Thuật toán Metaheuristic cho bài toán TSP

Nhóm 13

Khoa Toán - Cơ - Tin học
Trường Đại học Khoa học Tự nhiên
Đại học Quốc gia Hà Nội

Ngày 22 tháng 5 năm 2023

Nội dung

Giới thiệu về bài toán tối ưu tổ hợp

Phát biểu bài toán

Giới thiệu về bài toán tối ưu tổ hợp

Phát biểu bài toán

Một cách tổng quát, mỗi bài toán tối ưu tổ hợp có thể được phát biểu như sau: Cho bộ 3 (S, f, Ω) trong đó:

Giới thiệu về bài toán tối ưu tổ hợp

Phát biểu bài toán

Một cách tổng quát, mỗi bài toán tối ưu tổ hợp có thể được phát biểu như sau: Cho bộ 3 (S, f, Ω) trong đó:

- S là tập hữu hạn các trạng thái (phương án hay lời giải tiềm năng), thường là rất lớn.
- f là hàm mục tiêu xác định trên S
- Ω là tập các ràng buộc

Giới thiệu về bài toán tối ưu tổ hợp

Phát biểu bài toán

Một cách tổng quát, mỗi bài toán tối ưu tổ hợp có thể được phát biểu như sau: Cho bộ 3 (S, f, Ω) trong đó:

- S là tập hữu hạn các trạng thái (phương án hay lời giải tiềm năng), thường là rất lớn.
- f là hàm mục tiêu xác định trên S
- Ω là tập các ràng buộc

Mỗi phương án $s \in S$ thỏa mãn các ràng buộc Ω được gọi là phương án chấp nhận được. Mục đích của bài toán tối ưu tổ hợp là tìm ra phương án chấp nhận được s^* tối ưu hóa toàn cục hàm mục tiêu f.

Giới thiệu về bài toán tối ưu tổ hợp

Ví dụ

Giới thiệu về bài toán tối ưu tổ hợp

Ví dụ

- Tìm đường đi ngắn nhất giữa 2 đỉnh trên đồ thị
- Lập kế hoạch phân phối nguồn hàng tới nơi tiêu thụ với chi phí cực tiểu
- Lập thời khóa biểu cho giáo viên và học sinh thuận lợi nhất
- Bài toán N con hậu
- Bài toán người du lịch (Traveling salesman problem - TSP)

Giới thiệu về bài toán tối ưu tổ hợp

Các cách tiếp cận bài toán

Giới thiệu về bài toán tối ưu tổ hợp

Các cách tiếp cận bài toán

Đối với các bài toán tối ưu tổ hợp cỡ nhỏ, chúng ta có thể thu được lời giải tối ưu chính xác bằng các thuật toán vét cạn, tuy nhiên đối với các bài toán cỡ lớn thì chưa có thuật toán tìm lời giải chính xác trong thời gian đa thức nên chỉ có thể tìm lời giải gần đúng hoặc đủ tốt.

Giới thiệu về bài toán tối ưu tổ hợp

Các cách tiếp cận bài toán

Đối với các bài toán tối ưu tổ hợp cỡ nhỏ, chúng ta có thể thu được lời giải tối ưu chính xác bằng các thuật toán vét cạn, tuy nhiên đối với các bài toán cỡ lớn thì chưa có thuật toán tìm lời giải chính xác trong thời gian đa thức nên chỉ có thể tìm lời giải gần đúng hoặc đủ tốt.

Các thuật toán tìm lời giải gần đúng đòi hỏi phải được chứng minh tính hội tụ và ước lượng được sai số tại mỗi bước.

Giới thiệu về bài toán tối ưu tổ hợp

Các cách tiếp cận bài toán

Đối với các bài toán tối ưu tổ hợp cỡ nhỏ, chúng ta có thể thu được lời giải tối ưu chính xác bằng các thuật toán vét cạn, tuy nhiên đối với các bài toán cỡ lớn thì chưa có thuật toán tìm lời giải chính xác trong thời gian đa thức nên chỉ có thể tìm lời giải gần đúng hoặc đủ tốt.

Các thuật toán tìm lời giải gần đúng đòi hỏi phải được chứng minh tính hội tụ và ước lượng được sai số tại mỗi bước.

Các thuật toán tìm lời giải đủ tốt được phát triển để khắc phục các hạn chế của 2 cách tiếp cận trên $\Rightarrow$ Các thuật toán dựa trên thực nghiệm - Heurisitic Algorithms.

Heuristic và Metaheuristics

Heuristic Algorithms

Heuristic và Metaheuristics

Heuristic Algorithms

Heuristic Algorithms được phát triển theo 2 hướng:

- Metaheuristic: Là một lược đồ tính toán tổng quát đề xuất cho lớp bài toán rộng, khi dùng cho các bài toán cụ thể cần thêm các vận dụng chi tiết cho phù hợp. Nhờ các lược đồ này, người dùng có thể xây dựng được thuật toán cho bài toán trong thực tế nên hiện nay đang được dùng phổ biến.
- Problem-specific heuristics: Được đề xuất riêng biệt cho từng bài toán cụ thể, cho phép tìm nhanh một lời giải đủ tốt hoặc xấp xỉ tối ưu địa phương.

Heuristic và Metaheuristics

Metaheuristics

Heuristic và Metaheuristics

Metaheuristics

Ưu điểm:

Metaheuristics

Ưu điểm:

- Áp dụng được cho lớp bài toán rộng
- Hiệu quả trong thực nghiệm
- Dễ dàng cài đặt
- Dễ dàng thực hiện tính toán song song

Heuristic và Metaheuristics

Metaheuristics

Ưu điểm:

- Áp dụng được cho lớp bài toán rộng
- Hiệu quả trong thực nghiệm
- Dễ dàng cài đặt
- Dễ dàng thực hiện tính toán song song

Nhược điểm:

Heuristic và Metaheuristics

Metaheuristics

Ưu điểm:

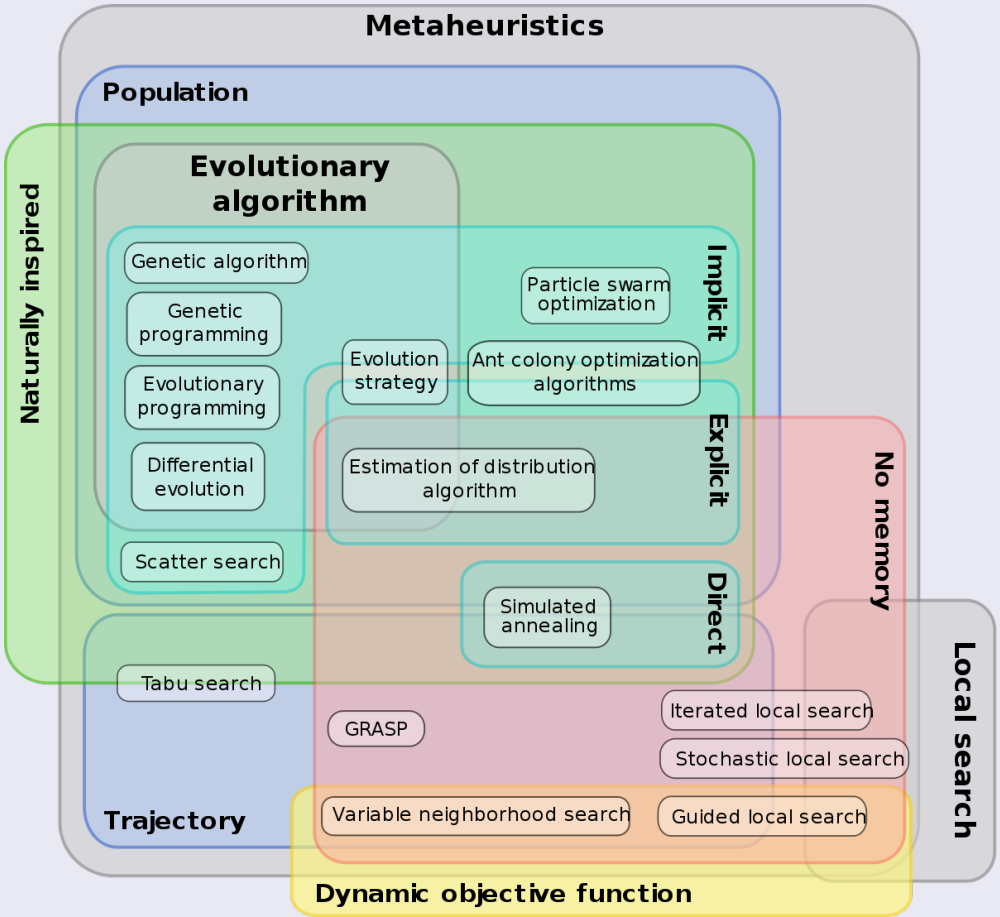
- Áp dụng được cho lớp bài toán rộng
- Hiệu quả trong thực nghiệm
- Dễ dàng cài đặt
- Dễ dàng thực hiện tính toán song song

Nhược điểm:

- Không phải là các thuật toán tìm ra lời giải chính xác
- Không nhất quán
- Cơ sở toán học khá yếu

Metaheuristics

Metaheuristics



Local Search

Local Search Pseudocode

Local Search

Local Search Pseudocode

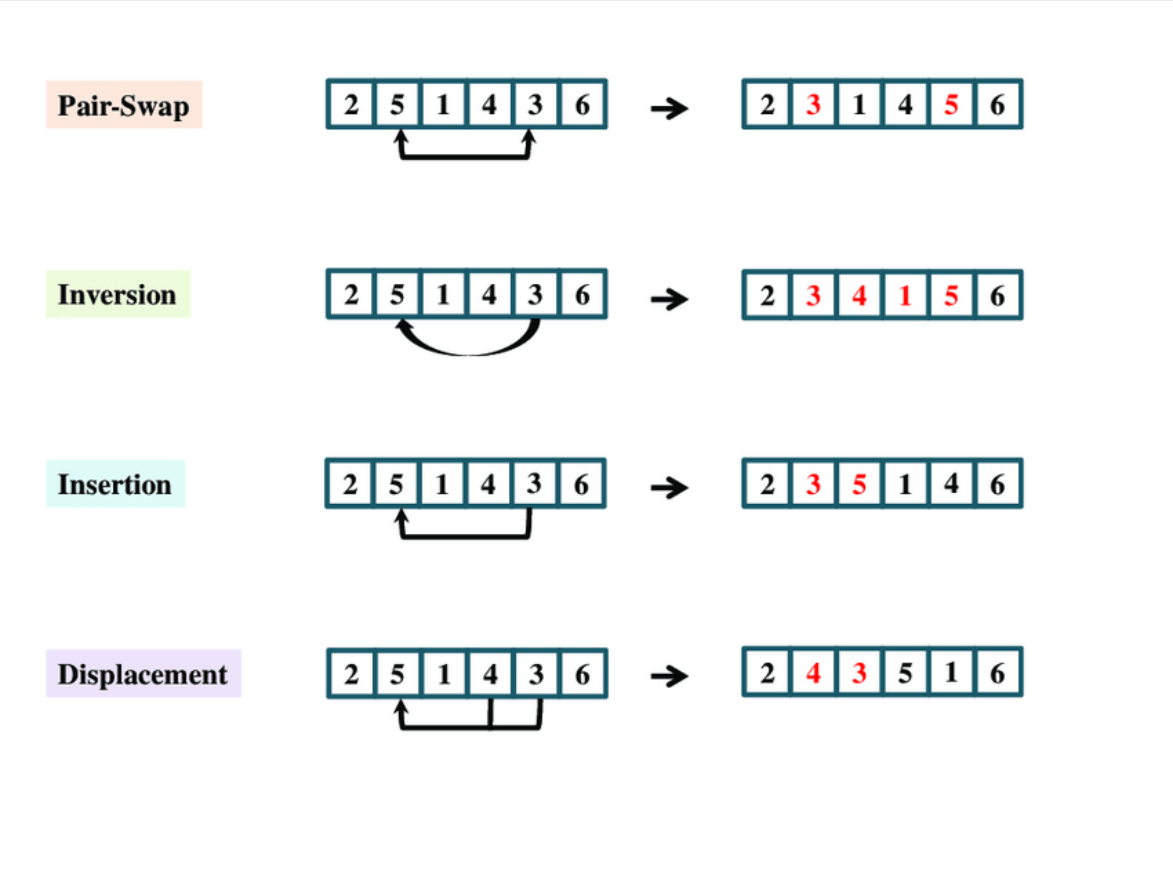
PROCEDURE localSearch(stopCriteria)

    /* Generate initial solution */
    $s = s_0$;
    while *stopCriteria is not achived* **do:**
        /* Generate candidate neighbors from s */
        $\mathcal{N}_s = generate(s)$;
        if *there is no better neighbor* **then** *stop*;
        /* Select best solution from $\mathcal{N}_s$ to replace the current solution s */
        $s = select(\mathcal{N}_s)$;
    end while
    return s;

END PROCEDURE

Neighborhood trong Local Search

Neighborhood trong Local Search

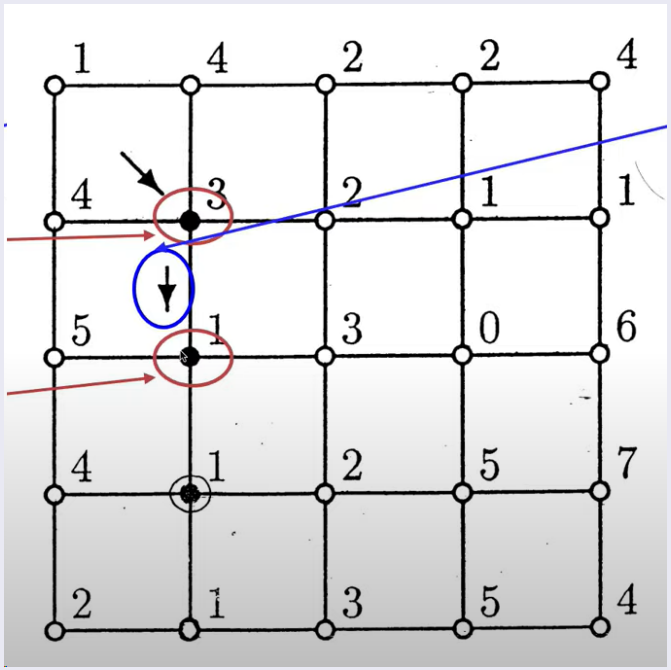


Local Search

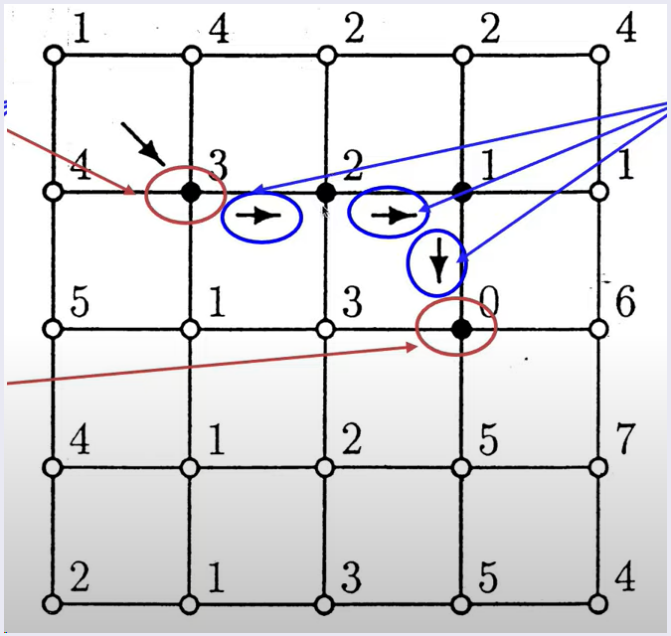
Local Search Example

Local Search

Local Search Example



Best improvement



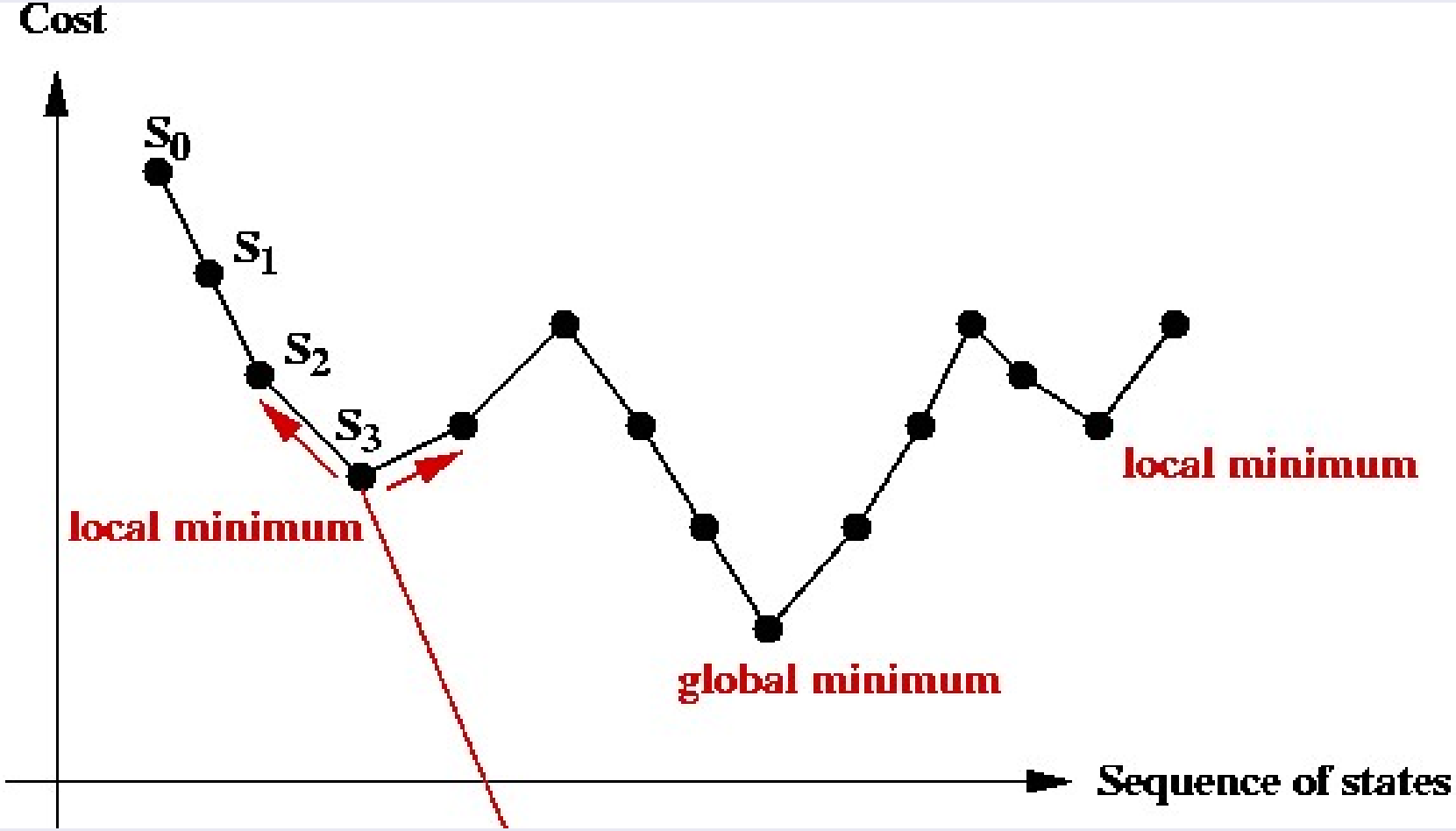
First improvement

Local Search

Local Search Example

Local Search

Local Search Example



Nhược điểm và giải pháp cải thiện Local Search

Local Search

Nhược điểm và giải pháp cải thiện Local Search

- Sau mỗi bước đều đòi hỏi phải có sự cải thiện nghiệm hiện tại nên hội tụ rất nhanh tới cực trị địa phương và rất nhạy cảm với nghiệm khởi tạo, có khả năng sẽ bị 'kẹt' lại tại một cực trị địa phương.

Local Search

Nhược điểm và giải pháp cải thiện Local Search

- Sau mỗi bước đều đòi hỏi phải có sự cải thiện nghiệm hiện tại nên hội tụ rất nhanh tới cực trị địa phương và rất nhạy cảm với nghiệm khởi tạo, có khả năng sẽ bị 'kẹt' lại tại một cực trị địa phương.
- Một số cách nhằm cải thiện Local Search:
  1. Thực hiện Local Search nhiều lần với các giá trị khởi tạo khác nhau: Iterated Local Search, Multistart Local Search
  2. Thay đổi hàm mục tiêu hoặc dữ liệu đầu vào: Guided Local Search, Noisy Method, Smoothing Method
  3. Sử dụng hàng xóm khác: Variable Neighborhood Search
  4. Chấp nhận sau mỗi bước có thể không nhận được nghiệm cải thiện: Simulated annealing, Tabu Search

Iterated Local Search

Iterated Local Search Pseudocode

Iterated Local Search

Iterated Local Search Pseudocode

PROCEDURE iteratedLocalSearch(stopCriteria)

/* Generate initial solution */

$s = s_0$;

/* Perform Local Search on initial solution */

$curBest = localSearch(s)$;

while *stopCriteria is not achived* **do:**

/* Getting perturbation of current best solution based on searched history */

$s = perturbation(curBest)$;

$s^* = localSearch(s)$;

if $f(s^*) < f(curBest)$ **do**:

$curBest = s^*$;

end if

end while

return *curBest*;

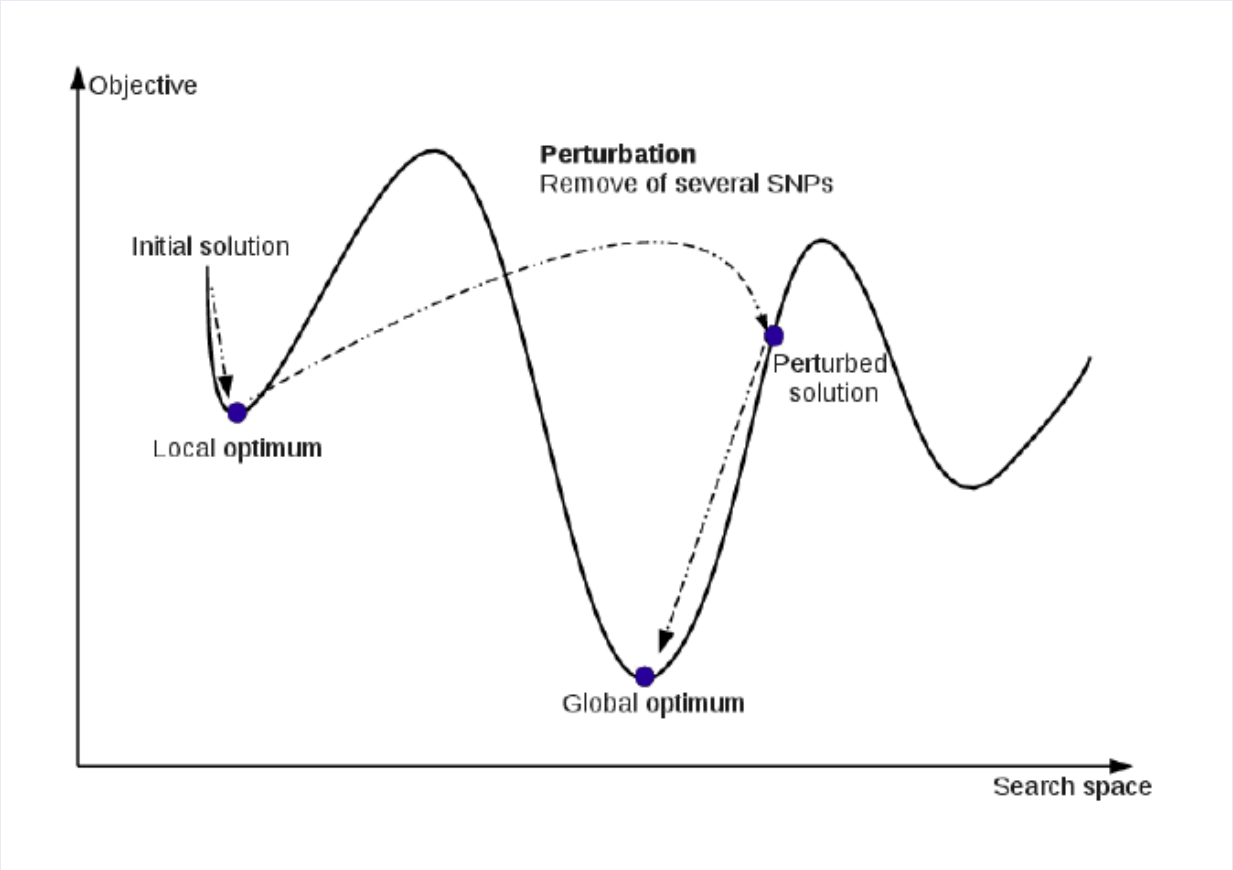
END PROCEDURE

Iterated Local Search

Iterated Local Search performance

Iterated Local Search

Iterated Local Search performance



Thuật toán Local Search cho bài toán TSP

Mô tả bài toán

Thuật toán Local Search cho bài toán TSP

Mô tả bài toán

Có n thành phố $T_1, T_2, T_3, ..., T_n$ với c_{ij} là chi phí khi di chuyển giữa 2 thành phố T_i và T_j.
Một người du lịch xuất phát từ một thành phố T_i bất kì, người đó muốn đi qua tất cả các thành phố còn lại, mỗi thành phố đi qua đúng 1 lần và quay trở về thành phố xuất phát.
Tìm hành trình (chu trình) có chi phí nhỏ nhất.

Thuật toán Local Search cho bài toán TSP

Xác định cấu hình của 1 nghiệm

Thuật toán Local Search cho bài toán TSP

Xác định cấu hình của 1 nghiệm

Một nghiệm chấp nhận được của bài toán được biểu diễn ở dạng hoán vị của tập gồm N số nguyên dương đầu tiên.

Ví dụ: $N = 10$

Một nghiệm chấp nhận được của bài toán được biểu diễn như sau:

$$C = [1, 3, 2, 4, 5, 10, 6, 8, 7, 9, 1]$$

Thuật toán Local Search cho bài toán TSP

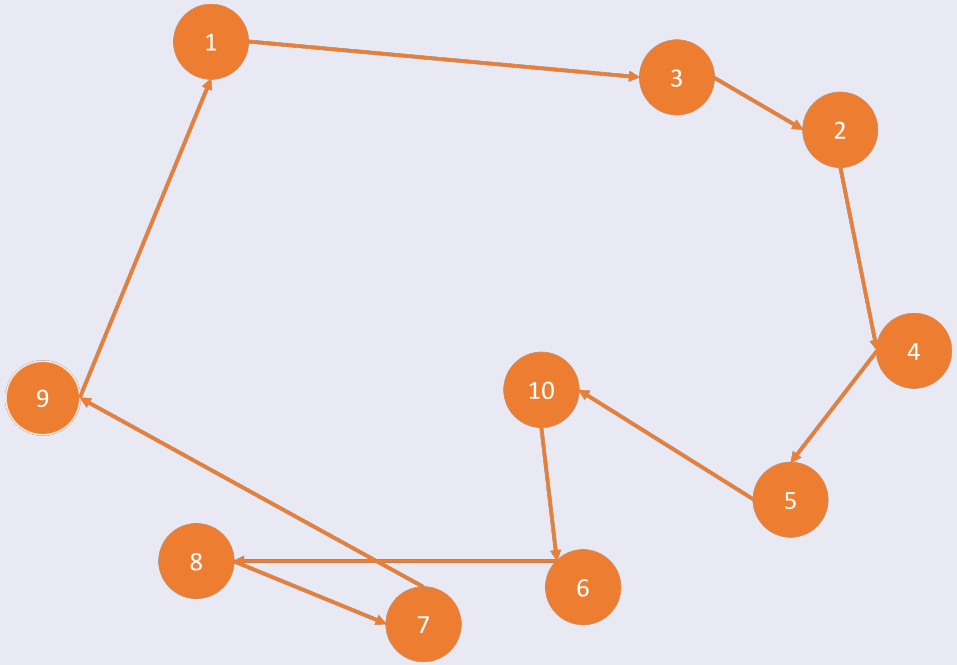
Xác định cấu hình của 1 nghiệm

Thuật toán Local Search cho bài toán TSP

Xác định cấu hình của 1 nghiệm

Ví dụ:

$$C = [1, 3, 2, 4, 5, 10, 6, 8, 7, 9, 1]$$



Thuật toán Local Search cho bài toán TSP

Định nghĩa hàng xóm của một cấu hình

Thuật toán Local Search cho bài toán TSP

Định nghĩa hàng xóm của một cấu hình

Hai cấu hình nghiệm là hàng xóm của nhau nếu chúng sai khác nhau 2 cạnh - 2-Opt.
Lựa chọn 2 cạnh: $[c_i, c_{i+1}]$ và $[c_j, c_{j+1}]$ từ cấu hình hiện tại sau đó tạo cấu hình mới chứa 2 cạnh $[c_i, c_j]$ và $[c_{i+1}, c_{j+1}]$.

Thuật toán Local Search cho bài toán TSP

Định nghĩa hàng xóm của một cấu hình

Hai cấu hình nghiệm là hàng xóm của nhau nếu chúng sai khác nhau 2 cạnh - 2-Opt.
Lựa chọn 2 cạnh: $[c_i, c_{i+1}]$ và $[c_j, c_{j+1}]$ từ cấu hình hiện tại sau đó tạo cấu hình mới chứa 2 cạnh $[c_i, c_j]$ và $[c_{i+1}, c_{j+1}]$.

$$\text{Cấu hình hiện tại: } [..., c_i, c_{i+1}, c_{n_1}, c_{n_2}, ..., c_{n_{s-1}}, c_{n_s}, c_j, c_{j+1}, ...]$$

$$\text{Cấu hình mới (hàng xóm): } [..., c_i, c_j, c_{n_s}, c_{n_{s-1}}, ..., c_{n_2}, c_{n_1}, c_{i+1}, c_{j+1}, ...]$$

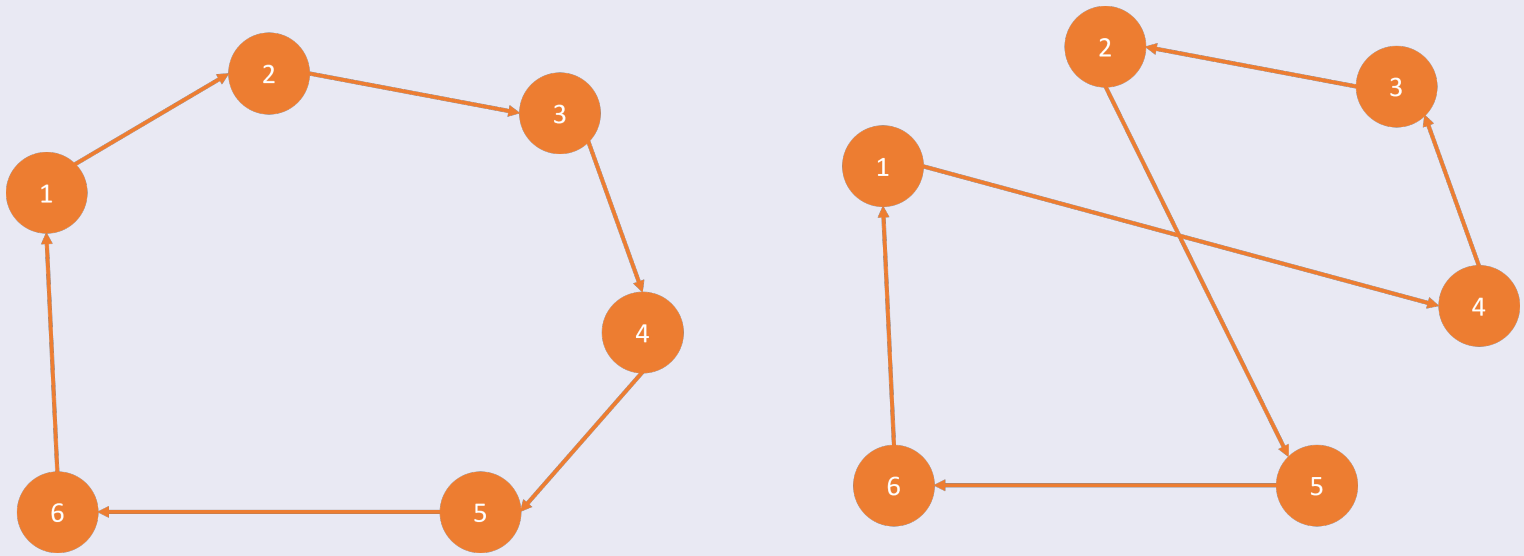
Thuật toán Local Search cho bài toán TSP

Định nghĩa hàng xóm của một cấu hình

Thuật toán Local Search cho bài toán TSP

Định nghĩa hàng xóm của một cấu hình

Ví dụ:



$$[1, 2, 3, 4, 5, 6, 1] \rightarrow [1, 4, 3, 2, 5, 6, 1]$$

Thuật toán Local Search cho bài toán TSP

Giá trị của một cấu hình

Thuật toán Local Search cho bài toán TSP

Giá trị của một cấu hình

Giá trị để đánh giá 1 cấu hình nghiệm là tổng trọng số của chu trình biểu diễn bởi cấu hình đó.
Mục tiêu là tìm cấu hình có giá trị nhỏ nhất.

Thuật toán Local Search cho bài toán TSP

Tiêu chuẩn dừng

Thuật toán Local Search cho bài toán TSP

Tiêu chuẩn dừng

Thuật toán dừng khi tại một cấu hình không tìm được hàng xóm nào có giá trị tốt hơn cấu hình hiện tại hoặc số bước lặp vượt quá số M cho trước.

Thuật toán Local Search cho bài toán TSP

Ví dụ cài đặt thuật toán

Thuật toán Local Search cho bài toán TSP

Ví dụ cài đặt thuật toán

Cho bài toán TSP có ma trận kề như sau:

$$\begin{bmatrix} 0 & 38 & 31 & 22 & 44 \\ 38 & 0 & 86 & 57 & 85 \\ 31 & 86 & 0 & 8 & 98 \\ 22 & 57 & 8 & 0 & 5 \\ 44 & 85 & 98 & 5 & 0 \end{bmatrix}$$

Cấu hình xuất phát:

$$S_0 = [1, 4, 2, 3, 5, 1]$$

Với $p(S_0) = 307$

Trực quan thuật toán.

Thuật toán Local Search cho bài toán TSP

Ví dụ cài đặt thuật toán

Thuật toán Local Search cho bài toán TSP

Ví dụ cài đặt thuật toán

Sử dụng thuật toán Local Search, tìm được nghiệm đủ tốt:

$$S = [1, 2, 5, 4, 3, 1], p(S) = 167$$

Thuật toán Local Search cho bài toán TSP

Ví dụ cài đặt thuật toán

Sử dụng thuật toán Local Search, tìm được nghiệm đủ tốt:

$$S = [1, 2, 5, 4, 3, 1], p(S) = 167$$

Sử dụng thuật toán vét cạn, tìm được nghiệm tối ưu:

$$S = [1, 2, 5, 4, 3, 1], p(S) = 167$$

Thuật toán Local Search cho bài toán TSP

Đánh giá độ phức tạp của thuật toán

Thuật toán Local Search cho bài toán TSP

Đánh giá độ phức tạp của thuật toán

Với số M cho trước là số bước lặp tối đa, tại mỗi bước lặp, ta duyệt các qua hàng xóm của cấu hình hiện tại bằng việc thay đổi 2 cạnh.

Với mỗi cấu hình có tất cả N cạnh nên số lần duyệt tối đa là số cách chọn ra 2 cạnh từ N cạnh để thay đổi $\Rightarrow C_N^2 = \frac{N(N-1)}{2} = O(N^2)$

Do đó độ phức tạp của thuật toán Local Search là: $M \times O(N^2) = O(N^2)$

Tài liệu tham khảo

[1] Alsheddy, Abdullah & Voudouris, Christos & Tsang, Edward & Alhindi, Ahmad. (2016). Guided Local Search. 10.1007/978-3-319-07153-4_2-1.

[2] https://uet.vnu.edu.vn/wp-content/uploads/2018/03/Tóm-tắt-luận-án-tiến-sĩ-của-NCS-Trần-Ngọc-Hà.pdf

[3] https://en.wikipedia.org/wiki/List_of_metaphor-based_metaheuristics

[4] https://www.researchgate.net/figure/Examples-of-the-different-operations-in-the-variable-neighborhood-search-VNS-local_fig1_332150258

[5] https://www2.seas.gwu.edu/ simhaweb/champalg/tsp/tsp.html

[6] https://www.youtube.com/watch?v=eekow29FSoc&list=PLN4kTzLXGGgWNf4CDyoZZOsjOCftW5ej6

[7] https://www.researchgate.net/figure/Iterated-local-search_fig1_282055629

[8] https://www.researchgate.net/figure/Guided-Local-Search-strategy-escaping-from-traps-increasing-the-relative-objective_fig7_49399511

THANK YOU FOR YOUR ATTENTION!

$$X_{ij} = \begin{cases} 1 \text{ nếu nhân viên } i \text{ làm việc ca } j \\ 0 \text{ nếu ngược lại} \end{cases} \Bigg| \begin{array}{l} i \in [1, days] \\ j \in [1, 2.days] \end{array}$$

Hàm mục tiêu

$$min \quad C_0 \sum_{i=1}^{n_1} \sum_{j=1}^{days \times 2} X_{ij} + C_1 \sum_{i=1}^{n_2} \sum_{j=1}^{days} Y_{ij}$$