BỘ QUỐC PHÒNG
HỌC VIỆN KỸ THUẬT QUÂN SỰ

PHẠM ĐÌNH THÀNH

NGHIÊN CỨU PHÁT TRIỂN MỘT SỐ THUẬT TOÁN TIẾN HÓA GIẢI BÀI TOÁN CÂY KHUNG PHÂN CỤM ĐƯỜNG ĐI NGẮN NHẤT

Chuyên ngành : Cơ sở toán học cho tin học
Mã số : 9 46 01 10

TÓM TẮT LUẬN ÁN TIẾN SĨ TOÁN HỌC

HÀ NỘI - NĂM 2021

CÔNG TRÌNH ĐƯỢC HOÀN THÀNH TẠI
HỌC VIỆN KỸ THUẬT QUÂN SỰ - BỘ QUỐC PHÒNG

Người hướng dẫn khoa học: PGS.TS. Huỳnh Thị Thanh Bình

Phản biện 1: PGS.TS Lê Trọng Vĩnh

Phản biện 2: PGS.TS Ngô Hồng Sơn

Phản biện 3: PGS.TS Nguyễn Quang Uy

Luận án được bảo vệ tại Hội đồng đánh giá luận án cấp Học viện theo quyết định số ...../....., ngày ..... tháng.....năm..... của Giám đốc Học viện Kỹ thuật Quân sự, họp tại Học viện Kỹ thuật Quân sự vào hồi.....giờ.....ngày.....tháng.....năm.....

Có thể tìm hiểu luận án tại:

- Thư viện Học viện Kỹ thuật Quân sự
- Thư viện Quốc gia

GIỚI THIỆU

Trong nhiều ứng dụng mạng, nhằm đảm bảo tính hiệu quả và bảo mật, các thiết bị đầu cuối có thể được chia vào các nhóm sao cho việc kết nối giữa các thiết bị đầu cuối trong cùng một nhóm có tính "cục bộ". Khi đó, việc đảm bảo liên kết giữa các thiết bị đầu cuối, tương ứng với việc cần phải tìm cây khung của đồ thị con với các đỉnh thuộc cùng một nhóm. Ví dụ, trong lĩnh vực nông nghiệp, con người từ rất sớm đã có nhu cầu tối ưu hệ thống dẫn nước tưới tiêu từ một giếng nước tới các ốc đảo trong sa mạc; trong mỗi ốc đảo lại cần tối ưu hệ thống dẫn nước tới các vị trí trồng cây. Trong lĩnh vực bưu chính, giao vận,... các công ty có nhu cầu tối ưu vận chuyển thư từ, hàng hóa,... từ trung tâm tới các tỉnh, rồi từ các tỉnh lại vận chuyển tới các huyện, xã.

Với những yêu cầu thực tiễn đó, một lớp các bài toán cây khung, trong đó tập đỉnh được phân chia thành các tập con đã được quan tâm nghiên cứu. Trong đó, bài toán cây phân cụm đường đi ngắn nhất (*Clustered Shortest-Path Tree Problem - CluSPT*) [20] là bài toán có vai trò quan trọng trong các ứng dụng thực tiễn và nhận được nhiều sự quan tâm của các nhà nghiên cứu.

Do CluSPT là bài toán thuộc lớp NP-Khó [19, 20] nên luận án lựa chọn hướng tiếp cận giải xấp xỉ, sử dụng các thuật toán meta-heuristic và heuristic. Hiện nay, các thuật toán thuộc lớp meta-heuristic và heuristic được sử dụng để giải các bài toán tối ưu rất đa dạng, từ các thuật toán dựa trên hướng giảm của hàm số (gradient descent) [45], cho tới các thuật toán tiến hóa (*Evolutionary Algorithm - EA*) [6-8], hay các thuật toán lấy ý tưởng từ tối ưu trong tự nhiên [9, 89]. Trong những năm gần đây, các thuật toán có ý tưởng bắt nguồn từ tự nhiên được sử dụng rộng rãi để giải các bài toán có mức độ phi tuyến cao hoặc các bài toán tối ưu rất khó [90]. Trong các thuật toán lấy ý tưởng từ quá trình tối ưu hóa trong tự nhiên, EA là một trong các nhóm thuật toán được quan tâm nghiên cứu nhiều nhất và là một trong những kỹ thuật tính toán thông minh quan trọng nhất hiện nay [16, 52, 93].

Các thuật toán EA sử dụng các khái niệm trong sinh học và áp dụng vào lĩnh vực khoa học máy tính. Tối ưu đa nhân tố (multi-factorial optimization) là một mô hình mới của tiến hóa đa nhiệm vụ (evolutionary

multi-tasking) [35, 48, 77]. Điểm khác biệt dễ nhận thấy giữa tối ưu đa nhân tố và thuật toán EA cơ bản đó là thuật toán EA cơ bản chỉ tập trung vào giải một bài toán tối ưu tại một thời điểm trong khi tối ưu đa nhân tố thường giải đồng thời nhiều bài toán tối ưu. Thuật toán tiến hóa đa nhân tố (*Multi-Factorial Evolutionary Algorithm - MFEA*) là nghiên cứu đầu tiên về kết hợp giữa tối ưu đa nhân tố với thuật toán di truyền [35, 36]. Do thuật toán MFEA được kế thừa các ưu điểm của quá trình trao đổi tri thức tiềm ẩn (implicit knowledge transfer) giữa các bài toán nên quá trình tìm kiếm lời giải của thuật toán MFEA được cải thiện về cả tốc độ và chất lượng so với lời giải tìm được khi sử dụng thuật toán tiến hóa cơ bản.

Mặc dù đã được áp dụng vào giải hiệu quả nhiều lớp bài toán, cũng như các ứng dụng thuộc nhiều lĩnh vực khác nhau trong thực tế, tuy nhiên, các nghiên cứu về ứng dụng thuật toán EA và thuật toán MFEA vào giải các bài toán trên đồ thị, đặc biệt là tìm lời giải cho bài toán cây khung phân cụm vẫn còn hạn chế. Vì vậy, luận án này tập trung vào việc xây dựng thuật toán tiến hóa và tiến hóa đa nhân tố hiệu quả để giải các bài toán cây khung phân cụm, bao gồm từ xây dựng các toán tử tiến hóa mới như lai ghép, đột biến, giải mã,... cho tới tìm cơ chế mới trong việc kết hợp hiệu quả giữa thuật toán MFEA với các thuật toán khác.

Mục tiêu nghiên cứu của luận án

Mục tiêu nghiên cứu chính của luận án là xây dựng các thuật toán xấp xỉ để giải bài toán CluSPT, trong đó luận án tập trung vào hai hướng: sử dụng thuật toán tiến hóa (chương 3) và sử dụng thuật toán tiến hóa đa nhân tố (chương 4).

Các mục tiêu cụ thể trong luận án như sau:

1. Nghiên cứu bài toán CluSPT.
2. Nghiên cứu, đề xuất các toán tử tiến hóa hiệu quả giải bài toán CluSPT, đặc biệt đối với các toán tử cần thiết để áp dụng thuật toán MFEA như toán tử mã hóa và giải mã.
3. Nghiên cứu, đề xuất cơ chế kết hợp giữa thuật toán MFEA với các thuật toán xấp xỉ.
4. Nghiên cứu, đề xuất các thuật toán xấp xỉ giúp tìm kiếm nhanh lời giải bài toán CluSPT, dễ cài đặt và là cơ sở để so sánh, đánh

giá các thuật toán EA và MFEA được đề xuất.

5. Nghiên cứu các phương pháp đánh giá thuật toán bao gồm xây dựng bộ dữ liệu và phương pháp đánh giá thực nghiệm.

Phương pháp nghiên cứu

Phương pháp nghiên cứu dựa trên nghiên cứu lý thuyết, phân tích tài liệu, mô hình toán học và thực nghiệm để đánh giá các thuật toán đề xuất, so sánh với các thuật toán đã được nghiên cứu đã có để giải bài toán CluSPT. Từ đó, có thể đề xuất hướng tiếp cận phù hợp và hiệu quả giải bài toán CluSPT.

Phạm vi nghiên cứu của luận án

Luận án tập trung nghiên cứu:

- Bài toán CluSPT.
- Các thuật toán heuristic hiệu quả trên bài toán đồ thị.
- Thuật toán di truyền giải bài toán tối ưu trên đồ thị, đặc biệt các bài toán có tập đỉnh được chia thành các tập nhỏ hơn.
- Thuật toán MFEA và áp dụng giải các bài toán tối ưu tổ hợp.
- Phương pháp xây dựng các bộ dữ liệu, phương pháp đánh giá và tiến hành thực nghiệm.

Các đóng góp của luận án

Về lý thuyết:

1. Đề xuất thuật toán chính xác SLA-M giải bài toán CluSPT trên đồ thị metric (*metric graph*).
2. Đề xuất thuật toán HB-RGA kết hợp giữa thuật toán tham lam ngẫu nhiên kết hợp với ý tưởng của thuật toán Dijkstra để giải bài toán CluSPT. Thuật toán HB-RGA tìm được lời giải trong thời gian ngắn và chất lượng lời giải tốt hơn thuật toán đã có trước đó.
3. Đề xuất hai thuật toán tiến hóa C-EA và N-EA để giải bài toán CluSPT. Với mỗi thuật toán, luận án đề xuất mới toán tử lai ghép, toán tử đột biến và phương pháp mã hóa lời giải. Luận án cũng đề xuất cách cài đặt thực nghiệm tính hàm mục tiêu của bài toán CluSPT để giảm chi phí tính toán.
4. Đề xuất thuật toán tiến hóa đa nhân tố G-MFEA giải bài toán CluSPT. Thuật toán G-MFEA cho kết quả tốt hơn các thuật toán

trong nghiên cứu trước đây, trong đó, thuật toán G-MFEA tìm được lời giải tối ưu trên nhiều tập dữ liệu khác nhau.

Về mặt ứng dụng: các thuật toán đề xuất của luận án có thể áp dụng trực tiếp vào giải các bài toán trong thực tế trong kỹ thuật, trong sản xuất, v.v.

Cấu trúc của luận án

Ngoài phần Giới thiệu, luận án gồm các phần chính sau:

- ***Chương 1*** trình bày hai vấn đề: kiến thức cơ bản về các thuật toán và bài toán CluSPT.
- ***Chương 2*** trình bày đề xuất thuật toán chính xác và thuật toán tham lam ngẫu nhiên để giải bài toán CluSPT.
- ***Chương 3*** trình bày hai thuật toán tiến hóa giải bài toán CluSPT.
- ***Chương 4*** trình bày thuật toán tiến hóa đa nhân tố giải bài toán CluSPT.
- ***Chương 5*** trình bày kết quả thực nghiệm của các thuật toán.

CHƯƠNG 1: TỔNG QUAN

1.1. Thuật toán di truyền

Thuật toán di truyền được giới thiệu lần đầu vào năm 1975 bởi John Holland [39] và là mô hình đầu tiên của thuật toán EA được xây dựng và sử dụng [3, 4, 79].

1.2. Thuật toán tiến hóa đa nhân tố

Ý tưởng chính của MFEA như sau:

- Tạo ra một không gian tìm kiếm duy nhất với cách biểu diễn chung cho tất cả các tác vụ
- Áp dụng các toán tử của thuật toán EA như khởi tạo quần thể, lai ghép, đột biến lên không gian không gian tìm kiếm chung (*Unified Search Space - USS*) để biến đổi quần thể.
- Đánh giá mỗi cá thể trong không gian tìm kiếm chung thông qua những tiêu chí đối với từng tác vụ trong quần thể.

1.3. Bài toán cây phân cụm đường đi ngắn nhất

1.3.1. Một số định nghĩa

Cho $G = (V, E, w)$ là một đồ thị vô hướng, liên thông, có trọng số cạnh không âm; trong đó V và E lần lượt là tập đỉnh và tập cạnh của

đồ thị; w là ma trận trọng số cạnh của đồ thị. Cho trước tập các đỉnh $S \subseteq V$, ký hiệu $G[S]$ là đồ thị con của G được cảm sinh bởi tập S. Tương tự, $T[S]$ là đồ thị con của cây khung T được cảm sinh bởi tập S.

Định nghĩa 1.1 (Phân hoạch của tập đỉnh của đồ thị [51]). *Cho $G = (V,E,w)$ là một đồ thị vô hướng, liên thông, các cạnh có trọng số không âm. Tập $C = \{C_1, C_2, \ldots, C_h\}$ được gọi là phân hoạch của V nếu $C_1 \cup C_2 \cup \ldots \cup C_h = V$ và $C_i \cap C_j = \emptyset, \forall i, j \in [1,h], i \neq j$.*

Định nghĩa 1.2 (Chi phí định tuyến giữa hai đỉnh [20]). *Cho $G = (V,E,w)$ là một đồ thị vô hướng, liên thông, các cạnh có trọng số không âm. Chi phí định tuyến giữa hai đỉnh $u, v \in V$ trên cây khung T (ký hiệu $d_T(u,v)$) của đồ thị G được tính bằng chi phí đường đi nối giữa hai đỉnh đó trên cây khung T.*

1.3.2. Phát biểu bài toán

Bài toán CluSPT được phát biểu như sau [19, 20]:

Cho một đơn đồ thị vô hướng $G = (V,E,w)$, một phân hoạch $C = \{C_1, C_2, \ldots, C_h\}$ của V và đỉnh nguồn $s \in V$. Mục tiêu của bài toán CluSPT là tìm một cây khung T của đồ thị G sao cho:

- Với mỗi cụm $C_i (i = 1, \ldots, h)$, đồ thị con $T[C_i]$ là một đồ thị liên thông.
- Tổng chi phí định tuyến giữa đỉnh nguồn s và các đỉnh còn lại trên cây khung T là nhỏ nhất, hay nói cách khác:

$$f(T) = \sum_{v \in V(T)} d_T(s,v) \rightarrow \min \tag{1.1}$$

Có hai trường hợp lời giải s' của bài toán CluSPT không hợp lệ:

- Lời giải s' không phải là một cây khung.
- Tồn tại một đồ thị con trong một cụm của lời giải s' là đồ thị không liên thông.

1.3.3. Tổng quan tình hình nghiên cứu

Các bài toán liên quan đến tập đỉnh được phân vào các cụm đã được biết đến từ những năm 70 của thế kỷ trước [5, 38, 54]. Gần đây,

tác giả D'Emidio và các cộng sự [19, 20] đã nghiên cứu một dạng khác của bài toán cây khung phân cụm, bài toán CluSPT. Bài toán CluSPT xuất hiện nhiều trong các ứng dụng cần tối ưu về thiết kế mạng, kết nối hệ thống cáp TV và hệ thống cáp quang. Tác giả đã đề xuất một thuật toán xấp xỉ AAL (Approximation Algorithm) để giải bài toán CluSPT. Ý tưởng chính của thuật toán AAL là lần lượt tìm cây khung nhỏ nhất cho đồ thị con được cảm sinh từ tập đỉnh của mỗi cụm và đồ thị được nhận được bằng cách coi mỗi cụm là một đỉnh.

1.4. Kết luận chương

Chương này đã trình bày: kiến thức cơ bản về các thuật toán (thuật toán tiến hóa và thuật toán tiến hóa đa nhân tố); giới thiệu bài toán CluSPT - là bài toán thuộc lớp bài toán NP-Khó; trình bày về ứng dụng của bài toán CluSPT trong lĩnh vực mạng truyền thông, trong nông nghiệp và trong phân phối hàng hóa, dịch vụ; giới thiệu một thuật toán xấp xỉ giải bài toán CluSPT. Chương này cũng giới thiệu một số nghiên cứu liên quan tới bài toán CluSPT.

CHƯƠNG 2: THUẬT TOÁN XẤP XỈ GIẢI BÀI TOÁN CÂY PHÂN CỤM VỚI ĐƯỜNG ĐI NGẮN NHẤT

Chương này trình bày về hai thuật toán xấp xỉ giải bài toán CluSPT: thuật toán dựa trên chiến lược tìm kiếm tham lam ngẫu nhiên và thuật toán tìm lời giải dựa trên chiến lược hình sao.

2.1. Thuật toán xây dựng cây khung hình sao

Phần này trình bày về thuật toán tìm lời giải bài toán CluSPT trên lớp đồ thị đầy đủ. Luận án cũng chứng minh rằng, đối với đồ thị metric [51], lời giải tìm được là tối ưu.

2.1.1. Lược đồ thuật toán

Luận án đề xuất thuật toán xấp xỉ (ký hiệu là SLA) giải bài toán CluSPT thông qua việc tìm các cây khung có dạng hình sao (star-like) gồm hai mức (two-level). Trong cây khung dạng hình sao, một đỉnh đóng vai trò là đỉnh trung tâm, các đỉnh còn lại nối với đỉnh này thông qua các cạnh nối giữa hai đỉnh hoặc đường nối giữa hai đỉnh.

Mã giả của thuật toán SLA được trình bày trong thuật toán 2.1.

Thuật toán 2.1: Thuật toán SLA

Input: Đồ thị phân cụm $G = (V, E, w, C)$;
Đỉnh nguồn s;
Output: Một lời giải của bài toán CluSPT $T = (V_T, E_T)$;

begin
  $V_T \leftarrow V$;
  $E_T \leftarrow \emptyset$;
  $C_t \leftarrow$ Cụm chứa đỉnh nguồn s;
  foreach *đỉnh* $v \in C_t, v \neq s$ **do**
    $p_v \leftarrow$ Đường đi ngắn nhất nối s và v trên đồ thị $G[C_t]$;
    $E_T \leftarrow E_T \cup p_v$;
  foreach *cụm* C_i *với* $i \neq t$ **do**
    foreach *đỉnh* $v \in C_i$ **do**
      $p_{v,u} \leftarrow$ Đường đi ngắn nhất nối v và $u (u \in C_i)$ trên $G[C_i]$;
      $d_{v,u} \leftarrow$ Chi phí của đường đi $p_{v,u}$;
      $f_v \leftarrow |C_i| \times d(s,v) + \sum_{u \in C_i} d_{v,u}$;
    $r_i = argmin\{f(v) | v \in C_i\}$;
    $E_T \leftarrow E_T \cup p_{r_i,u}, \forall u \in C_i, u \neq r_i$;
  foreach *cụm* $C_i, i \neq n$ **do**
    $E_T \leftarrow E_T \cup e = (s, r_i)$;
  return (V_T, E_T);

2.1.2. Thuật toán dạng hình sao trên đồ thị metric

Bổ đề 2.1.1. *Nếu cây T là một lời giải tối ưu của bài toán CluSPT trên đồ thị metric thì mỗi cây khung bộ phận (*local tree*) của T là một đồ thị dạng sao.*

Bổ đề 2.1.2. *Nếu cây khung T là một lời giải tối ưu của bài toán CluSPT trên đồ thị metric thì mỗi cạnh liên cụm (*inter-cluster edge*) trong lời giải T sẽ là cạnh nối giữa gốc của một cây khung bộ phận và đỉnh nguồn s của đồ thị G.*

Từ bổ đề (2.1.1) và (2.1.2) suy ra thuật toán SLA có thể tìm được lời giải tối ưu của bài toán CluSPT trong đồ thị metric (ký hiệu là SLA-M) khi thay khoảng cách giữa hai đỉnh v và u trong bằng trọng số cạnh nối giữa hai đỉnh v và u.

2.2. Thuật toán tham lam ngẫu nhiên

Thuật toán đề xuất (ký hiệu HB-RGA) dựa trên sự kết hợp giữa thuật toán tham lam ngẫu nhiên (*Randomized Greedy Algorithm - RGA*) và thuật toán cây đường đi ngắn nhất (*Shortest Path Tree Algorithm - SPTA*), trong đó:

- Thuật toán SPTA được sử dụng để tạo cây đường đi ngắn nhất cho mỗi cụm.
- Thuật toán RGA được sử dụng để tìm các cạnh nối giữa các cụm.

Lược đồ thuật toán HB-RGA được trình bày trong thuật toán 2.2.

Trong thuật toán 2.2, phương thức $Find\_Shortest\_Path\_Tree(x)$ sẽ sử dụng thuật toán Dijkstra để tìm cây đường đi ngắn nhất với đỉnh bắt đầu là đỉnh x.

2.3. Kết luận chương

Mặc dù thuật toán HB-RGA có nhiều ưu điểm như: ý tưởng và cài đặt đơn giản, độ phức tạp tính toán không cao, kết quả gần với kết quả tối ưu. Tuy nhiên, thuật toán HB-RGA có hạn chế là chiến lược tìm kiếm chỉ phù hợp với bài toán CluSPT, không thể sử dụng cho các bài toán khác, kể cả các bài toán mà lời giải có cùng cấu trúc (chỉ khác hàm mục tiêu).

Thuật toán SLA-M có thể tìm được lời giải tối ưu trên đồ thị metric. Tuy nhiên, thuật toán SLA-M có hạn chế là chỉ áp dụng được và có hiệu quả trên lớp đồ thị metric.

Nghiên cứu của chương này đã công bố trong công trình [III] và [VI].

CHƯƠNG 3: THUẬT TOÁN TIẾN HÓA GIẢI BÀI TOÁN CÂY PHÂN CỤM VỚI ĐƯỜNG ĐI NGẮN NHẤT

Chương này trình bày đề xuất dựa trên thuật toán tiến hóa để giải bài toán CluSPT. Thuật toán đề xuất đảm bảo lời giải tìm được luôn hợp lệ và được xây dựng dựa trên ý tưởng phân rã bài toán CluSPT

Thuật toán 2.2: Thuật toán HB-RGA

Input: Đồ thị phân cụm $G = (V, E, w, C)$;
Đỉnh nguồn s;
Output: Một lời giải của bài toán CluSPT $T = (V_T, E_T)$;

begin
  $V_T \leftarrow V$;
  $Q \leftarrow \{1, 2, \ldots, h\}$;
  $cur \leftarrow$ Chỉ số của cụm chứa đỉnh nguồn s;
  $T \leftarrow Find\_Shortest\_Path\_Tree(s)$;
  $dis[cur] = 0$ ▷ Khoảng cách từ cụm C_{cur} tới cụm gốc;
  $Q \leftarrow Q \backslash cur$;
  while $Q \neq \emptyset$ **do**
    foreach *cụm* C_i *với* $i \in Q$ **do**
      $m \leftarrow \text{random}(|C_i|, \sum_{j \in Q} |C_j|)$;
      foreach *cạnh* (u, v), $u \in C_{cur}$, $v \in C_i$ **do**
        $d[u] \leftarrow$ Chi phí của đường đi ngắn nhất trên cây T từ đỉnh gốc của cụm C_{cur} tới đỉnh u;
        $CostSPT(v) \leftarrow$ Tổng chi phí đường đi ngắn nhất từ đỉnh v tới các đỉnh khác trong cụm C_i;
        $f(u, v) = m \times (d[u] + w[u, v]) + CostSPT(v)$;
      $e = (a, b) \leftarrow$ Chọn cạnh phù hợp nhất trong tập $\{(u, v) | u \in C_{cur}, v \in C_i\}$ theo xác suất
      $$p(u, v) = \frac{f(u, v)^{\gamma}}{\sum_{u' \in C_{cur}, v' \in C_i} f(u', v')^{\gamma}};$$
      if $dis[i] > dis[cur] + d[a] + w[a, b]$ **then**
        $dis[i] \leftarrow dis[cur] + d[a] + w[a, b]$;
        $r_i \leftarrow b$;
        $temporaryEdge[i] \leftarrow e$;
    $cur \leftarrow \arg\min_{i \in Q} dis[i]$;
    $T \leftarrow T \cup temporaryEdge[cur]$;
    $T \leftarrow T \cup Find\_Shortest\_Path\_Tree(r_{cur})$;
    $Q \leftarrow Q \backslash cur$;

thành hai bài toán con, sau đó áp dụng thuật toán tiến hóa đối với cả hai bài toán con hoặc kết hợp áp dụng thuật toán tiến hóa giải bài toán con thứ nhất và áp dụng thuật toán đúng giải bài toán con thứ hai.

3.1. Thuật toán tiến hóa dựa trên mã Cayley

Phần này trình bày áp dụng thuật toán tiến hóa để giải bài toán CluSPT (ký hiệu là C-EA) dựa trên mã hóa cây khung trong mỗi cụm và cây khung nối giữa các cụm bằng mã Cayley.

3.1.1. Phương pháp phân rã bài toán CluSPT

Trong hướng tiếp cận này, mỗi bài toán được phân rã thành hai bài toán con: bài toán thứ nhất sẽ xác định cây khung của mỗi cụm; bài toán con thứ 2 sẽ xác định cây khung của đồ thị $G-Graph$.

Mã giả của hướng tiếp cận C-EA được trình bày trong thuật toán 3.1. Điểm lưu ý trong thuật toán này là để tính giá trị thích nghi của cá thể ind_j thì bắt buộc phải xác định trước tất cả các cây khung bộ phận và cây khung toàn cục (*global tree*) của lời giải bài toán CluSPT tương ứng s_j (xem dòng 4 và dòng 14 trong thuật toán 3.1). Sau đó, thuật toán mới tính giá trị thích nghi của lời giải s_j của bài toán CluSPT. Cuối cùng, giá trị thích nghi $fit(ind_j)$ của cá thể ind_j được tính thông qua giá trị chi phí của cá thể s_j.

3.1.2. Mã hóa cá thể

Một cá thể trong không gian tìm kiếm được mã hóa thông qua hai giai đoạn: Giai đoạn đầu sẽ mã hóa cây khung trong mỗi cụm, giai đoạn thứ hai sẽ mã hóa cây khung nối giữa các cụm.

Một nhiễm sắc thể gồm $h+2$ đoạn được đánh đánh số từ 1 tới $h+2$: h đoạn đầu tiên sẽ là các chuỗi mã Cayley biểu diễn các cây khung bộ phận tương ứng. Đoạn thứ $h+1$ là mã Cayley của cây khung toàn cục. Đoạn thứ $h+2$ là M-Seg chứa các đỉnh gốc của các cây khung bộ phận.

3.1.3. Phương pháp khởi tạo cá thể

Để tạo ra cá thể hợp lệ cho bài toán CluSPT, cây khung của mỗi cụm và của đồ thị $G-Graph$ lần lượt được xây dựng dựa trên tạo ngẫu nhiên mã Cayley.

3.1.4. Toán tử lai ghép

Thuật toán C-EA sử dụng lai ghép một điểm cắt [8].

Do lời giải của bài toán CluSPT được mã hóa thành các chuỗi Cayley nên phép lai ghép một điểm cắt luôn tạo ra cá thể con hợp lệ.

Thuật toán 3.1: Lược đồ thuật toán C-EA

Input: Đồ thị phân cụm $G = (V, E, w, C)$;
Đỉnh nguồn s;
Output: Lời giải của bài toán CluSPT;

begin
  $P_0 \leftarrow$ Tạo ngẫu nhiên N cá thể;
  foreach *cá thể* $ind_j \in P_0$ **do**
    Tạo cây cây khung bộ phận và cây khung toàn cục của lời giải bài toán CluSPT s_j tương ứng với cá thể ind_j;
    Tính giá trị thích nghi của cá thể ind_j dựa trên chi phí của lời giải s_j;
  $t \leftarrow 0$;
  while *điều kiện dừng chưa thỏa mãn* **do**
    $O_t \leftarrow \emptyset$;
    while *số_cá_thể_con_được_sinh* $< N$ **do**
      Chọn ngẫu nhiên hai cá thể p_a và p_b từ quần thể P_t;
      Lai ghép, đột biến cá thể p_a và p_b tạo ra hai cá thể con o_a và o_b;
      $O_t \leftarrow O_t \cup \{o_a, o_b\}$;
    foreach *cá thể* $o_j \in O_t$ **do**
      Tạo cây cây khung bộ phận và cây khung toàn cục của lời giải bài toán CluSPT s'_j tương ứng với cá thể o_j;
      Tính giá trị thích nghi của cá thể o_j dựa trên chi phí của lời giải s'_j;
    $R_t \leftarrow O_t \cup P_t$;
    $P_{t+1} \leftarrow$ Chọn N cá thể tốt nhất từ R_t;
    $t \leftarrow t + 1$;
  $ind^* \leftarrow$ cá thể tốt nhất từ P_t;
  Tạo cây khung bộ phận và cây khung toàn cục của lời giải bài toán CluSPT s^* tương ứng với cá thể ind^*;
  return s^*;

3.1.5. Toán tử đột biến

Toán tử đột biến thực hiện hai thay đổi khác nhau trên cá thể:

- Thay đổi đầu sẽ tạo ra chuỗi Cayley mới bằng cách hoán đổi vị trí của hai gen trên một đoạn.
- Thay đổi thứ hai chỉ tác động lên đoạn M-Seg thông qua thay đổi các gốc của cây khung bộ phận.

3.2. Hướng tiếp cận dựa trên giảm không gian tìm kiếm của thuật toán tiến hóa

3.2.1. Cách tiếp cận

Hướng tiếp cận dựa trên thuật toán EA để tìm kiếm lời giải hợp lệ và có chi phí nhỏ nhất có thể của bài toán CluSPT. Thông thường các hướng tiếp cận này sẽ tìm kiếm lời giải trên toàn bộ không gian lời giải của bài toán CluSPT. Do bài toán CluSPT thuộc lớp bài toán NP-Khó, nên việc tiếp cận như trên dẫn tới hao phí tài nguyên tính toán và thời gian, đặc biệt khi số chiều của đồ thị đầu vào lớn. Vì vậy, phần này sẽ giới thiệu hướng tiếp cận mới (gọi là N-EA) dựa trên việc phân rã bài toán CluSPT thành hai bài toán con nhỏ hơn.

3.2.2. Phương pháp phân rã bài toán CluSPT

Bài toán CluSPT được phân rã thành hai bài toán con (sub-problem): bài toán con thứ nhất (ký hiệu là *H-Problem*) sẽ tìm các cạnh nối giữa các cụm; bài toán con thứ hai (ký hiệu là *L-Problem*) xác định cây khung của đồ thị con trong mỗi cụm. Với cách tiếp cận này, thuật toán N-EA giải bài toán H-problem trước, sau đó ứng với lời giải của bài toán H-Problem sẽ tìm lời giải của bài toán L-Problem.

Mã giả của thuật toán đề xuất được trình bày trong thuật toán 3.2. Trong thuật toán 3.2, giá trị thích nghi của cá thể được tính thông qua chi phí của lời giải của bài toán CluSPT (dòng 4 và dòng 11 trong thuật

toán 3.2).

Thuật toán 3.2: Lược đồ thuật toán N-EA

Input: Đồ thị phân cụm $G = (V, E, w, C)$; Đỉnh nguồn s;
Output: Lời giải của bài toán CluSPT;

begin
  $P_0 \leftarrow$ Tạo ngẫu nhiên N cá thể của bài toán H-Problem;
  foreach *cá thể* $ind_j \in P_0$ **do**
    Xây dựng lời giải s_j của CluSPT dựa trên cá thể ind_j;
    Tính giá trị thích nghi của ind_j dựa trên chi phí của s_j;
  $t \leftarrow 0$;
  while *điều kiện dừng chưa thỏa mãn* **do**
    $P'_t \leftarrow$ Tournament Selection(P_t);
    $O_t \leftarrow$ Thực hiện lai ghép và đột biến(P'_t);
    foreach *cá thể* $c_j \in O_t$ **do**
      Xây dựng lời giải s'_j của CluSPT dựa trên c_j;
      Tính giá trị thích nghi của c_j từ chi phí của s'_j;
    $R_t \leftarrow O_t \cup P_t$;
    $P_{t+1} \leftarrow$ Chọn N cá thể tốt nhất từ R_t;
    $t \leftarrow t + 1$;
  $ind^* \leftarrow$ cá thể tốt nhất từ P_t;
  Xây dựng lời giải s^* của CluSPT dựa trên cá thể ind^*;
  return s^*;

3.2.3. Biểu diễn cá thể

Mỗi nhiễm sắc thể là một mảng các đỉnh, trong đó, phần tử thứ i của mảng là một đỉnh gốc r_i của cụm thứ i. Gốc của cụm thứ i được sử dụng để xây dựng các cạnh nối giữa cụm thứ i và các cụm khác.

3.2.4. Phương pháp khởi tạo cá thể

Thuật toán N-EA sẽ tạo ngẫu nhiên cá thể Ind=($ind_1, ind_2, \ldots, ind_h$), trong đó ind_i là gốc của cụm thứ i ($i = 1, \ldots, h$).

3.2.5. Toán tử lai ghép

Toán tử lai ghép trong thuật toán N-EA được xây dựng dựa trên toán tử lai ghép hai điểm cắt. Tuy nhiên, toán tử lai ghép vẫn có thể tạo ra cá thể con không hợp lệ do các gốc của các cụm không được nối với

nhau. Khi có một cá thể con không hợp lệ, toán tử lai ghép sẽ loại bỏ cá thể đó.

3.2.6. Toán tử đột biến

Ý tưởng chính của toán tử đột biến trong thuật toán N-EA là thay đổi gốc của một cụm trên nhiễm sắc thể.

3.2.7. Cách đánh giá cá thể mới

Chi phí của lời giải T được tính và biến đổi như sau:

$$f(T) = \sum_{u \in V} d_T(s,u) \tag{3.1}$$

$$= \sum_{i=1}^{k} \left(|C_i| * d_T(s,r_i) + \sum_{u \in C_i} d_T(r_i,u) \right) \tag{3.2}$$

Do số chiều của đồ thị con trong mỗi cụm nhỏ hơn số chiều của đồ thị đầu vào G, nên độ phức tạp khi tính toán khi tính chi phí lời giải bài toán CluSPT bằng công thức (3.2) sẽ nhỏ hơn khi tính bằng công thức (3.1).

3.3. Kết luận chương

Do thuật toán C-EA mã hóa cây khung bằng mã Cayley và sử dụng toán tử tiến hóa để tạo cây khung cho cả đồ thị trong mỗi cụm và đồ thị các cạnh nối giữa các cụm nên chất lượng lời giải của thuật toán C-EA vẫn còn hạn chế. Thuật toán C-EA có những ưu điểm như: các toán tử tiến hóa có ý tưởng rõ ràng và dễ cài đặt; các toán tử tiến hóa có thể áp dụng cho các bài toán có cấu trúc lời giải tương tự khác.

Do thuật toán N-EA sử dụng thuật toán Dijkstra để tìm cây khung trong mỗi cụm nên thuật toán EA được sử dụng để tối ưu cạnh nối giữa các cụm. Bên cạnh đó, thuật toán N-EA còn biến đổi công thức tính hàm mục tiêu của lời giải bài toán CluSPT giúp giảm chi phí tính toán khi cài đặt thực nghiệm. Tuy nhiên, thuật toán N-EA vẫn còn hạn chế như: Khi đồ thị đầu vào là đồ thị thưa, thuật toán tốn nhiều tài nguyên để tìm và kiểm tra đồ thị tương ứng với một cá thể có liên thông hay không? Trong lời giải tìm được bởi thuật toán N-EA mỗi cụm chỉ nối với cụm khác thông qua một đỉnh. Các toán tử tiến hóa sử dụng trong thuật toán N-EA chỉ sử được để giải bài toán CluSPT.

Thuật toán trình bày trong chương này được công bố trong công trình [I] và [IV].

CHƯƠNG 4: THUẬT TOÁN TIẾN HÓA ĐA NHÂN TỐ GIẢI BÀI TOÁN CÂY PHÂN CỤM VỚI ĐƯỜNG ĐI NGẮN NHẤT

Thuật toán MFEA được đề xuất (ký hiệu là G-MFEA) gồm có hai tác vụ: tác vụ thứ nhất xác định lời giải hợp lệ của bài toán CluSPT; tác vụ thứ hai sẽ tìm lời giải tốt nhất dựa trên tối ưu các cạnh nối giữa các cụm của mỗi lời giải bài toán CluSPT tìm được ở tác vụ thứ nhất.

4.1. Ý tưởng đề xuất thuật toán G-MFEA

Thuật toán N-EA vẫn có một số hạn chế như:

- Do một cụm được nối với các cụm khác thông qua chỉ một đỉnh nên trong một số trường hợp, lời giải tìm được không tốt.
- Trong trường hợp tồn tại nhiều hơn một cạnh nối giữa hai cụm, thuật toán N-EA không tiến hành đánh giá các cạnh này để chọn ra cạnh tốt nhất.
- Phương thức tạo ngẫu nhiên cá thể và toán tử lai ghép đòi hỏi một số lượng lớn tài nguyên để tìm đồ thị liên thông.

Để khắc phục hạn chế của thuật toán N-EA, thuật toán G-MFEA sử dụng mã hóa lời giải dựa trên biểu diễn cạnh. Điểm nổi bật trong cách mã hóa trong thuật toán G-MFEA là một cụm có thể nối với các cụm khác thông qua nhiều cạnh và nhiều đỉnh khác nhau. Cách mã hóa này có thể giúp cải thiện chất lượng lời giải được tạo ra, đặc biệt khi đồ thị đầu vào là đồ thị không đầy đủ.

4.2. Lược đồ của thuật toán G-MFEA

Thuật toán G-MFEA có các đặc trưng sau:

- Mỗi cá thể trong không gian USS là một lời giải của bài toán H-Problem.
- Thuật toán G-MFEA có hai tác vụ, đầu ra của cả hai tác vụ là lời giải bài toán CluSPT. Các lời giải này được xây dựng từ cá thể trong không gian USS thông qua hai thuật toán khác nhau.

4.3. Biểu diễn cá thể

Một cá thể trong không gian USS lưu các thông tin: thông tin về các cạnh nối giữa các cụm (lưu vào thuộc tính ES); thông tin về các

Thuật toán 4.1: Lược đồ thuật toán G-MFEA

Input: Đồ thị phân cụm $G = (V, E, C)$;
Đỉnh nguồn s;
Output: Lời giải của bài toán CluSPT;

begin
  $t \leftarrow 0$;
  $P_t \leftarrow$ Tạo ngẫu nhiên N cá thể của bài toán H-Problem;
  foreach *cá thể* $ind_j \in P_t$ **do**
    Gán ngẫu nhiên chỉ số kỹ năng phù hợp nhất τ_j cho ind_j;
    Tạo lời giải s_j của bài toán CluSPT dựa trên ind_j và τ_j;
    Tính chi phí đối với mỗi tác vụ của cá thể ind_j dựa trên chi phí của lời giải s_j;
    Tính xếp hạng đối với mỗi tác vụ, giá trị thích nghi vô hướng của cá thể ind_j;
  while *điều kiện dừng chưa thỏa mãn* **do**
    $P'_t \leftarrow$ Tournament Selection(P_t);
    $O_t \leftarrow$ Thực hiện lai ghép và đột biến (P'_t) ;
    foreach *cá thể* $c_j \in O_t$ **do**
      Tạo lời giải s'_j của bài toán CluSPT dựa trên c_j và chỉ số kỹ năng phù hợp nhất của c_j;
      Đánh giá chi phí đối với mỗi tác vụ của c_j dựa trên chi phí của lời giải s'_j;
    $R_t \leftarrow O_t \cup P_t$;
    Cập nhật xếp hạng đối với mỗi tác vụ, giá trị thích nghi vô hướng và chỉ số kỹ năng phù hợp nhất của các cá thể trong tập R_t;
    $P_{t+1} \leftarrow$ Chọn N cá thể tốt nhất trong tập R_t;
    $t \leftarrow t + 1$;
  return Lời giải bài toán CluSPT tốt hơn trong hai tác vụ;

đỉnh có cạnh nối ra các đỉnh thuộc các cụm khác (lưu vào thuộc tính IE) và thông tin đỉnh gốc của mỗi cụm (lưu vào thuộc tính LR).

4.4. Phương pháp khởi tạo cá thể

Cá thể được khởi tạo ngẫu nhiên thông qua ba bước chính như sau:

1. Tạo đồ thị $G - Graph$ G' = (V', E') từ đồ thị đầu vào G = (V, E).
2. Áp dụng thuật toán tạo cây khung ngẫu nhiên [69] cho đồ thị G'.
3. Với mỗi cạnh (C_i, C_j) của đồ thị G', tìm một cạnh ngẫu nhiên trên đồ thị G nối một đỉnh thuộc cụm C_i với một thuộc cụm C_j.

4.5. Toán tử lai ghép

Các cá thể con được sinh ra sẽ kế thừa các thông tin đó từ các cá thể cha mẹ theo các quy tắc sau:

- Nếu một cạnh trong cá thể con xuất hiện ở cả hai cá thể cha mẹ thì hai thuộc tính IE, LR của cá thể con sẽ được chọn ngẫu nhiên từ một trong hai cá thể cha mẹ.
- Nếu một cạnh của cá thể con có trên một cá thể cha mẹ thì thuộc tính IE, LR của các thể con được kế thừa từ cá thể cha mẹ đó.

4.6. Toán tử đột biến

Cá thể thực hiện đột biến thông qua hai bước: bước đầu tiên sẽ tạo một cây khung mới bằng cách thêm ngẫu nhiên một cạnh vào cá thể để tạo thành chu trình, sau đó xóa ngẫu nhiên một cạnh từ chu trình (cạnh xóa phải khác cạnh vừa thêm) để tạo thành một cây khung mới. Bước thứ hai sẽ cập nhập thông tin các thuộc tính IE và LR của cá thể mới.

4.7. Phương pháp giải mã

Phương pháp giải mã gồm các bước:

- Đối với tác vụ thứ nhất, lời giải bài toán CluSPT được xây dựng bằng cách sử dụng các thuộc tính LR và IE.
- Đối với tác vụ thứ hai, thuật toán HB-RGA được sử dụng để tìm cạnh tốt nhất nối giữa các đỉnh của hai cụm khác nhau.

4.8. Cách tác vụ thứ hai cải thiện chất lượng lời giải

Do thuật toán HB-RGA sử dụng chiến lược tham lam và vét cạn nên cạnh nối giữa các cụm do thuật toán HB-RGA tìm được thường tốt hơn cạnh được xây dựng bởi phương thức giải mã trong tác vụ thứ nhất.

4.9. Kết luận chương

Khác với các nghiên cứu trước đây về thuật toán MFEA, thuật toán G-MFEA chỉ có một bài toán đầu vào, từ bài toán đó thuật toán G-MFEA sẽ phân rã thành hai bài toán con để giải bằng hai tác vụ. Trong thuật toán G-MFEA, thuật toán HB-RGA đóng vai trò tương tự như thuật toán tìm kiếm cục bộ, giúp cải thiện chất lượng lời giải tìm được bằng tác vụ sử dụng thuật toán EA. Tuy nhiên, khác với thuật toán tìm kiếm cục bộ, quá trình trao đổi vật chất di truyền giữa hai tác vụ được liên tục thực hiện thông qua quá trình truyền lại đặc tính theo chiều dọc và cơ chế ghép đôi cùng loại, cũng như các toán tử tiến hóa trong thuật toán G-MFEA. Thuật toán G-MFEA có đặc điểm:

Điểm mạnh:

- Thuật toán G-MFEA sử dụng tác vụ thứ hai đóng vai trò tương tự như thuật toán tìm kiếm cục bộ nên chất lượng lời giải tìm được gần hơn với kết quả tối ưu.
- Thuật toán khắc phục được các hạn chế của các thuật toán khác.

Hạn chế:

- Mã hóa cá thể trong thuật toán G-MFEA cần lưu trữ nhiều thông tin.
- Do sử dụng kết hợp giữa hai thuật toán MFEA và HB-RGA nên khó để cài đặt thực nghiệm thuật toán G-MFEA.

Thuật toán được trình bày trong chương này được công bố trong công trình [V].

CHƯƠNG 5: KẾT QUẢ THỰC NGHIỆM

5.1. Dữ liệu thực nghiệm và đánh giá lời giải

5.1.1. Đồ thị metric

Thông tin về các bộ dữ liệu được cập nhật tại [74].

5.1.2. Đồ thị đầy đủ phi metric

Thông tin về các bộ dữ liệu đầy đủ phi metric được cập nhật tại [74]

5.1.3. Tiêu chí đánh giá

Chất lượng của một thuật toán được đánh giá qua chất lượng lời giải và thời gian tính.

5.1.4. Môi trường, tham số thực nghiệm

Luận án tiến hành hai nhóm thực nghiệm chính:

- Phân tích hiệu quả của các thuật toán mà luận án đề xuất với thuật toán đã được nghiên cứu trước đây trên hai phương diện chất lượng lời giải tìm được và thời gian thực hiện.
- Phân tích ảnh hưởng của một số tham số: số đỉnh của đồ thị đầu vào, số cụm của đồ thị đầu vào,... tới hiệu quả của các thuật toán đề xuất, cũng như tới kết quả so sánh giữa các thuật toán.

Với mỗi bộ dữ liệu, các thuật toán được thực nghiệm 30 lần trên máy tính: CPU - Intel Core i7 (4790M), RAM - 16GB. Thuật toán HB-RGA: $\gamma = 50$; các thuật toán EA và MFEA: số lần đánh giá là 5000 lần, kích thước quần thể $P = 100$, $p_c = 0.5$, $p_m = 0.05$, $rmp = 0.5$.

5.2. Kết quả thực nghiệm

Luận án tiến hành phân tích kết quả thực nghiệm theo ba hình thức:

- **Phân tích thống kê:** luận án sử dụng phân tích thống kê để phân tích hiệu quả của các đề xuất.
- **Phân tích chi tiết:** so sánh chi tiết kết quả tìm được của các thuật toán theo từng bộ dữ liệu thuộc các thuật toán khác nhau.
- **Phân tích nhân tố ảnh hưởng:** phân tích sự ảnh hưởng của các đặc trưng của dữ liệu đầu vào tới hiệu năng của các thuật toán.

5.2.1. Đồ thị metric

a) Phân tích thống kê

Luận án sử dụng thống kê phi tham số (*Non-parametric statistic*) để phân tích kết quả trong hai bước chính:

- Bước 1 sử dụng kiểm định Friedman, Aligned Friedman, Quade để kiểm tra sự khác biệt giữa kết quả của các thuật toán có ý nghĩa thống kê hay không.
- Bước 2 sử dụng các phân tích thống kê hậu kiểm (*Post-hoc statistical*) để xác định chi tiết sự khác biệt giữa kết quả của các thuật toán, xác định thuật toán tốt nhất (theo chất lượng lời giải).

Kết quả thống kê của các kiểm định Friedman và Iman-Davenport trong bảng 5.3 cho thấy sự khác biệt giữa kết quả các thuật toán có ý nghĩa thống kê với ngưỡng $\alpha = 0.05$. Giá trị xếp hạng trung bình của các thuật toán đánh giá theo kiểm định Friedman, Friednman Aligned

và Quade trong bảng 5.4 cho thấy thuật toán G-MFEA có thứ hạng nhỏ nhất nên thuật toán G-MFEA được chọn làm thuật toán điều khiển (*control algorithm*) trong các phân tích thống kê hậu kiểm.

Giá trị đã hiệu chỉnh của trị số p thu được từ các kiểm định Friedman và Quade trong bảng 5.6 cho thấy G-MFEA tốt hơn ba thuật toán AAL, C-EA và N-EA ở ngưỡng được xem xét $\alpha = 0.05$.

b) So sánh kết quả trong từng tập dữ liệu

Dữ liệu trong bảng 5.7 cho thấy thuật toán G-MFEA có nhiều nhất số lần tìm được lời giải tối ưu (29 lần) và số lần tìm được lời giải tốt nhất (95 lần); thuật toán HB-RGA có số lần tìm được lời giải tốt nhất lớn thứ 2 với 47 lần, tiếp sau đó đến thuật toán N-EA với 2 lần tìm được lời giải tốt nhất; hai thuật toán AAL và C-EA không lần nào có lời giải tốt nhất hoặc tìm được lời giải tối ưu.

Kết quả trong các bảng 5.19 – bảng 5.21 cho thấy lời giải tìm của các thuật toán kém nhất là thuật toán AAL, tiếp đến thuật toán C-EA và N-EA. Tương tự, xếp hạng trung bình của thuật toán N-EA kém hơn đáng kể so với hai thuật toán G-MFEA và HB-RGA.

Giá trị trung bình RPD trên các tập dữ liệu của các thuật toán được trình bày trong bảng 5.8 cho thấy thuật toán AAL tìm được lời giải có chất lượng kém nhất trong năm thuật toán được so sánh. Đối với các giải thuật đề xuất, lời giải nhận được từ thuật toán C-EA thường kém hơn so với ba thuật toán HB-RGA, N-EA và G-MFEA. Giá trị RPD của ba thuật toán HB-RGA, N-EA và G-MFEA không chênh lệch nhiều như đối với hai thuật toán AAL và C-EA.

Chất lượng lời giải tìm được bởi các thuật toán xếp theo thứ tự giảm dần là: G-MFEA $\rightarrow$ HB-RGA $\rightarrow$ N-EA $\rightarrow$ C-EA $\rightarrow$ AAL.

c) Các yếu tố ảnh hưởng tới hiệu quả của thuật toán

Hình 5.3 ta thấy với các bộ dữ liệu có nhiều hơn 25 cụm thì thuật toán HB-RGA tìm được lời giải tốt hơn thuật toán G-MFEA. Hình 5.4 cho thấy thuật toán G-MFEA có hiệu quả cao hơn đối với các bộ dữ liệu có số cụm bé.

Hình 5.5(a) và 5.5(b) cho thấy, với các bộ dữ liệu có số đỉnh nhỏ hơn 99 (đối với Type 1), 300 (đối với Type 5) thuật toán G-MFEA tìm được kết quả tốt hơn thuật toán HB-RGA.

Các phân tích trên cho thấy thuật toán N-EA và HB-RGA không hiệu quả bằng thuật toán G-MFEA khi số cụm hoặc số đỉnh của đồ thị đầu vào nhỏ. Nguyên nhân chính dẫn tới kết quả trên là do thuật toán G-MFEA sử dụng đồng thời cả hai tác vụ để tìm lời giải cho một bộ dữ liệu nên có sự trao đổi thông tin và hỗ trợ lẫn nhau giữa các tác vụ trong quá trình tìm lời giải. Bên cạnh đó, thuật toán G-MFEA kết hợp được các thế mạnh của quá trình khai phá (*exploration*) không gian tìm kiếm lời giải mới của thuật toán EA và khai thác (*exploitation*) không gian lời giải đã có của thuật toán HB-RGA. Với mỗi thế hệ, tác vụ có thuật toán EA giúp tạo ra quần thể mới đa dạng, sau đó lựa chọn các cá thể tốt hơn để tạo thành quần thể cho thế hệ sau. Trong khi, tác vụ có thuật toán HB-RGA đóng vai trò như bước tìm kiếm cục bộ. Thuật toán HB-RGA lựa chọn cá thể từ không gian USS của quần thể đang xét để cải thiện chất lượng lời giải.

d) So sánh thời gian thực hiện

Thuật toán N-EA có thời gian tính toán thấp hơn các thuật toán còn lại, trong khi thuật toán G-MFEA có thời gian tính toán lớn nhất. Thuật toán N-EA có thời gian thực hiện thấp là do cá thể trong thuật toán này chỉ mã hóa số cụm của đồ thị đầu vào nên các toán tử tiến hóa thực hiện trên nhiễm sắc thể có chiều dài bằng số lượng cụm. Thời gian tính toán của thuật toán G-MFEA lớn do các toán tử tiến hóa cần thời gian để xác định thông tin về các cụm, các cạnh nối giữa các cụm và thông tin về các đỉnh gốc cục bộ. Bên cạnh đó, thuật toán G-MFEA sẽ xây dựng đồ thị đầu vào cho thuật toán HB-RGA, sau đó mới áp dụng thuật toán HB-RGA để tìm lời giải của bài toán CluSPT nên chi phí tính toán cũng sẽ tăng thêm.

Thời gian tính toán của ba thuật toán N-EA, HB-RGA và C-EA khác nhau không nhiều.

5.2.2. Đồ thị đầy đủ phi metric

Do thuật toán SLA-M không áp dụng được với đồ thị phi metric nên trong phần này, luận án chỉ đánh giá năm thuật toán AAL, C-EA, N-EA, HB-RGA và G-MFEA.

a) Phân tích thống kê

Các giá trị Friedman và Iman-Davenport trong bảng 5.10 cho thấy sự khác biệt giữa kết quả các thuật toán trên tập dữ liệu đầy đủ phi

metric có ý nghĩa thống kê với ngưỡng $\alpha = 0.05$. Trong bảng 5.11, thuật toán G-MFEA có thứ hạng nhỏ nhất nên thuật toán G-MFEA có hiệu năng tốt nhất.

Ước lượng kết quả so sánh đối kháng giữa các thuật toán trong bảng 5.12 cho thấy thuật toán G-MFEA có hiệu quả tốt nhất. Thuật toán AAL kém hiệu quả nhất. Thuật toán HB-RGA tốt hơn hai thuật toán N-EA và C-EA; thuật toán N-EA cho kết quả tốt hơn thuật toán C-EA.

b) So sánh kết quả trong từng tập dữ liệu

Khác với kết quả trên bộ dữ liệu metric, hiệu quả của các thuật toán có sự khác biệt rõ ràng hơn trên bộ dữ liệu đầy đủ phi metric.

Kết quả trong bảng 5.14 cũng cho thấy:

- Thuật toán AAL kém hơn các thuật toán khác trên tất cả các bộ dữ liệu đã thực nghiệm.
- Thuật toán C-EA kém ba thuật toán N-EA, HB-RGA và G-MFEA trên tất cả các bộ dữ liệu.
- Số bộ dữ liệu mà thuật toán N-EA kém hơn thuật toán HB-RGA và thuật toán G-MFEA là 20/23 và 18/23 (Type 1), 18/18 (Type 5), 30/34 và 28/34 (Type 6).
- Số bộ dữ liệu mà thuật toán HB-RGA kém hơn thuật toán G-MFEA là 17/23 (Type 1), 15/18 (Type 5) và 22/34 (Type 6).

Thứ tự chất lượng lời giải tìm được bởi các thuật toán là: G-MFEA $\rightarrow$ HB-RGA $\rightarrow$ N-EA $\rightarrow$ C-EA $\rightarrow$ AAL.

c) Các yếu tố ảnh hưởng tới hiệu quả của thuật toán

Biểu đồ phân tán theo số cụm của đồ thị trong hình 5.7(a) cho thấy khi số cụm nhỏ hơn 10 thì thuật toán G-MFEA tốt hơn cả hai thuật toán HB-RGA và N-EA; giá trị nhỏ nhất của số cụm để thuật toán G-MFEA kém hơn một trong hai thuật toán N-EA và HB-RGA là 10. Trong hình 5.7(b) cho thấy trên tập dữ liệu Type 5, không có trường hợp nào thuật toán G-MFEA có kết quả kém hơn thuật toán N-EA. Hình 5.7(c) chỉ ra rằng khi số cụm nhỏ hơn 9 thuật toán G-MFEA tốt hơn cả hai thuật toán HB-RGA và N-EA; thuật toán HB-RGA (N-EA) có kết quả tốt hơn thuật toán G-MFEA chỉ khi số cụm lớn hơn hoặc bằng 9 (20).

5.3. Kết luận chương

Các phân tích được tiến hành dựa trên kết quả thực nghiệm của các thuật toán thu được từ 215 bộ dữ liệu thuộc hai dạng đồ thị: metric và đầy đủ phi metric.

Các phân tích kết quả thực nghiệm cũng chỉ ra rằng thuật toán G-MFEA tìm được lời giải tốt nhất, trong khi thuật toán C-EA tìm được lời giải kém nhất. Lời giải tìm được bởi thuật toán HB-RGA và N-EA có chất lượng gần nhau, tuy nhiên thuật toán HB-RGA có xu hướng tốt hơn thuật toán N-EA.

Luận án cũng đã chỉ ra sự ảnh hưởng của số cụm, số đỉnh của đồ thị đầu vào tới kết quả của các thuật toán. Từ đó có thể một phần dự đoán kết quả nhận được của các thuật toán theo hai thuộc tính trên của đồ thị đầu vào.

KẾT LUẬN

Các đóng góp mới

Đối với một bài toán NP-khó như bài toán CluSPT, hiện tại có ba hướng tiếp cận chính để phát triển thuật toán giải: 1) hướng tiếp cận đúng, 2) hướng tiếp cận heuristic, 3) hướng tiếp cận meta-heuristic. Đóng góp của luận án là đề xuất các thuật toán giải bài toán CluSPT theo cả ba hướng tiếp cận:

- Đối với hướng tiếp cận đúng, luận án đề xuất thuật toán đúng dựa trên phương pháp duyệt đồ thị theo chiều rộng và xây dựng đồ thị có dạng hình sao. Kết quả thực nghiệm cho thấy, thuật toán đề xuất giải được bài toán CluSPT nhanh với đồ thị khoảng 500 đỉnh và 50 cụm.
- Đối với cách tiếp cận heuristic, thuật toán HB-RGA dựa trên thuật toán tham lam ngẫu nhiên được đề xuất để giải bài toán CluSPT. Thuật toán HB-RGA có ưu điểm về thời gian thực hiện nhanh, không phụ thuộc nhiều vào số lượng cụm của đồ thị đầu vào và chất lượng lời giải tìm được có tính "ổn định" cao.
- Đối với hướng tiếp cận meta-heuristic, luận án đề xuất ba thuật toán dựa trên thuật toán tiến hóa (thuật toán C-EA và thuật toán N-EA) và tiến hóa đa nhân tố (thuật toán G-MFEA).
  - Thuật toán C-EA sử dụng mã hóa Cayley để mã hóa lời giải nên có ưu điểm về thời gian thực hiện, về cài đặt đơn giản

và do có thể sử dụng các toán tử tiến hóa đã có nên không cần xây dựng thêm các toán tử tiến hóa đặc trưng của bài toán CluSPT. Hơn nữa, do sử dụng các toán tử tiến hóa không phải thiết kế riêng cho bài toán CluSPT nên thuật toán có thể áp dụng để giải các bài toán cây khung phân cụm (*clustered spanning tree*).

- Thuật toán N-EA dựa trên sự kết hợp giữa thuật toán EA và thuật toán Dijkstra. Do sử dụng thuật toán Dijkstra để tìm cây khung nhỏ nhất trong các cụm và sử dụng thuật toán EA để tối ưu hóa cạnh nối giữa các cụm nên lời giải tìm được bằng thuật toán N-EA được cải thiện nhiều so với thuật toán C-EA.
- Thuật toán G-MFEA dựa trên sự kết hợp giữa thuật toán MFEA và thuật toán HB-RGA. Điểm đóng góp chính của thuật toán G-MFEA so với các nghiên cứu đã có là đề xuất cơ chế giúp phân rã bài toán đầu vào thành hai bài toán riêng biệt cho hai tác vụ thực hiện. Do một tác vụ giúp khai phá không gian lời mới, còn một tác vụ giúp khai thác không gian lời giải đã có nên lời giải tìm được bằng thuật toán G-MFEA tiệm cận với lời giải tối ưu.

Kết quả thực nghiệm trên nhiều bộ dữ liệu cho thấy rằng các đề xuất trong luận án cho chất lượng lời giải tốt hơn thuật toán đã được công bố trước đó.

Hạn chế: Thuật toán G-MFEA có độ phức tạp về bộ nhớ sử dụng còn cao.

Hướng phát triển

Trong những nghiên cứu tiếp theo, luận án sẽ tiếp tục mở rộng nghiên cứu về các vấn đề:

- Đề xuất các toán tử tiến hóa để áp dụng thuật toán MFEA-II giải bài toán CluSPT.
- Nghiên cứu bài toán CluSPT với trọng số các cạnh của đồ thị đầu vào thay đổi theo thời gian.

DANH MỤC CÔNG TRÌNH CÔNG BỐ

I. Dinh, Thanh Pham, Binh Huynh Thi Thanh, Trung Tran Ba, and Long Nguyen Binh. "Multifactorial evolutionary algorithm for solving clustered tree problems: competition among Cayley codes." Memetic Computing 12, no. 3 (2020): pp. 185-217. *(Q1, IF: 3.860)*

II. P. D. Thanh, D. A. Dung, T. N. Tien and H. T. T. Binh, "An Effective Representation Scheme in Multifactorial Evolutionary Algorithm for Solving Cluster Shortest-Path Tree Problem," 2018 IEEE Congress on Evolutionary Computation (CEC), Rio de Janeiro, 2018, pp. 1-8.

III. P. D. Thanh, H. Thi Thanh Binh, D. D. Dac, N. Binh Long and L. M. Hai Phong, "A Heuristic Based on Randomized Greedy Algorithms for the Clustered Shortest-Path Tree Problem," 2019 IEEE Congress on Evolutionary Computation (CEC), Wellington, New Zealand, 2019, pp. 2915-2922.

IV. Huynh Thi Thanh Binh, Pham Dinh Thanh, and Ta Bao Thang. "New approach to solving the clustered shortest-path tree problem based on reducing the search space of evolutionary algorithm". Knowledge-Based Systems, 180:12–25, 2019. *(Q1, IF: 5.921)*

V. Thanh, P.D., Binh, H.T.T. and Trung, T.B. "An efficient strategy for using multifactorial optimization to solve the clustered shortest path tree problem." Applied Intelligence 50, 1233–1258 (2020). *(Q2, IF: 3.325)*

VI. Hanh, Phan Thi Hong, Pham Dinh Thanh, and Huynh Thi Thanh Binh. "Evolutionary algorithm and multifactorial evolutionary algorithm on clustered shortest-path tree problem." Information Sciences 553 (2021): 280-304. *(Q1, IF: 5.91)*

VII. Huynh, Thi Thanh Binh, Dinh Thanh Pham, Ba Trung Tran, Cong Thanh Le, Minh Hai Phong Le, Ananthram Swami, and Thu Lam Bui. "A multifactorial optimization paradigm for linkage tree genetic algorithm." Information Sciences 540 (2020): 325-344. *(Q1, IF: 5.91)*