0,479%	0,188%	0,319%	0,048%	0,210%
Mean	0,056%	0,021%	0,203%	0,114%	0,144%	0,019%	0,093%

(b) Primal Integral (PI)

[1]https://www.sintef.no/projectweb/top/vrptw/

Experiments