GIẢI THUẬT META-HEURISTIC GIẢI BÀI TOÁN NGƯỜI DU LỊCH

META-HEURISTIC ALGORITHM FOR SOLVING TRAVELLING SALESMAN PROBLEM

Võ Khánh Trung
Trường Đại học Bách khoa Hà Nội
Email: trungvokhanh@yahoo.com

Đặng Đại Thọ
Trường Cao đẳng Công nghệ Thông tin, Đại học Đà Nẵng
Email: ddtho.dt@gmail.com

TÓM TẮT

Bài toán người du lịch là một bài toán NP-khó thuộc thể loại tối ưu rời rạc hay tổ hợp được nghiên cứu trong vận trù học hoặc lý thuyết khoa học máy tính. Bài toán được phát biểu như sau: cho trước một danh sách các thành phố và khoảng cách giữa chúng, tìm chu trình ngắn nhất thăm mỗi thành phố đúng một lần. Hiện nay chưa có giải thuật chính xác nào để giải quyết bài toán này trong trường hợp tổng quát. Vì vậy, các giải thuật gần đúng đặc biệt được quan tâm. Trong bài báo này, chúng tôi đề xuất một giải thuật meta-heuristic sử dụng ý tưởng tìm kiếm địa phương để giải bài toán người du lịch. Giải thuật đã được cài đặt, thử nghiệm trên bộ dữ liệu chuẩn lấy từ TSPLIB và thu được những kết quả khá tốt.

Từ khóa: bài toán người du lịch; NP-khó; giải thuật meta-heuristic; tìm kiếm cục bộ; tối ưu hóa tổ hợp

ABSTRACT

The travelling salesman problem (TSP) is an NP-hard problem in combinatorial optimization, important in operations research and theoretical computer science. It asks the following question: given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city? Hower, today there are not exact algorithms to solve it in general case. Therefore, heuristic and approximation algorithms are especially interested. This paper proposes a new meta-heuristic algorithm based on local search for solving traveling salesman problem. This algorithm has been installed, tested on standard data sets taken from TSPLIB and obtained with quite good results.

Key words: travelling salesman problem (TSP); NP-hard problem; meta-heuristic algorithm; local search; combinatorial

1. Đặt vấn đề

Bài toán người du lịch là một bài toán tối ưu hóa tổ hợp thuộc lớp NP-khó, được giới thiệu lần đầu tiên vào năm 1930. Đây là một trong những bài toán được nghiên cứu chuyên sâu trong tối ưu hóa và lý thuyết khoa học máy tính.

Bài toán được phát biểu dưới dạng đồ thị như sau: Cho đồ thị vô hướng có trọng số G = (V, E, C) gồm n đỉnh với $V = \{v_1, v_2, ..., v_n\}$. Hãy tìm một hành trình $s = v_1 \to v_2 \to ... \to v_n \to v_1$ sao cho:

- Mỗi đỉnh chỉ xuất hiện 1 lần $(v_i \neq v_j, \forall i \neq j)$

- Tổng chi phí của hành trình $Cost(s) = \sum_{i=1}^{n-1} c_{v_i, v_{i+1}} + c_{v_n, v_1}$ là nhỏ nhất.

Cho đến nay, chưa có giải thuật chính xác để giải bài toán này trong trường hợp tổng quát. Vì vậy, các giải thuật gần đúng đặc biệt được quan tâm. Nghiên cứu này đề xuất giải thuật meta-heuristic sử dụng ý tưởng tìm kiếm địa phương để giải bài toán. Giải thuật đã được thử nghiệm trên 25 bộ dữ liệu chuẩn lấy từ TSPLIP, kết quả thực nghiệm được so sánh với kết quả tối ưu của các bộ dữ liệu chuẩn này.

2. Đề xuất giải thuật

Ý tưởng của giải thuật là sử dụng tìm kiếm cục bộ. Giải thuật cục bộ có thể tóm tắt như sau: xuất phát từ một lời giải ứng cử viên, thực hiện các bước lặp, mỗi bước lặp xét các ứng cử viên có thể sinh ra từ ứng cử viên hiện tại thông qua một số phép biến đổi (gọi là ứng cử viên láng giềng), di chuyển đến một ứng cử viên láng giềng tốt hơn ứng cử viên hiện tại.

Vấn đề đặt ra là quá trình tìm kiếm cục bộ kết thúc khi nào?

Một trong những lựa chọn phổ biến là kết thúc quá trình tìm kiếm nếu sau một số bước lặp mà không tìm được lời giải tốt hơn. Bởi vì trong toàn bộ quá trình tìm kiếm giải thuật cục bộ luôn chọn những lời giải ứng cử viên hợp lệ nên luôn có một lời giải hợp lệ dù giải thuật bị dừng ở bất kỳ thời điểm nào.

Dưới đây là mã giả của giải thuật tìm kiếm cục bộ:

1. Khởi tạo một ứng cử viên $s_0 \in S$
2. **while** (s_0 chưa phải là cực trị địa phương) **do**
3. Lựa chọn $s \in N(s)$ sao cho $f(s)<f(s_0)$
4. Thay thế s_0 bằng s
5. **end while**

Chú thích:

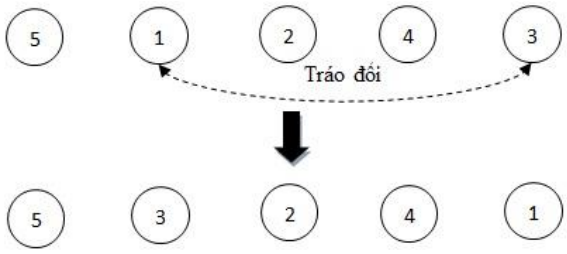
- S là tập các lời giải ứng cử viên.

- N(s) là tập các lời giải ứng cử viên hàng xóm của s.

- *f(s)* là hàm lượng giá của lời giải ứng cử viên s.

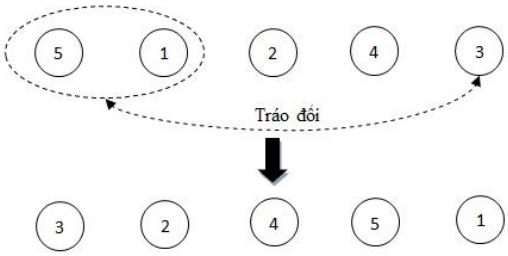
Để áp dụng giải thuật tìm kiếm cục bộ giải bài toán người du lịch, chúng tôi biểu diễn mỗi ứng cử viên s là một hoán vị của n số tự nhiên đầu tiên, kí hiệu (1, 2,…, n). Chẳng hạn, lời giải của bài toán người du lịch với 6 đỉnh được biểu diễn bằng hoán vị (1, 3, 2, 5, 4, 6) thì chu trình mà người du lịch sẽ đi là $1 \to 3 \to 2 \to 5 \to 4 \to 6 \to 1$. Hàm lượng giá *f(s)* là chi phí người du lịch phải trả cho hành trình của mình. Vấn đề then chốt của giải thuật tìm kiếm cục bộ là giải thuật tìm các ứng cử viên láng giềng của ứng cử viên s. Dưới đây chúng tôi đề xuất một số phương pháp tìm các ứng cử viên láng giềng một cách nhanh chóng và hiệu quả:

Cách thứ nhất: Tráo đổi 2 vị trí ngẫu nhiên trong hoán vị s.



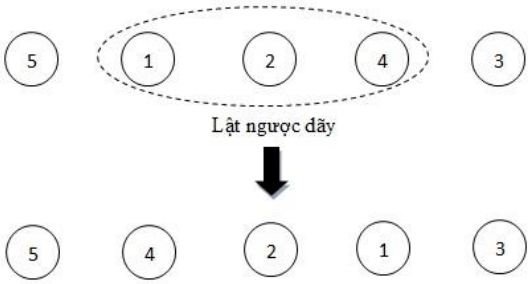
Hình 1. *Tráo đổi 2 vị trí trong hoán vị*

Cách thứ hai: Tráo đổi 2 dãy số ngẫu nhiên trong hoán vị s.



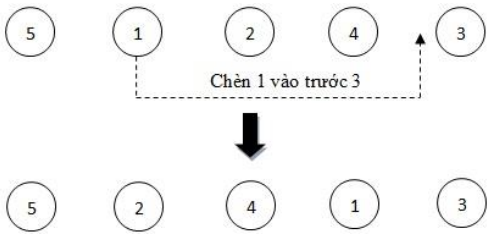
Hình 2. *Tráo đổi 2 dãy trong hoán vị*

Cách thứ ba: Chọn ngẫu nhiên một dãy số liên tiếp trong s và lật ngược dãy này.



Hình 3. *Lật ngược một dãy số liên tiếp ngẫu nhiên trong s*

Cách thứ tư: Chọn ngẫu nhiên một vị trí trong dãy, và di chuyển nó đến một vị trí mới.



Hình 4. *Di chuyển một vị trí trong s đến một vị trí mới*

Nhận xét:

- Giải thuật tìm kiếm cục bộ có thể hội tụ đến cực trị địa phương mà không phải là cực trị

toàn cục. Khi đó giải thuật có thể sẽ không trả về được lời giải tốt nhất.

- Thực nghiệm cho thấy trên đồ thị lớn, nếu sinh ra một lời giải khá tồi thì sẽ phải lặp rất nhiều bước để đến được cực trị địa phương. Nếu quá trình tìm kiếm cục bộ kết thúc do giới hạn thời gian mà vẫn chưa hội tụ đến cực trị địa phương thì lời giải trả về sẽ không tốt.

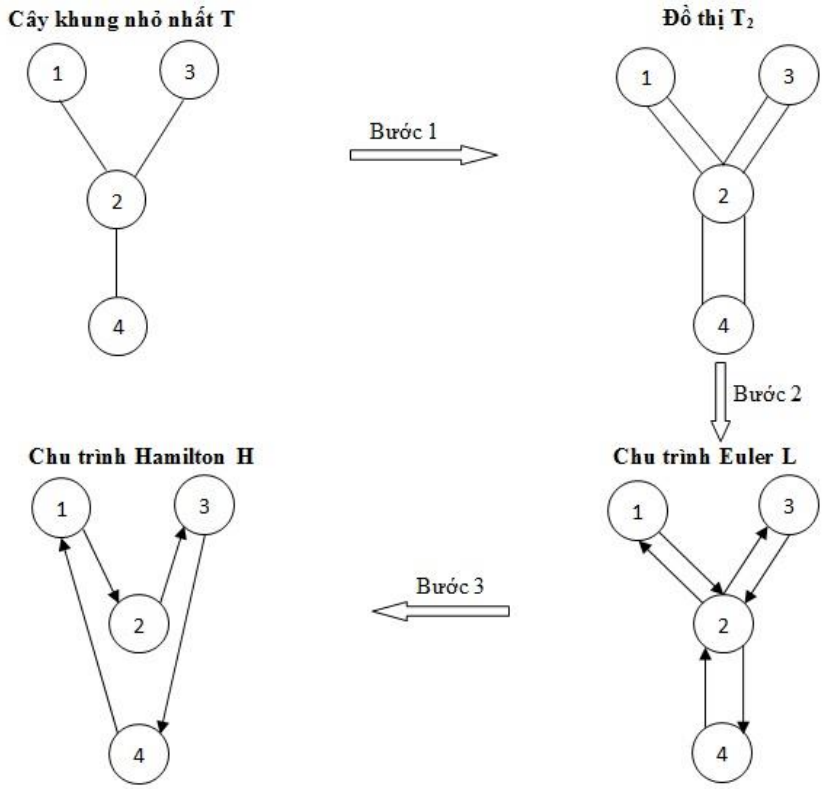
Sau đây là một số đề xuất để giải quyết những vấn đề nêu trên:

- Thay vì chỉ tìm kiếm cục bộ một lần, tiến hành tìm kiếm cục bộ k lần trên k lời giải được sinh ngẫu nhiên và chọn ra lời giải tốt nhất trong k cực trị địa phương.

- Trong k lần tìm kiếm cục bộ, sẽ có một lần tìm kiếm đặc biệt với lời giải khởi tạo s_0 là khá tốt.

Chúng tôi sử dụng giải thuật xấp xỉ tỷ lệ 2 dựa trên bài toán cây khung nhỏ nhất [11] để tìm một lời giải s_0 khá tốt:

Gọi $T_2 = (V, E_2)$ là đồ thị được xây dựng từ cây khung T bằng cách với mỗi cạnh $e = (v_1, v_2)$, bổ sung thêm một cạnh $e_1 = (v_1, v_2)$. Vì T_2 là đa đồ thị có tất cả các đỉnh đều là bậc chẵn, suy ra T_2 tồn tại chu trình Euler. Tìm chu trình Euler $L = (V, E_2)$ trên đồ thị T_2 đi qua lần lượt các đỉnh trên chu trình Euler L, chỉ lấy lần đầu tiên xuất hiện của mỗi đỉnh sẽ nhận được lời giải cho bài toán người du lịch với tỷ lệ xấp xỉ 2.



Hình 5. *Minh họa giải thuật xấp xỉ tỷ lệ 2 giải bài toán người du lịch*

Với ví dụ minh họa trên, chu trình Euler là $L = 1 \to 2 \to 3 \to 2 \to 4 \to 2 \to 1$

Chỉ giữ lại lần đầu tiên xuất hiện của các đỉnh trên chu trình L, ta có chu trình Hamilton như sau: $H = 1 \to 2 \to 3 \to 4 \to 1.$

Dưới đây là chứng minh lời giải tìm được bằng giải thuật nêu ở trên không quá hai lần so với kết quả tối ưu:

Giả sử đường đi Euler có dạng $L = v_1 \to v_2 \to ... \to v_m$, và chu trình Hamilton $H = v_{i_1} \to v_{i_2} \to ... \to v_{i_n}$

Với bất kỳ 2 đỉnh i, j liên tiếp trên chu trình H, gọi $a_1, a_2, ..., a_k$ là các đỉnh nằm giữa 2 đỉnh i, j trên chu trình Euler L, ta có:

$C_{ia_2} < C_{ia_1} +$

$C_{ia_3} < C_{ia_2} +$

....

$C_{ij} < C_{ia_k} + C_{a_k j}$

Suy ra $Cost(H) < Cost(L) < 2 * Cost(T)$

$\Rightarrow \frac{Cost(H)}{Cost(T)} < 2$

Giải thuật tìm cây khung nhỏ nhất là giải thuật chính xác. Từ đó suy ra kết quả của giải thuật xấp xỉ không quá hai lần kết quả tối ưu. Hay giải thuật trên là giải thuật xấp xỉ với tỷ lệ 2.

Dưới đây là mã giả của giải thuật đề xuất:

```
1.program SolingTheTravallingSalesmanProblem
2.      k ← số lần sử dụng tìm kiếm cục bộ
3. solution1 ← lời giải xấp xỉ tỷ lệ là 2 cho TSP
4. for i ← 2 to k do
5.    solutioni ← sinh lời giải ngẫu nhiên cho TSP
6. end for
7. for i ← 1 to k do
8.    s0 ← solutioni // Khởi tạo lời giải ban đầu
9.    while (s0 chưa phải là cực trị địa phương) do
10.       for sT ← N(s0) do
11.          if ( f(s0) > f(sT) ) then
12.             s0 ← sT
13.          end if
14.       end for
15.   end while
16.   if ( f(sbest) > f(s0) )then
17.          best ← s0
18.    end if
23. end for
24. return best
25. end program
```

3. Thực nghiệm

3.1. Dữ liệu thực nghiệm

Dữ liệu thử nghiệm là các bộ dữ liệu chuẩn được lấy từ thư viện TSPLIB, ở địa chỉ http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/tsp/. Các đỉnh được mô tả bằng các điểm trên mặt phẳng, trọng số giữa các đỉnh là khoảng cách Euclidena giữa chúng. Kích thước của bộ dữ liệu nhỏ nhất là 51 đỉnh và lớn nhất là 299 đỉnh. Các bộ dữ liệu đều cung cấp kết quả tối ưu để so sánh với kết quả tìm được bởi giải thuật đề xuất.

3.2. Thiết lập hệ thống

Chương trình cài đặt giải thuật đề xuất sử dụng ngôn ngữ lập trình Java, được thiết lập chạy 5 lần trên mỗi bộ dữ liệu và lấy kết quả tốt nhất. Máy tính sử dụng có cấu hình Intel Core 2 Duo T6400 2GHz, 4Gb RAM.

3.3. Kết quả thực nghiệm

Kết quả thực nghiệm được thống kê dưới bảng sau (*kết quả được làm tròn thành số nguyên*):

Bảng 1. *Kết quả giải thuật đề xuất giải bài toán người du lịch*

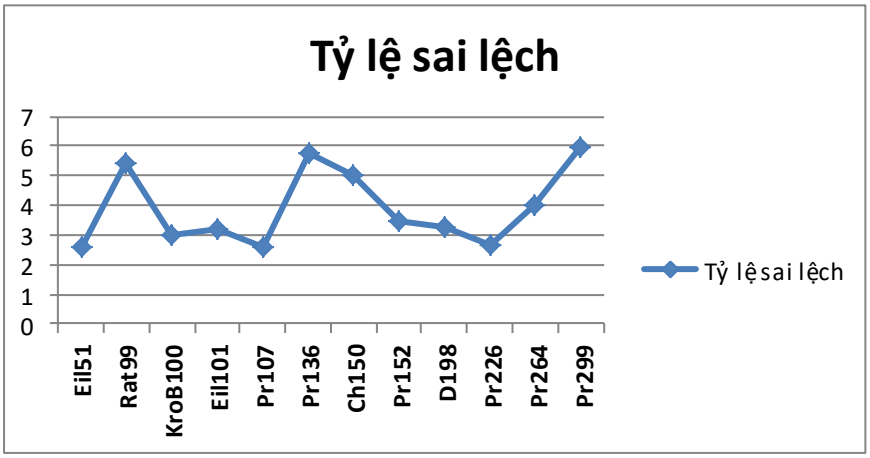
Index	Instance	N	Best Solution	Solution	Gap
1	Eil51	51	426	437	2.58216
2	Eil76	76	538	556	3.345725
3	Pr76	76	108159	110972	2.600801
4	Rat99	99	1211	1277	5.450041
5	KroA100	100	21282	22235	4.477963
6	KroB100	100	22141	22806	3.003478
7	Rd100	100	7910	8220	3.91909
8	Eil101	101	629	649	3.17965
9	Lin105	105	14379	14775	2.754016

10	Pr107	107	44303	45449	2.586732
11	Pr124	124	59030	60759	2.929019
12	Bier127	127	118282	121390	2.627619
13	Ch130	130	6110	6382	4.451718
14	Pr136	136	96772	102357	5.771297
15	Pr144	144	58537	59934	2.386525
16	Ch150	150	6528	6854	4.993873
17	KroA150	150	26524	27888	5.142512
18	Pr152	152	73682	76235	3.46489
19	Rat195	195	2323	2446	5.294877
20	D198	198	15780	16300	3.295311
21	KroA200	200	29368	31063	5.771588
22	KroB200	200	29437	31110	5.683324
23	Pr226	226	80369	82501	2.652764
24	Pr264	264	49135	51093	3.984939
25	Pr299	299	48191	51076	5.986595

Chú thích:

- Index: Chỉ số của bộ dữ liệu;
- Instance: Tên của bộ dữ liệu;
- N: Số lượng đỉnh;
- Best Solution: Kết quả tối ưu của bộ dữ liệu;
- Solution: Kết quả tìm được bởi giải thuật đề xuất;
- Gap: Sai lệch của kết quả tìm được so với kết quả tối ưu, được tính theo công thức: $gap = 100 * \frac{(s - s^*)}{s^*}\%$ [12] trong đó s là kết quả tìm được, s* là kết quả tối ưu.

Thực nghiệm cho thấy giải thuật đề xuất tìm được lời giải cho bài toán người du lịch với thời gian thực hiện khá nhanh. Kết quả tìm sai lệch không quá 6% so với lời giải tối ưu, và trung bình sai lệch là 3,9%.



Hình 6. *Biểu đồ tỷ lệ sai lệch giữa kết quả tìm được và kết quả tối ưu (đơn vị: %)*

4. Kết luận

Trong bài báo này chúng tôi đã đề xuất một giải thuật meta-heuristic để giải bài toán người du lịch dựa trên tìm kiếm cục bộ. Giải thuật được thiết kế với những bước di chuyển lời giải ngẫu nhiên, không quá phức tạp để cài đặt.

Kết quả thực nghiệm cho thấy thời gian thực hiện giải thuật khá nhanh. Kết quả tìm được khá tốt với sai số không quá 6% và trung bình sai lệch là 3,9% so với kết quả tối ưu trên bộ dữ liệu chuẩn. Vì thế, chúng ta có thể áp dụng giải thuật đề xuất để giải bài toán người du lịch có kích thước lớn trong thời gian cho phép mà vẫn nhận được kết quả khá tốt.

Trong tương lai, chúng tôi tiếp tục thực hiện cải tiến giải thuật này để thu được kết quả tốt hơn.

TÀI LIỆU THAM KHẢO

[1] R. Bellman, "Combinatorial processes and dynamic programming", in: *Combinatorial Analysis* (R. Bellman and M. Hall, Jr., eds.), American Mathematical Society, pp. 217-249.

[2] Woeginger, G.J. (2003), "Exact Algorithms for NP-Hard Problems: A Survey", *Combinatorial Optimization – Eureka, You Shrink! Lecture notes in computer science, vol. 2570*, Springer, pp. 185–207.

[3] Applegate, D. L.; Bixby, R. M.; Chvátal, V.; Cook, W. J. (2006), The *Traveling Salesman Problem*, ISBN 0-691-12993-2.

[4] Avriel, Mordecai (2003), *Nonlinear Programming: Analysis and Methods*, Dover Publications, ISBN 0-486-43227-0.

[5] P. Chundi and D. J. *Rosenkrantz*, Efficient Algorithms for Segmentation of Item-Set Time Series, Data Mining, An Analysis of Several Heuristics for the *Traveling Salesman Problem*, SIAM Journal on Computing, 6, 3, Sept. *1977*, 563-581.

[6] Johnson, D.S. and McGeoch, L.A., "*The traveling salesman problem: A case study in local optimization"*, Local search in combinatorial optimization, 1997, 215-310.

[7] S. S. Ray, S. Bandyopadhyay and S. K. Pal, *"Genetic Operators for Combinatorial Optimization in TSP and Microarray Gene Ordering*", Applied Intelligence, 2007, 26(3). pp. 183-195.

[8] A. B. Kahng and S. Reda, *"Match Twice and Stitch: A New TSP Tour Construction Heuristic"*, Operations Research Letters, 2004, 32(6). pp. 499–509.

[9] AARTS, E.H.L., AND J.K. LENSTRA (eds.) [1997], *Local Search in Combinatorial Optimization*, Wiley.

[10] N. Christofides, *Worst-case analysis of a new heuristic for the travelling salesman problem*, Report 388, Graduate School of Industrial Administration, Carnegie Mellon University, 1976.

[11] Matthias Englert, Heiko Roglin, and Berthold Vocking. *Worst case and probabilistic analysis of the 2-Opt algorithm for the TSP*, InProc. of the 18th Ann. ACM-SIAM Symp, On Discrete Algorithms (SODA), pages 1295–1304. SIAM, 2007.

[12] Mohammad Ahmadvand, Majid Yousefikhoshbakht, Narges Mahmoodi Darani. *Solving the Traveling Salesman Problem by an Efficient Hybrid Metaheuristic Algorithm*, Article 7, Volume 3, Number 3, Summer 2012, Page 75-84, 2012.

(BBT nhận bài: 29/09/2013, phản biện xong: 26/12/2013)