SOICT Hackathon 2023: Routing Optimization

Tri Phan Khang Tran An Vo Huy M. Le Gia Khang Le

University of Information Technology (UIT)
Vietnam National University - Ho Chi Minh City (VNU-HCM)

Table of Contents

Introduction

Table of Contents

Original

- Có M hub được đánh số từ 1 đến M.
- Khoảng cách giữa hai hub u và v là $d_{u,v}$. Ma trận khoảng cách thỏa mãn bất đẳng thức tam giác
- Có N yêu cầu vận chuyển, mỗi yêu cầu cần nhận hàng tại hub p_i và trả hàng tại hub d_i trong khoảng thời cho phép. Các đơn hàng có thông tin về thể tích và khối lượng.
- Có K xe tải thực hiện việc vận chuyển hàng hóa. Mỗi xe có thể tích, khối lượng hàng tối đa có thể chở, có vận tốc di chuyển, thời gian hoạt động và bắt đầu, kết thúc tại hub de_i.
- Hàm mục tiêu: $\min_S \vec{F}(S) = 10^9 \times \frac{O_N}{N} - 10^6 \times \frac{O_K}{K} - \frac{O_T}{10^3}$
- Với O_N là số đơn hàng xử lí thành công, O_K là số xe sử dụng, O_T là tổng thời gian hoạt động của các xe.

Redefination

- Xem như các đỉnh $p_i, d_i, (i \in \{1 \dots N\}), de_j, (j \in \{1 \dots k\})$ là đôi một khác nhau
- Biểu diễn lại bằng đồ thị $G'(V', E')$:
  - $\mathcal{P} = \{p_1, p_2, \dots, p_N\}$ là các đỉnh lấy hàng.
  - $\mathcal{D} = \{d_1, d_2, \dots, d_N\}$ là các đỉnh dỡ hàng.
  - $\mathcal{T} = \{t_1, t_2, \dots, t_K\}$ là các đỉnh bắt đầu và kết thúc của xe tải.
- Tập cạnh $E' = V' \times V'$, cạnh có độ dài giống đồ thị gốc.
- Mỗi đỉnh i sẽ có thêm thông tin:
  - **Thời gian hoạt động** $[e_i, l_i]$: thời gian bốc hàng, dỡ hàng, hoặc thời gian hoạt động của xe tải.
  - **Thời gian phục vụ** s_i: bằng thời gian bốc hàng, dỡ hàng hoặc bằng 0.
  - **Tải trọng** c: khối lượng cần xử lí thêm: $c_{p_i} = cap_i$, $c_{d_i} = -cap_i$
  - **Thể tích** v: thể tích cần xử lí thêm: $v_{p_i} = vol_i$, $v_{d_i} = -vol_i$

Solution presentation

Một lời giải cho bài toán là một lịch trình cho các xe. Gọi $\pi^{(i)}$ là lịch trình cho xe thứ i, $\pi^{(i)}_j$ là đỉnh thứ i trong lịch trình của xe i. Một lời giải $S = \{\pi^{(1)}, \ldots, \pi^{(K)}\}$ gọi là hợp lệ nếu thỏa mãn tất cả các ràng buộc:

1. $\pi^{(i)}_1 = \pi^{(i)}_{|\pi^{(i)}|} = t_i$
2. $p_i \in \pi^{(i)} \Rightarrow p_i \notin \pi^{(j)}, \quad \forall i \neq j$
3. $p_i \in \pi^{(i)} \Leftrightarrow d_i \in \pi^{(i)}$
4. $\begin{cases} \pi^{(i)}_j = p_x \\ \pi^{(i)}_k = d_x \end{cases} \Rightarrow j < k$

Solution presentation

5. $\forall i \leq K, j \leq |\pi^{(i)}| : \quad \sum_{k=1}^{j} c_{\pi_k^{(i)}} \leq mcap_i$
6. $\forall i \leq K, j \leq |\pi^{(i)}| : \quad \sum_{k=1}^{j} v_{\pi_k^{(i)}} \leq mvol_i$
7. $ar_j^{(i)} \geq e_{\pi_j^{(i)}}$
8. $ar_j^{(i)} \leq l_{\pi_j^{(i)}}$
9. $ar_{j+1}^{(i)} \geq ar_j^{(i)} + s_{\pi_j^{(i)}} + time_i(\pi_j^{(i)}, \pi_{j+1}^{(i)})$

Table of Contents

Heuristic construction

Algorithm 1 Initial Solution

```
1: P ← {1, 2, ..., N}
2: for each vehicle do
3:     while True do
4:         bestPen ← ∞
5:         for each p ∈ P do
6:             newPen ← Penalty if insert p to vehicle
7:             bestPen ← min(bestPen, newPen)
8:         end for
9:         if bestPen < ∞ then
10:            Insert request i has lowest penalty
11:            Remove i from P
12:        else
13:            Break
14:        end if
15:    end while
16: end for
```

Hình: Heuristic construction

Heuristic construction

- Penalty của một lịch trình di chuyển là thời điểm xe hoàn thành đơn cuối cùng
- Nếu lịch trình không hợp lệ, penalty là ∞
- Phương pháp thêm request vào lịch trình:
  - Thêm d_i vào cuối lịch trình hiện tại, chọn vị trí thêm p_i để penalty nhỏ nhất
  - Thử tất cả các cặp vị trí có thể để thêm p_i và d_i. Chọn cặp vị trí tạo thành lịch trình có penalty nhỏ nhất.

Large Neighborhood Search

Ý tưởng của Large Neighborhood Search (LNS) là tại mỗi vòng lặp, xóa bỏ q cặp (p_i, d_i) sau đó thêm lại các cặp (p_i, d_i) này và cố gắng thêm các cặp (p_i, d_i) khác trong số các yêu cầu chưa được xử lí. Để giảm thời gian tính toán, ta cũng chỉ thử trong $maxTry$ cặp (p_i, d_i). Solution mới được tạo ra sẽ được xem xét lấy hay không theo simulated annealing.

Large Neighborhood Search

Algorithm 2 Large Neighborhood Search

Require: solution S, parameter $q, maxTry$

$S_{best} \leftarrow S$
while not terminated **do**
    $S' \leftarrow S$
    $D \leftarrow q$ inserted request
    Remove requests from D of S
    Reinsert requests from D to S
    $E \leftarrow maxTry$ request not inserted not in D
    for each $e \in E$ **do**
        if can insert e to S **then**
            insert e to S
        end if
    end for
    if $F(S) > F(S_{best})$ **then**
        $S_{best} = S$
    end if
    if not accept(S, S') **then**
        $S \leftarrow S'$
    end if
end while

Hình: Large Neighborhood Search

Guided Ejection Search

Thuật toán Guided Ejection Search (GES) sẽ remove tất cả request của một xe nào đó, sau đó cố gắng thêm các request chưa được dùng vào các xe khác. Nếu không thể thêm request vào các xe khác, xem xét xóa bỏ một số request hiện có để thêm request đó vào và perturbation solution hiện tại thông qua một số hàm pertubation cơ bản. Việc chọn cách xóa sẽ thông qua một hệ thông penalty h như sau:

- Ban đầu, $h_i = 0, (\forall i)$
- Khi được xem xét thêm vào lại solution, nếu không thể thêm trực tiếp request i vào bất kì xe tải nào thì tăng h_i lên 1
- Trong tất cả các cách xóa sẽ tạo ra solution mới có thể thêm request i vào ta sẽ chọn cách xóa có tổng h của các request bị xóa là nhỏ nhất.

Guided Ejection Search

Algorithm 3 Guided Ejection Search

Require: solution S

$S_{best} \leftarrow S$
Remove all request of random truck r
$EP \leftarrow$ all non-inserted request
Initialize penalty $h_i = 1, \forall i \leq N$
while not terminated **do**
    $pd \leftarrow$ a request from EP
    if can insert pd to S **then**
        Insert pd to S
        Remove pd from EP
    else
        $h_{pd} = h_{pd} + 1$
        if can insert pd by remove some request **then**
            Remove requests have minimum penalty
            Insert these requests to EP
            Insert pd to S
            Remove pd from EP
        end if
        $S = pertubation(S)$
    end if
end while
Insert requests to truck r via Initial process

Hình: Guided Ejection Search

Table of Contents

Experiment

	LNS	**AGES**	**LNS + AGES**
Test 1	**538997935.9**	538997926.7	546997902.3
Test 2	**627997558.2**	606997606.5	605997611.3
Test 3	**662997213.6**	653997260.8	661997235.8
Test 4	**720996927.7**	714996942.0	719996905.8
Test 5	793996524.1	794996566.7	**802996532.5**
Total	**3344986159.5**	3309986302.7	3337986187.7

Table of Contents

Conclusion