Phương pháp tìm kiếm lân cận rộng thích ứng cho một số lớp bài toán định tuyến phương tiện

Nguyễn Mạnh Linh

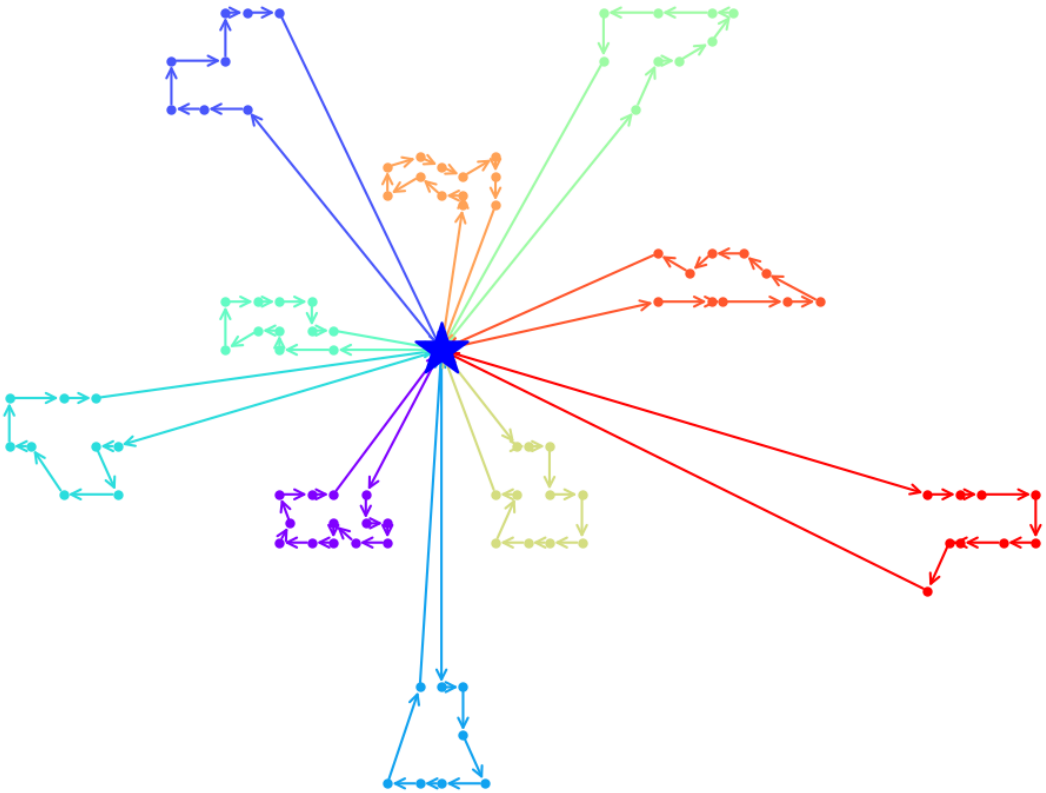
Khoa Toán-Cơ-Tin học
Đại học Khoa học Tự nhiên

2023/12

Outline

Giới thiệu

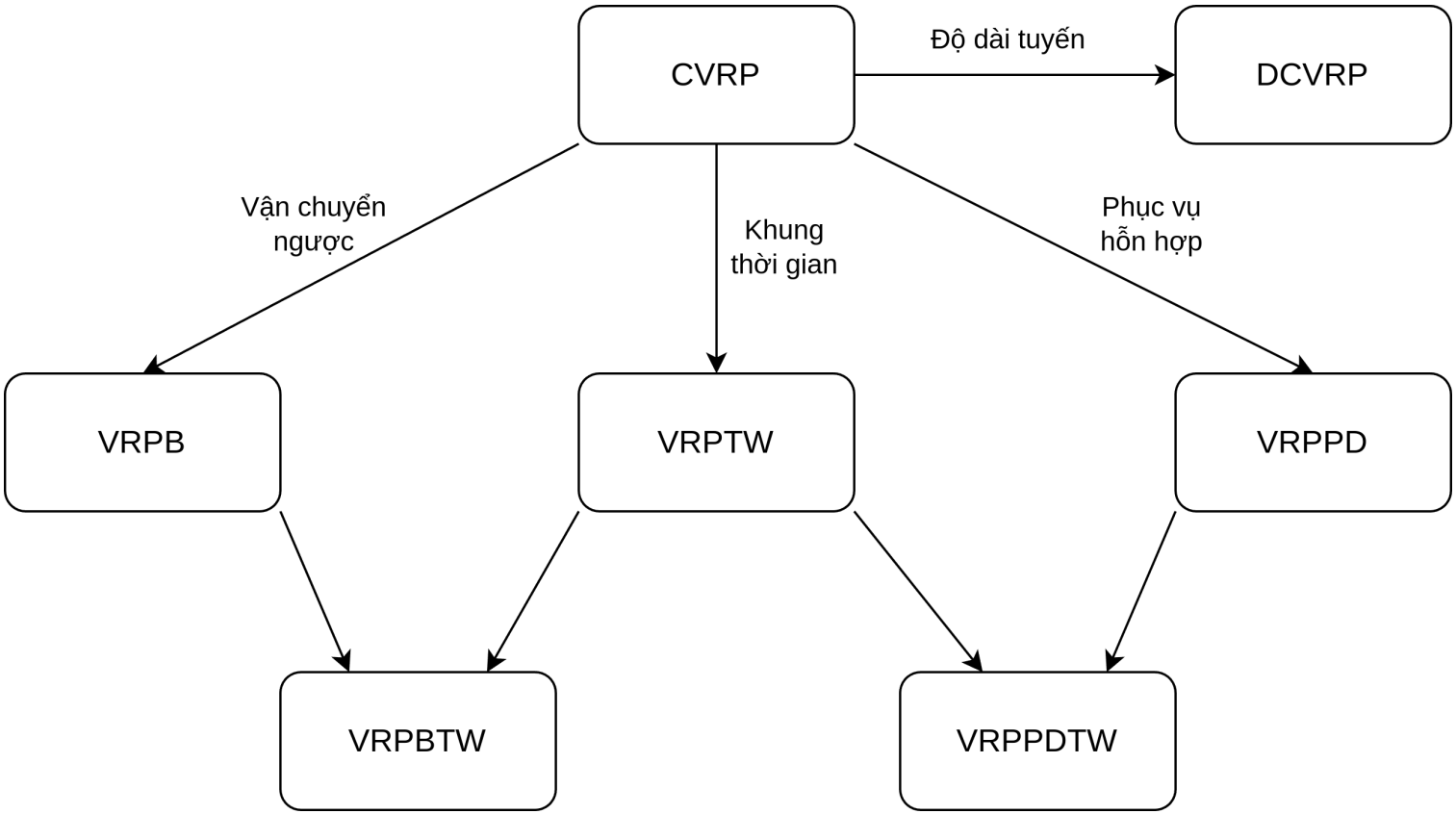
Giới thiệu



Hình 1: VRP với 10 xe phục vụ 100 khách hàng (cấu hình Solomon C101)

Outline

Định nghĩa



Hình 2: Các bài toán, biến thể của VRP

Định nghĩa

Định nghĩa theo ngôn ngữ lý thuyết đồ thị

- Gọi $G = (V, A)$ là một độ thị đầy đủ với $V = \{0, ..., n\}$ là tập nút và A là tập cung. Các nút $i = 1, ..., n$ đại diện cho các yêu cầu cần phục vụ, nút 0 là kho hàng.
- Một số không âm được gọi là chi phí c_{ij} đại diện cho mỗi cung $(i, j) \in A$.
- Bất đẳng thức tam giác cho chi phí

$$c_{ij} + c_{jk} \geq c_{ik}, \quad \forall i, j, k \in V. \tag{1}$$

Định nghĩa

Định nghĩa theo ngôn ngữ lý thuyết đồ thị

- Mỗi yêu cầu i có một nhu cầu (về tải trọng) là d_i và nhu cầu của kho $d_0 = 0$. Gọi $d(S) = \sum_{i \in S} d_i$ là tổng nhu cầu của tập.
- Tập K đại diện cho các xe, mỗi xe có tải trọng C, các xe ở kho tại thời điểm bắt đầu. Giả thiết $d_i \leq C$ với mỗi $i = 1, ..., n$.
- K_{min} là số xe ít nhất cần để phục vụ toàn bộ khách hàng.
- Với một tập $S \subseteq V \setminus \{0\}$, gọi $r(S)$ là số xe ít nhất để phục vụ toàn bộ khách hàng thuộc tập S.

Định nghĩa - CVRP

Bài toán định tuyến xe với ràng buộc tải trọng (*Capacited Vehicle Routing Problem - CVRP*) yêu cầu tìm một tập chính xác K các chu trình (mỗi chu trình ứng với một tuyến đường) với tổng chi phí của tất cả các cung thuộc các chu trình này là nhỏ nhất.

Các ràng buộc

(i) Mỗi chu trình đều đi qua nút ứng với kho hàng.

(ii) Mỗi nút ứng với một khách hàng được đi qua bởi đúng một chu trình.

(iii) Tổng nhu cầu của các khách hàng trong mỗi chu trình không được vượt quá tải trọng của xe.

Định nghĩa - VRPTW

VRPTW

- Bài toán định tuyến xe với ràng buộc thời gian - VRPTW (*VRP with time windows*) mở rộng CVRP. Mỗi khách hàng i bị ràng buộc bởi một khoảng thời gian $[a_i, b_i]$ (*time window*), thời gian phục vụ khách hàng i là s_i.
- Nếu xe đến nút i sớm thì phải chờ đến thời gian a_i mới được phục vụ. Nếu xe đến nút i muộn hơn b_i thì khách hàng sẽ không được phục vụ.

Định nghĩa - VRPTW

VRPTW yêu cầu tìm một tập chính xác K chu trình với tổng chi phí là nhỏ nhất.

Các ràng buộc

(i) Mỗi chu trình đều đi qua nút ứng với kho hàng.

(ii) Mỗi nút ứng với một khách hàng được đi qua bởi đúng một chu trình.

(iii) Tổng nhu cầu của các khách hàng trong mỗi chu trình không được vượt quá tải trọng của xe.

(iv) Với mỗi khách hàng i, thời gian bắt đầu phục vụ phải nằm trong khung thời gian $[a_i, b_i]$ và xe hoàn thành việc phục vụ sau khoảng thời gian s_i.

Định nghĩa - kí hiệu

Mô hình - CVRP

Công thức dòng xe hai chỉ số

x_{ij} nhận giá trị bằng 1 nếu cung $(i, j) \in A$ nằm trong nghiệm tối ưu và 0 nếu trong trường hợp còn lại.

$$\text{(VRP1)} \quad \min \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij} \tag{2}$$

s.t.

$$\sum_{i \in V} x_{ij} = 1, \quad \forall j \in V \setminus \{0\}, \tag{3}$$

$$\sum_{j \in V} x_{ij} = 1, \quad \forall i \in V \setminus \{0\}, \tag{4}$$

$$\sum_{i \in V} x_{i0} = K, \tag{5}$$

Mô hình - CVRP

$$\sum_{j \in V} x_{0j} = K, \tag{6}$$

$$\sum_{i \notin S} \sum_{j \in S} x_{ij} \geq r(S), \quad \forall S \subseteq V \setminus \{0\}, S \neq \emptyset, \tag{7}$$

$$x_{ij} \in \{0, 1\}, \quad \forall i, j \in V. \tag{8}$$

Mô hình - VRPTW

Công thức dòng xe ba chỉ số

x_{ijk} nhận giá trị 1 nếu xe k đi trực tiếp từ nút i tới nút j và 0 nếu ngược lại.

$$\min \sum_{k \in K} \sum_{(i,j) \in A} c_{ij} x_{ijk} \tag{9}$$

s.t.

$$\sum_{k \in K} \sum_{j \in \Delta^+(i)} x_{ijk} = 1, \quad \forall i \in N, \tag{10}$$

$$\sum_{j \in \Delta^+(0)} x_{0jk} = 1, \quad \forall k \in K, \tag{11}$$

$$\sum_{i \in \Delta^-(j)} x_{ijk} - \sum_{i \in \Delta^+(j)} x_{jik} = 0, \quad \forall k \in K, j \in N, \tag{12}$$

$$\sum_{i \in \Delta^-(n+1)} x_{i,n+1,k} = 1, \quad \forall k \in K, \tag{13}$$

$$x_{ijk}(w_{ik} + s_i + t_{ij} - w_{jk}) \leq 0, \quad \forall k \in K, (i,j) \in A, \tag{14}$$

$$a_i \sum_{j \in \Delta^+(i)} x_{ijk} \leq w_{ik} \leq b_i \sum_{j \in \Delta^+(i)} x_{ijk}, \quad \forall k \in K, i \in N, \tag{15}$$

$$E \leq w_{ik} \leq L, \quad \forall k \in K, i \in \{0, n+1\}, \tag{16}$$

$$\sum_{i \in N} d_i \sum_{j \in \Delta^+(i)} x_{ijk} \leq C, \quad \forall k \in K, \tag{17}$$

$$x_{ijk} \geq 0, \quad \forall k \in K, (i,j) \in A, \tag{18}$$

$$x_{ijk} \in \{0,1\}, \quad \forall k \in K, (i,j) \in A. \tag{19}$$

Outline

Phương pháp - Thuật toán chính xác

- Phương pháp nhánh cận
- Quy hoạch động
- Công thức dòng xe
- Công thức dòng hàng
- Công thức phân hoạch tập hợp

Phương pháp - Heuristic cổ điển

- Thuật toán tiết kiệm
- Phân cụm trước, định tuyến sau
- Heuristic cải tiến

Phương pháp - Metaheuristics

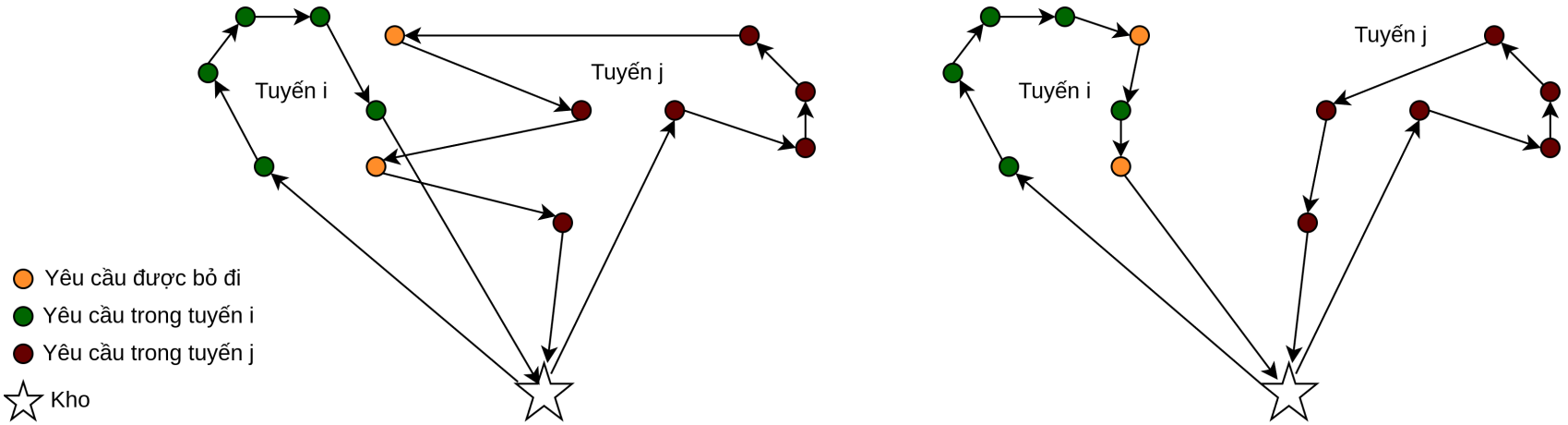
- Tìm kiếm địa phương
- Tìm kiếm quần thể
- Cơ chế học

Outline

Tìm kiếm lân cận

- Lân cận
- Tối ưu địa phương
- Tiêu chí chấp nhận nghiệm
- Điều kiện dừng

Tìm kiếm lân cận rộng



Hình 3: Lược đồ LNS

Tìm kiếm lân cận rộng (*LNS*)

Thuật toán 1 LNS Heuristic

Require: $s \in \{\text{nghiệm chấp nhận được}\}, q \in \mathbb{N}$

1: nghiệm $s_{best} = s$;
2: **repeat**
3: $s' = s$;
4: bỏ q yêu cầu từ s';
5: thêm lại các yêu cầu đã bỏ đi vào s';
6: **if** $f(s') < f(s)$ **then**
7: $s_{best} = s'$;
8: **if** $accept(s', s)$ **then**
9: $s = s'$;
10: **until** đạt điều kiện dừng
11: **return** s_{best};

Hình 4: Mã giả LNS

LNS - Thuật toán hủy

- Xóa yêu cầu tệ nhất
- Xóa ngẫu nhiên
- Thuật toán xóa Shaw
- Thuật toán xóa tuyến tệ

LNS - Thuật toán hủy

Xóa yêu cầu tệ nhất

- Mảng L: sắp xếp các yêu cầu theo thứ tự giảm dần của chi phí (sau khi bỏ đi yêu cầu đó).
- Chọn một số ngẫu nhiên $y \in (0, 1)$.
- Xóa yêu cầu $r = L[y^p|L|]$.

LNS - Thuật toán hủy

Xóa tuyến tệ nhất

Chi phí trung bình trên mỗi yêu cầu với nhiễu

$$\text{avg\_cost}(r, s) = \frac{\text{cost}(r, s)}{\text{size}(r) - 1} + \lambda \phi d_{max} \tag{20}$$

với r là tuyến được chọn, $\text{cost}(r, s)$ là chi phí của tuyến r trong nghiệm s, $\text{size}(r)$ là số phần tử của tuyến r, λ là hệ số nhiễu, ϕ là một số ngẫu nhiên trong khoảng $(0, 1)$, d_{max} là khoảng cách lớn nhất trong tập yêu cầu.
Xóa đi tuyến có chi phí trung bình trên mỗi yêu cầu lớn nhất và có số yêu cầu nhỏ hơn số yêu cầu trung bình trên mỗi tuyến (với một ngưỡng nhất định).

LNS - Thuật toán hủy

Phương pháp xóa Shaw

Chỉ số độ tương đồng

$$R(i,j) = \varphi d_{ij} + \chi|t_i - t_j| + \psi|l_i - l_j| \tag{21}$$

với d_{ij} là khoảng cách từ i tới j, t_i là thời gian khi đến địa điểm i, l_i là tải của xe tại i.

LNS - Thuật toán hủy

Thuật toán 3 Xóa yêu cầu - Shaw

Require: $s \in \{\text{nghiệm chấp nhận được}\}, q \in \mathbb{N}, p \in \mathbb{R}_+$

1: yêu cầu r được chọn ngẫu nhiên từ s;
2: tập hợp yêu cầu $\mathbb{D} = \{r\}$;
3: **while** $|\mathbb{D}| < q$ **do**
4: r được chọn ngẫu nhiên từ $\mathbb{D}$;
5: Mảng L chứa tất cả các yêu cầu thuộc s và không thuộc $\mathbb{D}$;
6: sắp xếp L sao cho $i < j \Rightarrow R(r, L[i]) < R(r, L[j])$;
7: chọn một số ngẫu nhiên y trong khoảng $[0, 1)$;
8: $\mathbb{D} = \mathbb{D} \cup L[y^p|L|]$
9: xóa các yêu cầu trong $\mathbb{D}$ khỏi s;

Hình 5: Thuật toán xóa Shaw

LNS - Thuật toán sửa

- Chèn yêu cầu tham lam (với nhiễu)
- Heuristic hối tiếc

LNS - Thuật toán sửa

Chèn yêu cầu tham lam

- Khi chèn thêm yêu cầu u và giữa hai yêu cầu i và j trong tuyến k thì chi phí tăng thêm là $\Delta f_{u,i,j,k} = d_{iu} + d_{uj} - d_{ij}$.
- Chèn thêm yêu cầu vào vị trí làm tăng ít chi phí nhất.
- Có thể thêm nhiễu vào hàm tăng chi phí.

$$\Delta f_{u,i,j,k} := \Delta f_{u,i,j,k} + \lambda p d_{\max} \tag{22}$$

với $d_{\max}$ là khoảng cách lớn nhất giữa hai yêu cầu, p là một số ngẫu nhiên trong khoảng $(-1, 1)$ và λ là một hằng số điều khiển.

LNS - Thuật toán sửa

Heuristic hối tiếc

- Đặt $x_{ik} \in \{1, ..., m\}$ là biến biểu diễn tuyến đường cho yêu cầu i có chi phí chèn thêm vào thấp thứ k.
- Giá trị *regret* $c_i^* = \Delta f_{i,x_{i2}} - \Delta f_{i,x_{i1}}$.
- Chọn ra yêu cầu i thỏa mãn $\max_{i \in V \setminus \{0\}} \{c_i^*\}$ và chèn vào vị trí tương ứng.
- Mở rộng regret-k, chọn yêu cầu thỏa mãn

$$\max_{i \in V \setminus \{0\}} \{\sum_{j=1}^{k} (\Delta f_{i,x_{ij}} - \Delta f_{i,x_{i1}})\}. \tag{23}$$

LNS - Tiêu chí chấp nhận nghiệm

Bước ngẫu nhiên - Ramdom Walk

Mọi nghiệm s' đều được chấp nhận.

Chấp nhận tham lam - Greedy Acceptance

Nghiệm s' được chấp nhận nếu chi phí của nó là nhỏ hơn so với nghiệm hiện tại.

Mô phỏng luyện kim - Simulated Annealing

Mọi nghiệm cải thiện s' được chấp nhận. Nếu $c(s') > c(s)$ thì s' được chấp nhận với xác suất $\exp\{\frac{c(s)-c(s')}{T}\}$ với T là nhiệt độ. Nhiệt độ T giảm sau mỗi vòng lặp với một hệ số Φ.

LNS - Tiêu chí chấp nhận nghiệm

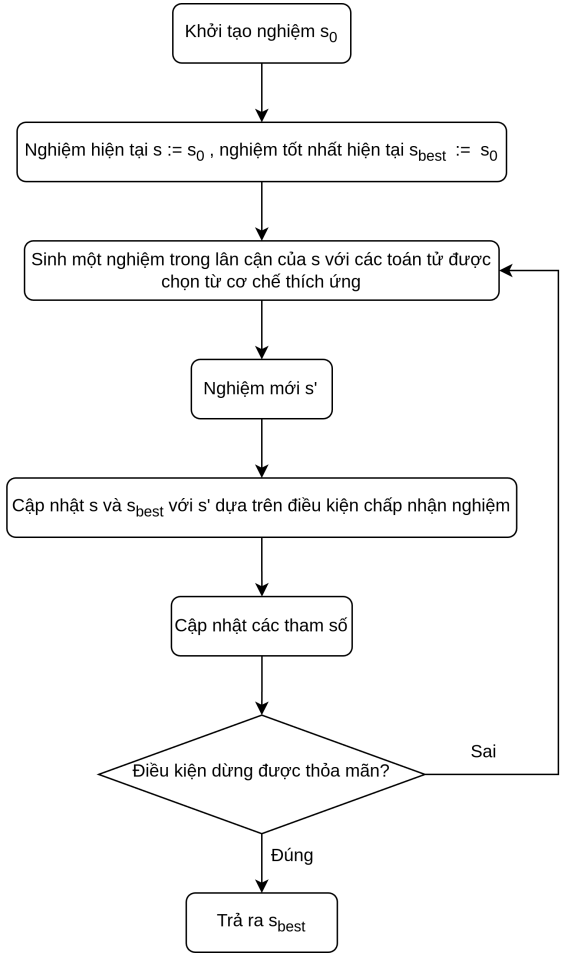
Chấp nhận với ngưỡng - Threshold Acceptance

Nghiệm s' được chấp nhận nếu $c(s') - c(s) < T$ với T là ngưỡng, ngưỡng này được giảm sau mỗi vòng lặp với hệ số Φ.

Đại hồng thủy - Great Deluge Algorithm

Nghiệm s' được chấp nhận nếu $c(s') < L$ với một ngưỡng L, ngưỡng này chỉ giảm nếu nghiệm được chấp nhận, và giảm với hệ số Φ.

Tìm kiếm lân cận rộng thích ứng (*ALNS*)



Hình 6: Lược đồ chính của ALNS

ALNS - Lựa chọn thuật toán

Lựa chọn thuật toán hủy và thêm lại

Gán cho mỗi heuristic một trọng số khác nhau và sử dụng nguyên tắc "bánh xe lựa chọn". Nếu có k heuristic với trọng số $w_i, i \in \{1, ..., k\}$, ta chọn heuristic j với xác suất

$$p_j = \frac{w_j}{\sum_{i=1}^{k} w_i}. \tag{24}$$

ALNS - Điều chỉnh tham số tự động

- Trọng số được điều chỉnh mỗi khi có nghiệm mới được chấp nhận.
- Mỗi heuristic được gán điểm khác nhau và được điều chỉnh tùy thuộc vào tình huống.
- Cập nhật trọng số sau mỗi bước.

ALNS - Điều chỉnh tham số tự động

Điểm của mỗi heuristic được đặt là 0 khi bắt đầu và được tăng thêm $\sigma_1, \sigma_2, \sigma_3$ tùy thuộc vào tình huống.

- σ_1 khi hành động xóa-chèn cuối cùng dẫn đến một nghiệm mới tốt hơn nghiệm tốt nhất toàn cục.
- σ_2 khi hành động xóa-chèn cuối cùng dẫn đến một nghiệm chưa được chấp nhận trước đó, chi phí tốt hơn chi phí của nghiệm hiện tại.
- σ_3 khi hành động xóa-chèn cuối cùng dẫn đến một nghiệm chưa được chấp nhận trước đó, chi phí của nghiệm mới tệ hơn chi phí của nghiệm hiện tại nhưng thỏa mãn điều kiện chấp nhận nghiệm.

ALNS - Điều chỉnh tham số tự động

ω_{ij} là trọng số của heuristic i được sử dụng tại bước j
Khi bước j kết thúc, ta tính toán trọng số cho tất cả heuristic i để sử dụng cho bước thứ $j+1$

$$\omega_{i,j+1} = \omega_{ij}(1-r) + r\frac{\pi_i}{\theta_i}. \tag{25}$$

Trong đó, π_i là điểm số của heuristic i được nhận trong bước cuối cùng, θ_i là số lần ta cố gắng sử dụng heuristic i trong bước thực hiện đó, r là tham số điều khiển.

ALNS - B-ALNS

B-ALNS

Thêm nhiễu khi điều chỉnh tham số tự động. Giả sử sau m vòng lặp, chúng ta mới lại có một nghiệm được chấp nhận từ lần cuối cùng nghiệm được chấp nhận.

$$\omega_{i,j+1} = \omega_{ij}(1-r) + r\frac{\pi_i}{\theta_i} + \alpha\beta(1-e^{-\gamma m}) \tag{26}$$

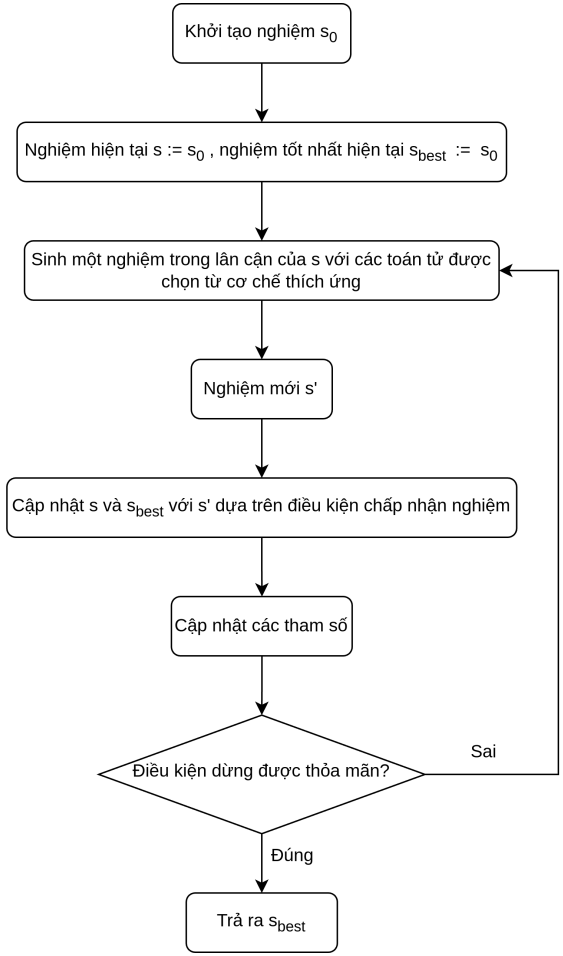
với α (có thể âm hoặc dương) và γ (dương) là các tham số điều khiển, β là một số ngẫu nhiên trong khoảng $(0, 1)$.

ALNS - Số lượng yêu cầu bỏ đi, thêm lại

- Bỏ đi một số lượng yêu cầu cố định.
- Chọn ngẫu nhiên số lượng bỏ đi trong một khoảng nhất định.

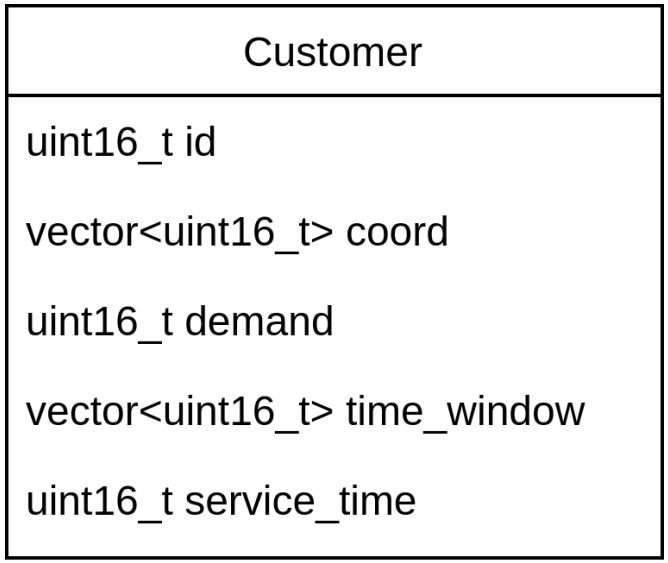
Outline

Ứng dụng



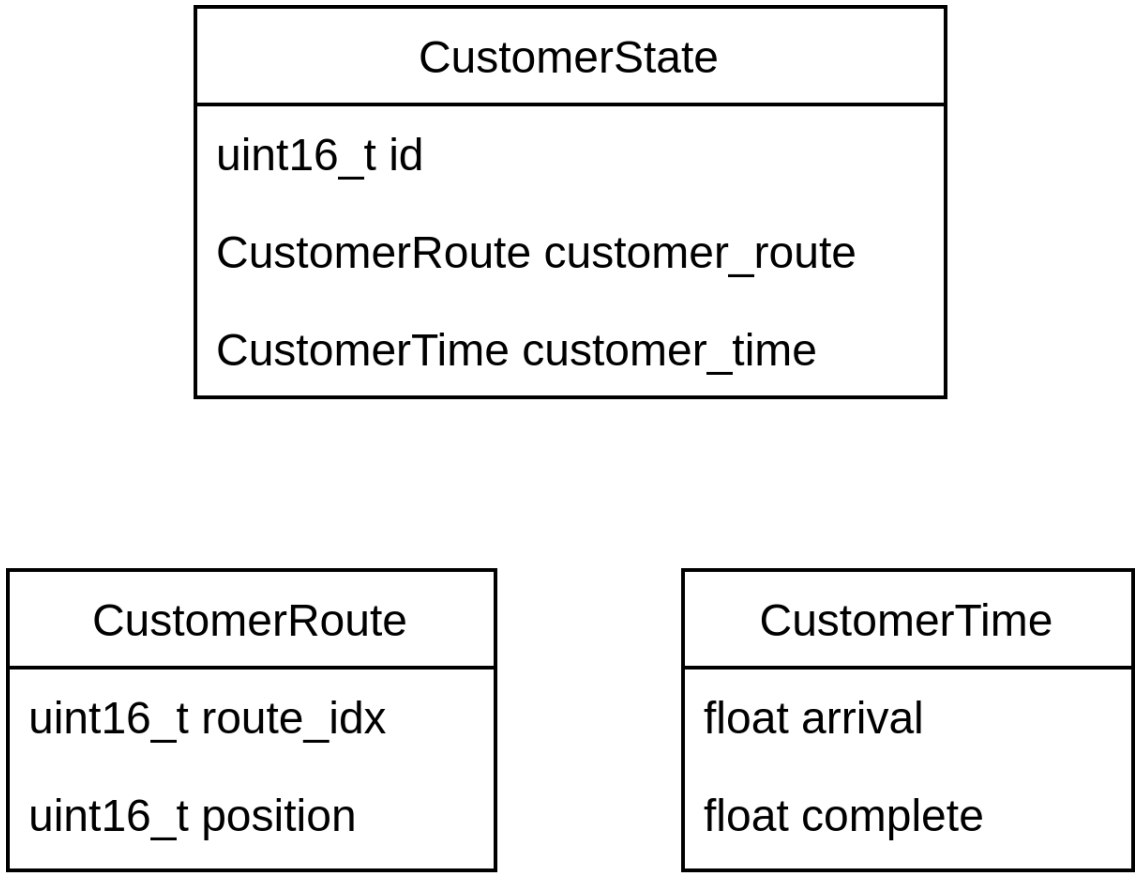
Hình 7: Lược đồ chính của ALNS

Ứng dụng - Các lớp chính



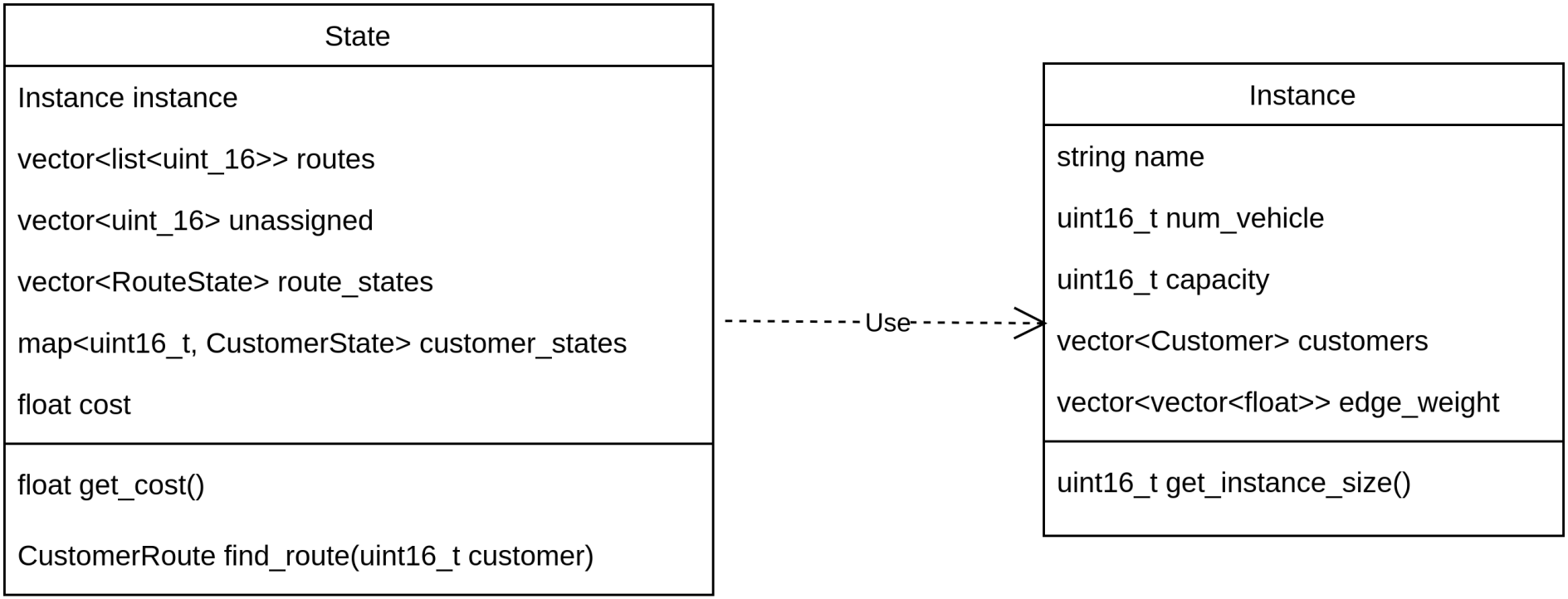
Hình 8: Lớp thuộc tính của khách hàng

Ứng dụng - Các lớp chính



Hình 9: Lớp trạng thái của khách hàng

Ứng dụng - Các lớp chính



Hình 10: Lớp trạng thái của hệ

Ứng dụng - Triển khai thuật toán

Chương trình được chia làm ba giai đoạn *Tiền xử lý*, *Chương trình chính*, *Đo đạc*.

- *Tiền xử lý* : đọc cấu hình, tính toán ma trận khoảng cách, lưu trữ vào các tệp.
- *Chương trình chính*: triển khai ALNS.
- *Đo đạc*: phân tích logs, tính toán các độ đo.

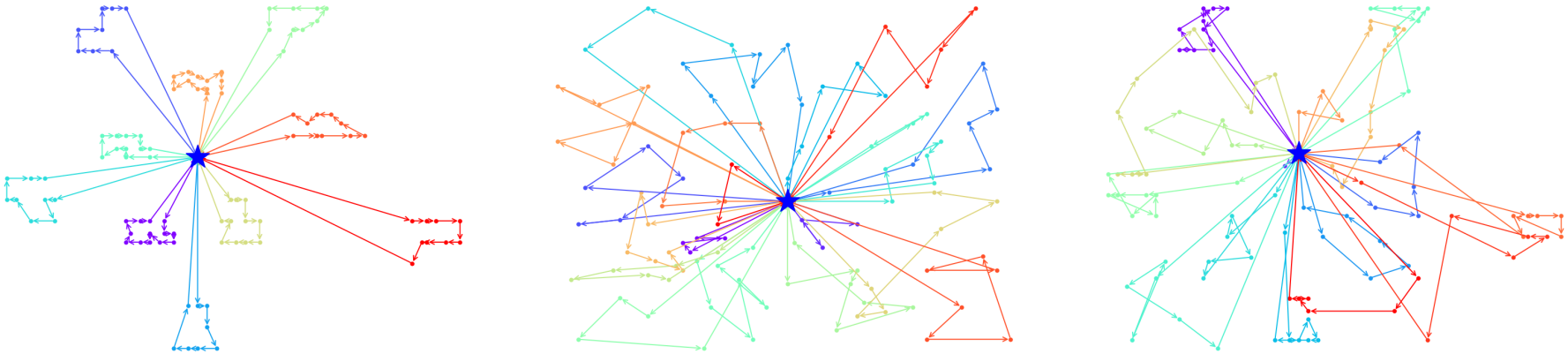
Outline

Thực nghiệm - Phân loại cấu hình

(a) Lớp C (b) Lớp R (c) Lớp RC

Hình 11: Lớp các cấu hình

Thực nghiệm - Phân loại cấu hình



(a) Lớp C (b) Lớp R (c) Lớp RC

Hình 12: Minh họa lời giải cho các lớp cấu hình

Thực nghiệm

Định dạng cấu hình Solomon

- Tên cấu hình
- Số xe được sử dụng
- Tải trọng của mỗi xe
- ID của yêu cầu
- Tọa độ các yêu cầu
- Nhu cầu (về tải) của mỗi yêu cầu
- Khung thời gian của mỗi yêu cầu
- Thời gian phục vụ của mỗi yêu cầu

Thực nghiệm - Tập Solomon

ins	ALNS	best known	gap (%)
c1	828.38	826.70	0.20
c2	589.86	587.38	0.42
r1	1,180.36	1,173.61	0.61
r2	883.92	872.51	1.38
rc1	1,339.14	1,334.49	0.36
rc2	1,007.99	1,000.68	0.74

Bảng 1: Kết quả thực nghiệm trên tập Solomon

Thực nghiệm - Tập Solomon

instance	alns best	nv	bk cost	bk nv	gap (%)
r201	1,152.96	**7**	1,143.20	8	0.32
r202	1,035.32	**7**	1,029.60	8	0.42
r203	880.90	6	870.80	6	0.41
r204	743.91	**4**	731.30	5	0.53
r205	958.81	5	949.80	5	0.40
r206	883.92	5	875.90	5	0.39
r207	806.31	5	794.00	4	0.85
r208	948.57	4	701.00	4	1.77
r209	717.53	5	854.80	5	0.48
r210	909.32	**5**	900.50	6	0.99
r211	1,053.50	5	746.70	4	0.46
avg					1.38

Bảng 2: Kết quả đo với tập Solomon R2

Thực nghiệm - Tập Solomon

instance	alns best	nv	bk cost	bk nv	gap (%)
rc201	1,274.61	**8**	1,261.80	9	1.02
rc202	1,099.54	**6**	1,092.30	8	0.66
rc203	931.16	5	923.70	5	0.81
rc204	788.66	4	783.50	4	0.66
rc205	1,157.66	7	1,154.00	7	0.32
rc206	1,060.50	**6**	1,051.10	7	0.89
rc207	966.08	6	962.90	6	0.33
rc208	785.73	4	776.10	4	1.24
avg					0.74

Bảng 3: Kết quả đo với tập Solomon RC2

Thực nghiệm - Tập HG

ins	best	avg	std	bk	gap (%)
C1_2	2,679.01	2,686.94	11.44	2,672.73	0.23
C1_4	7,158.13	7,216.72	58.54	7,021.10	1.99
C1_6	14,626.95	14,867.10	202.71	13,879.77	5.44
C1_8	27,131.40	27,633.31	393.53	24,652.04	10.11
C1_10	47,005.59	47,904.60	632.61	41,232.21	14.06

Bảng 4: Kết quả thực nghiệm trên tập HG - C1 từ 200 đến 1000 yêu cầu

Thực nghiệm - Tập HG

ins	best	avg	std	bk	gap (%)
R1_2	3,628.58	3,657.67	20.33	3,572.90	1.60
R1_4	8,691.74	8,767.74	56.40	8,334.37	4.35
R1_6	19,079.78	19,234.81	109.56	17,758.66	7.47
R1_8	34,272.76	34,595.79	230.14	30,959.89	10.69
R1_10	54,042.04	54,636.52	376.85	46,854.07	15.40

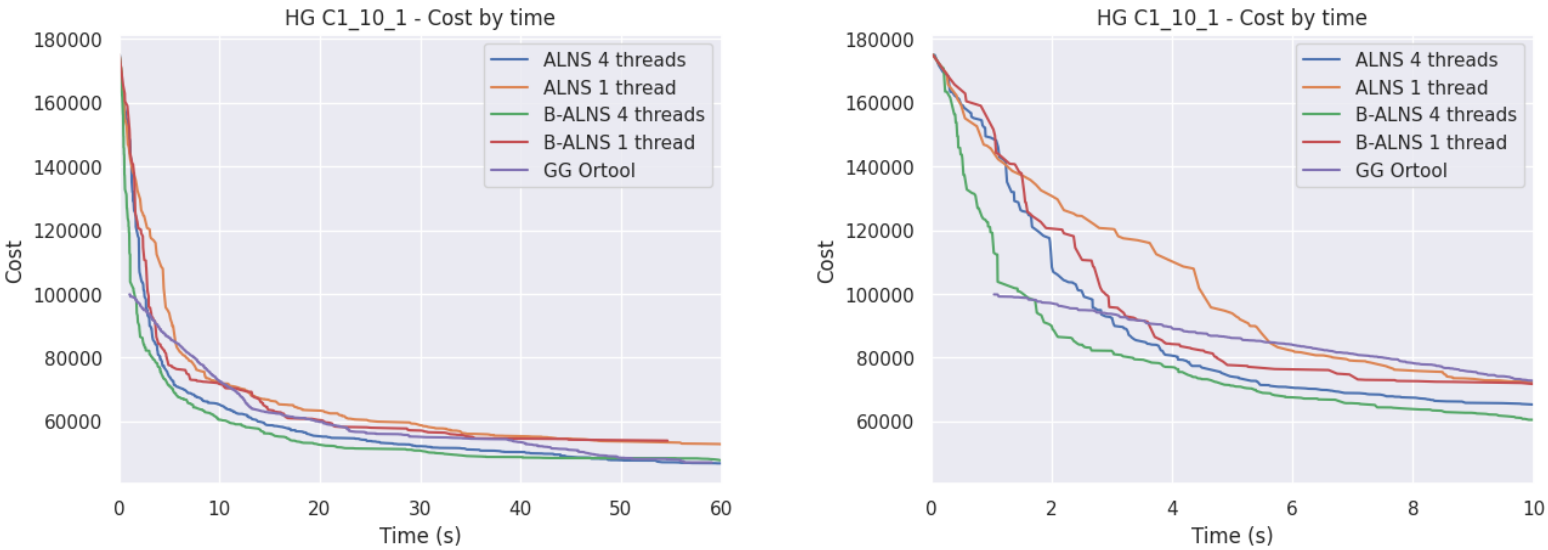
Bảng 5: Kết quả thực nghiệm trên tập HG - R1 từ 200 đến 1000 yêu cầu

Thực nghiệm - Tập HG

ins	best	avg	std	bk	gap (%)
RC1_2	3,205.24	3,235.20	25.30	3,151.30	1.73
RC1_4	8,304.23	8,380.06	60.39	7,849.98	5.78
RC1_6	17,185.75	17,322.09	112.87	15,918.08	7.98
RC1_8	31,245.28	31,491.67	191.38	28,424.05	9.91
RC1_10	49,633.08	50,050.37	321.19	43,859.18	13.15

Bảng 6: Kết quả thực nghiệm trên tập HG - RC1 từ 200 đến 1000 yêu cầu

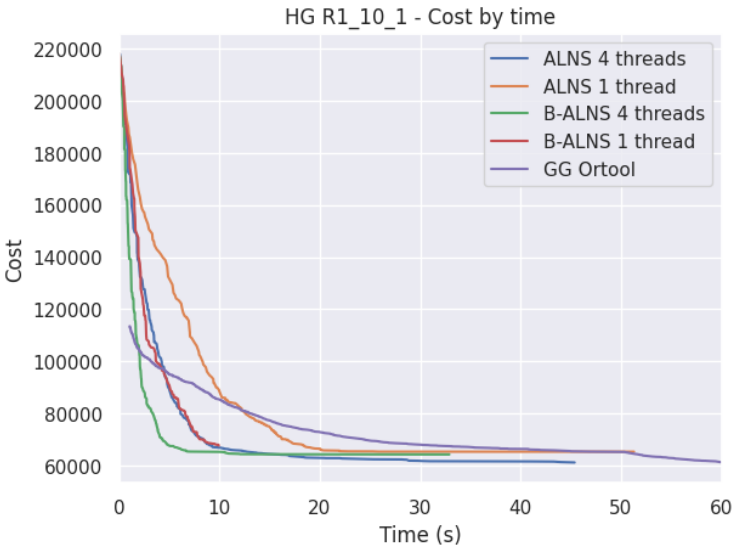
Thực nghiệm - Hiệu năng



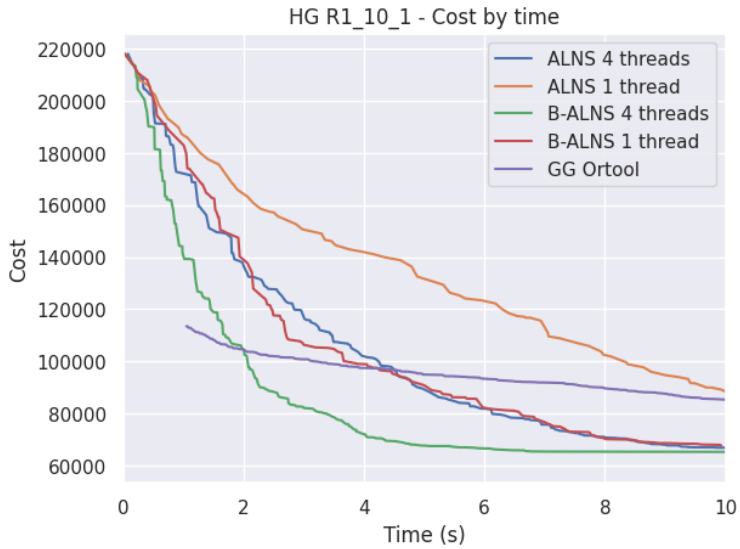
(a) 60s (b) 10s

Hình 13: Giá trị hàm mục tiêu theo thời gian, cấu hình C1_10_1

Thực nghiệm - Hiệu năng



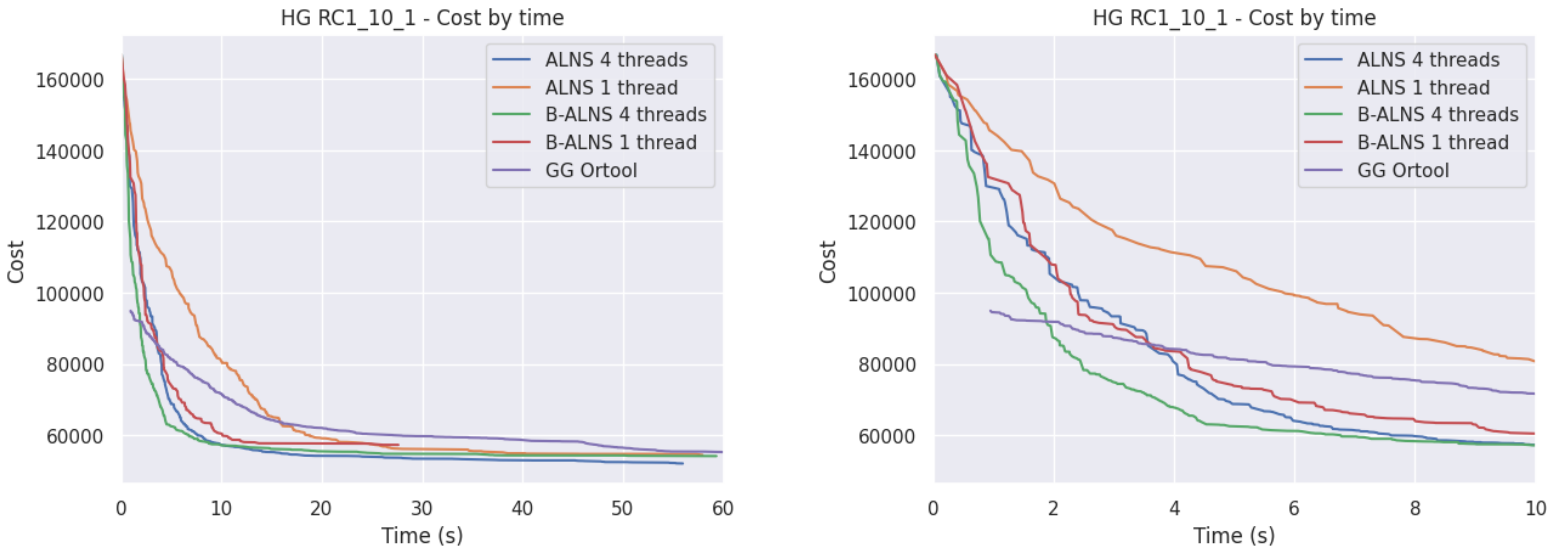
(a) 60s



(b) 10s

Hình 14: Giá trị hàm mục tiêu theo thời gian, cấu hình R1_10_1

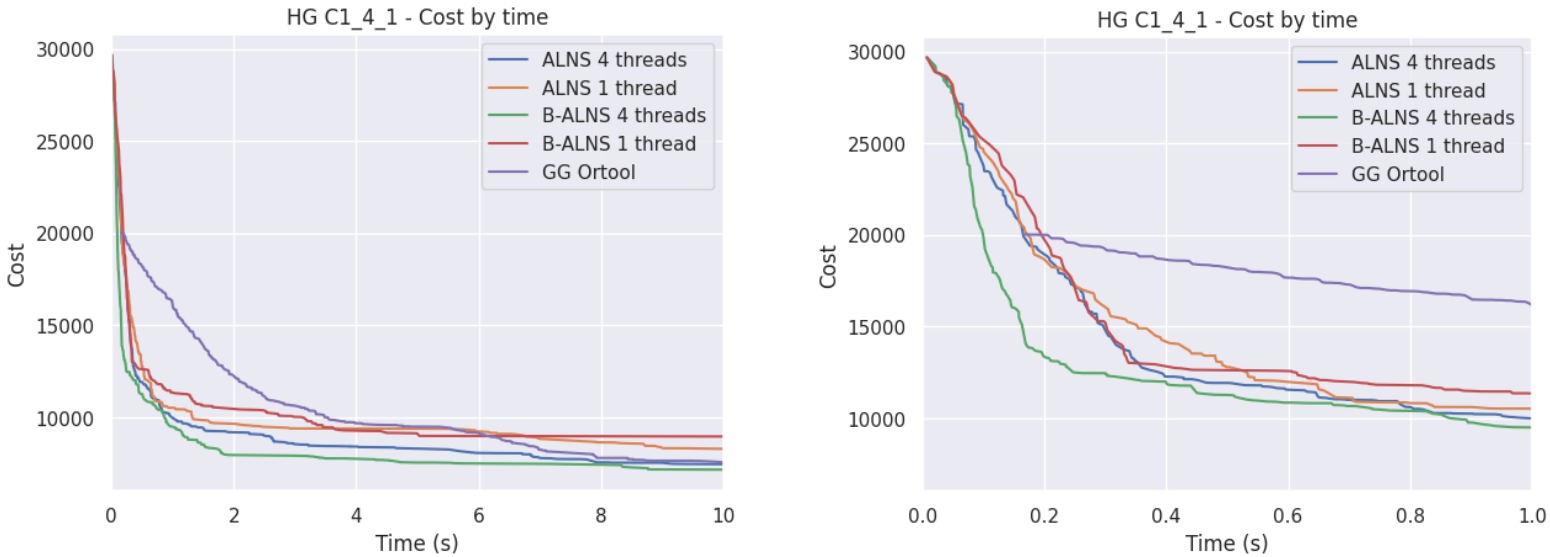
Thực nghiệm - Hiệu năng



(a) 60s (b) 10s

Hình 15: Giá trị hàm mục tiêu theo thời gian, cấu hình RC1_10_1

Thực nghiệm - Hiệu năng

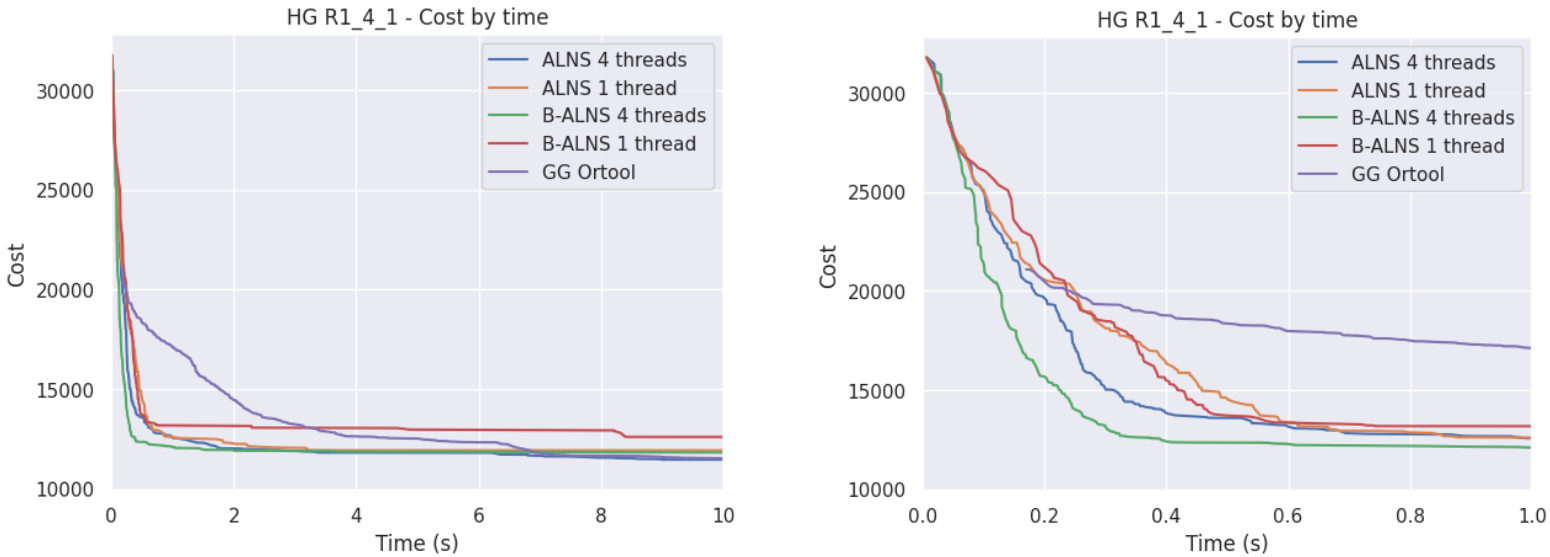


(a) 10s

(b) 1s

Hình 16: Giá trị hàm mục tiêu theo thời gian, cấu hình C1_4_1

Thực nghiệm - Hiệu năng

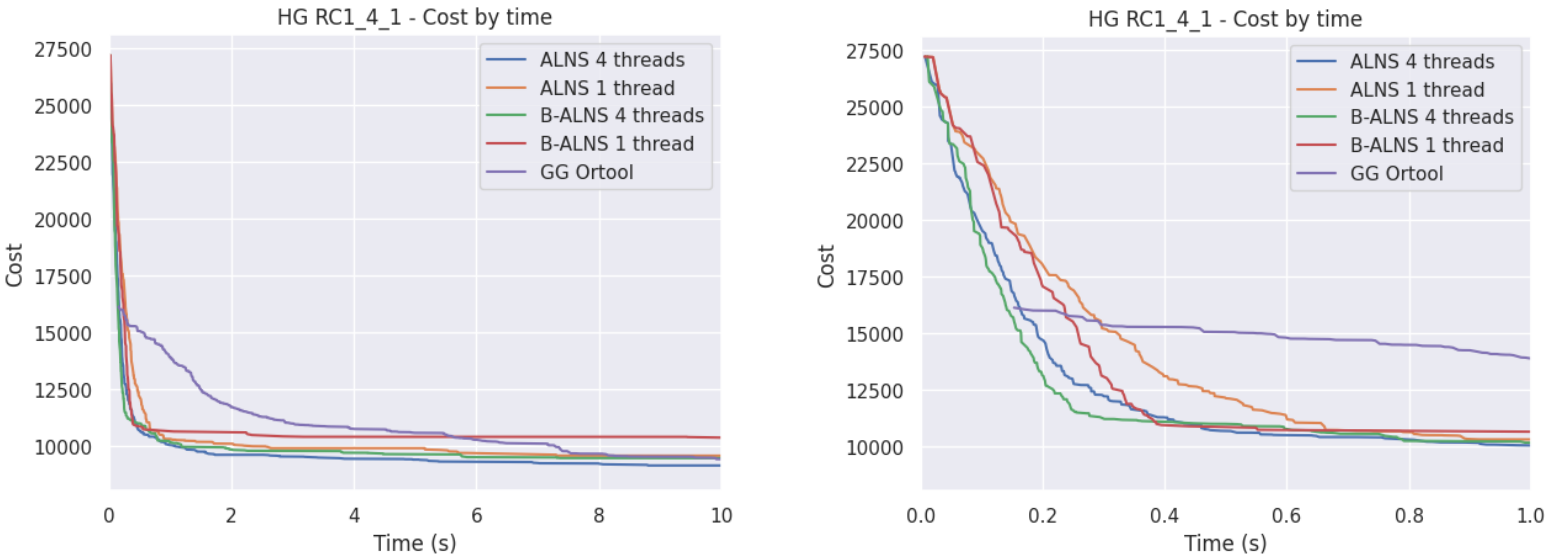


(a) 10s

(b) 1s

Hình 17: Giá trị hàm mục tiêu theo thời gian, cấu hình R1_4_1

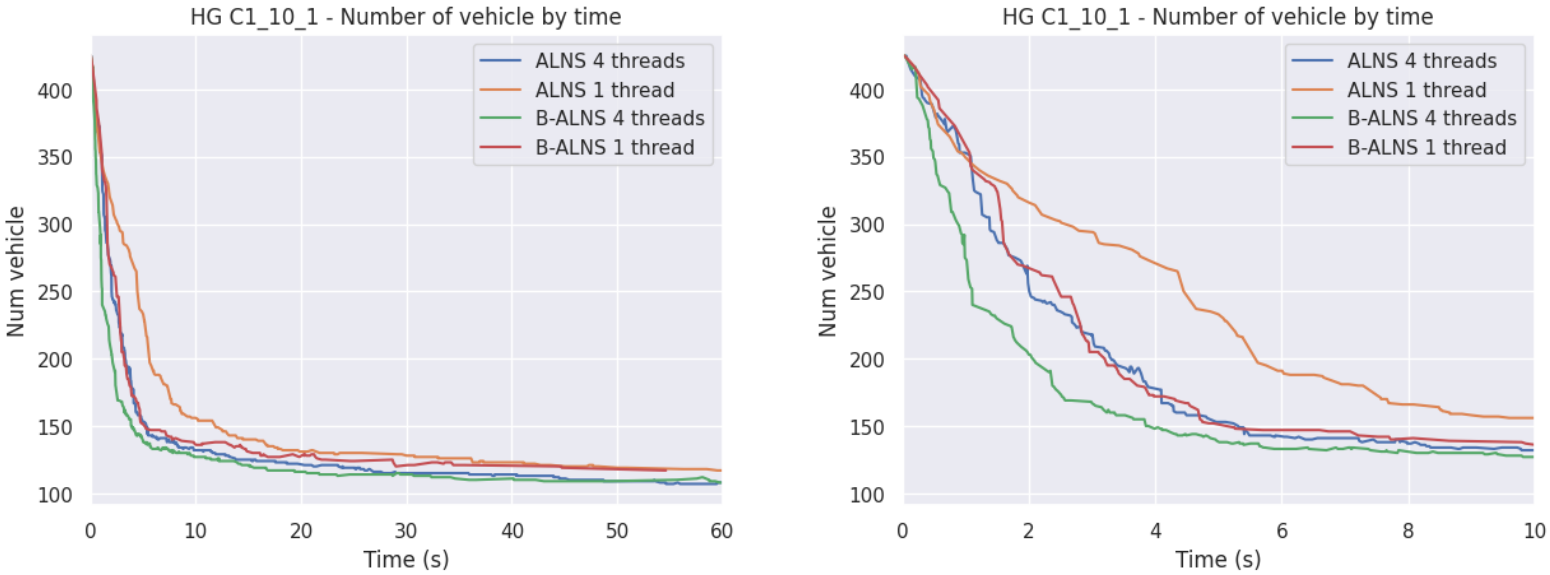
Thực nghiệm - Hiệu năng



(a) 10s (b) 1s

Hình 18: Giá trị hàm mục tiêu theo thời gian, cấu hình RC1_4_1

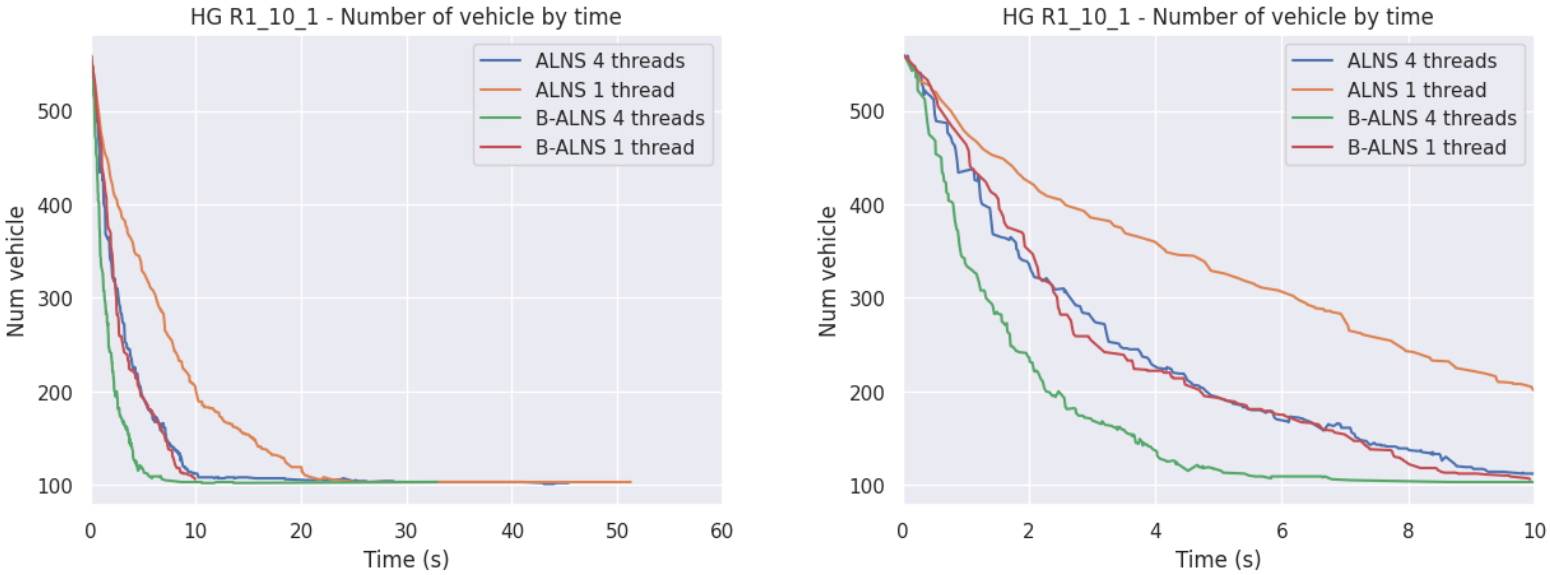
Thực nghiệm - Hiệu năng



(a) 60s (b) 10s

Hình 19: Số xe sử dụng theo thời gian, cấu hình C1_10_1

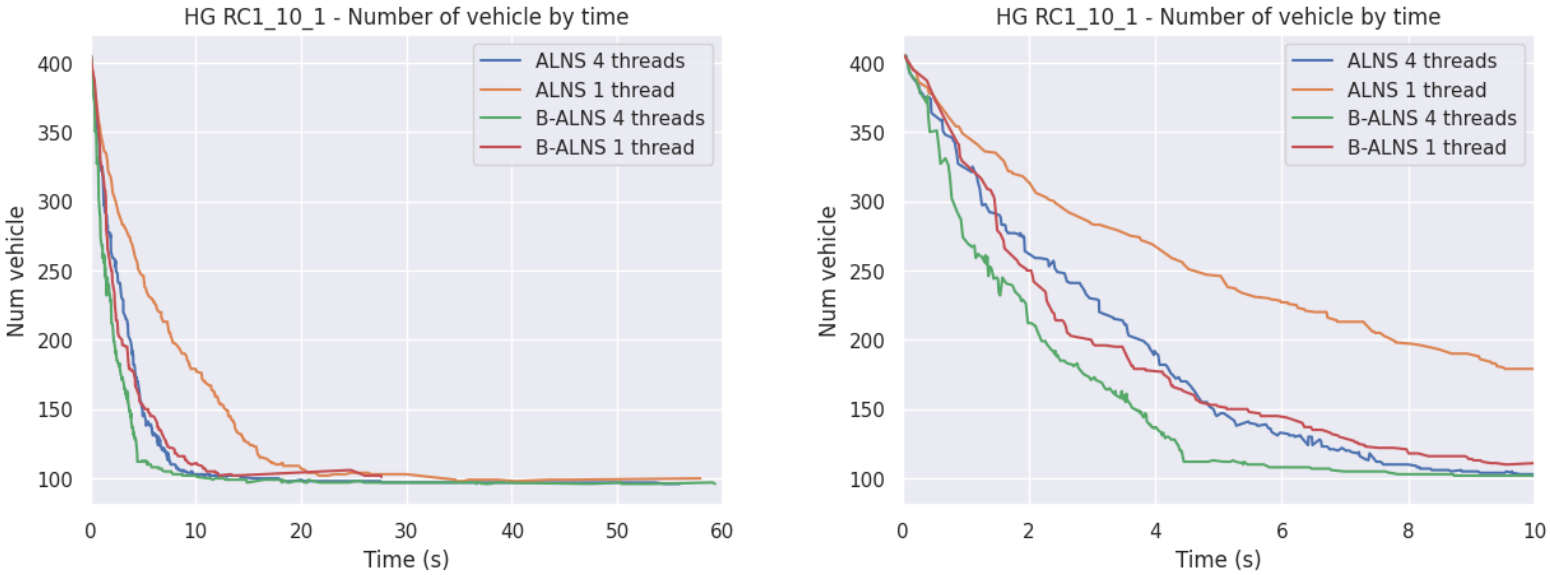
Thực nghiệm - Hiệu năng



(a) 60s (b) 10s

Hình 20: Số xe sử dụng theo thời gian, cấu hình R1_10_1

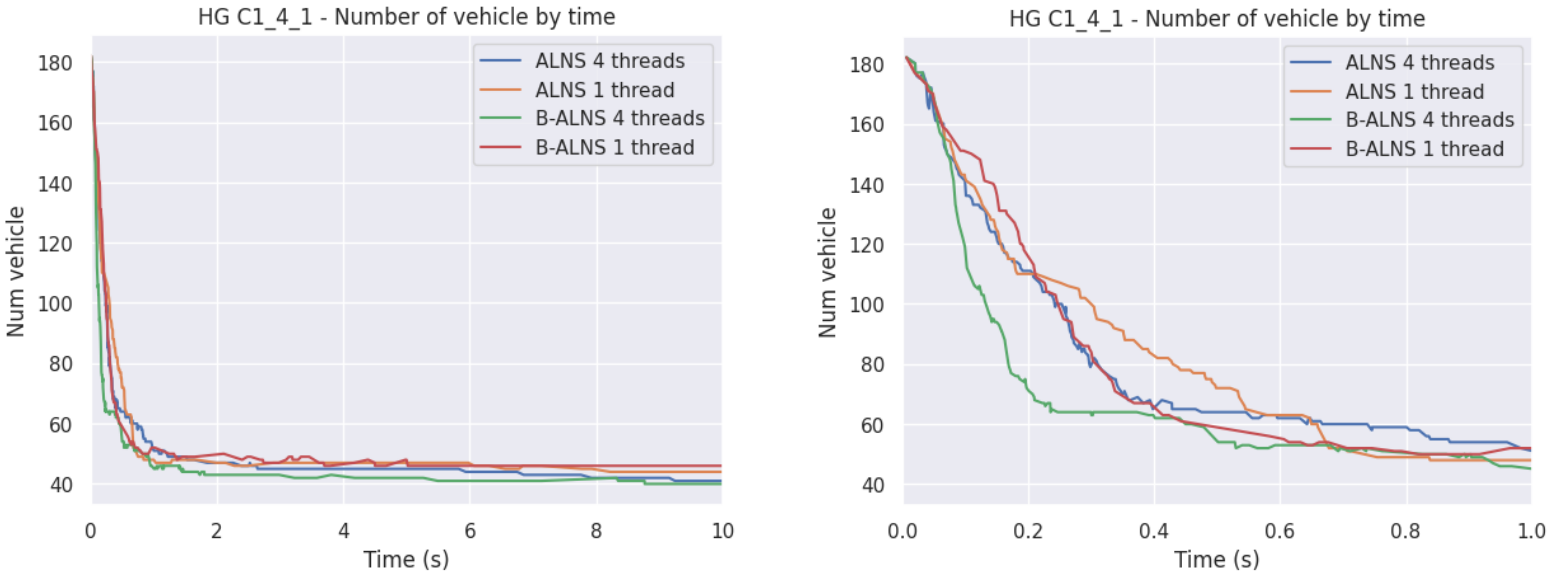
Thực nghiệm - Hiệu năng



(a) 60s (b) 10s

Hình 21: Số xe sử dụng theo thời gian, cấu hình RC1_10_1

Thực nghiệm - Hiệu năng

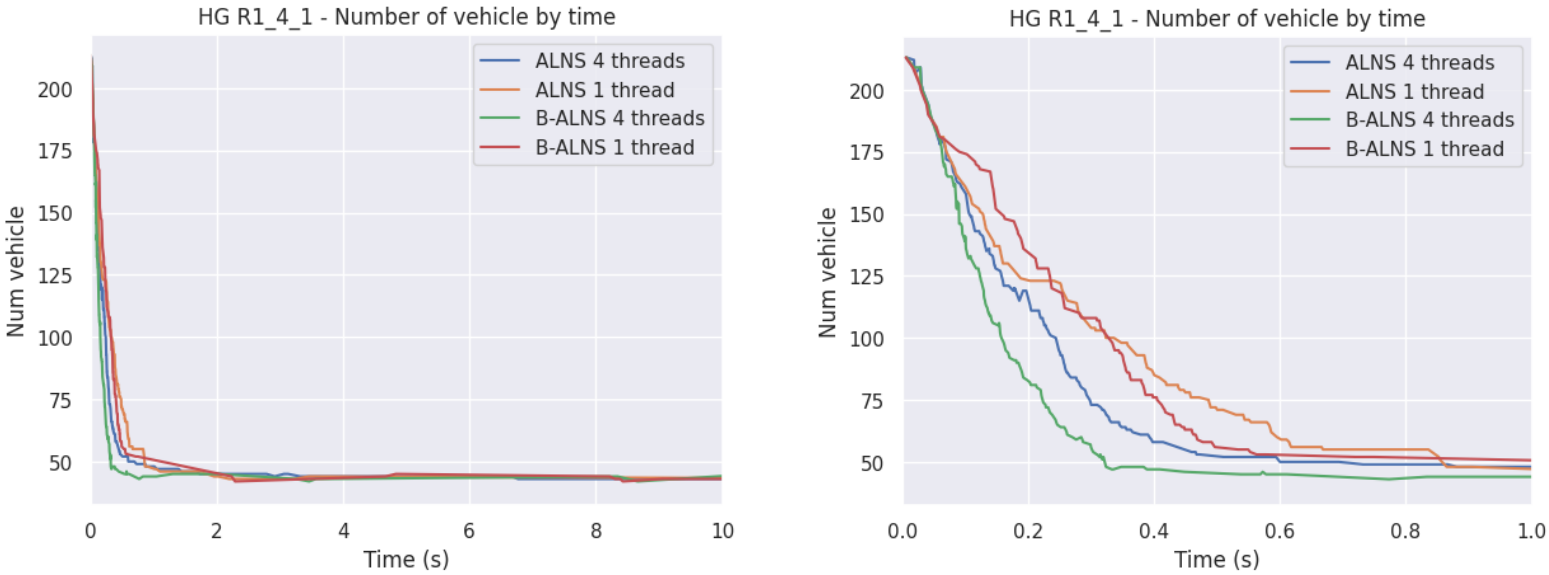


(a) 10s

(b) 1s

Hình 22: Số xe sử dụng theo thời gian, cấu hình C1_4_1

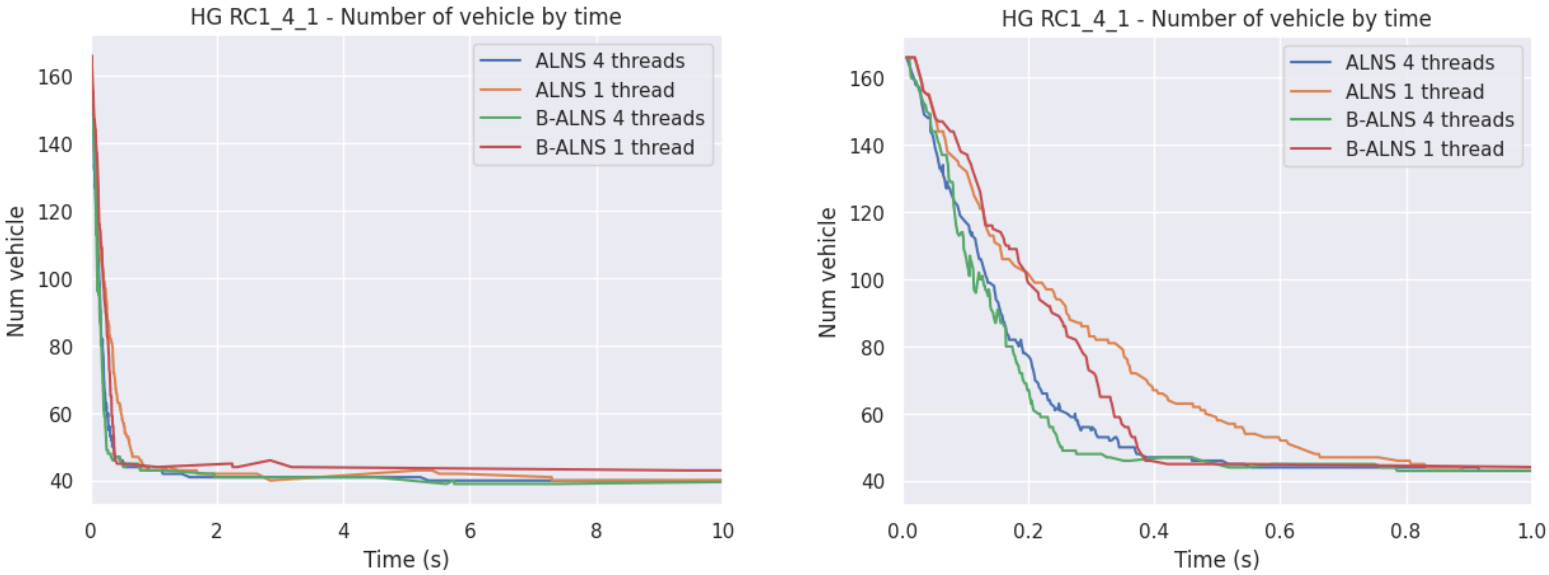
Thực nghiệm - Hiệu năng



(a) 10s (b) 1s

Hình 23: Số xe sử dụng theo thời gian, cấu hình R1_4_1

Thực nghiệm - Hiệu năng



(a) 10s (b) 1s

Hình 24: Số xe sử dụng theo thời gian, cấu hình RC1_4_1

Outline

Kết luận

Tóm tắt

- Mô hình toán học cho lớp các bài toán định tuyến phương tiện.
- Thuật toán tìm kiếm lân cận rộng thích ứng.
- Đề xuất phương pháp *xóa tuyến tệ* cho LNS.
- Đề xuất hiệu chỉnh B-ALNS giúp tăng tốc thuật toán trong thời gian đầu.
- Thực nghiệm trên các tập dữ liệu có kích thước từ nhỏ tới lớn.

Gợi ý áp dụng

- B-ALNS cho các bài toán yêu cầu độ trễ thấp.
- Sử dụng kết hợp B-ALNS và ALNS cho các bài toán yêu cầu chất lượng nghiệm tốt.

Mở rộng nghiên cứu

- Cải tiến hiệu chỉnh B-ALNS để thu được nghiệm tốt hơn trong thời gian chạy dài.
- Cơ chế thích ứng để chuyển đổi B-ALNS và ALNS.
- Cơ chế thích ứng xác định số lượng yêu cầu bỏ đi, thêm lại.
- Nghiên cứu các thuật toán lai ALNS.