BỘ GIÁO DỤC ĐÀO TẠO 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

LUẬN ÁN TIẾN SĨ TOÁN HỌC

HÀ NỘI - NĂM 2021

BỘ GIÁO DỤC ĐÀO TẠO 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

LUẬN ÁN TIẾN SĨ TOÁN HỌC

NGƯỜI HƯỚNG DẪN KHOA HỌC:

PGS. TS. Huỳnh Thị Thanh Bình

HÀ NỘI - NĂM 2021

LỜI CAM ĐOAN

Nghiên cứu sinh cam đoan luận án là công trình nghiên cứu của chính mình dưới sự hướng dẫn của PGS.TS. Huỳnh Thị Thanh Bình. Luận án có sử dụng thông tin trích dẫn từ nhiều nguồn tham khảo khác nhau và các thông tin trích dẫn được ghi rõ nguồn gốc. Các số liệu, kết quả trong luận án là trung thực và chưa từng được công bố trong các công trình nghiên cứu của bất kỳ tác giả nào khác.

GIẢNG VIÊN HƯỚNG DẪN	NGHIÊN CỨU SINH
PGS.TS. Huỳnh Thị Thanh Bình	*Phạm Đình Thành*

LỜI CÁM ƠN

Lời đầu tiên, nghiên cứu sinh xin gửi lời cảm ơn chân thành và sâu sắc tới giáo viên hướng dẫn PGS. TS. Huỳnh Thị Thanh Bình đã tận tình dạy bảo và cung cấp những gợi ý quý báu giúp tôi nâng cao kiến thức và hoàn thành tốt luận án này. Nghiên cứu sinh cũng xin bày tỏ lòng biết ơn sâu sắc tới PGS.TS. Bùi Thu Lâm, Học viện Kỹ thuật Quân sự đã nhiệt tình hỗ trợ và đưa ra những định hướng, những lời khuyên trong suốt quá trình tôi thực hiện luận án.

Nghiên cứu sinh xin bày tỏ lòng biết ơn tới Ban giám đốc Học viện Kỹ thuật Quân sự, Ban chủ nhiệm và đặc biệt các thầy cô đang công tác tại khoa Công nghệ thông tin đã hết lòng truyền đạt kiến thức và tạo điều kiện thuận lợi nhất để tôi hoàn thành chương trình học tập và thực hiện luận án nghiên cứu của mình.

Nghiên cứu sinh cũng xin trân trọng cảm ơn Ban Giám hiệu trường Đại học Tây Bắc, Ban chủ nhiệm và các đồng nghiệp tại khoa Khoa học tự nhiên - Công nghệ, trường Đại học Tây Bắc đã tạo điều kiện và giúp đỡ nghiên cứu sinh trong thời gian học tập.

Tôi xin trân trọng cảm ơn các thầy cô trong và ngoài trường đã tham gia đọc và nhận xét luận án ở các cấp Bộ môn, cấp Cơ sở, cấp phản biện độc lập, cấp Trường, đã cho tôi những ý kiến quý báu để tôi hoàn thiện luận án này.

Tôi cũng xin gửi lời cảm ơn tới các thành viên của phòng thí nghiệm Mô hình hóa, mô phỏng và tối ưu hóa(Modelling, Simulation and Optimization lab – MSO Lab), trường Đại học Bách khoa Hà Nội, những người đã luôn nhiệt tình giúp đỡ tôi trong suốt quá trình học tập và nghiên cứu.

Cuối cùng, nghiên cứu sinh chân thành bày tỏ lòng cám ơn tới gia đình và bạn bè đã kiên trì, chia sẻ, động viên nghiên cứu sinh trong suốt quá trình học tập và hoàn thành luận án này.

Hà Nội, ngày tháng năm 2021

NGHIÊN CỨU SINH

Phạm Đình Thành

MỤC LỤC

DANH MỤC THUẬT NGỮ VÀ TỪ VIẾT TẮT **v**

DANH MỤC BẢNG BIỂU **viii**

DANH MỤC HÌNH ẢNH **x**

GIỚI THIỆU **1**

Chương 1 TỔNG QUAN **7**

1.1 Thuật toán tiến hóa 7

1.1.1 Tổng quan về thuật toán tiến hóa 7

1.1.2 Mã hóa lời giải trong thuật toán tiến hóa 8

1.1.3 Khởi tạo quần thể 8

1.1.4 Chọn lọc cá thể cha mẹ 8

1.1.5 Toán tử lai ghép 9

1.1.6 Toán tử đột biến 10

1.1.7 Chọn lọc cá thể cho thế hệ tiếp theo 10

1.1.8 Điều kiện dừng của thuật toán 11

1.2 Thuật toán tiến hóa đa nhân tố 11

1.2.1 Bài toán tiến hóa đa nhân tố 11

1.2.2 Cơ bản về giải thuật tiến hóa đa nhân tố 13

1.2.3 Mã hóa cá thể 14

1.2.4 Các toán tử lai ghép và đột biến 15

1.2.5 Cơ chế đánh giá có chọn lọc 16

1.3 Bài toán cây phân cụm đường đi ngắn nhất 17

1.3.1 Một số định nghĩa 17

1.3.2 Phát biểu bài toán 21

1.3.3 Ứng dụng của bài toán 24

1.3.4 Tổng quan tình hình nghiên cứu 24

1.4 Kết luận chương 30

Chương 2 THUẬT TOÁN XẤP XỈ GIẢI BÀI TOÁN CÂY PHÂN CỤM VỚI ĐƯỜNG ĐI NGẮN NHẤT **31**

2.1 Thuật toán xây dựng cây khung hình sao 31
2.1.1 Lược đồ thuật toán 31
2.1.2 Độ phức tạp của thuật toán 33
2.1.3 Thuật toán dạng hình sao trên đồ thị metric 34
2.2 Thuật toán tham lam ngẫu nhiên 36
2.2.1 Các bước của thuật toán HB-RGA 37
2.2.2 Thuật toán tham lam ngẫu nhiên tìm cạnh nối giữa các cụm 39
2.2.3 Độ phức tạp của thuật toán 41
2.3 Đánh giá thuật toán 43
2.4 Kết luận chương 43
Chương 3 THUẬT TOÁN TIẾN HÓA GIẢI BÀI TOÁN CÂY PHÂN CỤM VỚI ĐƯỜNG ĐI NGẮN NHẤT 45
3.1 Thuật toán tiến hóa dựa trên mã Cayley 45
3.1.1 Lược đồ thuật toán 45
3.1.2 Mã hóa cá thể 46
3.1.3 Phương pháp khởi tạo cá thể 49
3.1.4 Toán tử lai ghép 49
3.1.5 Toán tử đột biến 51
3.2 Hướng tiếp cận dựa trên giảm không gian tìm kiếm của thuật toán tiến hóa 52
3.2.1 Cách tiếp cận 52
3.2.2 Phương pháp phân rã bài toán CluSPT 52
3.2.3 Biểu diễn cá thể 55
3.2.4 Phương pháp khởi tạo cá thể 58
3.2.5 Toán tử lai ghép 60
3.2.6 Toán tử đột biến 61
3.2.7 Cách đánh giá cá thể mới 62
3.3 Đánh giá thuật toán 63

3.4 Kết luận chương 64

Chương 4 THUẬT TOÁN TIẾN HÓA ĐA NHÂN TỐ GIẢI BÀI TOÁN CÂY PHÂN CỤM VỚI ĐƯỜNG ĐI NGẮN NHẤT 65

4.1 Ý tưởng đề xuất thuật toán G-MFEA 66
4.2 Lược đồ của thuật toán G-MFEA 68
4.3 Biểu diễn cá thể 70
4.4 Phương pháp khởi tạo cá thể 72
4.5 Toán tử lai ghép 74
4.6 Toán tử đột biến 77
4.7 Phương pháp giải mã 78
4.8 Cách tác vụ thứ hai cải thiện chất lượng lời giải 80
4.9 Đánh giá thuật toán 83
4.10 Kết luận chương 84

Chương 5 KẾT QUẢ THỰC NGHIỆM 86

5.1 Dữ liệu thực nghiệm và đánh giá lời giải 86
5.1.1 Đồ thị metric 86
5.1.2 Đồ thị đầy đủ phi metric 87
5.1.3 Tiêu chí đánh giá 88
5.1.4 Môi trường, tham số thực nghiệm 89
5.2 Kết quả thực nghiệm 89
5.2.1 Đồ thị metric 90
5.2.2 Đồ thị đầy đủ phi metric 104
5.3 Kết luận chương 110

DANH MỤC CÔNG TRÌNH CÔNG BỐ 116

TÀI LIỆU THAM KHẢO 117

PHỤ LỤC 129

DANH MỤC THUẬT NGỮ VÀ TỪ VIẾT TẮT

Các thuật ngữ

Tiếng Anh	**Tiếng Việt**
Age based selection	Chọn lọc dựa theo tuổi
Assortative mating	Cơ chế ghép đôi cùng loại
Chromosome	Nhiễm sắc thể
Clustered spanning tree	Cây khung phân cụm
Control algorithm	Thuật toán điều khiển
Cost	Chi phí
Critical value	Giá trị tới hạn
Decompose	Phân rã
Exploitation	Khai thác
Exploration	Khai phá
Factorial rank	Xếp hạng đối với mỗi tác vụ
Factorial cost	Chi phí đối với mỗi tác vụ
Fitness	Giá trị thích nghi
Fitness based selection	Chọn lọc dựa theo giá trị thích nghi
Genotype	Kiểu gen
Global tree	Cây khung toàn cục
Implicit genetic transfer	Trao đổi vật chất di truyền tiềm ẩn
Individual	Cá thể
Inter-cluster edge	Cạnh liên cụm
Locality	Cục bộ
Local root	Gốc cục bộ
Local tree	Cây khung bộ phận
Metric graph	Đồ thị metric
Non-parametric statistic	Thống kê phi tham số
Phenotype	Kiểu hình
Population	Quần thể
Post-hoc statistical	Thống kê hậu kiểm
Random mating probability	Xác suất ghép cặp ngẫu nhiên
Random selection	Chọn lọc ngẫu nhiên
Rank selection	Chọn lọc dựa theo thứ hạng trong quần thể
Scalar fitness	Giá trị thích nghi vô hướng

Tiếng Anh	Tiếng Việt
Selective evaluation	Cơ chế đánh giá có chọn lọc
Skill factor	Chỉ số kỹ năng phù hợp nhất
Sparse graph	Đồ thị thưa
Termination condition	Điều kiện dừng
Tournament selection	Chọn lọc cạnh tranh
Vertical cultural transmission	Truyền lại đặc tính theo chiều dọc

Các từ viết tắt

Từ viết tắt	Ý nghĩa
AAL	Thuật toán xấp xỉ AAL
CluSPT-Lib	CluSPT Library
C-MFEA	Thuật toán tiến hóa đa nhân tố C-MFEA
C-EA	Thuật toán tiến hóa sử dụng mã Cayley C-EA
CluSteinerTP	Bài toán cây Steiner phân cụm
CluTSP	Bài toán người đi du lịch phân cụm
CluSPT	Bài toán cây phân cụm đường đi ngắn nhất
CESA	Xây dựng tập cạnh của lời giải
EM	Tiến hóa đa nhiệm
EA	Thuật toán tiến hóa
GTSP	Bài toán người du lịch tổng quát
GMSTP	Bài toán cây khung nhỏ nhất tổng quát
GA	Thuật toán di truyền
G-MFEA	Thuật toán tiến hóa đa nhân tố G-MFEA
HB-RGA	Thuật toán xấp xỉ dựa trên thuật toán tham lam ngẫu nhiên
M4CRST	Thuật toán tạo cây khung ngẫu nhiên
MCST	Tìm cây khung có chi phí nhỏ nhất
InterCluMRCT	Bài toán cây khung phân cụm với chi phí định tuyến liên cụm nhỏ nhất
CluMRCT	Bài toán cây khung phân cụm có chi phí định tuyến nhỏ nhất
MSTP	Bài toán cây khung nhỏ nhất
MFO	Tối ưu hóa đa nhân tố
MOO	Tối ưu hóa đa mục tiêu

Từ viết tắt	**Ý nghĩa**
MFEA	Thuật toán tiến hóa đa nhân tố
NCX	Toán tử lai ghép trong thuật toán N-EA
N-EA	Thuật toán tiến hóa N-EA
NMO	Toán tử đột biến mới
PSO	Tối ưu bầy đàn
RGA	Thuật toán tham lam ngẫu nhiên
RPD	Tỉ lệ phần trăm chênh lệch tương đối
SPTA	Thuật toán cây đường đi ngắn nhất
SOO	Tối ưu hóa đơn mục tiêu
SLA	Thuật toán dạng hình sao SLA
SLA-M	Thuật toán chính xác SLA-M
STP	Bài toán cây Steiner
TSP	Bài toán người đi du lịch
USS	Không gian tìm kiếm chung

DANH MỤC BẢNG BIỂU

5.1 Tóm tắt thông tin về các bộ dữ liệu đồ thị metric 87
5.2 Các tiêu chí đánh giá thuật toán 88
5.3 Bảng kết quả phân tích kiểm định Friedman và Iman-Davenport (α=0.05) 91
5.4 Xếp hạng của các thuật toán theo phương thức Friedman, Friednman Aligned và Quade 91
5.5 Các trị số z và p của các kiểm định Friedman, Quade (G-MFEA là thuật toán điều khiển (*control algorithm*)) 92
5.6 Bảng giá trị đã hiệu chỉnh của trị số p của các kiểm định Friedman và Quade (G-MFEA là thuật toán điều khiển) . . 92
5.7 Bảng tóm tắt số lần tìm được lời giải tốt nhất và số lần tìm được lời giải tối ưu của các thuật toán 93
5.8 Bảng giá trị trung bình RPD của các thuật toán trên các tập dữ liệu 94
5.9 Bảng tóm tắt so sánh kết quả dựa trên số bộ dữ liệu của các tập dữ liệu 96
5.10 Bảng kết quả kiểm định Friedman và Iman-Davenport (α=0.05) 105
5.11 Giá trị xếp hạng trung bình của các thuật toán nhận được từ được đánh giá bởi các kiểm định Friedman, Friednman Aligned và Quade 105
5.12 Ước lượng đối kháng giữa các thuật toán 106
5.13 Bảng giá trị đã hiệu chỉnh của trị số p của các kiểm định Friedman và Quade (G-MFEA là thuật toán điều khiển) . . 106

5.14 Bảng tóm tắt so sánh kết quả dựa trên số bộ dữ liệu phi metric 108
15 Thông tin về các bộ dữ liệu đồ thị metric thuộc Type 3 và Type 4 129
16 Thông tin về các bộ dữ liệu đồ thị metric thuộc Type 1 . . . 130
17 Thông tin về các bộ dữ liệu đồ thị metric thuộc Type 5 . . . 131
18 Thông tin về các bộ dữ liệu đồ thị metric thuộc Type 6 . . . 132
19 Kết quả thực nghiệm của các thuật toán trên bộ dữ liệu đồ thị metric thuộc Type 1 133
20 Kết quả thực nghiệm của các thuật toán trên bộ dữ liệu đồ thị metric thuộc Type 5 134
21 Kết quả thực nghiệm của các thuật toán trên bộ dữ liệu đồ thị metric thuộc Type 6 136
22 Kết quả thực nghiệm của các thuật toán trên bộ dữ liệu đồ thị metric thuộc Type 3 và Type 4 138
23 Thời gian tính trung bình của các thuật toán trên bộ dữ liệu đồ thị metric thuộc Type 1 140
24 Thời gian tính trung bình của các thuật toán trên bộ dữ liệu đồ thị metric thuộc Type 5 141
25 Thời gian tính trung bình của các thuật toán trên bộ dữ liệu đồ thị metric thuộc Type 6 142
26 Thời gian tính trung bình của các thuật toán trên bộ dữ liệu đồ thị metric thuộc Type 3 and Type 4 143
27 Kết quả thực nghiệm của các thuật toán trên tập đồ thị đầy đủ phi metric thuộc Type 1 144
28 Kết quả thực nghiệm của các thuật toán trên tập đồ thị đầy đủ phi metric thuộc Type 5 145
29 Kết quả thực nghiệm của các thuật toán trên tập đồ thị đầy đủ phi metric thuộc Type 6 146

DANH MỤC HÌNH ẢNH

1.1 Minh họa các định nghĩa về đồ thị 19
1.2 Cây khung phân cụm của bài toán CluSPT cho đồ thị gồm 4 cụm và 15 đỉnh 22
1.3 Minh họa về lời giải không hợp lệ của bài toán CluSPT . . . 22
1.4 Ứng dụng của bài toán CluSPT trong nông nghiệp 24
1.5 Ứng dụng của bài toán CluSPT trong tối ưu hệ thống cấp điện và thông tin liên lạc 25
1.6 Ví dụ minh họa các bước của thuật toán AAL 29
1.7 Ví dụ minh họa hạn chế của thuật toán AAL 29

2.1 Ví dụ về lời giải của bài toán CluSPT trên đồ thị metric . . . 36
2.2 Ví dụ minh họa các bước của thuật toán HB-RGA 42

3.1 Ví dụ về biểu diễn cá thể trong thuật toán C-EA 48
3.2 Ví dụ về toán tử lai ghép trong thuật toán C-EA 50
3.3 Ví dụ về toán tử đột biến trong thuật toán C-EA 52
3.4 Ví dụ về cách phân rã bài toán CluSPT thành các bài toán bé hơn 53
3.5 Ví dụ về lời giải hợp lệ và không hợp lệ đối với cách tiếp cận mới 55
3.6 Ví dụ minh họa biểu diễn cá thể trong thuật toán N-EA . . . 58
3.7 Ví dụ về phương pháp khởi tạo cá thể với đồ thị đầu vào gồm 4 cụm 59
3.8 Ví dụ về toán tử lai ghép sử dụng trong thuật toán N-EA . . 61

3.9 Ví dụ minh họa toán tử đột biến sử dụng trong thuật toán N-EA 62

4.1 Ví dụ về các hạn chế của thuật toán N-EA 67
4.2 Ví dụ về hạn chế của biểu diễn lời giải trong thuật toán N-EA 68
4.3 Ví dụ về mã hóa lời giải trong thuật toán G-MFEA 71
4.4 Ví dụ minh họa các bước của toán tử lai ghép trong thuật toán G-MFEA 76
4.5 Ví dụ minh họa các bước của toán tử đột biến 79
4.6 Ví dụ minh họa về phương pháp giải mã trong thuật toán G-MFEA 81
4.7 Ví dụ minh họa cách cải thiện chất lượng lời giải của tác vụ thứ hai 83

5.1 Tỉ lệ phần trăm số lời giải tối ưu và số lời giải tốt nhất tìm được của các thuật toán (NBS: số lời giải tốt nhất tìm được; NOS: số lời giải tối ưu tìm được) 94
5.2 Giá trị trung bình RPD của các thuật toán trên các tập dữ liệu 95
5.3 Biểu đồ phân tán của kết quả so sánh hai thuật toán G-MFEA và HB-RGA với số cụm của đồ thị. 98
5.4 Biểu đồ phân tán của kết quả so sánh hai thuật toán G-MFEA và N-EA với số cụm của đồ thị. 100
5.5 Biểu đồ phân tán của kết quả so sánh hai thuật toán G-MFEA và HB-RGA với số đỉnh của đồ thị. 101
5.6 Biểu đồ phân tán của kết quả so sánh hai thuật toán G-MFEA và N-EA với số đỉnh của đồ thị. 103
5.7 Biểu đồ phân tán của kết quả so sánh các thuật toán và số cụm của đồ thị đối với các bộ dữ liệu đầy đủ phi metric. . . . 112
5.8 Biểu đồ phân tán của kết quả so sánh các thuật toán và số đỉnh của đồ thị đối với các bộ dữ liệu đầy đủ phi metric. . . . 113

GIỚI THIỆU

Bài toán tìm cây khung có chi phí nhỏ nhất (*Minimal-Cost Spanning Tree - MCST*) trên đồ thị có trọng số là một trong các bài toán nổi tiếng trong lĩnh vực tối ưu rời rạc cũng như trong khoa học máy tính. Bài toán MCST được ứng dụng trong nhiều lĩnh vực thực tế như tối ưu hệ thống truyền thông, tối ưu hệ thống giao vận, v.v. Trong nghiên cứu, đã có rất nhiều biến thể của bài toán MCST (khi thay đổi hàm mục tiêu hoặc thêm các rằng buộc) được nghiên cứu như: bài toán cây khung nhỏ nhất (minimum spanning trees) [46, 71], cây Steiner nhỏ nhất (Steiner minimum trees) [28, 82], cây đường đi ngắn nhất (shortest-path trees) [24, 46] và cây khung với chi phí định tuyến nhỏ nhất (minimum routing cost spanning trees) [44, 55].

Tuy nhiên, 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 khung phân cụm đường đi ngắn nhất (Clustered Shortest-path Tree Problem) [22] 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 bài toán cây khung phân cụm đường đi ngắn nhất là bài toán thuộc

lớp NP-Khó [21, 22] 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) [47], cho tới các thuật toán tiến hóa [6–8], hay các thuật toán lấy ý tưởng từ tối ưu trong tự nhiên [9, 91]. 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 [92]. Trong các thuật toán lấy ý tưởng từ quá trình tối ưu hóa trong tự nhiên, thuật toán tiến hóa (evolutionary algorithm) 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 [18, 54, 95].

Các thuật toán tiến hóa sử dụng các khái niệm trong sinh học và áp dụng vào trong khoa học máy tính. Các thuật toán tiến hóa được đánh giá là phù hợp để giải nhiều bài toán phức tạp hoặc phi tuyến ngay cả đối với nhiều bài toán được đánh giá là khó giải khi sử dụng một số thuật toán tối ưu trước đó.

Tối ưu đa nhân tố (multi-factorial optimization) là một mô hình mới của tiến hóa đa nhiệm vụ (evolutionary multi-tasking) [37, 50, 79]. Điểm khác biệt dễ nhận thấy giữa tối ưu đa nhân tố và thuật toán tiến hóa cơ bản (classical evolutionary algorithm) đó là thuật toán tiến hóa 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) là nghiên cứu đầu tiên về kết hợp giữa tối ưu đa nhân tố với thuật toán di truyền [37, 58]. Do thuật toán tiến hóa đa nhân tố 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 tiến hóa đa nhân tố 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 tiến hóa và thuật toán tiến hóa đa nhân tố 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 tiến hóa đa nhân tố 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 cây phân cụm đường đi ngắn nhất (*Clustered Shortest-Path Tree Problem - CluSPT*), trong đó luận án tập trung vào hai hướng: sử dụng thuật toán tiến hóa (chương 3) và thuật toán tiến hóa đa nhân tố (chương 4).

Các mục tiêu cụ thể trong luận án như sau:

1. **Xây dựng các toán tử tiến hóa:** mặc dù đã có nhiều nghiên cứu về các toán tử tiến hóa để giải các bài toán có lời giải là cây khung, song nếu áp dụng các toán tử tiến hóa đã có vào bài toán cây khung phân cụm, các toán tử này có thể tạo ra lời giải không hợp lệ. Vì vậy, việc xây dựng các thuật toán tiến hóa (*Evolutionary Algorithm - EA*) để giải bài toán cây khung phân cụm là cần thiết, trong đó luận án chú trọng xây dựng toán tử tiến hóa hiệu quả giải bài toán CluSPT.
2. **Xây dựng toán tử tiến hóa đặc trưng của thuật toán tiến hóa đa nhân tố (*Multi-Factorial Evolutionary Algorithm - MFEA*):** khác với EA cơ bản chỉ tìm lời giải của một bài toán, thuật toán MFEA tìm lời giải đồng thời của nhiều bài toán nên để áp dụng thuật toán MFEA, trước tiên cần xây dựng toán tử mã hóa để biểu

diễn lời giải của các bài toán vào một biểu diễn chung. Khi đánh giá cá thể trong mỗi tác vụ, toán tử giải mã cũng cần được đề xuất, để tìm lời giải của từng bài toán từ cá thể trong không gian tìm kiếm chung (*Unified Search Space - USS*).

3. **Kết hợp thuật toán tiến hóa đa nhân tố với các thuật toán xấp xỉ:** nghiên cứu cơ chế kết hợp giữa thuật toán tiến hóa đa nhân tố với một trong các thuật toán xấp xỉ như: Thuật toán tham lam ngẫu nhiên (*Randomized Greedy Algorithm - RGA*), tối ưu bầy đàn (*Particle Swarm Optimization - PSO*),... dựa trên cơ sở lựa chọn các ưu điểm của từng thuật toán thành phần, để cải thiện chất lượng lời giải của bài toán CluSPT.
4. **Xây dựng thuật toán xấp xỉ:** luận án hướng tới xây dựng 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 các phương pháp đánh giá thực nghiệm một cách khách quan, khái quát được hầu hết các trường hợp của bài toán.

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:

- 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 tiến hóa đa nhân tố và áp dụng giải các bài toán tối ưu tổ hợp.
- Phương pháp xây dựng các bộ dữ liệu, cách đánh giá và tiến hành thực nghiệm để so sánh chính xác hiệu quả của các thuật toán.

Các đóng góp của luận án

Các kết quả nghiên cứu chính của luận án đã được công bố ở các tạp chí và hội nghị chuyên ngành. Các kết quả chính đạt được của luận án có ý nghĩa khoa học bao gồm cả hai khía cạnh lý thuyết và ứng dụng.

- **Về lý thuyết:**
  - Đề xuất thuật toán chính xác SLA-M giải bài toán CluSPT trên đồ thị metric (*metric graph*).
  - Đề xuất thuật toán tham lam ngẫu nhiên kết hợp với ý tưởng của thuật toán Dijkstra để giải bài toán CluSPT. Thuật toán đề xuất tìm được lời giải trong thời gian ngắn và chất lượng lời giải tốt hơn thuật toán đã có trước đó.
  - Đề xuất hai thuật toán tiến hóa C-EA và N-EA để giải bài toán CluSPT. Đối với mỗi thuật toán, luận án đề xuất mới toán tử lai ghép, toán tử đột biến và phương pháp mã hóa lời giải. Các toán tử tiến hóa được đề xuất còn có thể áp dụng cho các bài toán khác trong nhóm cây khung phân cụm. Luận án cũng đề xuất cách cài đặt thực nghiệm tính hàm mục tiêu của bài toán CluSPT để giảm chi phí tính toán.
  - Đề xuất thuật toán tiến hóa đa nhân tố G-MFEA để giải bài toán CluSPT. Thuật toán G-MFEA cho kết quả tốt hơn các thuật toán trong nghiên cứu trước đây, trong đó, một thuật toán G-MFEA tìm được các lời giải tối ưu trên nhiều tập dữ liệu khác nhau.
- **Về mặt ứng dụng:** Do bài toán CluSPT có nhiều ứng dụng trong thực tiễn, nên 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 (tối ưu hóa

mạng cáp quang, tối ưu hóa mạng lưới cáp đồng trục), trong sản xuất (tối ưu mạng lưới vận chuyển hàng hóa từ nhà máy sản xuất tới các kho hàng), v.v.

Cấu trúc của luận án

Luận án gồm phần mở đầu, năm chương và phần kết luận.

Phần mở đầu trình bày ý nghĩa, tổng quan tình hình nghiên cứu thuộc lĩnh vực luận án quan tâm giải quyết, mục đích nghiên cứu của luận án, phương pháp nghiên cứu, phạm vi nghiên cứu và các đóng góp của luận án.

Chương 1 trình bày hai vấn đề: vấn đề thứ nhất trình bày kiến thức cơ bản về các thuật toán tiến hóa; vấn đề thứ hai trình bày về bài toán CluSPT: các khái niệm liên quan về đồ thị, phát biểu bài toán, ứng dụng của bài toán.

Chương 2 trình bày đề xuất về 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 đề xuất.

Chương 1
TỔNG QUAN

Chương này sẽ trình bày lý thuyết cơ bản về các thuật toán tiến hóa bao gồm các khái niệm cơ bản, các toán tử tiến hóa và nguyên tắc cơ bản của thuật toán tiến hóa (*Evolutionary Algorithm - EA*) và thuật toán tiến hóa đa nhân tố (*Multi-Factorial Evolutionary Algorithm - MFEA*). Bên cạnh đó, các khái niệm cơ bản về lý thuyết đồ thị, phát biểu bài toán cây phân cụm đường đi ngắn nhất (*Clustered Shortest-Path Tree Problem - CluSPT*) cũng như các nghiên cứu liên quan và ứng dụng của bài toán CluSPT cũng được trình bày trong chương này.

1.1. Thuật toán tiến hóa

1.1.1. Tổng quan về thuật toán tiến hóa

Thuật toán EA được xây dựng bằng cách mô phỏng và sử dụng các khái niệm quen thuộc trong sinh học và trong tiến hóa như: Quần thể (*population*) tiến hóa bao gồm các cá thể (*individual*) - đại diện cho những lời giải hợp lệ đối với bài toán [4, 12]. Nhiễm sắc thể (*chromosome*) hay bộ gen (genome) bao gồm nhiều gen (genes). Mỗi gen này thể hiện một đặc trưng của cá thể, đó có thể là một đặc trưng về kiểu gen (*genotype*) hoặc một đặc trưng về kiểu hình (*phenotype*).

Thuật toán EA bao gồm rất nhiều mô hình khác nhau như: thuật toán di truyền, lập trình di truyền, lập trình tiến hóa, chiến lược tiến hóa... [20]. Với ý tưởng chủ đạo là sử dụng một hoặc nhiều tác động trên một quần thể có sẵn để biến đổi quần thể, nâng cao khả năng thích nghi của quần thể [3, 4]. Ý tưởng này được thể hiện cụ thể trong từng mô hình, theo những cách đặc trưng khác nhau của từng kĩ thuật đó.

Thuật toán di truyền (*Genetic Algorithm - GA*) là một mô hình của thuật toán tiến hóa, sử dụng các toán tử di truyền như lai ghép, đột biến, chọn lọc, v.v. để biến đổi quần thể ban đầu. Thuật toán di truyền được giới thiệu lần đầu vào năm 1975 bởi John Holland [41] và là mô hình đầu tiên của thuật toán tiến hóa được xây dựng và sử dụng [3, 4, 81].

Thuật toán 1.1: Thuật toán di truyền

Input: Đầu vào của bài toán tối ưu;
Output: Lời giải tốt nhất tìm được;

```
1  begin
2      Khởi tạo quần thể ban đầu P;
3      Đánh giá cá thể;
4      while chưa đạt điều kiện dừng do
5          Chọn các cặp cha mẹ từ P;
6          Áp dụng toán tử lại ghép cho cặp cá thể cha mẹ và tạo ra tập cá thể con O;
7          Áp dụng toán tử đột biến lên O;
8          Chọn lọc các cá thể cho thế hệ sau từ tập O ∪ P để tạo thành quần thể P mới;
9      end
10     return Cá thể tốt nhất trong quần thể;
11 end
```

1.1.2. Mã hóa lời giải trong thuật toán tiến hóa

Trong thuật toán GA, mỗi cá thể sẽ tương đương với một lời giải của bài toán. Do có nhiều phương pháp mã hóa lời giải khác nhau và mỗi phương pháp mã hóa đều có những điểm mạnh và điểm yếu riêng nên lựa chọn cách mã hóa nào sẽ tùy thuộc vào từng bài toán cụ thể.

Do phương pháp mã hóa lời giải khác nhau sẽ cần các toán tử tiến hóa khác nhau nên khi lựa chọn phương pháp mã hóa lời giải cũng cần xem xét tới các toán tử lai ghép, đột biến hoặc phép chọn lọc sẽ sử dụng.

1.1.3. Khởi tạo quần thể

Quần thể trong thuật toán GA là tập hợp những cá thể - lời giải hợp lệ hoặc khả thi đối với bài toán đang được xem xét. Việc khởi tạo quần thể cùng với việc chọn lọc cá thể cho thế hệ tiếp theo đóng vai trò quan trọng trong việc đảm bảo các cá thể trong không gian tìm kiếm cũng như đầu vào của các toán tử tiến hóa là hợp lệ.

Hiện nay, có nhiều phương pháp khởi tạo quần thể cho một thuật toán GA. Việc lựa chọn phương pháp khởi tạo nào sẽ tùy thuộc vào bài toán đang được giải và cách biểu diễn (mã hóa) lời giải.

1.1.4. Chọn lọc cá thể cha mẹ

Một số phương pháp chọn lọc cá thể cha mẹ để tiến hành lai ghép:

a) Chọn lọc dựa theo giá trị thích nghi (*fitness based selection*)

Chọn lọc cá thể cha mẹ dựa theo giá trị thích nghi thường gồm các phương pháp sau: lựa chọn xén (truncation selection), lựa chọn theo vòng quay roulette (roulette wheel selection), lựa chọn theo kiểu rải (stochastic universal sampling),v.v.

b) Chọn lọc ngẫu nhiên (*random selection*)

Các cá thể cha mẹ được chọn lọc hoàn toàn ngẫu nhiên theo số lượng hoặc tỉ lệ xác định sẵn. Trong một số bài toán, có những cá thể cha mẹ không phải là tốt nhất trong quần thể nhưng khi lai ghép với nhau lại có thể cho ra những cá thể con rất tốt.

c) Chọn lọc dựa theo thứ hạng trong quần thể (*rank selection*)

Đôi khi việc chọn lọc cá thể cha mẹ dựa vào độ thích nghi gặp khó khăn khi mà độ thích nghi của các cá thể chênh lệch không nhiều, dẫn đến xác suất chúng được lựa chọn gần như là bằng nhau. Khi đó, hiệu quả của phép lựa chọn không khác gì phép lựa chọn ngẫu nhiên. Để tránh tình huống đó, độ thích nghi của cá thể thường không được xét đến mà thay đó việc chọn lọc sẽ dựa vào thứ hạng của cá thể trong quần thể.

Thứ hạng thường được tính dựa trên giá trị thích nghi đối với môi trường. Những cá thể có thứ hạng cao thì được ưu tiên chọn lọc để đưa vào sinh sản tạo thế hệ tiếp theo.

d) Chọn lọc cạnh tranh (*tournament selection*)

Đây cũng là một phép lựa chọn được sử dụng nhiều trong thuật toán tiến hóa. Trong cách chọn này, số nguyên $k > 0$ được xác định làm tham số cho phép chọn lọc giao đấu. Ở mỗi lượt chọn, k cá thể được chọn ngẫu nhiên từ quần thể đang xét. Dựa vào độ thích nghi, cá thể tốt nhất trong k cá thể này được chọn làm cá thể cha mẹ.

1.1.5. Toán tử lai ghép

Trong tự nhiên, con người và các loài động vật khi sinh ra đã mang những đặc điểm thừa hưởng từ cả bố và mẹ. Những đặc điểm này là do

gen của con được sao chép một phần từ bố và phần còn lại từ mẹ. Tương tự như vậy, phép lai ghép trong thuật toán di truyền cho phép tạo ra những cá thể con từ một phần của cá thể cha ghép với một phần của cá thể mẹ. Cách lựa chọn phần gen nào để ghép và ghép như thế nào được phụ thuộc vào từng bài toán, từng phương pháp tiếp cận.

1.1.6. Toán tử đột biến

Mặc dù lai ghép trong thuật toán tiến hóa cho phép "trộn" các đặc điểm di truyền của các cá thể có sẵn, từ đó tạo ra một số cá thể thừa hưởng được đặc điểm tốt từ cá thể cha và mẹ. Tuy nhiên, phép lai ghép không đem lại sự đa dạng và nguồn gen mới cho quần thể.

Trong khi đó, đột biến giải quyết được yêu cầu này. Đột biến làm thay đổi ngẫu nhiên một phần nhỏ của cá thể, đảm bảo cho việc duy trì và phát triển tính đa dạng của nguồn nguyên liệu gen. Xác suất đột biến thường được đặt giá trị nhỏ, vì với một xác suất đột biến lớn thì thuật toán tiến hóa mất đi ưu thế và trở thành tìm kiếm ngẫu nhiên.

1.1.7. Chọn lọc cá thể cho thế hệ tiếp theo

Chọn lọc cá thể cho thế hệ tiếp theo (hay còn gọi là chọn lọc tồn tại - Survivor selection) là bước quyết định cá thể nào sẽ bị loại bỏ, cá thể nào sẽ được giữ lại cho quần thể của thế hệ tiếp theo. Dễ dàng có thể nhận thấy rằng những cá thể có độ thích nghi cao cần được giữ lại vì những cá thể này truyền lại những vật chất di truyền tốt cho thế hệ sau. Tuy nhiên, việc chọn lọc cá thể không nên chỉ dừng lại ở chọn tất cả các cá thể tốt nhất, mà cần xem xét đến việc duy trì và phát triển tính đa dạng trong nguồn nguyên liệu gen của quần thể. Có hai phương pháp chọn lọc phổ biến là dựa vào tuổi và dựa vào độ thích nghi.

a) Chọn lọc dựa vào tuổi

Chọn lọc dựa theo tuổi (*age based selection*) không xem xét đến độ thích nghi của cá thể mà thay vào đó là tuổi hay thời gian tồn tại của cá thể. Khi cá thể đã trải qua số thế hệ tối đa được quy định, chúng sẽ bị loại bỏ khỏi quần thể cho dù độ thích nghi cao hay thấp.

b) Chọn lọc dựa vào giá trị thích nghi

Khác với chọn lọc dựa vào tuổi cá thể, chọn lọc dựa theo giá trị thích nghi không quan tâm tới thời gian tồn tại của cá thể trong quần thể mà chỉ hoàn toàn dựa vào giá trị thích nghi. Do đó, những cá thể có giá trị thích nghi cao có thể được giữ lại trong quần thể qua rất nhiều thế hệ tiến hóa.

Các phương pháp chọn lọc dựa vào giá trị thích nghi như: chọn lọc xén, chọn lọc theo vòng quay roulette, chọn lọc theo kiểu rải hay chọn lọc giao đấu.

1.1.8. Điều kiện dừng của thuật toán

Điều kiện dừng (*termination condition*) của bài toán được lựa chọn với mục đích xác định thời điểm lời giải gần nhất với lời giải đúng được tìm ra và dung hòa được yếu tố chi phí thời gian. Để tìm được thời điểm đó, thông thường các điều kiện sau có thể được chọn làm điều kiện dừng: khi chất lượng lời giải không tiếp tục cải thiện, sau một số thế hệ nhất định, khi hàm mục tiêu hoặc độ thích nghi của cá thể đạt ngưỡng nhất định, v.v.

1.2. Thuật toán tiến hóa đa nhân tố

1.2.1. Bài toán tiến hóa đa nhân tố

Trong nhiều năm qua, các thuật toán tiến hóa đã được áp dụng hiệu quả để tìm lời giải của nhiều dạng bài toán thuộc nhiều lĩnh vực khác nhau trong khoa học, kỹ thuật và vận trù học. Về cơ bản, các bài toán này có thể được phân vào hai nhóm [37, 38]:

- Tối ưu hóa đơn mục tiêu (*Single-objective Optimization - SOO*) [17, 34]: mỗi điểm trong không gian tìm kiếm sẽ tương ứng một giá trị vô hướng của hàm mục tiêu.
- Tối ưu hóa đa mục tiêu (*Multi-Objective Optimization - MOO*) [88, 90]: mỗi điểm trong không gian tìm kiếm sẽ ánh xạ tới một véc tơ các giá trị (vô hướng) của các hàm mục tiêu.

Tuy nhiên, với sự phát triển mạnh mẽ của lĩnh vực điện toán đám mây trong thời gian gần đây, các thuật toán tiến hóa hiện có tỏ ra không hiệu quả khi được áp dụng vào giải các bài toán thuộc lĩnh vực này, do hầu hết các thuật toán tiến hóa được thiết kế để giải một bài toán trong một lần thực hiện, trong khi các bài toán thuộc lĩnh vực điện toán đám mây thường cần phải được giải đồng thời [17, 99].

Để giải quyết vấn đề trên, tác giả Abhisek Gupta và các cộng sự [37] đã giới thiệu lớp bài toán mới, được gọi là tối ưu hóa đa nhân tố (*Multi-Factorial Optimization - MFO*). Đặc trưng quan trọng nhất xuất hiện trong MFO nhưng chưa từng có trong các lĩnh vực tính toán tiến hóa khác đó là bài toán MFO cung cấp một "vùng" để thực hiện gián tiếp quá trình song song việc tìm kiếm lời giải dựa trên quần thể [36].

Do MFO giải đồng thời hai hoặc nhiều bài toán tối ưu nên tác giả A. Gupta và các cộng sự nhận định rằng, các bài toán tối ưu khác nhau có nhưng điểm tương đồng nhất định, và việc giải bài toán này cùng lúc có thể giúp bổ trợ cho giải bài toán khác [94]. Từ đó, các tác giả đề xuất mô hình MFO có khả năng tương hỗ giữa các tác vụ trong quá trình thực hiện việc tìm kiếm tập lời giải cho cùng lúc nhiều bài toán tối ưu hóa [35, 78].

Định nghĩa 1.1 (Bài toán tối ưu hóa đa nhân tố)**.** *Bài toán tối ưu hóa đa nhân tố là bài toán cùng lúc cực tiểu hóa (hoặc cực đại hóa) K hàm $f_i(x^{(i)}), x^{(i)} \in \mathcal{X}_i \subset \mathbb{R}^n, i \in \{1, 2, \ldots, K\}$, ký hiệu $\min_{x^{(i)} \in \mathbb{X}} f_i(x^{(i)})$. Trong đó:*

- *$x^{(i)} \in \mathbb{R}^n$ là các biến quyết định của bài toán thứ i.*
- *$f_i(x^{(i)}) : \mathcal{X}_i \to \mathbb{R}^n$ là hàm mục tiêu của bài toán thứ i.*
- *$\mathcal{X}_i$ là không gian tìm kiếm của bài toán thứ i.*

Mục tiêu của bài toán là tìm bộ nghiệm tối ưu $(x^{(1)*}, x^{(2)*}, \ldots, x^{(K)*})$ sao cho với mọi lời giải thành phần $x^{(j)*}$ ta có $f_i(x^{(j)}) \leq f_i(x^{(i)})\ \forall x^{(i)} \in \mathcal{X}_i$.

Do MFO là một lược đồ của tiến hóa đa nhiệm (*Evolutionary Multi-tasking - EM*) [39, 98] nên MFO có đặc điểm cơ bản của EM đó là khả năng giải quyết đồng thời hai hoặc nhiều tác vụ tối ưu. Các tác vụ này có

thể giống hay khác nhau ở loại bài toán, số chiều và cách biểu diễn lời giải và có thể phụ thuộc hay không phụ thuộc lẫn nhau. MFO do đó được đặc trưng bởi sự tồn tại đồng thời của nhiều không gian tìm kiếm với số chiều và cách biểu diễn khác nhau, cũng như những hàm mục tiêu khác nhau cho mỗi tác vụ (bài toán) con.

1.2.2. Cơ bản về giải thuật tiến hóa đa nhân tố

Hiện nay có rất nhiều những hướng tiếp cận khác nhau để giải bài toán MFO. Mô hình đầu tiên của tối ưu đa nhân tố đối với tiến hóa đa nhiệm được đề xuất bởi bởi tác giả A. Gupta và các cộng sự [37] (ký hiệu là MFEA). Ý tưởng chính của thuật toán 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 tiến hóa như khởi tạo quần thể, lai ghép, đột biến lên không gian tìm kiếm chung để biến đổi quần thể.
- Đánh giá mỗi lời giải – cá thể trong không gian tìm kiếm chung thông qua những tiêu chí thể hiện chất lượng của lời giải đối với từng tác vụ hoặc vai trò của lời giải đối với quần thể.
- Để thuận tiện cho việc đánh giá trên cũng như để tìm ra lời giải của từng tác vụ sau khi áp dụng thuật toán MFEA, cần có cách chuyển đổi giữa cá thể trong không gian tìm kiếm chung và lời giải cho từng tác vụ.

a) Các tiêu chí đánh giá cá thể

Do dựa trên cơ sở của thuật toán EA nên thuật toán MFEA cũng gồm các bước tương tự như thuật toán EA như: mã hóa, khởi tạo quần thể, lai ghép, đột biến, chọn lọc, nhưng ở một số bước sẽ có thêm một số đặc trưng riêng của thuật toán nhằm thực hiện nhiệm vụ tối ưu hai hoặc nhiều bài toán có tính chất giống hoặc khác nhau cùng một lúc [11, 14, 73, 97]. Thuật toán MFEA giải đồng thời nhiều bài toán bằng cách biểu diễn lời

giải của các bài toán riêng biệt đó về một biểu diễn trong một không gian tìm kiếm chung. Từ đó, MFEA thực hiện các toán tử tiến hóa ở trên biểu diễn chung đó. Giả sử, áp dụng MFEA để giải K bài toán tối ưu một lúc. Để thực hiện việc này, MFEA đưa ra một số tham số đặc trưng ứng với mỗi cá thể, cụ thể như sau:

- Chi phí đối với mỗi tác vụ (*factorial cost*): mức độ hiệu quả của cá thể khi giải mỗi bài toán. Xét tác vụ T_j, chi phí đối với mỗi tác vụ ψ_j^i của cá thể p_i chính là chi phí của cá thể p_i đối với bài toán T_j.
- Xếp hạng đối với mỗi tác vụ (*factorial rank*): thứ hạng dựa trên chi phí đối với tác vụ, khi so sánh với tất cả các cá thể trong quần thể đang xét. Xếp hạng đối với mỗi tác vụ r_j^i của cá thể p_i đối với tác vụ T_j được định nghĩa là thứ tự của cá thể p_i trong quần thể sau khi đã được sắp xếp theo thứ tự tăng hoặc giảm dần của ψ_j.
- Giá trị thích nghi vô hướng (*scalar fitness*): được tính bằng nghịch đảo của xếp hạng tốt nhất (xếp hạng "cao" nhất) trong các xếp hạng của cá thể đó đối với các tác vụ.
- Chỉ số kỹ năng phù hợp nhất (*skill factor*): tác vụ mà cá thể có xếp hạng cao nhất trong tất cả các tác vụ của bài toán.

b) Cấu trúc của thuật toán tiến hóa đa nhân tố

Các bước của thuật toán MFEA được mô tả trong thuật toán 1.2.

Một trong các điểm khác biệt chính của MFEA so với GA cơ bản là cho phép lai ghép các cá thể của các bài toán tối ưu hóa khác nhau phụ thuộc theo giá trị xác suất ghép cặp ngẫu nhiên (*random mating probability*). Bên cạnh đó, phép lai ghép ngẫu nhiên đa nhân tố này đem đến một khả năng chia sẻ các gen tốt giữa các bài toán tối ưu hóa, cũng như cùng lúc giúp tăng tốc độ hội tụ của các bài toán.

1.2.3. Mã hóa cá thể

Không giống như thuật toán GA cơ bản, thuật toán MFEA giải đồng thời nhiều bài toán tối ưu, nên cần phải có một không gian tìm kiếm chung

Thuật toán 1.2: Thuật toán tiến hóa đa nhân tố

Input: Đầu vào của các bài toán tối ưu;
Output: Cá thể tốt nhất của mỗi tác vụ;

```
begin
    Khởi tạo một quần thể ngẫu nhiên P;
    Đánh giá mỗi cá thể đối với từng tác vụ;
    Tính chỉ số kỹ năng phù hợp nhất của mỗi cá thể;
    while chưa đạt điều kiện dừng do
        Áp dụng các toán tử tiến hóa lên quần thể P để tạo ra quần thể con C;
        Đánh giá các cá thể trong C đối với một số tác vụ nhất định;
        Hợp quần thể P và C để tạo thành quần thể trung gian I;
        Cập nhật giá trị thích nghi vô hướng và chỉ số kỹ năng phù hợp nhất của tất
         cả mọi cá thể trong I;
        Lựa chọn những cá thể có giá trị thích nghi vô hướng cao nhất từ quần thể I
         để tạo nên quần thể P mới;
    end
    return Cá thể tốt nhất trong quần thể của mỗi tác vụ;
end
```

để đưa không gian tìm kiếm của từng bài toán về giải trên không gian tìm kiếm chung đó.

Với bài toán tối ưu đa nhân tố gồm K tác vụ và số chiều của tác vụ thứ i là D_i, thuật toán MFEA thường đặt số chiều của lời giải trong không gian tìm kiếm chung là số lớn nhất trong số các số chiều của các bài toán thành phần, nghĩa là $D_{multitask} = \max\{D_i\}_{i=1,...,K}$. Sau khi xác định được số chiều của lời giải, mã hóa cá thể trong thuật toán MFEA còn tùy thuộc vào từng bài toán và các tác vụ trong bài toán đó.

Sau khi mã hóa được cá thể trong không gian tìm kiếm chung, bước khởi tạo quần thể được thực hiện tương tự như thuật toán GA, sau khi khởi tạo được quần thể thì sẽ tiến hành tính chỉ số kỹ năng phù hợp nhất, giá trị thích nghi vô hướng của mỗi cá thể trong quần thể đó.

1.2.4. Các toán tử lai ghép và đột biến

Các toán tử tiến hóa trong thuật toán tiến hóa đa nhân tố sử dụng thêm cơ chế ghép đôi cùng loại (*assortative mating*) trong quá trình tạo ra cá thể con [49]. Cơ chế này phân định rõ trường hợp nào toán tử lai ghép

sẽ xảy ra và trường hợp nào không. Những cặp cá thể cha mẹ đã được chọn ngẫu nhiên sẽ được so sánh về chỉ số kỹ năng phù hợp nhất: nếu hai cá thể cha mẹ có cùng chỉ số kỹ năng phù hợp nhất (có sự tương đồng về văn hóa) thì chúng sẽ được lai ghép với nhau để tạo ra hai cá thể con. Các cặp cha mẹ không có cùng chỉ số kỹ năng phù hợp nhất thì sẽ được lai ghép theo một tỉ lệ % xác định. Tỉ lệ đó được gọi là xác suất ghép cặp ngẫu nhiên. Những cá thể cha mẹ không được lai ghép sẽ đột biến để tạo ra các cá thể con.

Các bước của cơ chế ghép đôi cùng loại được trình bày trong thuật toán 1.3 [37].

Thuật toán 1.3: Cơ chế ghép đôi cùng loại

Input: Hai cá thể cha mẹ p_a, p_b;
Output: Hai cá thể con c_a, c_b;

```
1  begin
2      Tạo một số ngẫu nhiên rand nằm trong khoảng (0, 1);
3      if (cá thể cha mẹ cùng chỉ số kỹ năng phù hợp nhất) or (rand < rmp) then
4          Cá thể cha mẹ p_a và p_b lai ghép tạo ra hai cá thể con c_a và c_b;
5      end
6      else
7          Cá thể cha mẹ p_a đột biến tạo ra cá thể con c_a;
8          Cá thể mẹ p_b đột biến tạo ra cá thể con c_b;
9      end
10     return c_a, c_b;
11 end
```

1.2.5. Cơ chế đánh giá có chọn lọc

Mỗi cá thể trong quần thể MFEA khó có thể đem tới lời giải tốt cho tất các tác vụ MFEA. Do đó, việc đánh giá tất cả các cá thể cho tất cả các tác vụ ở mỗi một thế hệ sẽ lãng phí tài nguyên tính toán. Lý tưởng hơn cả nếu như mỗi cá thể chỉ được đánh giá cho những tác vụ nào mà chúng có khả năng giải quyết tốt nhất. Vì vậy, thuật toán MFEA sử dụng cơ chế đánh giá có chọn lọc (*selective evaluation*), dựa trên hiện tượng truyền lại đặc tính theo chiều dọc (*vertical cultural transmission*) trong sinh học [29, 80]. Cụ thể, một cá thể có thể bị ảnh hưởng trực tiếp bởi

kiểu hình của cha hoặc mẹ, thay vì chỉ thừa hưởng kiểu gen. Trong thuật toán MFEA, nguyên lí này được thể hiện thông qua việc mô phỏng có chọn lọc chỉ số kỹ năng phù hợp nhất của cha hoặc mẹ:

- Đối với các cá thể con được tạo ra từ quá trình lai ghép: một số cá thể sẽ được đánh giá dựa trên chỉ số kỹ năng phù hợp nhất của cá thể bố, số còn lại được đánh giá dựa trên chỉ số kỹ năng phù hợp nhất của cá thể mẹ. Khả năng được đánh giá theo cá thể bố và khả năng được đánh giá theo cá thể mẹ là bằng nhau (tỉ lệ 50:50).
- Đối với các cá thể con được tạo ra từ quá trình lai ghép: một số cá thể sẽ được đánh giá dựa trên chỉ số kỹ năng phù hợp nhất của cá thể bố, số còn lại được đánh giá dựa trên chỉ số kỹ năng phù hợp nhất của cá thể mẹ.
- Đối với các cá thể con được tạo ra từ quá trình đột biến: cá thể con sẽ được đánh giá dựa trên chỉ số kỹ năng phù hợp nhất từ cá thể đã đột biến ra nó.

Các bước của cơ chế đánh giá có chọn lọc được trình bày trong thuật toán 1.4.

1.3. Bài toán cây phân cụm đường đi ngắn nhất

Phần này giới thiệu tổng quan về CluSPT, các ứng dụng của bài toán CluSPT và khảo sát các thuật toán giải bài toán CluSPT.

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. Ký hiệu:

- Cạnh nối giữa đỉnh u và đỉnh v ký hiệu là $e = (u, v)$ và trọng số của cạnh ký hiệu là $w(u, v)$ hoặc $w(e)$.
- $V(G)$ và $E(G)$ ký hiệu là tập đỉnh và tập cạnh của đồ thị G.
- 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.

Thuật toán 1.4: Cơ chế đánh giá cá thể

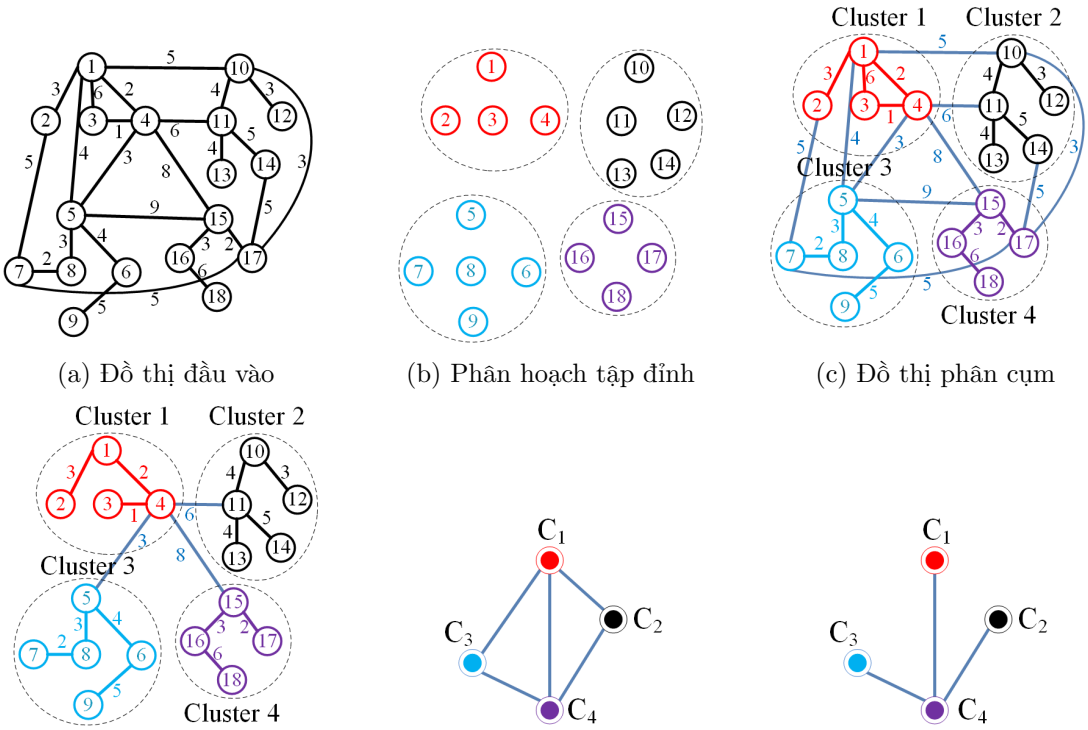
Input: Một cá thể con c được sinh ra từ hai cá thể cha mẹ (p_a và p_b) hoặc chỉ một cá thể cha mẹ (p_a hoặc p_b) sau bước Ghép đôi cùng loại;
Output: Cá thể c sau khi đánh giá

```
1  begin
2      if c được sinh ra từ hai cá thể cha mẹ then
3          Tạo một số ngẫu nhiên rand trong khoảng (0, 1);
4          if rand < 0.5 then
               /* Chỉ đánh giá c theo tác vụ τ_a (chỉ số kỹ năng phù hợp nhất
                  của p_a).                                                  */
5              c mô phỏng theo p_a;
6          end
7          else
               /* Chỉ đánh giá c theo tác vụ τ_b (chỉ số kỹ năng phù hợp nhất
                  của p_b).                                                  */
8              c mô phỏng theo p_b;
9          end
10     end
11     else
           /* Chỉ đánh giá c theo chỉ số kỹ năng phù hợp nhất của cha mẹ  */
12         c mô phỏng theo cá thể cha duy nhất;
13     end
14     Chi phí của tất cả các tác vụ còn lại của c được gán một giá trị rất lớn;
15     return c;
16 end
```

Định nghĩa 1.2 (Phân hoạch của tập đỉnh của đồ thị [53])**.** *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 $C_i \cap C_j = \emptyset, \forall i, j \in [1, h], i \neq j$.*

Hình 1.1(b) minh họa một cách phân hoạch của tập đỉnh của đồ thị trong hình 1.1(a), trong đó tập đỉnh V được phân hoạch thành các tập: $C_1 = \{1, 2, 3, 4\}$, $C_2 = \{10, 11, 12, 13, 14\}$, $C_3 = \{5, 6, 7, 8, 9\}$, $C_4 = \{15, 16, 17, 18\}$.

Định nghĩa 1.3 (Chi phí định tuyến giữa hai đỉnh [22])**.** *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*



(a) Đồ thị đầu vào (b) Phân hoạch tập đỉnh (c) Đồ thị phân cụm

(d) Cây khung phân cụm (e) Đồ thị $G - Graph$ (f) Cây khung toàn cục

Hình 1.1: Minh họa các định nghĩa về đồ thị

đó trên cây khung T.

Định nghĩa 1.4 (Đồ thị phân cụm [84]). *Cho $G = (V,\ E,\ w)$ là đồ thị vô hướng, liên thông, các cạnh có trọng số không âm. Nếu tồn tại tập phân hoạch $C = \{C_1, C_2, \ldots, C_h\}$ của V thì G được gọi là đồ thị phân cụm, tập $C_1, C_2, \ldots, C_h$ được gọi là các cụm (cluster) của đồ thị. Đồ thị phân cụm $G = (V,\ E,\ w)$ với tập phân hoạch C được ký hiệu là $G = (V,\ E,\ w,\ C)$.*

Hình 1.1(c) minh họa đồ thị phân cụm nhận được từ đồ thị trong hình 1.1(a), với phân hoạch $C = \{C_1, C_2, C_3, C_4\}$ của tập đỉnh V là: $C_1 = \{1, 2, 3, 4\}$, $C_2 = \{10, 11, 12, 13, 14\}$, $C_3 = \{5, 6, 7, 8, 9\}$, $C_4 = \{15, 16, 17, 18\}$.

Định nghĩa 1.5 (Cây khung phân cụm [53]). *Cho đồ thị có trọng số cạnh $G = (V,\ E,\ w)$ trong đó các đỉnh được phân hoạch thành h cụm $C = \{C_1, C_2, \ldots, C_h\}$, một cây khung T của G là một cây khung phân cụm, nếu T có thể cắt thành h cây con bằng cách xóa đi $h - 1$ cạnh, sao*

cho mỗi cây con đó là một cây khung của đồ thị con của G được cảm sinh bởi các cụm $C_i, i \in [1, h]$.

Xét đồ thị phân cụm và cây khung lần lượt được minh họa trong các hình 1.1(c), hình 1.1(d). Do sau khi xóa các cạnh $(4, 11), (4, 15)$ và $(4, 5)$, đồ thị nhận được trong mỗi cụm đều là cây khung nên cây khung trong hình 1.1(c) là cây khung phân cụm.

Định nghĩa 1.6 (Đồ thị G-Graph)**.** *Cho* $G = (V,\ E,\ w)$ *là đồ thị phân cụm. G-Graph là đồ thị được suy ra từ đồ thị G trong đó mỗi đỉnh của G-Graph tương ứng với một cụm trong đồ thị G, giữa hai đỉnh của đồ thị G-Graph có cạnh nối khi có ít nhất một cạnh nối giữa các đỉnh của các cụm tương ứng trong đồ thị G.*

Đồ thị $G - Graph$ tương ứng với đồ thị phân cụm trong hình 1.1(c) được minh họa trong hình 1.1(e), trong đó đỉnh $C_i (i = 1, \ldots, 4)$ của đồ thị $G - Graph$ tương ứng cụm i của đồ thị phân cụm. Do không có cạnh nào nối giữa hai đỉnh của cụm 1 và cụm 3 nên trong đồ thị $G - Graph$ không có cạnh nào nối giữa hai đỉnh C_1 và C_3.

Định nghĩa 1.7 (Đỉnh gốc của cây khung)**.** *Cho đồ thị có trọng số cạnh* $G = (V,\ E,\ w)$ *với đỉnh nguồn* $s \in V$ *và tập đỉnh* V *được phân hoạch thành* h *cụm* $C = \{C_1, C_2, \ldots, C_h\}$*. Nếu* T *là một cây khung phân cụm của đồ thị* G *thì* T *còn được gọi là cây khung có đỉnh gốc tại* s*.*

Định nghĩa 1.8 (Cây khung toàn cục)**.** *Cho* $G = (V,\ E,\ w)$ *là đồ thị phân cụm và* G' *là đồ thị G-Graph của đồ thị G. Khi đó, cây khung của đồ thị* G' *được gọi là cây khung toàn cục (*global tree*).*

Hình 1.1(f) minh họa một cây khung toàn cục của đồ thị $G - Graph$ trong hình 1.1(e).

Định nghĩa 1.9 (Cây khung bộ phận [53])**.** *Xét cây khung* T *và một tập con các đỉnh* $V^* \subseteq V(T)$*, cây khung bộ phận (*local tree*) của tập* V^* *trên*

cây khung T là đồ thị con $T[V^*]$. Để dễ theo dõi, trong luận án này, mỗi cây khung bộ phận của cây khung phân cụm T là một cây con của cây T được cảm sinh bởi tập đỉnh trong một cụm.

Định nghĩa 1.10 (Đỉnh gốc của cây khung bộ phận)**.** *Cho $G = (V,\ E,\ w)$ là đồ thị phân cụm và T là cây khung phân cụm với gốc tại đỉnh $r \in V$. Với mỗi cụm $C_t (1 \leq t \leq h)$ bất kỳ, đỉnh $r_t \in C_t$ được gọi là gốc của cây khung bộ phận của cụm C_t của T nếu $\forall u \in C_t$, $u \neq r_t$ thì u là con hoặc cháu của đỉnh r_t trên T.*

Giả sử đỉnh 1 là đỉnh gốc của cây khung trong hình 1.1(d), các đỉnh 11, 5, 15 lần lượt là các đỉnh gốc của các cây khung bộ phận của cụm 2, 3 và 4.

Định nghĩa 1.11 (Cạnh liên cụm [53])**.** *Cho $G = (V,\ E,\ w)$ là đồ thị phân cụm và T là cây khung phân cụm với gốc tại đỉnh r của đồ thị G. Xét hai đỉnh bất kỳ $u, v \in V$ ($u \in C_i, v \in C_j, 1 \leq i, j \leq h$), cạnh $e = (u, v)$ được gọi cạnh liên cụm (*inter-cluster edge*) nếu hai đỉnh u và v thuộc hai cụm khác nhau, hay nói cách khác $i \neq j$. Dễ thấy, nếu đỉnh u là cha mẹ của đỉnh v, thì đỉnh v là gốc của cây khung bộ phận của cụm C_j, khi đó đỉnh u được gọi là cổng (port) của cụm C_j.*

Cạnh liên cụm của đồ thị phân cụm trong hình 1.1(c) gồm: (1, 10), (4, 11), (4, 15), (1, 5), (4,5), v.v.

Định nghĩa 1.12 (Đồ thị metric [52, 83])**.** *Đồ thị metric là một đồ thị đầy đủ với khoảng cách giữa ba đỉnh bất kỳ $x, y, z \in V$ luôn thỏa mãn bất đẳng thức tam giác: $w(x, y) + w(y, z) \geq w(x, z)$*

1.3.2. Phát biểu bài toán

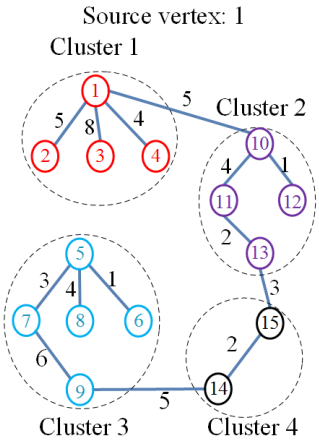
Có nhiều bài toán cây khung phân cụm được quan tâm nghiên cứu trong thời gian gần đây, tuy nhiên, luận án đi sâu nghiên cứu bài toán CluSPT.

Bài toán CluSPT được phát biểu như sau [22]:

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:

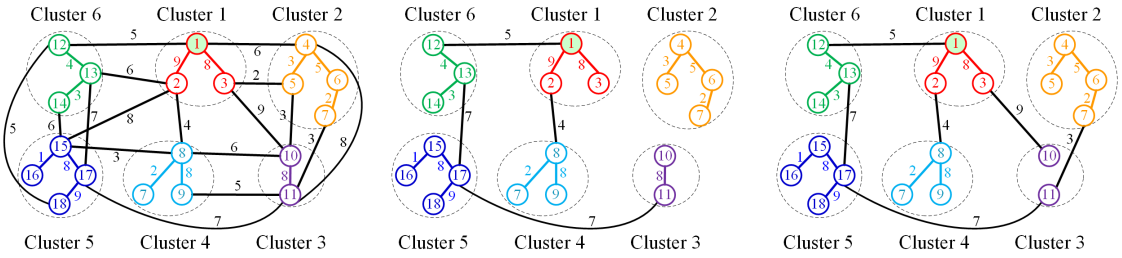
- Với mỗi cụm $C_i (i = 1, \ldots, h)$, đồ thị con $T[C_i]$ là một đồ thị liên thông.
- Tổng chi phí định tuyến giữa đỉnh nguồn s và các đỉnh còn lại trên cây khung T là nhỏ nhất, hay nói cách khác:

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



Hình 1.2: Cây khung phân cụm của bài toán CluSPT cho đồ thị gồm 4 cụm và 15 đỉnh

Hình 1.2 minh họa một cây phân cụm của bài toán CluSPT, trong đó, đồ thị đầu vào gồm 15 đỉnh, 4 cụm và đỉnh 1 là đỉnh nguồn. Chi phí định tuyến của cây khung phân cụm là 186.



(a) (b) (c)

Hình 1.3: Minh họa về lời giải không hợp lệ của bài toán CluSPT

Phát biểu bài toán cho thấy rằng, lời giải s' của bài toán CluSPT là hợp lệ nếu thỏa mãn đồng thời hai rằng buộc: s' là một cây khung và đồ thị con trong mỗi cụm của s' là một đồ thị liên thông. Điều đó có nghĩa rằng, có hai khả năng dẫn tới lời giải s' của bài toán CluSPT là 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.

Hình 1.3 minh họa các trường hợp không hợp lệ của lời giải bài toán CluSPT. Hình 1.3(a) minh họa đồ thị đầu vào (input graph) G với 6 cụm, 18 đỉnh và đỉnh 1 là đỉnh nguồn. Trong hình 1.3(b), do các đỉnh trong cụm thứ 2 không nối với bất kỳ đỉnh nào thuộc cụm khác, nên lời giải trong hình này không phải là lời giải hợp lệ của bài toán CluSPT (do lời giải không phải là cây khung). Trong hình 1.3(c), do đỉnh 10 và đỉnh 11 trong cụm thứ 3 không được nối với nhau, nên vi phạm rằng buộc về tính liên thông của đồ thị con trong mỗi cụm. Do đó, lời giải trong hình này cũng là lời giải không hợp lệ.

Không mất tính tổng quát, giả sử rằng:

- Đỉnh nguồn $s \in C_1$.
- Đỉnh nguồn s là gốc của cây khung phân cụm (*clustered spanning tree*) T của đồ thị đầu vào G. Cụm chứa đỉnh s gọi là cụm gốc.
- Các cây $T_1, T_2, \ldots, T_h$ là cây khung bộ phận tương ứng với các cụm $C_1, C_2, \ldots, C_h$ trên cây khung phân cụm T.
- Các đỉnh $r_1, r_2, \ldots, r_h$ lần lượt là các gốc của các cây khung bộ phận $T_1, T_2, \ldots, T_h$.

Mệnh đề 1.3.1. *Nếu T là lời giải tối ưu của bài toán CluSPT thì các cây $T_1, T_2, \ldots, T_h$ là các cây đường đi ngắn nhất với gốc tại các đỉnh $r_1, r_2, \ldots, r_h$ [86].*

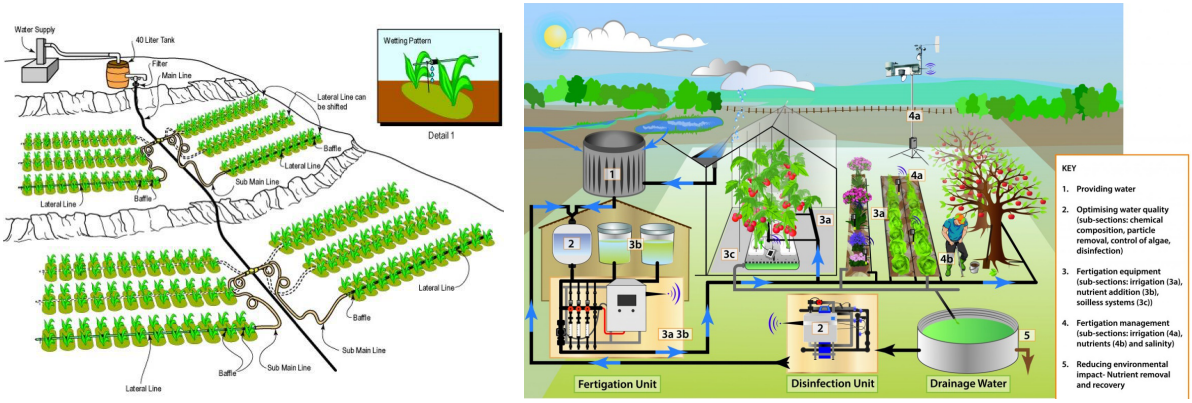
Mệnh đề (1.3.1) cho thấy rằng, tương ứng với mỗi tập gồm tất cả các

cạnh liên cụm, ta có thể tìm được các cạnh còn lại của lời giải T bài toán CluSPT sao cho giá trị hàm mục tiêu $f(T)$ là nhỏ nhất.

1.3.3. Ứng dụng của bài toán

Bài toán CluSPT xuất hiện từ rất sớm trong nhiều ứng dụng thực tế thuộc các lĩnh vực như: trong nông nghiệp [27], trong viễn thông [33, 70], trong quân sự [30], trong phân phối hàng hóa và dịch vụ [57],v.v.

Một trong các ứng dụng của bài toán CluSPT xuất hiện sớm nhất là ứng dụng trong lĩnh vực nông nghiệp. Ngay từ rất sớm, con người đã có nhu cầu cần tối ưu hệ thống dẫn nước từ nguồn nước tới các địa điểm canh tác. Đối với hệ thống dẫn nước trong sa mạc, yêu cầu hạn chế lượng nước bị mất trong quá trình dẫn nước được đặt nên hàng đầu. Hình 1.4 minh họa hệ thống dẫn nước tưới từ nguồn nước tới các cánh đồng, trong đó các cánh đồng đóng vai trò như các cụm.



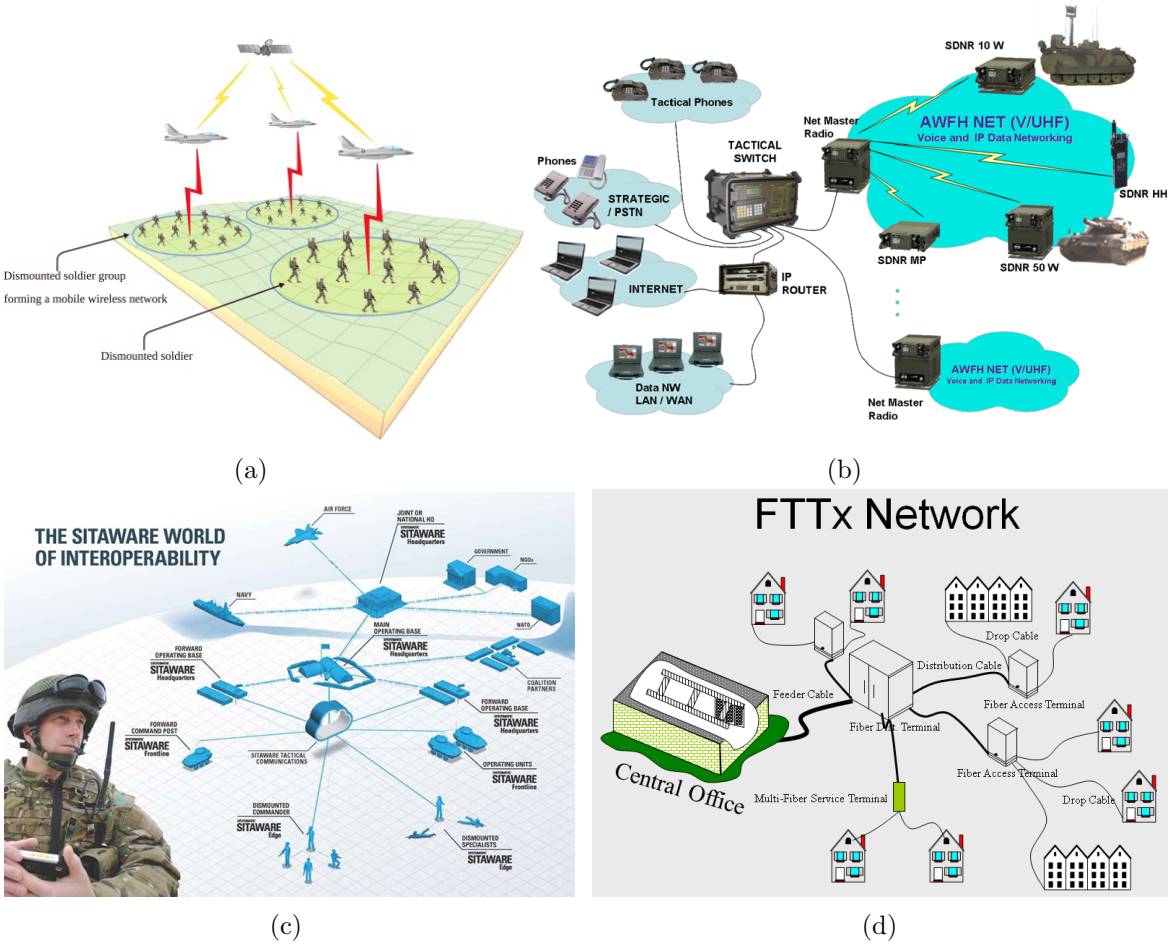
(a) (b)

Hình 1.4: Ứng dụng của bài toán CluSPT trong nông nghiệp (nguồn: Internet)

Bài toán CluSPT còn xuất hiện nhiều trong các ứng dụng tối ưu hệ thống thông tin liên lạc hay hệ thống cấp điện như minh họa trong hình 1.5.

1.3.4. 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, một trong các bài toán được nghiên cứu sớm nhất là bài toán người đi du lịch phân cụm (*Clustered*



Hình 1.5: Ứng dụng của bài toán CluSPT trong tối ưu hệ thống cấp điện và thông tin liên lạc (nguồn: Internet)

Traveling Salesman Problem - CluTSP) [5, 40, 56]. Bài toán CluTSP là một biến thể của bài toán tối ưu tổ hợp nổi tiếng bài toán người đi du lịch (*Traveling Salesman Problem - TSP*) [72, 93, 96]. Ngày nay, xuất phát từ yêu cầu cần tối ưu hệ thống mạng, bài toán cây phân cụm (clustered tree problem) nhận được nhiều sự quan tâm trong cộng đồng các nhà nghiên cứu [13, 23, 84].

Tác giả Y.S. Myung và nhóm nghiên cứu [57] đã nghiên cứu bài toán cây khung nhỏ nhất tổng quát (*Generalized Minimum Spanning Tree Problem - GMSTP*), là bài toán tổng quát của bài toán cây khung nhỏ nhất (*Minimum Spanning Tree Problem*) [2, 48] với các đỉnh được chia vào các nhóm. Trong đó, lời giải của bài toán GMSTP là cây có chi phí nhỏ nhất

và mỗi nhóm chỉ chứa một đỉnh duy nhất. Trong nghiên cứu này, sau khi chứng minh GMSTP là bài toán thuộc lớp NP–Khó, các tác giả đã đề xuất hai mô hình quy hoạch nguyên tuyến tính và so sánh chúng trong việc giải mô hình nới lỏng quy hoạch nguyên khi áp dụng vào bài toán GMSTP. Để đánh giá thuật toán đề xuất, tác giả sử dụng hai tập dữ liệu: tập dữ liệu nhỏ được chọn từ dữ liệu bài toán người du lịch tổng quát (*Generalized Traveling Salesman Problem - GTSP*) [45, 66] với số đỉnh nhỏ hơn 100; tập dữ liệu lớn được tạo ngẫu nhiên. Kết quả thực nghiệm cho thấy khoảng cách giữa cận trên và cận dưới tăng khi số lượng nhóm và số đỉnh trong mỗi nhóm tăng.

Một trong các bài toán cây phân cụm khác nhận được nhiều sự quan tâm là bài toán cây Steiner phân cụm (*Clustered Steiner Tree Problem - CluSteinerTP*) [86, 87]. Bài toán CluSteinerTP là một biến thể của bài toán cây Steiner (*Steiner tree problem - STP*) [42, 82]. Trong bài toán CluSteinerTP các đỉnh cũng đươc chia vào các cụm, một bài toán STP là một bài toán CluSteinerTP nếu các cụm không có phần tử chung [84]. Trong [86], dựa trên kết quả thực nghiệm, các tác giả B. Y. Wu và C. W. Lin đã chỉ ra rằng tỉ lệ Steiner nằm trong khoảng (3, 4) thì những kết quả tốt nhất và từ đó đề xuất một thuật toán gần đúng cho bài toán CluSteinerTP. Thuật toán này chuyển bài toán ban đầu về một bài toán cây Steiner sao cho cây Steiner tìm được không có đỉnh Steiner nào thuộc vào cây cục bộ của các cụm. Thuật toán được đề xuất có độ phức tạp $O(n * log(n) + f(n))$ trong đó n là số đỉnh.

Một biến thể khác của bài toán cây phân cụm, bài toán cây khung phân cụm có chi phí định tuyến nhỏ nhất (*Minimum Routing Cost Clustered Tree Problem - CluMRCT*) [52, 53]. Trong nghiên cứu của mình các tác giả đã chỉ ra rằng bài toán CluMRCT thuộc lớp NP-Khó nếu bài toán có ít nhất hai cụm. Các tác giả cũng đã đề xuất một thuật toán xấp xỉ cận tỉ lệ 2 (2-approximation) để giải bài toán CluMRCT bằng cách tạo đồ thị gồm hai mức dựa trên cây khung R-star (R-star spanning tree) và dựa trên hai

đặc trưng của cây khung R-star: một cây khung R-star với chi phí định tuyến nhỏ nhất có thể được tạo trong $O(n^2)$ với n là số đỉnh của bài toán và tồn tại một cây khung R-star với chi phí lớn nhất gấp hai lần chi phí của lời giải tối ưu của bài toán CluMRCT.

Gần đây, tác giả D'Emidio và các cộng sự [21, 22] 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.

Các tác giả đã trong nghiên cứu [15] đã đề xuất thuật toán hai mức dựa trên thuật toán GA để giải bài toán CluSPT. Trong đó, mức thứ nhất sẽ xác định cây khung nối giữa các cụm còn mức thứ hai sẽ tìm cây khung trong các cụm. Một trong các đóng góp chính của nghiên cứu là đã đề xuất sử dụng mảng hai chiều để mã hóa lời giải bài toán CluSPT. Ưu điểm của cách mã hóa này giúp lưu được nhiều thông tin về các cạnh nối giữa các cụm nên có thể xây dựng được cây khung tốt nhất nối giữa các cụm. Bên cạnh đó các tác giả cũng đề xuất phương pháp xây dựng cá thể, các toán tử lai ghép và đột biến dựa trên mã hóa được đề xuất. Dù kết quả thực nghiệm chỉ ra tính hiệu quả của đề xuất khi so sánh với thuật toán GA được đề xuất trước đó áp dụng vào giải bài toán CluSPT. Tuy nhiên, thuật toán đề xuất vẫn có hạn chế khi số lượng cá thể được tạo khi khởi tạo quần thể và số phép lai ghép thực hiện trong mỗi thế hệ rất lớn (gấp 3 lần kích thước quần thể) nên khi tiêu chí so sánh giữa các thuật toán là số lần đánh giá thì thuật toán đề xuất chưa chắc đã hiệu quả bằng các thuật toán đã có.

Trong nghiên cứu [16], tác giả Ovidiu Cosma và cộng sự đã đề xuất cải tiến thuật toán GA trong nghiên cứu [15]: sử dụng kết hợp hai cơ chế khởi tạo cá thể dựa trên ngẫu nhiên và dựa trên thuật toán tham lam để tạo quần thể bản đầu. Điểm khác biệt nữa so với nghiên cứu trước đây là thuật toán cải tiến sử dụng cả phương pháp chọn lọc ngẫu nhiên và chọn lọc cá thể ưu tú để chọn lọc cá thể cha mẹ thực hiện lai ghép. Thông qua

phân tích kết quả so sánh với thuật toán GA và thuật toán MFEA được nghiên cứu trước đó, các tác giả đã chỉ ra hiệu quả của thuật toán đề xuất.

Các tác giả trong nghiên cứu [21, 22] 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.

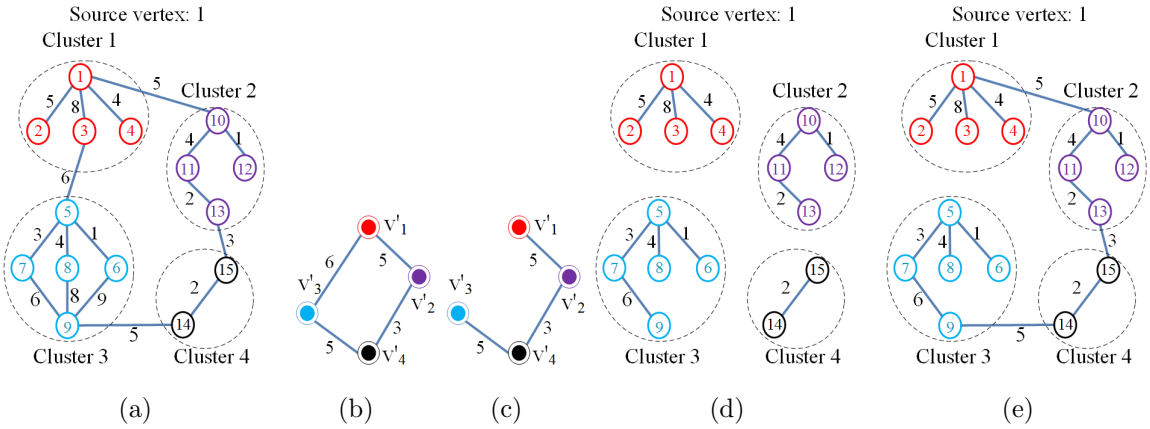
Thuật toán AAL chia bài toán CluSPT thành 2 bài toán con và tìm lời giải mỗi bài toán con trong một giai đoạn khác nhau. Thuật toán AAL sử dụng thuật toán Prim [69] để tìm lời giải của bài toán con thứ nhất (là cây khung nhỏ nhất của đồ thị nối giữa các cụm) và lời giải bài toán con thứ 2 (là cây khung nhỏ nhất của đồ thị con trong mỗi cụm). Chi tiết các bước của thật toán AAL như sau:

Bước 1: Xác định đồ thị $G' = (V', E')$, trong đó mỗi đỉnh thuộc tập V' tương ứng với một cụm V_i. Giữa hai đỉnh của đồ thị G' có cạnh nối khi và chỉ khi có ít nhất một cạnh của đồ thị G nối giữa các đỉnh thuộc hai cụm tương ứng với hai đỉnh đó.

Bước 2: Tìm cây khung nhỏ nhất T' của đồ thị G' và tìm h cây khung nhỏ nhất $T_1, \ldots, T_h$ tương ứng của các đồ thị con $G[C_1], \ldots, G[C_h]$.

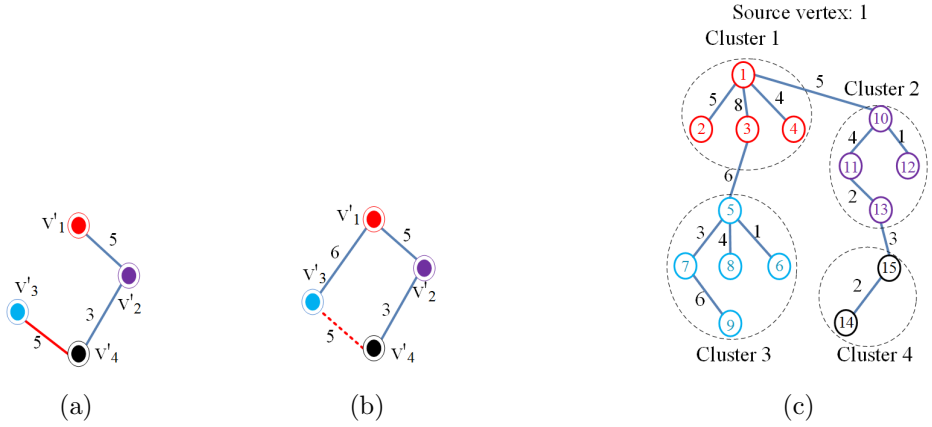
Bước 3: Tạo cây khung của đồ thị G bằng cách hợp các cây khung nhỏ nhất T' và $T_i (i = 1, \ldots, h)$: $\left\{\cup_{i=1}^{h} E(T_i)\right\} \cup E(T')$

Hình 1.6 minh họa các bước của thuật toán AAL. Hình 1.6(a) minh họa đồ thị đầu vào G gồm 15 đỉnh, 4 cụm và đỉnh nguồn là 1. Hình 1.6(b) minh họa đồ thị G' nhận được từ đồ thị G trong bước 1 của thuật toán. Do đồ thị G có 4 cụm nên đồ thị G' có 4 đỉnh v'_1, v'_2, v'_3, v'_4. Do trên đồ thị G có các cạnh nối giữa cụm 1 với cụm 2; cụm 1 với cụm 3; cụm 2 với cụm 4; cụm 3 với cụm 4, nên đồ thị G' có các cạnh (v'_1, v'_2); (v'_1, v'_3); (v'_2, v'_4) và (v'_3, v'_4). Hình 1.6(c) minh họa cây khung nhỏ nhất của đồ thị G' được tạo ra bởi áp dụng thuật toán Prim [69]. Đồ thị sau khi thực hiện bước 2 của thuật toán AAL được minh họa trong hình 1.6(d) (sau khi áp dụng thuật



Hình 1.6: Ví dụ minh họa các bước của thuật toán AAL

toán Prim để tìm cây khung nhỏ nhất của đồ thị con trong mỗi cụm của đồ thị G). Hình 1.6(e) minh họa đồ thị nhận được sau khi thực hiện hợp của các cây khung trong hình 1.6(c) và hình 1.6(d) (bước 3 của thuật toán AAL). Lời giải của bài toán CluSPT sau khi áp dụng thuật toán AAL cho đồ thị G có chi phí là 186.



Hình 1.7: Ví dụ minh họa hạn chế của thuật toán AAL

Mặc dù thuật toán AAL có ưu điểm về thời gian tính toán, cấu trúc thuật toán đơn giản,... nhưng thuật toán AAL vẫn có hạn chế trong việc tìm lời giải của bài toán CluSPT, hạn chế này do thuật toán AAL có gắng cực tiểu tổng trọng số của cây khung của các đồ thị con mà không xem xét tới khoảng cách từ các đỉnh tới đỉnh nguồn. Do đó, lời giải tìm được bằng thuật toán AAL có thể không tốt. Để rõ hơn về nhận định này, luận

án xét ví dụ trong hình 1.7 với đồ thị đầu vào và lời giải tìm được bằng thuật toán AAL như trong hình 1.6. Để cho dễ theo dõi, cây khung của đồ thị G' trong hình 1.6(c) được minh họa lại trong hình 1.7(a). Nếu cạnh (v'_1, v'_3) (cạnh nét đứt, màu đỏ trong hình 1.7(b)) được thay bằng bằng cạnh (v'_3, v'_4), thì khi đó cây khung trong hình 1.7(a) có giá trị 13 sẽ thấp hơn cây khung trong hình 1.7(b) (có giá trị 14). Mặc dù vậy, chi phí của lời giải tương ứng với cây khung trong hình 1.7(b) là 154 (hình 1.7(c)), thấp hơn chi phí 186 của cây khung trong hình 1.6(e). Qua ví dụ trên có thể thấy rằng, thuật toán AAL có thể tìm ra lời giải không tốt. Từ các phân tích trên, tác giả nhận thấy rằng các thuật toán trong các nghiên cứu trước đó vẫn chưa thực sự hiệu quả, đặc biệt chưa sử dụng các thuật toán meta-heuristic để giải bài toán CluSPT.

1.4. Kết luận chương

Chương này đã trình bày một số nội dung chính sau: thứ nhất, trình bày kiến thức cơ bản về thuật toán tiến hóa và thuật toán tiến hóa đa nhân tố. Trong đó, thuật toán tiến hóa và tiến hóa đa nhân tố được nghiên cứu chính trong luận án này. Thứ hai, giới thiệu bài toán CluSPT - là bài toán thuộc lớp bài toán NP-Khó. Thứ ba, 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. Thứ tư, 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 như các nghiên cứu về bài toán CluSteinerTP, bài toán CluTSP, bài toán CluMRCT.

Chương 2

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

Do có ưu điểm về thời gian thực hiện và dễ cài đặt nên các thuật toán xấp xỉ được sử dụng nhiều trong các bài toán tối ưu trên đồ thị. Chương này trình bày về hai thuật toán xấp xỉ để 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. Nghiên cứu của chương này đã công bố trong công trình [III] và [VI].

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 (*metric graph*) [53], lời giải tìm được là tối ưu.

2.1.1. Lược đồ thuật toán

Một trong các hướng tiếp cận đơn giản nhất để đảm bảo lời giải của bài toán CluSPT luôn hợp lệ đó là lần lượt xây dựng cây khung cho các cụm, sau đó xây dựng cây khung nối giữa các cụm. Tuy nhiên, do chi phí lời giải của bài toán CluSPT được tính dựa trên đường đi ngắn nhất từ mỗi đỉnh tới đỉnh nguồn, nên việc xây dựng cây khung cho từng cụm và cây khung nối giữa các cụm cần đồng thời gắn với việc tối ưu chi phí đó.

Dựa trên các nhận xét trên, 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. Trong trường hợp đồ thị dạng sao có ba đỉnh, do chỉ có duy nhất một đỉnh là đỉnh trong, nên đỉnh này là tâm của đồ thị. Đối với trường hợp đồ thị dạng sao chỉ có một đỉnh hoặc hai đỉnh, thì một đỉnh bất kỳ là tâm của đồ thị.

Thuật toán SLA tìm cây khung phân cụm T thông qua các bước như sau:

Bước 1 (xác định cây khung trong cụm gốc): Giả sử đỉnh nguồn s thuộc cụm C_t. Tìm đường đi ngắn nhất nối các đỉnh trong cụm C_t với đỉnh s. Cây khung dạng sao của cụm C_t được tạo bởi hợp của các được đi ngắn nhất tìm được.

Bước 2 (xác định cây khung trong các cụm không phải cụm gốc): Cây khung bộ phận của mỗi cụm $C_i, (1 \leq i \leq h, i \neq t)$ là một cây khung dạng sao với đỉnh trung tâm $r_i \in C_i$ thỏa mãn $r_i = argmin\{f(v)|v \in C_i\}$ trong đó:

$$f(v) = |C_i| \times d(s, v) + \sum_{u \in C_i} d(v, u) \tag{2.1}$$

với $|C_i|$, $d(x, y)$ lần lượt là số đỉnh trong cụm $|C_i|$ và giá trị đường đi ngắn nhất nối giữa đỉnh x và đỉnh y của đồ thị đầu vào G.

Bước 3 (xác định cây khung nối giữa các cụm): Sau khi xây dựng được cây khung bộ phận của các cụm C_i, cạnh nối giữa đỉnh trung tâm r_i và đỉnh nguồn s chính là cạnh liên cụm.

Công thức (2.1) tính chi phí từ các đỉnh thuộc cụm C_i tới đỉnh nguồn s khi đỉnh v là gốc cục bộ (*local root*) của cụm C_i. Hàm $f(v)$ được tính dựa trên chi phí từ đỉnh nguồn s tới đỉnh v và chi phí từ đỉnh v tới tất cả các đỉnh khác trong cùng cụm. Do v là đỉnh gốc cục bộ nên tổng chi phí từ đỉnh nguồn s tới các đỉnh trong cụm C_i là:

$$\sum_{u \in C_i} d(s, u) = \sum_{u \in C_i} (d(s, v) + d(v, u)) \tag{2.2}$$

$$= |C_i| \times d(s, v) + \sum_{u \in C_i} d(v, u) \tag{2.3}$$

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: $T = (V_T, E_T)$;

1 **begin**
2 $V_T \leftarrow V$;
3 $E_T \leftarrow \emptyset$;
4 $C_t \leftarrow$ Cụm chứa đỉnh nguồn s;
`/* Xác định cây khung trong cụm chứa đỉnh nguồn` s `*/`
5 **foreach** *đỉnh* $v \in C_t, v \neq s$ **do**
6 $p_v \leftarrow$ Đường đi ngắn nhất nối s và v trên đồ thị con $G[C_t]$;
7 $E_T \leftarrow E_T \cup p_v$;
8 **end**
`/* Xác định cây khung trong cụm không chứa đỉnh nguồn` s `*/`
9 **foreach** *cụm* C_i *với* $i \neq t$ **do**
10 **foreach** *đỉnh* $v \in C_i$ **do**
11 $p_{v,u} \leftarrow$ Đường đi ngắn nhất nối v và $u (u \in C_i)$ trên đồ thị con $G[C_i]$;
12 $d_{v,u} \leftarrow$ Chi phí của đường đi $p_{v,u}$;
13 $f_v \leftarrow |C_i| \times d(s, v) + \sum_{u \in C_i} d_{v,u}$ ▷ Chi phí tới đỉnh s và các đỉnh cùng cụm;
14 **end**
15 $r_i = argmin\{f(v) | v \in C_i\}$ ▷ Đỉnh có tổng chi phí nhỏ nhất trong cụm;
16 $E_T \leftarrow E_T \cup p_{r_i,u}, \forall u \in C_i, u \neq r_i$;
17 **end**
`/* Xác định cây khung nối giữa các cụm */`
18 **foreach** *cụm* $C_i, i \neq n$ **do**
19 $E_T \leftarrow E_T \cup e = (s, r_i)$ ▷ Thêm cạnh nối giữa s và đỉnh r_i vào tập cạnh;
20 **end**
21 **return** (V_T, E_T);
22 **end**

2.1.2. Độ phức tạp của thuật toán

Chi phí xác định cây khung trong mỗi cụm là $O(|C_i|)_{i=1,..,h}$ với $|C_i|$ là số đỉnh của cụm thứ i. Do chỉ cần xây dựng cây hình sao (nối một đỉnh gốc cục bộ được chọn với các đỉnh còn lại) nên độ phức tạp thực hiện dòng 5 – 7 và 10 – 13 đều là $O(|C_i|)$.

Các dòng lệnh 10 – 13 được thực hiện $|C_i|$ lần trong vòng lặp **for** ở dòng 10 – 17 nên độ phức tạp của khối dòng lệnh 10 – 17 là $O(|C_i|^2)$.

Do các dòng lệnh 10 – 17 được thực hiện một lần trong mỗi cụm còn lại nên độ phức tạp của khối dòng lệnh 9 – 17 là $\sum_{i=1..h, i \neq t} |C_i|^2$.

Do đó, độ phức tạp thuật toán là $O(|C_t| + \sum_{i=1..h, i \neq t} |C_i|^2)$ với C_t là

cụm chứa đỉnh nguồn.

Nếu dùng cận lỏng hơn, do $|C_t| + \sum_{i=1..h, i\neq t} |C_i|^2 < \sum_{i=1..h} |C_i|^2 < \left(\sum_{i=1..h} |C_i|\right)^2 = |V|^2$ nên độ phức tạp của thuật toán SLA có thể coi là $O(|V|^2)$.

2.1.3. Thuật toán dạng hình sao trên đồ thị metric

Phần này sẽ nghiên cứu bài toán CluSPT trên một lớp đồ thị đặc biệt - đồ thị metric. Dựa trên các đặc trưng của đồ thị metric, luận án chỉ ra rằng thuật toán SLA sẽ tìm được lời giải tối ưu trên lớp đồ thị này.

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 của T là một đồ thị dạng sao.*

Chứng minh. Cho trước một đồ thị liên thông G và một đỉnh $v \in G$, một cây T được gọi là cây đường đi ngắn nhất (shortest path tree) có gốc tại đỉnh v nếu cây T thỏa mãn điều kiện: đường đi từ gốc v tới mỗi đỉnh $u \in T$ cũng chính đường đi ngắn nhất từ đỉnh v tới đỉnh u trên đồ thị G.

Các nghiên cứu trong [74, 89] đã chỉ ra rằng: cây đường đi ngắn nhất T với gốc v của đồ thị G cũng là một cây khung của đồ thị G.

Vì đồ thị G là đồ thị metric nên đường đi ngắn nhất nối giữa hai đỉnh bất kỳ u và v trong G cũng chính là cạnh $e = (u, v)$ nối giữa hai đỉnh này. Do đó, cây đường đi ngắn nhất có gốc tại đỉnh v trong đồ thị metric sẽ có dạng sao với tâm là đỉnh v. □

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 trong lời giải T sẽ là cạnh nối giữa gốc của một cây khung bộ phận và đỉnh nguồn s của đồ thị G.*

Chứng minh. Xét cây khung bộ phận T' của cây khung T với đỉnh gốc của T' là đỉnh v. Do mỗi đường nối từ đỉnh nguồn s tới đỉnh bất kỳ thuộc cây khung T' đều phải đi qua đỉnh v, nên chi phí đường đi ngắn nhất từ đỉnh nguồn s tới các đỉnh thuộc cây khung T' trong giá trị hàm mục tiêu của

lời giải T được tính theo công thức:

$$\sum_{u \in V(T')} d(u,s) = \sum_{u \in V(T')} d(u,v) + |V(T')| \times d(v,s) \tag{2.4}$$

Công thức (2.4) cho thấy, lời giải T là lời giải tối ưu khi và chỉ khi $d(v,s)$ là khoảng cách ngắn nhất giữa đỉnh v và đỉnh nguồn s. Do đang xét trên đồ thị metric nên đường đi ngắn nhất nối giữa hai đỉnh v và s chính là cạnh (v,s). Do đó, cạnh liên cụm nối giữa đỉnh gốc của cây khung bộ phận T' tới đỉnh nguồn s là cạnh nối giữa đỉnh v và đỉnh s. Hay nói cách khác, mỗi cạnh liên cụm của lời giải T sẽ nối đỉnh gốc của một cây khung bộ phận với đỉnh nguồn s. □

Từ các bổ đề (2.1.1) và (2.1.2) suy ra: một lời giải tối ưu của bài toán CluSPT trong đồ thị metric là một cây khung phân cụm gồm các đặc trưng:

1. Mỗi cây khung bộ phận là một cây khung dạng sao;
2. Mỗi cạnh liên cụm là một cạnh nối giữa đỉnh nguồn của đồ thị và đỉnh tâm của cây khung bộ phận.

Các lập luận trên chỉ ra rằng, thuật toán SLA có thể tìm được lời giải tối ưu của bài toán CluSPT trong đồ thị metric. Thuật toán SLA trên đồ thị metric (ký hiệu là SLA-M) chỉ thay khoảng cách giữa hai đỉnh v và u trong **Bước 2** bằng trọng số cạnh nối giữa hai đỉnh v và u. Các bước của thuật toán SLA-M như sau:

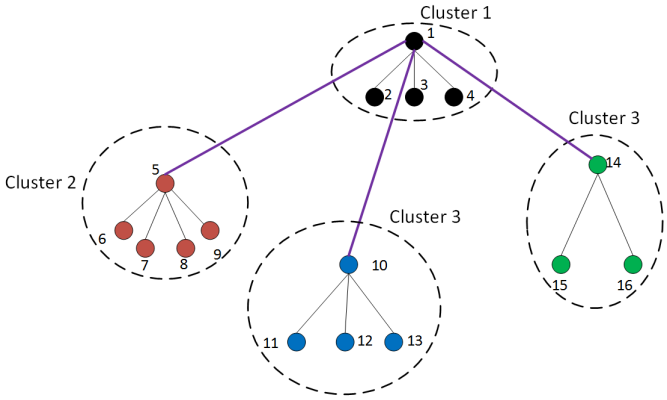
Bước 1: Giả sử đỉnh nguồn s thuộc cụm C_t, do cây khung bộ phận của lời giải T tại cụm C_t là một cây khung dạng sao với đỉnh trung tâm là đỉnh s, nên từ đỉnh s chỉ cần tìm cạnh nối tới các đỉnh khác thuộc cụm C_t.

Bước 2: Cây khung bộ phận của mỗi cụm $C_i, (1 \leq i \leq h, i \neq t)$ là một cây khung dạng sao với đỉnh trung tâm $r_i \in C_i$ thỏa mãn $r_i = argmin\{f(v)|v \in C_i\}$ trong đó $f(v) = |C_i| \times w(s,v) + \sum_{u \in C_i} w(v,u)$

và $w(v, u)$ là trọng số của cạnh nối giữa đỉnh v và đỉnh u của đồ thị đầu vào G.

Bước 3: Sau khi xây dựng được cây khung bộ phận của các cụm C_i, cạnh nối giữa đỉnh trung tâm r_i và đỉnh nguồn s chính là cạnh liên cụm.

Hình 2.1 minh họa một lời giải tối ưu của bài toán CluSPT trên đồ thị metric với đồ thị đầu vào gồm 4 cụm, 16 đỉnh và đỉnh nguồn là đỉnh 1.



Hình 2.1: Ví dụ về lời giải của bài toán CluSPT trên đồ thị metric

2.2. Thuật toán tham lam ngẫu nhiên

Phần này trình bày thuật toán xấp xỉ dựa trên thuật toán tham lam ngẫu nhiên (*Heuristic Based on Randomized Greedy Algorithm - HB-RGA*) giải bài toán CluSPT. Thuật toán HB-RGA dựa trên sự kết hợp giữa thuật toán tham lam ngẫu nhiên (*Randomized Greedy Algorithm - RGA*) [32, 63] và thuật toán cây đường đi ngắn nhất (*Shortest Path Tree Algorithm - SPTA*) [31, 74], trong đó:

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

Để tìm cạnh "tốt nhất" nối giữa một cụm đã được xét và một cụm chưa được xét, ứng với mỗi cạnh e của đồ thị đầu vào, thuật toán HB-RGA định nghĩa thêm tham số *phần thưởng* được tính theo trọng số của cạnh e và giá trị của cây đường đi ngắn nhất của cụm chưa được xét. Thuật toán

HB-RGA sẽ chọn cạnh nối giữa các cụm theo xác suất được tính từ giá trị *phần thưởng* của mỗi cạnh.

Trong phần này, ta định nghĩa:

- Tập Q là tập các cụm chưa được đưa vào lời giải đang xét. Ban đầu, tập Q gồm tất cả các cụm.
- Khoảng cách từ cụm gốc tới cụm bất kỳ là tổng chi phí của đường đi ngắn nhất từ đỉnh nguồn tới đỉnh gốc của cụm đó. Không mất tính tổng quát, ta giả sử đỉnh nguồn thuộc cụm 1. Nếu gọi $dis[i]$ là khoảng cách từ cụm gốc tới cụm $C_i (i = 1, \ldots, h)$ thì tại thời điểm ban đầu $dis[1] = 0$, $dis[i] = +\infty$, $i \in \{2, \ldots, h\}$.
- Với mỗi cụm C_i, ký hiệu $d(r_i, v)$ là khoảng cách từ đỉnh gốc của cụm C_i tới đỉnh $v \in C_i$.

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 [25, 74] để tìm cây đường đi ngắn nhất với đỉnh bắt đầu là đỉnh x. Phương thức $CostSPT(x)$ sẽ tính tổng chi phí các đường đi ngắn nhất từ đỉnh gốc x tới tất cả các đỉnh khác thuộc cùng cụm với đỉnh x.

2.2.1. Các bước của thuật toán HB-RGA

Thuật toán HB-RGA gồm các bước như sau:

1. **Bước 1: xét cụm gốc**
   - **Bước 1.1**: tạo cây khung đường đi ngắn nhất của cụm gốc bằng cách áp dụng thuật toán Dijkstra bắt đầu từ đỉnh nguồn s.
   - **Bước 1.2**: thêm cây khung đường đi ngắn nhất nhận được ở **Bước 1.1**. vào lời giải của bài toán CluSPT và đánh dấu cụm gốc là cụm đang xét, ký hiệu C_c.
   - **Bước 1.3**: xóa cụm gốc khỏi tập Q.
2. **Bước 2: với mỗi cụm khác**

   Sau khi thực hiện các bước sau, thuật toán HB-RGA sẽ thêm lần lượt các cụm còn lại vào lời giải trung gian.

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 ▷ Gán cụm đang xét là cụm chứa đỉnh nguồn;
  $T \leftarrow Find_Shortest_Path_Tree(s)$ ▷ Tạo cây đường đi ngắn nhất cho cụm gốc với đỉnh bắt đầu là đỉnh s;
  $dis[cur] = 0$ ▷ Khoảng cách từ cụm đang xét 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;
        $f(u, v) = m \times (d[u] + w[u, v]) + CostSPT(v)$;
      end
      $e = (a, b) \leftarrow$ Chọn cạnh phù hợp nhất trong tập $\{(u, v) | u \in C_{cur}, v \in C_i\}$ với xác suất $p(u, v) = \dfrac{f(u, v)^\gamma}{\sum_{u' \in C_{cur}, v' \in C_i} f(u', v')^\gamma}$;
      if $dis[i] > dis[cur] + d[a] + w[a, b]$ **then**
        $dis[i] \leftarrow dis[cur] + d[a] + w[a, b]$;
        $r_i \leftarrow b$;
        $temporaryEdge[i] \leftarrow e$;
      end
    end
    $cur \leftarrow \arg\min_{i \in Q} dis[i]$;
    $T \leftarrow T \cup temporaryEdge[cur]$;
    $T \leftarrow T \cup Find_Shortest_Path_Tree(r_{cur})$;
    $Q \leftarrow Q \backslash cur$;
  end
end

- **Bước 2.1: tìm cạnh trung gian nối giữa hai cụm**. Với mỗi cụm C_i trong tập Q, tìm một cạnh (u, v) nối cụm C_i với cụm đang xét C_c bằng thuật toán đề xuất dựa trên thuật toán RGA (trình bày trong phần 2.2.2) (đỉnh u thuộc cụm đang xét C_c, đỉnh

v thuộc một trong các cụm $C_i \in Q$)

- **Bước 2.2: cập nhật khoảng cách giữa cụm gốc và mỗi cụm** C_i. Gọi $w(u, v)$ là trọng số cạnh nối giữa hai cụm trong **Bước 2.1**. Nếu $dis[c] + d[r_c, u] + w[u, v] < dis[i]$ thì khoảng cách $dis[i]$ được cập nhật theo công thức $dis[i] = dis[c] + d[r_c, u] + w[u, v]$ và đặt $r_i = v$.
- **Bước 2.3:**
  - Tìm cụm $C_i \in Q$ có giá trị $dis[i]$ nhỏ nhất, sau đó thêm cụm này vào lời giải trung gian với cạnh nối là cạnh (u, v) được xác định trong **Bước 2.1**.
  - Cây khung trong cụm C_i được xây dựng bằng cách áp dụng thuật toán Dijkstra với đỉnh gốc v của cụm C_i được xác định trong **Bước 2.2**.
  - Đặt cụm C_i là cụm đang xét và xóa cụm C_i khỏi tập Q.
- **Bước 2.4:** Nếu tập Q chưa rỗng thì chuyển tới **Bước 2.1**, ngược lại thì dừng thuật toán.

2.2.2. Thuật toán tham lam ngẫu nhiên tìm cạnh nối giữa các cụm

Phần này trình bày về đề xuất áp dụng thuật toán RGA vào tính giá trị $d[i]$ của mỗi cụm C_i chưa được đưa vào lời giải trung gian (các cụm đang thuộc tập Q).

Với mỗi cụm chưa được chọn C_i thuộc Q:

- Sinh ngẫu nhiên số m có giá trị trong khoảng $[\alpha, \beta]$, trong đó α là số đỉnh của cụm C_i; β là tổng số đỉnh của tất cả các cụm thuộc tập Q. Nghĩa là $|C_i| \leq m \leq \sum_{C_j \in Q} |C_j|$ với C_j là cụm thuộc tập Q, $|C_j|$ là số đỉnh thuộc cụm thứ j.
- Gọi $E' = \{(u, v) | u \in C_c, v \in C_i\}$ là tập tất cả các cạnh nối giữa cụm đang xét C_c và cụm C_i. Hàm f tính chi phí các cạnh $(u, v) \in E'$ được

định nghĩa như sau:

$$f : E' \to \mathbb{R}$$
$$(u, v) \mapsto m \times (d[r_c, u] + w[u, v]) + costSPT(v);$$

trong đó $costSPT(v)$ là tổng chi phí của các đường đi ngắn nhất từ đỉnh v tới tất cả các đỉnh khác thuộc cụm C_i.

- Xác suất cạnh (u, v) được chọn làm cạnh nối giữa cụm đang xét C_c và cụm C_i được tính theo công thức:

$$p_{(u,v)} = \frac{f(u, v)^\gamma}{\sum_{(u',v')\in E'} f(u', v')^\gamma} \tag{2.5}$$

trong đó $\gamma \leq 0$ là tham số được sử dụng để xác định *mức độ tham* của thuật toán.

Hình 2.2 minh họa các bước của thuật toán HB-RGA, trong đó đồ thị đầu vào được minh họa trong hình 2.2(a) và đỉnh nguồn là đỉnh 1.

- Hình 2.2(b) minh họa **Bước 1** và **Bước 2.1**, trong đó đồ thị con của cụm 1 được tạo ra sau khi kết thúc **Bước 1**. Trong **Bước 2.1**, thuật toán HB-RGA xem xét các cạnh liên cụm có nối với cụm đang xét (cụm 1 - đường bao quanh màu đỏ) và các cụm 2, cụm 3 (đường bao là hình ô van nét đứt):
  - Do có hai cạnh nối giữa cụm 1 và cụm 2 là cạnh (1, 10) và cạnh (4, 10), nên giả sử cạnh (1, 10) được chọn là cạnh tạm thời nối giữa cụm 1 và cụm 2.
  - Do chỉ có một cạnh (3, 5) nối giữa cụm 1 và cụm 3 nên cạnh này được chọn là cạnh tạm thời nối giữa hai cụm này.
- Hình 2.2(c) minh họa **Bước 2.3** khi:
  - Do khoảng cách giữa cụm 1 và cụm 2 là 5, còn giữa cụm 1 và cụm 3 là 8 + 3 = 11. Nên cụm 2 được chọn là cụm đang xét và cạnh (1, 10) được thêm vào lời giải trung gian.
  - Áp dụng thuật toán Dijkstra vào tìm cây đường đi ngắn nhất của

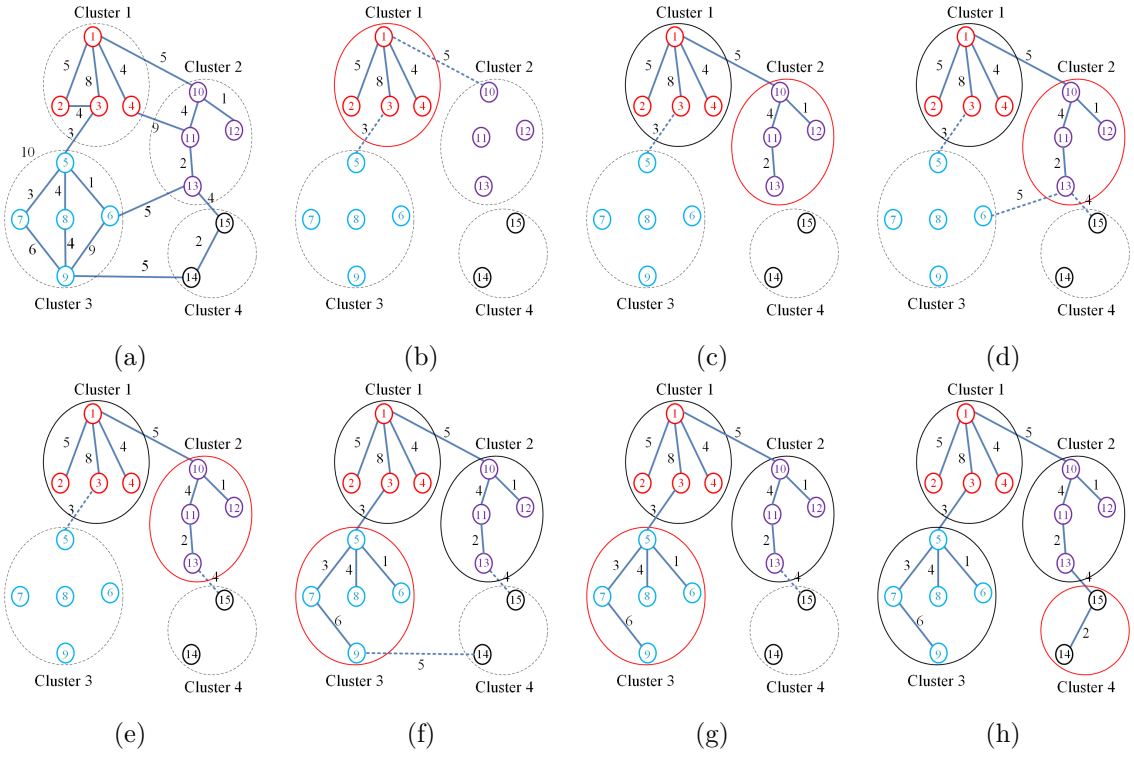
cụm 2 với đỉnh bắt đầu là đỉnh 10. Sau đó thêm cây đường đi ngắn nhất tìm được vào lời giải trung gian.

- Hình 2.2(d) minh họa **Bước 2.1** của thuật toán HB-RGA, trong đó cụm 2 là cụm đang xét. Do cạnh (13, 6) và cạnh (13, 15) lần lượt nối cụm 2 với cụm 3 và cụm 4, nên các cạnh này được sử dụng để tính khoảng cách từ cụm 3 và cụm 4 tới cụm gốc.
- Hình 2.2(e) minh họa **Bước 2.2** của thuật toán HB-RGA:
  - Cập nhật khoảng cách từ cụm gốc tới các cụm chưa được đưa vào lời giải trung gian.
  - Có hai cạnh nối giữa cụm 3 và các cụm đã được đưa vào lời giải tạm thời trong các bước trước (cạnh (3, 5)) nối với cụm 1 và cạnh (6, 13) nối với cụm 2). Do chiều dài đường đi ngắn nhất từ đỉnh 5 tới đỉnh nguồn 1 là $3 + 8 = 11$ và độ dài đường đi ngắn nhất từ đỉnh 6 tới đỉnh nguồn 1 là $5 + 2 + 4 + 5 = 16$, nên cạnh (3, 5) được giữ lại để đánh giá trong bước tiếp theo.
- Các bước tính toán trong hình 2.2(f) tương tự các bước đã miêu tả trong hình 2.2(c), trong đó khoảng cách giữa cụm 3 và cụm gốc (bằng 11) nhỏ hơn khoảng cách giữa cụm 4 và cụm gốc (bằng $5 + 2 + 4 + 4 = 15$), nên cụm 3 được chọn là cụm đang xét (hình ô van màu đỏ).
- Hình 2.2(g) minh họa lời giải trung gian sau khi cạnh (13, 15) được chọn để nối giữa cụm 2 và cụm 4.
- Hình 2.2(h) minh họa lời giải của bài toán CluSPT tìm được sau khi áp dụng thuật toán đề xuất.

2.2.3. Độ phức tạp của thuật toán

Để đánh giá độ phức tạp của thuật toán SLA, luận án ký hiệu: E_i là tập cạnh của đồ thị con $G[C_i]$; $n_{E_{max}} = \max\{|E_i|\}_{i=1,..,h}$; $n_{C_{max}} = \max\{|C_i|\}_{i=1,..,h}$.

Độ phức tạp về thời gian của việc xây dựng cây đường đi ngắn nhất dựa trên thuật toán Dijkstra trong cụm chứa đỉnh nguồn (dòng 5) là



Hình 2.2: Ví dụ minh họa các bước của thuật toán HB-RGA

$O(|E_m| + |C_m| \log |C_m|)$ với $|E_m|$ và $|C_m|$ lần lượt là số cạnh và số đỉnh của cụm chứa đỉnh nguồn s.

Giả sử cụm đang xét là cụm C_{cur}, tiến hành duyệt qua các cụm còn lại chưa xét. Duyệt cụm C_i:

- Dòng 11: số lần lặp của vòng **foreach** ở dòng 11 bằng số cạnh nối giữa hai cụm C_i và C_{cur} giả sử là $E_{cur,i}$. Trong mỗi lần lặp: chi phí chủ yếu do việc tính chi phí đường đi ngắn nhất từ đỉnh v đến các đỉnh trong C_i dùng thuật toán Dijkstra ở dòng 13:
  $O(|E_i| + |C_i| \log |C_i|) < O(n_{E_{max}} + n_{C_{max}} \log n_{C_{max}})$
- Độ phức tạp của dòng 11 – 15 là:

$$O(|E_{cur,i}|(n_{E_{max}} + n_{C_{max}} \log n_{C_{max}})) < O(|C_{cur}||C_i|(n_{E_{max}} + n_{C_{max}} \log n_{C_{max}})) < O(n_{C_{max}}^2(n_{E_{max}} + n_{C_{max}} \log n_{C_{max}}))$$

- Độ phức tạp của khối dòng lệnh 8 – 17:

– Khối dòng lệnh 10 – 17 thực hiện $\sum_{t=1..h}(h-t) = \frac{(h-1)*h}{2}$ lần.

– Độ phức tạp khối lệnh 8 – 17 là $O(h^2 n^2_{C_{max}}(n_{E_{max}}+n_{C_{max}} \log n_{C_{max}}))$.

Độ phức tạp thuật toán HB-RGA là $O(h^2 n^2_{C_{max}}(n_{E_{max}}+n_{C_{max}} \log n_{C_{max}}))$.

2.3. Đánh giá thuật toán

Để phân tích hiệu quả của thuật toán HB-RGA và thuật toán SLA-M luận án đã tiến hành:

- Thực nghiệm thuật toán HB-RGA được thực nghiệm trên 141 bộ dữ liệu thuộc 6 tập đồ thị metric và 75 bộ dữ liệu thuộc 3 tập đồ thị đầy đủ phi metric. Kết quả nhận được của thuật toán HB-RGA được so sánh với lời giải tối ưu và bốn thuật toán xấp xỉ khác.
- Sử dụng thuật toán SLA-M tìm lời giải tối ưu trên đồ thị metric và so sánh lời giải tìm được của các thuật toán xấp xỉ được đề xuất trong luận án với lời giải tối ưu.

Bên cạnh đó, luận án cũng tiến hành phân tích ảnh hưởng của các tham số của bộ dữ liệu đầu vào (số cụm và số đỉnh của đồ thị đầu vào) tới hiệu quả của thuật toán HB-RGA.

Do kết quả nghiên cứu trong công trình [III] đã chỉ ra rằng: thuật toán HB-RGA đạt hiệu quả tốt khi tham số $\gamma = 50$ nên trong các thực nghiệm của thuật toán HB-RGA luận án sử dụng giá trị $\gamma = 50$.

Để đánh giá các thuật toán đề xuất được chi tiết, trên nhiều khía cạnh và trên đa dạng các tiêu chí so sánh, đặc biệt là để thực hiện các phân tích thống kê và phân tích sự ảnh hưởng của các tham số dữ liệu đầu vào tới hiệu quả so sánh giữa nhiều thuật toán. Vì vậy, luận án phân tích kết quả các thuật toán, cũng như mô tả thực nghiệm chi tiết tại chương 5.

2.4. Kết luận chương

Chương này tập trung trình bày thuật toán đúng SLA-M áp dụng cho đồ thị metric và phương pháp xấp xỉ dựa trên thuật toán RGA.

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) như bài toán CluMRCT [53] và bài toán cây khung phân cụm với chi phí định tuyến liên cụm nhỏ nhất (*Minimum Inter-cluster Routing Cost Clustered Tree Problem - InterCluMRCT*) [85].

Thuật toán SLA-M được đề xuất dựa trên ý tưởng xây dựng lời giải bằng cách xác định đỉnh trung tâm của mỗi cụm và đỉnh trung tâm nối giữa các cụm. 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.

Kết quả tối ưu của các bộ dữ liệu trên đồ thị metric có được từ thuật toán đúng là cơ sở giúp luận án đánh giá một cách chính xác tính hiệu quả của các thuật toán xấp xỉ sau này. Bên cạnh đó, kết quả thuật toán HB-RGA cũng là cơ sở để luận án so sánh với các thuật toán tiến hóa và tiến hóa đa nhân tố ở phần sau.

Chương 3

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

Hiện nay, một số toán tử tiến hóa tìm cây khung của đồ thị đã được đề xuất. Tuy nhiên, nếu áp dụng các toán tử tiến hóa này để tìm lời giải bài toán CluSPT thì có thể tạo ra lời giải không hợp lệ. Vì vậy, chương này trình bày đề xuất dựa trên thuật toán tiến hóa để giải bài toán CluSPT. Thuật toán đề xuất luôn đảm bảo lời giải tìm được luôn hợp lệ và được xây dựng dựa trên ý tưởng phân rã bài toán CluSPT 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 chính xác giải bài toán con thứ hai.

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

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.

Để giải bài toán CluSPT, thuật toán C-EA sử dụng lược đồ mã hóa lời giải dựa trên hai lớp khác nhau. Cả hai lớp này đều sử dụng mã Cayley để biểu diễn các cây khung của mỗi lớp, tuy nhiên vị trí của các mã Cayley biểu diễn các cây khung bộ phận được đặt trước mã Cayley biểu diễn cây khung của đồ thị G-Graph. Bên cạnh đó, phần này còn trình bày các toán tử tiến hóa được xây dựng dựa trên cách tiếp cận hai lớp.

3.1.1. Lược đồ thuật toán

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 thuật toán 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 của lời giải bài toán CluSPT tương ứng s_j (xem dòng 4 và dòng 16 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 (dòng lệnh 5 và dòng lệnh 17 trong thuật toán 3.1) theo công thức sau:

$$fit(ind_j) = \frac{1}{f(s_j)} \tag{3.1}$$

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.

Để biểu diễn cả cây khung bộ phận và cây khung toàn cục, phương thức mã hóa sẽ chuyển một cá thể (ở dạng đồ thị) về dưới dạng là một chuỗi của các gen thông qua sử dụng thuật toán mã hóa của mã Cayley. Các cây khung bộ phận và cây khung toàn cục được nối với nhau thông qua tập các đỉnh là gốc của các cây khung bộ phận. Các thành phần của một cá thể sau khi mã hóa bằng mã Cayley bao gồm:

- Cây khung của cụm thứ t được biểu diễn bởi một đoạn gen thứ t, được ký hiệu là s_t. Giá trị của các gen trong mỗi đoạn s_t sẽ thuộc đoạn [1, $|C_t|$].
- Cây khung của đồ thị $G - Graph$ được biểu diễn bởi một đoạn gen riêng, ký hiệu là G-Seg.
- Các gen trong đoạn gen cuối cùng (ký hiệu M-Seg) được sử dụng để biểu diễn tập các gốc Lr của các cây khung bộ phận. Gen thứ i của đoạn M-Seg biểu diễn một đỉnh trong cụm thứ i.
- Cây khung được biểu diễn bằng mã Cayley [59, 60, 62].

Thuật toán 3.1: Lược đồ thuật toán C-EA

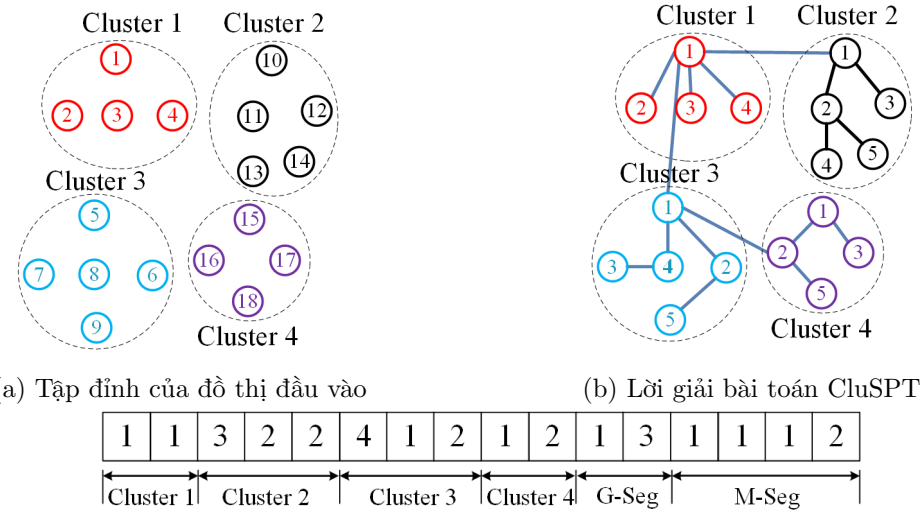
Input: Đồ thị phân cụm $G = (V, E, w, C)$; Đỉnh nguồn s;
Output: Lời giải của bài toán CluSPT;

```
1  begin
       /* Mỗi cá thể được biểu diễn là một mảng của các đỉnh và các cụm  */
2      P_0 ← Tạo ngẫu nhiên N cá thể            ▷ Chi tiết trong thuật toán 3.2;
3      foreach cá thể ind_j ∈ P_0 do
4          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;
5          Tính giá trị thích nghi (fitness) của cá thể ind_j dựa trên chi phí (cost) của lời
             giải s_j;
6      end
7      t ← 0;
8      while điều kiện dừng chưa thỏa mãn do
9          O_t ← ∅;
10         while số_cá_thể_con_được_sinh < N do
11             Chọn ngẫu nhiên hai cá thể p_a và p_b từ quần thể P_t;
12             Lai ghép, đột biến cá thể p_a và p_b tạo ra hai cá thể con o_a và o_b ▷ Chi tiết
                 xem phần 3.1.4 và phần 3.1.5;
13             O_t ← O_t ∪ {o_a, o_b};
14         end
15         foreach cá thể o_j ∈ O_t do
16             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;
17             Tính giá trị thích nghi của cá thể o_j dựa trên chi phí của lời giải s'_j;
18         end
19         R_t ← O_t ∪ P_t;
20         P_{t+1} ← Chọn N cá thể tốt nhất từ R_t;
21         t ← t + 1;
22     end
23     ind* ← cá thể tốt nhất từ P_t;
24     Tạo cây cây khung bộ phận và cây khung toàn cục của lời giải bài toán CluSPT
         s* tương ứng với cá thể ind*;
25     return s*;
26 end
```

Như vậy, một nhiễm sắc thể được chia thành $h + 2$ đoạn. Để thuận tiện cho theo dõi, $h + 2$ đoạn sẽ được đánh đánh số từ 1 tới $h + 2$. Do mã hóa các cây khung nên 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.

Ví dụ về mã hóa cá thể trong thuật toán C-EA được minh họa trong hình 3.1. Giả sử hình 3.1(b) minh họa đồ thị của cá thể, khi đó hình 3.1(c) minh họa mã hóa của đồ thị trong hình 3.1(b) của cá thể dưới dạng mã Prüfer Code. Cá thể trong hình 3.1(c) gồm 6 đoạn trong đó số đỉnh trong các cụm 1, 2, 3 và 4 lần lượt là 4, 5, 5, và 4; đoạn G-Seg gồm hai gen có giá trị là 1 và 3; đoạn M-Seg gồm bốn gen có giá trị là 1, 1, 1 và 2.



(a) Tập đỉnh của đồ thị đầu vào

(b) Lời giải bài toán CluSPT

(c) Cá thể

Hình 3.1: Ví dụ về biểu diễn cá thể trong thuật toán C-EA

Do nhãn của các đỉnh của cá thể thường không phải là nhãn của các đỉnh trong các đồ thị đầu vào, nên để tìm lại nhãn của các đỉnh trong các cụm, thuật toán C-EA đồng thời phải xây dựng hàm (gọi là *M-function*) có chức năng ánh xạ các đỉnh thuộc mỗi cụm của cá thể tới các đỉnh thuộc cụm tương ứng của đồ thị đầu vào. Ví dụ, xét cá thể như trong hình 3.1(a) và hình 3.1(b), hàm *M-function* sẽ ánh xạ các đỉnh thuộc đoạn thứ 2 của cá thể này với các đỉnh thuộc cụm 2 như sau: đỉnh 1 tương ứng với đỉnh 10, đỉnh 2 tương ứng với đỉnh 11, đỉnh 3 tương ứng với đỉnh 12, đỉnh 4 tương ứng với đỉnh 13, đỉnh 5 tương ứng với đỉnh 14. Các thông tin ánh xạ giữa các đỉnh của hàm *M-function* sẽ được sử dụng trong phương thức đánh giá cá thể.

3.1.3. Phương pháp khởi tạo cá thể

Để tạo ra cá thể hợp lệ cho một bài toán, ứng với mỗi biểu diễn khác nhau đòi hỏi cần có một phương pháp khởi tạo tương ứng. Phần này sẽ giới thiệu phương pháp tạo ra cá thể hợp lệ của bài toán CluSPT, trong đó mỗi đoạn của cá thể tương ứng là một chuỗi Cayley ngẫu nhiên. 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. Chi tiết của phương pháp tạo ngẫu nhiên cá thể được minh họa trong thuật toán 3.2.

Thuật toán 3.2: Phương pháp khởi tạo cá thể trong thuật toán C-EA

Input: Các đồ thị phân cụm $G = (V, E, C)$; Đỉnh nguồn s.
Output: Cá thể Ind.

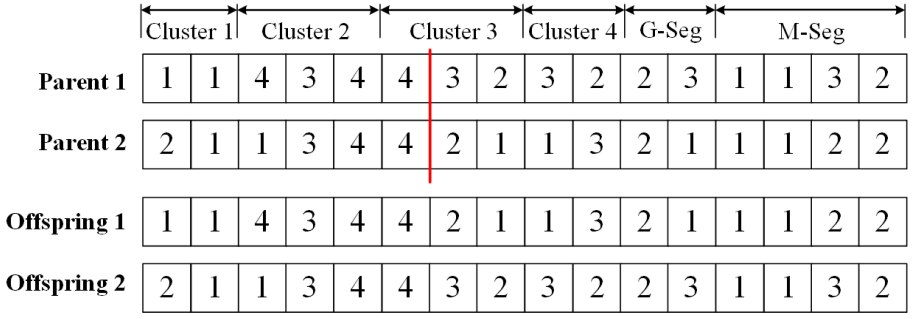
```
1  begin
2      for i ← 1 to h do
3          S_i ← Generate_Random_Cayley_Code(|C_i|);
4          Thêm chuỗi S_i vào cuối cá thể Ind;
5      end
6      S_{N+1} ← Generate_Random_Cayley_Code(h);
7      Thêm chuỗi S_{N+1} vào cuối cá thể Ind;
8      S_{N+2} ← Select_Random_Vertices_In_Clusters();
9      Thêm chuỗi S_{n+2} vào cuối cá thể Ind;
10     return Ind;
11 end
```

Trong thuật toán 3.2, phương thức Generate_Random_Cayley_Code(l) sẽ tạo ra chuỗi mã Cayley ngẫu nhiên có độ dài $l - 2$. Phương thức Select_Random_Vertices_In_Clusters() sẽ lựa chọn ngẫu nhiên một đỉnh trong mỗi cụm để tạo cạnh nối giữa các cụm.

3.1.4. Toán tử lai ghép

Điểm thuận lợi của việc sử dụng mã Cayley để mã hóa lời giải trong thuật toán EA là có thể sử dụng các toán tử tiến hóa đã có sẵn như: lai ghép một điểm cắt [8], phép lai ghép PMX [3, 4], v.v. Để khai thác lợi thế này, thuật toán C-EA sử dụng lai ghép một điểm cắt (one-point crossover operator) [4]. Phép lai ghép một điểm cắt sẽ lựa chọn ngẫu nhiên một điểm

trên nhiễm sắc thể, sau đó trao đổi hai đoạn gen của các cá thể cha mẹ để tạo ra hai cá thể con mới.



Hình 3.2: Ví dụ về toán tử lai ghép trong thuật toán C-EA

Hình 3.2 minh họa cơ chế của toán tử lai ghép trong thuật toán C-EA, trong đó, đường thẳng đứng màu đỏ là vị trí điểm lai ghép. Theo cách mã hóa được trình bày ở trên thì các chuỗi Cayley của các cây khung cây khung bộ phận và của cây khung toàn cục liền kề nhau và đoạn cuối cùng lưu thông tin các đỉnh được sử dụng để tạo lời giải thông qua kết hợp các cây khung bộ phận và cây khung toàn cục. Phép lai ghép một điểm cắt có khả năng giữ lại các đặc trưng di truyền của cha mẹ, tuy nhiên tùy vào vị trí của điểm lai ghép mà các đặc trưng được di truyền được giữ lại sẽ khác nhau. Nếu vị trí điểm lai ghép thuộc vào một chuỗi Cayley mã hóa cây khung bộ phận thì các chuỗi Cayley của cá thể con sẽ khác của cá thể cha mẹ. Nếu vị trí điểm lai ghép thuộc chuỗi Cayley của cây khung toàn cục thì cá thể con sẽ khác cá thể cha mẹ tại chuỗi Cayley này. Còn nếu vị trí điểm lai ghép thuộc đoạn M-Seg thì toán tử lai ghép sẽ tạo ra tập các điểm gốc mới. Do xác suất để vị trí điểm lai ghép thuộc đoạn gen của cây khung bộ phận, của cây khung toàn cục và của đoạn M-Seg phụ thuộc vào chiều dài của đoạn gen đó (bằng số gen của đoạn đó), nên xác suất để cá thể con có cây khung trong một cụm khác với cá thể cha mẹ phụ thuộc tỉ lệ chiều dài của đoạn gen biểu diễn cây khung của cụm đó và chiều dài của nhiễm sắc thể; còn xác suất để cá thể con có cây khung toàn cục khác với cha mẹ sẽ phụ thuộc vào tỉ lệ tổng chiều dài hai đoạn G-Seg, M-Seg và chiều dài của nhiễm sắc thể.

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.

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. Tác động của thay đổi này tới lời giải mới được tạo thành phụ thuộc nhiều vào mức độ cục bộ (*locality*) [51, 61] của mã Cayley được sử dụng để mã hóa cá thể. Nếu mã Cayley được sử dụng có tính cục bộ cao thì sự hoán đổi hai gen sẽ tạo ra cá thể mới ít thay đổi so với cá thể cha mẹ. Ngược lại, nếu mã Cayley có tính cục bộ thấp thì sự hoán đổi sẽ tạo ra cây khung mới có nhiều sự khác biệt so với cá thể cha mẹ.
- Thay đổi thứ hai chỉ tác động lên đoạn M-Seg thông qua thay đổi các gốc của cây khung bộ phận. Nếu chỉ có thay đổi thứ nhất thì cá thể con tạo ra chỉ khác cá thể mẹ ở cạnh nối trong một cụm hoặc cạnh nối giữa các cụm, không có sự khác nhau giữa các đỉnh nối giữa các cụm. Vì vậy, mục đích của thay đổi này là tạo ra cá thể con có thể có đỉnh nối giữa các cụm khác nhau.

Mã giả của toán tử đột biến được trình bày trong thuật toán 3.3.

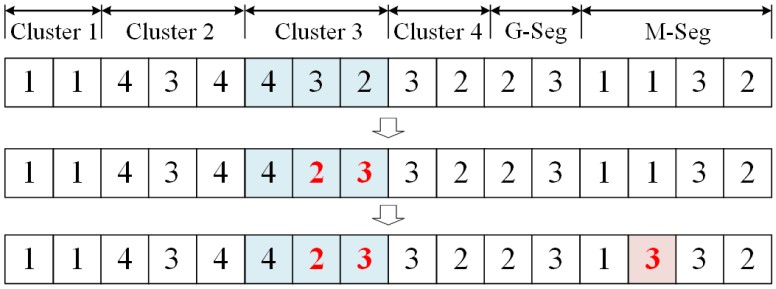
Thuật toán 3.3: Toán tử đột biến trong thuật toán C-EA

Input: Cá thể I.
Output: Cá thể mới I'.

```
1 begin
2     N ← Số các đoạn của cá thể;
      /* Tạo thay đổi trên cây khung trong cụm                    */
3     Chọn ngẫu nhiên một đoạn S_r, r ∈ [1, N − 2];
4     Đổi vị trí của hai gen được chọn ngẫu nhiên trong đoạn S_r;
      /* Tạo thay đổi trên cây khung nối giữa các cụm             */
5     v ← Chọn ngẫu nhiên một đỉnh trong đoạn S_r;
6     Thay thế gen thứ r trong đoạn M-Seg bởi đỉnh v;
7     return;
8 end
```

Xét ví dụ như trong hình 3.3, giả sử cụm thứ 3 được chọn ngẫu nhiên.

Giả sử hai gen của cụm thứ 3 (hai gen có giá trị 2 và 3 - các gen có giá trị đỏ trong hình) được trao đổi. Gen thứ 2 của đoạn M-Seg (giá trị đang là 1) được thay bằng giá trị 3 (giá trị có màu đỏ trong hình).



Hình 3.3: Ví dụ về toán tử đột biến trong thuật toán C-EA

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

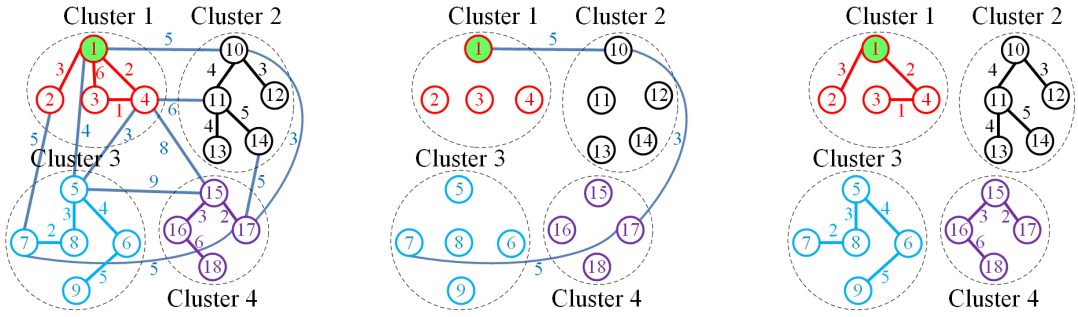
3.2.1. Cách tiếp cận

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, sau đó chỉ áp dụng thuật toán EA cho một bài toán con.

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, các cây khung của đồ thị con trong các cụm được xác định sau khi cây khung nối giữa cụm được xác định. Hay nói cách khác, 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.



(a) (b) (c)

Hình 3.4: Ví dụ về cách phân rã bài toán CluSPT thành các bài toán bé hơn

Hình 3.4 minh họa ví dụ về cách tiếp dựa trên phân rã bài toán CluSPT thành hai bài toán con. Trong đó, hình 3.4(a) là đồ thị đầu vào với đỉnh nguồn là đỉnh 1; Hình 3.4(b) minh họa đồ thị là lời giải của bài toán H-Problem; Hình 3.4(c) minh họa các cây khung của các đồ thị con trong các cụm.

Sau khi bài toán CluSPT được phân rã thành hai bài toán con, một số thuật toán như thuật toán di truyền hai mức (two-level genetic algorithm) [26, 64, 67, 68], thuật toán tiến hóa bi-level (evolutionary bi-level optimization) [19, 75],... có thể áp dụng để giải các bài toán con này. Mặc dù các thuật toán này sử dụng nhiều chiến lược khác nhau để cải thiện chất lượng lời giải tìm được, cũng như giảm hao phí tài nguyên tính toán trong khi tìm lời giải. Tuy nhiên, về bản chất, các thuật toán này vẫn xét hết tất cả các trường hợp (tổ hợp) có thể của các cạnh nối với các đỉnh nên số trường hợp phải xét đến vẫn rất lớn. Hơn nữa, một số hướng tiếp cận mã hóa lời giải của bài toán CluSPT dưới dạng một nhiễm sắc thể với số gen bằng số chiều của đồ thị đầu vào, nên cách mã hóa này sẽ gây lãng phí tài nguyên.

Để khắc phục các hạn chế trên, thuật toán được đề xuất cần có hai đặc trưng quan trọng:

1. Với mỗi lời giải của bài toán H-Problem, thuật toán phải xác định

được lời giải tương ứng của bài toán L-Problem.

2. Để xây dựng lời giải của bài toán CluSPT, thuật toán chỉ cần dựa trên các thông tin của lời giải của bài toán H-Problem.

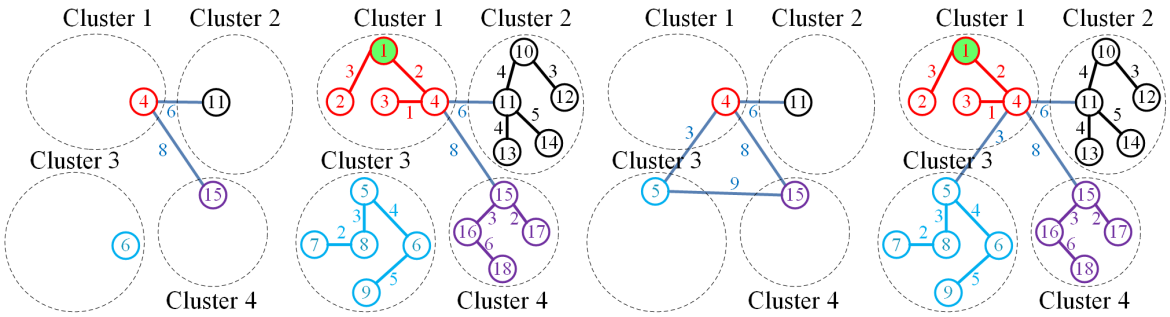
Để thỏa mãn đặc trưng đầu tiên, đối với mỗi cụm, lời giải của bài toán H-Problem cần có thông tin về đỉnh gốc của cây đường đi ngắn nhất của cụm đó. Bởi vì từ các đỉnh gốc này ta có thể sử dụng các thuật toán chính xác như thuật toán Dijstra [43, 74, 89] để tạo cây đường đi ngắn nhất cho bài toán L-Problem. Từ nhận xét trên ta thấy rằng, thuật toán chỉ cần tập trung vào các cạnh nối giữa các đỉnh của các cụm. Do đó, thuật toán chỉ mã hóa các đỉnh là các nút của các cạnh nối giữa các cụm.

Xét ví dụ trong hình 3.4, trong đó lời giải của bài toán H-Problem trong minh họa trong hình 3.4(b). Thuật toán sẽ lưu các cạnh nối giữa các đỉnh 1 với đỉnh 10, đỉnh 10 với đỉnh 17, đỉnh 17 với đỉnh 7. Khi đó, các đỉnh 1, 10, 17 và 7 sẽ được sử dụng để tìm các cây đường đi ngắn nhất của đồ thị con trong các cụm tương ứng chứa các đỉnh này (cụm 1, cụm 2, cụm 4 và cụm 3).

Với cách tiếp cận này, nếu một đồ thị con $G[C_i], i \in [1, h]$ của đồ thị G là đồ thị liên thông thì cây đường đi ngắn nhất của đồ thị con $G[C_i]$ luôn tìm được. Hay nói cách khác, nếu các đồ thị con $G[C_i], i \in [1, h]$ đều là đồ thị liên thông thì điều kiện thứ 2 của lời giải của bài toán CluSPT luôn được thỏa mãn. Mặc dù vậy, trong một số trường hợp, đồ thị cảm sinh bởi tập các đỉnh gốc của các cụm có thể không liên thông, khi đó điều kiện đầu tiên của lời giải bài toán CluSPT sẽ không được thỏa mãn.

Các ví dụ về lời giải hợp lệ và không hợp lệ được minh họa trong hình 3.5, trong đó đồ thị đầu vào được minh họa trong hình 3.4(a). Trong hình 3.5(a), do đồ thị con $G[V_r]$ của đồ thị G được cảm sinh bởi tập các đỉnh gốc $V_r = \{4, 11, 6, 15\}$ của các cụm không liên thông, nên không thể xây đựng được cây khung cho đồ thị con $G[V_r]$. Do đó, lời giải của bài toán CluSPT được xây dựng từ tập V_r là không hợp lệ bởi vì lời giải này không thỏa mãn điều kiện thứ nhất của lời giải của bài toán CluSPT. Lời giải

của bài toán CluSPT được xây dựng từ tập các đỉnh gốc V_r được minh họa trong hình 3.5(b). Hình 3.5(c) minh họa đồ thị con $G[V_{r'}]$ của đồ thị G được cảm sinh bởi tập các đỉnh gốc $V_{r'} = \{4, 11, 5, 15\}$ của các cụm. Do đồ thị con $G[V_{r'}]$ là đồ thị liên thông, nên luôn tìm cây khung của đồ thị $G[V_{r'}]$. Do đó, có thể tìm được lời giải hợp lệ của bài toán CluSPT, hình 3.5(d) minh họa một trong các lời giải hợp lệ có thể được xây dựng từ tập $V_{r'}$.



Hình 3.5: Ví dụ về lời giải hợp lệ và không hợp lệ đối với cách tiếp cận mới

Mã giả của thuật toán đề xuất được trình bày trong thuật toán 3.4. Điểm đáng chú ý trong thuật toán 3.4 là giá trị thích nghi của cá thể được tính thông qua chi phí của lời giải của bài toán CluSPT (dòng 5 và dòng 13 trong thuật toán 3.4). Điều đó có nghĩa rằng, để tính giá trị thích nghi của cá thể ind_j, trước tiên, thuật toán 3.4 sẽ xác định và tính chi phí lời giải s_j của bài toán CluSPT tương ứng với cá thể ind_j, sau đó, thuật toán 3.4 mới tính giá trị thích nghi $fit(ind_j)$ của cá thể ind_j bởi công thức được xác định trước như:

$$fit(ind_j) = \frac{1}{f(s_j)} \tag{3.2}$$

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.

Để tìm tập cạnh của lời giải của bài toán CluSPT từ tập gốc của các

Thuật toán 3.4: 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;

```
1  begin
2      P_0 ← Tạo ngẫu nhiên N cá thể của bài toán H-Problem   ▷ Mỗi cá thể là một
         mảng gồm các đỉnh gốc của các cụm;
3      foreach cá thể ind_j ∈ P_0 do
4          Xây dựng lời giải s_j của bài toán CluSPT dựa trên cá thể ind_j   ▷ Chi tiết
             trong thuật toán 3.5;
5          Tính giá trị thích nghi của ind_j dựa trên chi phí của lời giải s_j;
6      end
7      t ← 0;
8      while điều kiện dừng chưa thỏa mãn do
9          P'_t ← Tournament Selection(P_t)   ▷ Chọn các cá thể cha mẹ;
10         O_t ← Thực hiện lai ghép và đột biến(P'_t)   ▷ Chi tiết trong thuật toán 3.7 và
             3.8;
11         foreach cá thể c_j ∈ O_t do
12             Xây dựng lời giải s'_j của bài toán CluSPT dựa trên cá thể c_j   ▷ Chi tiết
                 trong thuật toán 3.5;
13             Tính giá trị thích nghi của c_j dựa trên chi phí của lời giải s'_j;
14         end
15         R_t ← O_t ∪ P_t;
16         P_{t+1} ← Chọn N cá thể tốt nhất từ R_t;
17         t ← t + 1;
18     end
19     ind* ← cá thể tốt nhất từ P_t;
20     Xây dựng lời giải s* của bài toán CluSPT dựa trên cá thể ind*   ▷ Chi tiết trong
         thuật toán 3.5;
21     return s*;
22 end
```

cụm, luận án đề xuất thuật toán xây dựng tập cạnh của lời giải (*Construct Edge Set of Solution - CESA*) dựa trên thuật toán Dijkstra [25] và đặc trưng của đồ thị được tạo từ cây đường đi ngắn nhất (cây đường đi ngắn nhất có thể được xác định sau khi thực hiện thuật toán Dijkstra). Các bước của thuật toán CESA được trình bày trong thuật toán 3.5.

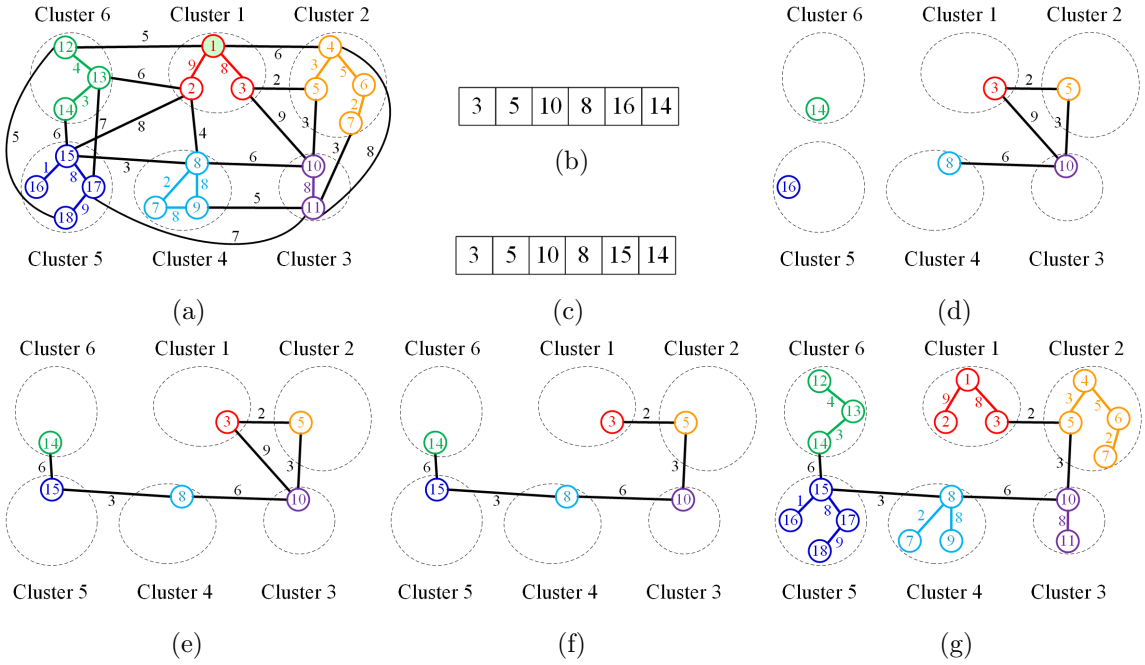
Do rằng đỉnh gốc của cụm có thể khác đỉnh nguồn s nên đối với cụm chứa đỉnh nguồn s, cây đường đi ngắn nhất luôn được xác định bằng cách áp dụng thuật toán Dijkstra với đỉnh bắt đầu là đỉnh s (dòng 11 trong thuật toán 3.5).

Thuật toán 3.5: Xây dựng tập cạnh của lời giải trong thuật toán N-EA

Input: Đồ thị phân cụm $G = (V, E, C)$;
Đỉnh nguồn s;
Cá thể $I = (r_1, r_2, \ldots, r_h)$.
Output: Một cây $T' = (V', E')$

```
1  begin
2      V' ← V;
3      S ← {r_1, r_2, ..., r_h};
4      C_m ← Xác định cụm chứa đỉnh nguồn s;
5      T ← Áp dụng thuật toán Dijkstra với đỉnh bắt đầu r_m để tìm cây đường đi ngắn
         nhất của đồ thị G[S];
6      foreach cụm C_j do
7          if C_j ≠ C_m then
8              T_j ← Áp dụng thuật toán Dijkstra với đỉnh bắt đầu r_j để tìm cây đường
                 đi ngắn nhất của đồ thị G[C_j];
9          end
10         else
11             T_m ← Áp dụng thuật toán Dijkstra với đỉnh bắt đầu s để tìm cây đường
                 đi ngắn nhất của đồ thị G[C_m];
12         end
13     end
14     E' ← (∪_{i=1}^{h} E(T_i)) ∪ E(T);
15     return T'
16 end
```

Ví dụ về thuật toán mã hóa lời giải CESA được minh họa trong hình 3.6, trong đó hình 3.6(a) minh họa đồ thị đầu vào G có 6 cụm. Do đồ thị G có 6 cụm nên số gen của nhiễm sắc thể cũng là 6. Hình 3.6(b) minh họa một cá thể được khởi tạo ngẫu nhiên, trong đó đỉnh 3 thuộc cụm 1 được chọn ngẫu nhiên làm đỉnh nguồn, đỉnh 5 thuộc cụm 2 được chọn ngẫu nhiễn là đỉnh nguồn,... Hình 3.6(d) minh họa đồ thị con của đồ thị G được cảm sinh bởi tập các đỉnh là các gen trên nhiễm sắc thể trong hình 3.6(b). Do đồ thị con (hình 3.6(d)) của đồ thị G được cảm sinh từ tập các gen trong hình 3.6(b) không liên thông nên cá thể trong hình 3.6(b) không hợp lệ. Hình 3.6(c) minh họa cá thể khác của đồ thị đầu vào G, cá thể này khác cá thể trong hình 3.6(b) tại gen thứ 5, khi đỉnh 16 được thay bằng đỉnh 15. Hình 3.6(e) minh họa đồ thị con của đồ thị G được cảm sinh bởi tập



Hình 3.6: Ví dụ minh họa biểu diễn cá thể trong thuật toán N-EA

đỉnh là các gen của nhiễm sắc thể trong hình 3.6(c). Cây đường đi ngắn nhất được tạo ra từ đồ thị con trong hình 3.6(e) sau khi thực hiện dòng lệnh 5 của thuật toán 3.5 được minh họa trong hình 3.6(f). Hình 3.6(g) minh họa một lời giải của bài toán CluSPT, lời giải này được tạo từ cá thể trong hình 3.6(c) bằng cách áp dụng thuật toán 3.5, trong đó các cây đường đi ngắn nhất của các đồ thị con của tất cả các cụm được xác định thông qua áp dụng thuật toán Dijkstra với các đỉnh bắt đầu lần lượt là 1, 5, 10, 8, 15 và 14.

3.2.4. Phương pháp khởi tạo cá thể

Phương pháp khởi tạo cá thể trong 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 $(i = 1, \ldots, h)$. Chi tiết phương pháp khởi tạo cá thể này được trình bày trong thuật toán 3.6.

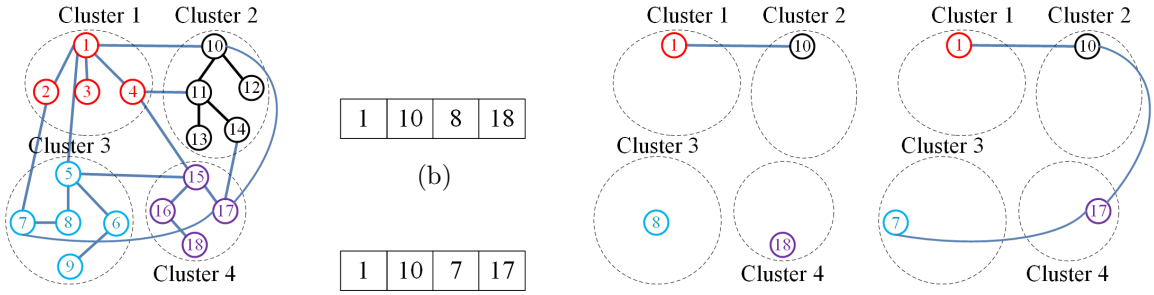
Hình 3.7 minh họa phương pháp khởi tạo cá thể cho đồ thị đầu vào gồm 5 cụm, trong đó hình 3.7(a) minh họa đồ thị đầu vào G. Hình 3.7(b) minh họa một lời giải trung gian không hợp lệ do đỉnh gốc của cụm thứ

Thuật toán 3.6: Phương thức khởi tạo cá thể trong thuật toán N-EA

Input: Đồ thị phân cụm $G = (V, E, C)$
Output: Một cá thể $Ind = (ind_1, ind_2, \ldots, ind_h)$

```
1  begin
2      repeat
3          V' ← ∅;
4          for i ← 1 to h do
5              ind_i ← Chọn ngẫu nhiên một đỉnh thuộc C_i;
6              V' ← V' ∪ {ind_i};
7          end
8      until G[V'] là đồ thị liên thông;
9      return Ind = (ind_1, ind_2, ..., ind_h)
10 end
```

3 (đỉnh 8) không được nối với bất kỳ đỉnh gốc nào khác (minh họa trong hình 3.7(d)) nên lời giải trung gian này sẽ bị loại bỏ. Hình 3.7(e) minh họa đồ thị con của đồ thị G cảm sinh bởi tập các đỉnh là các gen của nhiễm sắc thể trong hình 3.7(c). Do đồ thị con trong hình 3.7(e) liên thông, nên cá thể trong hình 3.7(c) được đưa vào quần thể.



Hình 3.7: Ví dụ về phương pháp khởi tạo cá thể với đồ thị đầu vào gồm 4 cụm

Nguyên nhân dẫn tới các cá thể trung gian không hợp lệ trong thuật toán 3.6 sẽ bị loại bỏ thay vì cố gắng tìm cách khắc phục các điểm không hợp lệ là do: mặc dù có thể có các phương pháp đơn giản để khắc phục các rằng buộc bị vi phạm trong đồ thị đầy đủ, tuy nhiên đối với đồ thị không đầy đủ các phương pháp đơn giản này có thể không hiệu quả hoặc không có khả năng khắc phục các rằng buộc bị vi phạm. Xét ví dụ trong hình 3.7(b), nếu chỉ thay thế gen thứ 3 trên nhiễm sắc thể thì cá thể vẫn

không hợp lệ do đỉnh 18 của gen thứ 4 vẫn là đỉnh cô lập. Cho nên cả gen thứ 3 và gen thứ 4 phải được đồng thời thay thế thì mới có khả năng tạo ra cá thể hợp lệ. Quá trình thay thế các gen này là không đơn giản, đặc biệt khi số lượng các cụm và số lượng các đỉnh của mỗi cụm của đồ thị đầu vào lớn.

3.2.5. Toán tử lai ghép

Toán tử lai ghép trong thuật toán N-EA (ký hiệu NCX) được xây dựng dựa trên toán tử lai ghép hai điểm cắt (two-point crossover operator) [4]. Tuy nhiên, toán tử NCX 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 NCX sẽ loại bỏ cá thể đó. Chi tiết về toán tử NCX được trình bày trong thuật toán 3.7.

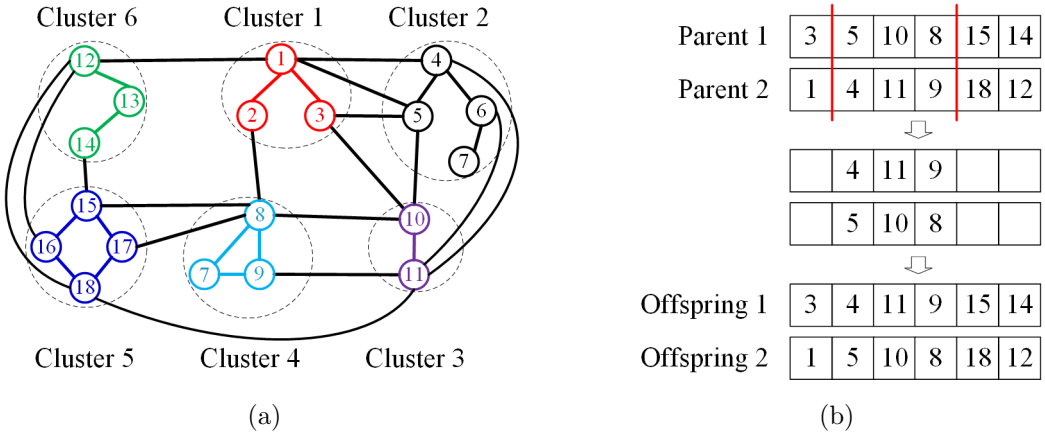
Thuật toán 3.7: Toán tử lai ghép trong thuật toán N-EA

Input: Đồ thị phân cụm $G = (V, E, C)$;
Cá thể cha mẹ: $P_i = (p_{i1}, \ldots, p_{ih}), i = 1, 2$;
Output: Các cá thể con $P_i^* = (p_{i1}^*, \ldots, p_{ih}^*), i = 1, 2$;

```
1  begin
2  |  P_i^* ← P_i, i = 1, 2;
3  |  x_{1,2} ← Chọn ngẫu nhiên từ tập {1, ..., h};
4  |  if x_1 > x_2 then swap(x_1, x_2)        ▷ Hoán đổi giá trị của hai biến ;
5  |  for j ← x_1 to x_2 do
6  |  |  swap(p*_{1j}, p*_{2j});
7  |  end
8  |  foreach cá thể con P_i^* do
9  |  |  if P_i^* là cá thể hợp lệ then
10 |  |  |  P_i^* ← Null
11 |  |  end
12 |  end
13 |  return P_i^*
14 end
```

Hình 3.8 minh họa các bước của toán tử NCX với đồ thị đầu vào G được minh họa trong hình 3.8(a). Do đồ thị G có 6 cụm nên nhiễm sắc thể có 6 gen. Hình 3.8(b) minh họa các bước của toán tử NCX với 2 đường thẳng đứng màu minh họa 2 điểm lại ghép. Toán tử NCX sẽ tạo ra 2 cá thể con. Tuy nhiên, cá thể con 1 không hợp lệ do đồ thị con của đồ thị G

cảm sinh bởi tập các đỉnh {3, 4, 11, 9, 15, 14} của cá thể con 1 không là đồ thị liên thông. Do đó, cá thể con 1 bị loại bỏ.



Hình 3.8: Ví dụ về toán tử lai ghép sử dụng trong thuật toán N-EA

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. Chi tiết về toán tử đột biến này được trình bày trong thuật toán 3.8.

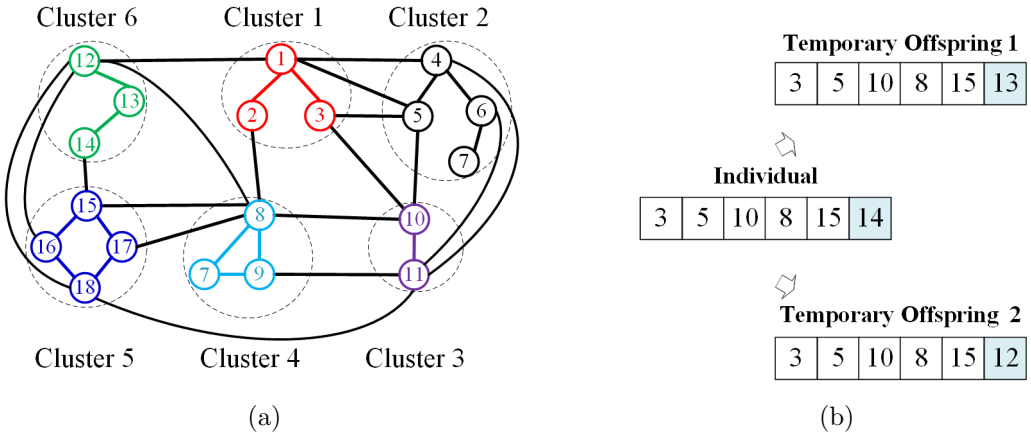
Thuật toán 3.8: Toán tử đột biến trong thuật toán N-EA

Input: Đồ thị phân cụm $G = (V, E, C)$;
Cá thể $P = (p_1, \ldots, p_h)$;
Output: Cá thể mới $P^* = (p_1^*, \ldots, p_h^*)$;

```
1  begin
2      P* ← P;
3      Chọn ngẫu nhiên một cụm C_j;
4      repeat
5          if tất cả các đỉnh trong cụm C_j đã được xét then
6              return P;
7          end
8          x ← Chọn ngẫu nhiên một đỉnh từ cụm C_j;
9          swap(x, p*_j);
10     until P* là cá thể hợp lệ;
11     return P*;
12 end
```

Hình 3.9 minh họa một ví dụ về các bước của toán tử đột biến. Giả sử đồ thị đầu vào gồm 6 cụm và 18 đỉnh được minh họa trong hình 3.9(a). Xét cá thể trong hình 3.9(b), giả sử cụm được chọn ngẫu nhiên là cụm thứ 6 (gen có màu nền xám), khi đó:

- Giả sử cá thể con trung gian thứ nhất được tạo từ các thể cha mẹ bằng cách thay thế đỉnh 14 bằng đỉnh 13. Do cá thể trung gian thứ nhất tạo được là cá thể không hợp lệ nên cá thể này bị loại bỏ.
- Giả sử cá thể con trung gian thứ hai được tạo bằng cách thay thế đỉnh 14 bằng đỉnh 12. Do cá thể con thứ hai là hợp lệ nên cá thể này là đầu ra của toán tử toán tử đột biến mới (*New mutation Operato - NMO*).



(a) (b)

Hình 3.9: Ví dụ minh họa toán tử đột biến sử dụng trong thuật toán N-EA

3.2.7. Cách đánh giá cá thể mới

Luận án giới thiệu cách tiếp cận mới trong việc tính chi phí của lời giải bài toán CluSPT. So với cách tính theo công thức trong hàm mục tiêu của bài toán CluSPT, cách tính chi phí mới này giúp giảm hao tốn tài nguyên tính toán.

Trong định nghĩa bài toán được giới thiệu ở phần 1.3.2, chi phí của lời

giải T được tính và biến đổi như sau:

$$f(T) = \sum_{u \in V} d_T(s, u) \tag{3.3}$$

$$= \sum_{i=1}^{k} \sum_{u \in C_i} d_T(s, u)$$

$$= \sum_{i=1}^{k} \sum_{u \in C_i} (d_T(s, r_i) + d_T(r_i, u))$$

$$= \sum_{i=1}^{k} \left(|C_i| * d_T(s, r_i) + \sum_{u \in C_i} d_T(r_i, u) \right) \tag{3.4}$$

Công thức (3.4) chỉ ra rằng, chi phí của lời giải T có thể được tính thông qua các giá trị của độ dài các đường đi từ đỉnh nguồn tới đỉnh gốc của các cụm và từ mỗi đỉnh gốc của một cụm tới các đỉnh khác trong cùng cụm. 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 chi phí lời giải bài toán CluSPT bằng công thức (3.4) sẽ nhỏ hơn khi tính bằng công thức (3.3).

3.3. Đánh giá thuật toán

Để phân tích hiệu quả của thuật toán C-MFEA và thuật toán N-EA luận án đã tiến hành thực nghiệm hai thuật toán trên 141 bộ dữ liệu thuộc 6 tập đồ thị metric và 75 bộ dữ liệu thuộc 3 tập đồ thị đầy đủ phi metric. Kết quả nhận được của thuật toán C-MFEA và thuật toán N-EA được so sánh với lời giải tối ưu và bốn thuật toán xấp xỉ khác.

Bên cạnh đó, luận án cũng tiến hành phân tích ảnh hưởng của các tham số của bộ dữ liệu đầu vào (số cụm và số đỉnh của đồ thị đầu vào) tới hiệu quả của thuật toán N-EA khi so sánh với các thuật toán đề xuất khác.

Để đánh giá các thuật toán đề xuất được chi tiết, trên nhiều khía cạnh và trên đa dạng các tiêu chí so sánh, đặc biệt là để thực hiện các phân tích thống kê và phân tích sự ảnh hưởng của các tham số dữ liệu đầu vào tới hiệu quả so sánh giữa nhiều thuật toán. Vì vậy, luận án phân tích kết

quả các thuật toán, cũng như mô tả thực nghiệm chi tiết tại chương 5.

3.4. Kết luận chương

Chương này trình bày hai đề xuất áp dụng thuật toán C-EA và N-EA để giải bài toán CluSPT.

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. Tuy nhiên thuật toán C-EA vẫn có những ưu điểm nhất định 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 như các bài toán bài toán cây khung phân cụm có chi phí định tuyến nhỏ nhất (*Minimum Routing Cost Clustered Tree Problem - CluMRCT*) và bài toán cây khung phân cụm với chi phí định tuyến liên cụm nhỏ nhất (*Minimum Inter-cluster Routing Cost Clustered Tree Problem - InterCluMRCT*).

Để cải thiện chất lượng lời giải tìm được, luận án đề xuất thuật toán N-EA. 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, trong 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 một số 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 và đỉnh nối với các cụm khác của một cụm cũng chính là đỉnh gốc cây khung tìm kiếm được bằng thuật toán Dijkstra. Vì vậy, chất lượng lời giải của thuật toán N-EA trong một số trường hợp vẫn chưa tốt. Các toán tử tiến hóa sử dụng trong thuật toán N-EA chỉ sử được để giải bài toán CluSPT.

Chương 4

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

Phần này trình bày hướng tiếp cận dựa trên thuật toán MFEA để giải bài toán CluSPT. Khác với thuật toán EA cơ bản, để áp dụng thuật toán MFEA vào giải các bài toán, ngoài việc cần xác định cách mã hóa lời giải, toán tử lai ghép, toán tử đột biến, còn cần phải xác định phương pháp giải mã để chuyển cá thể trong không gian tìm kiếm chung (*Unified Search Space - USS*) về thành lời giải của từng bài toán trong mỗi tác vụ. Bên cạnh đó, do thuật toán MFEA cần phải mã hóa lời giải của nhiều bài toán vào một cá thể nên cách thức mã hóa lời giải trong thuật toán MFEA cũng khác trong thuật toán EA cơ bản. Cũng do mã hóa lời giải của nhiều tác vụ nên các toán tử lai ghép, đột biến đã được giới thiệu trong thuật toán EA có thể không sử dụng được trong thuật toán MFEA.

Thuật toán MFEA được trình bày trong chương này (ký hiệu là G-MFEA) gồm có hai tác vụ: nhiệm vụ của tác vụ thứ nhất là xác định lời giải hợp lệ của bài toán CluSPT, trong khi nhiệm vụ của tác vụ thứ hai là cải thiện chất lượng lời giải tìm được trong tác vụ thứ nhất thông qua cơ chế trao đổi vật chất di truyền tiềm ẩn (*implicit genetic transfer*) giữa các tác vụ. 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.

Các toán tử tiến hóa được đề xuất trong thuật toán G-MFEA sẽ tác động lên lời giải trong hai giai đoạn: giai đoạn đầu sẽ bảo đảm rằng cá thể nhận được sẽ là một cây khung; giai đoạn sau sẽ bảo đảm rằng mỗi đồ thị con trong một cụm sẽ là một cây khung. Với cách tiếp cận này, lời giải được tạo ra bởi các toán tử này luôn được bảo đảm là lời giải hợp lệ của bài toán CluSPT.

Điểm khác biệt lớn nhất giữa thuật toán G-MFEA và các nghiên cứu khác sử dụng thuật toán MFEA để giải bài toán CluSPT, cũng như các

nghiên cứu về áp dụng thuật toán MFEA khác trước đây là: trong các nghiên cứu trước đây mỗi tác vụ sẽ tìm lời giải của một bài toán khác nhau (khác nhau về phát biểu bài toán hoặc số chiều của dữ liệu đầu vào). Trong khi thuật toán G-MFEA sẽ giải một bài toán bằng cách chia bài toán vào hai tác vụ: tác vụ thứ nhất sẽ tập trung vào giải bài toán ban đầu, còn tác vụ thứ sẽ giải một bài toán mới được phân rã (*decompose*) từ bài toán ban đầu. Thoạt nhìn sẽ thấy tác vụ thứ hai đóng vài trò giống như các thuật toán tìm kiếm cục bộ. Tuy nhiên, khác với thuật toán tìm kiếm cục bộ, quá trình trao đổi vật chất di truyền được liên tục thực hiện thông qua các phương thức và toán tử tiến hóa trong thuật toán G-MFEA.

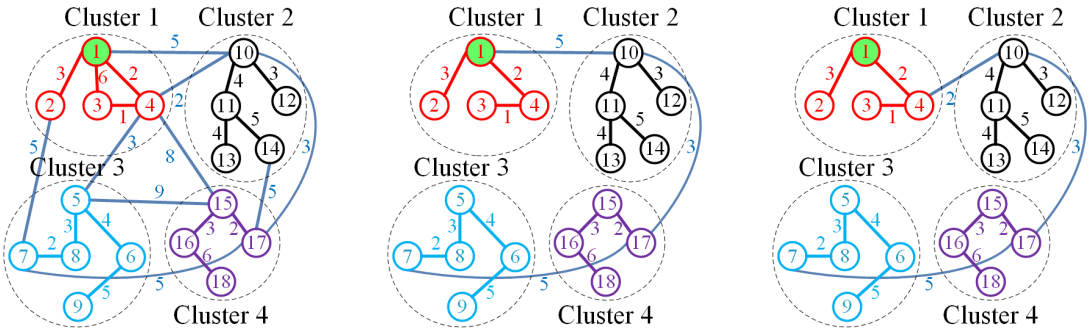
Thuật toán được trình bày trong chương này được công bố trong công trình [V].

4.1. Ý tưởng đề xuất thuật toán G-MFEA

Thuật toán N-EA được trình bày trong phần 3.2.1 đã tìm lời giải bài toán CluSPT bằng cách chia bài toán ban đầu thành hai bài toán nhỏ hơn: H-Problem và L-Problem. Mặc dù thuật toán N-EA giúp cải thiện chất lượng lời giải tìm được so với một số hướng tiếp cận khác, song hướng tiếp cận này vẫn có một số hạn chế như:

- Do một cụm được nối với các cụm khác thông qua chỉ một đỉnh duy nhất nên trong một số trường hợp, lời giải tìm được không tốt. Xét ví dụ như trong hình 4.1 với đồ thị đầu vào gồm 4 cụm, 18 đỉnh và đỉnh nguồn là đỉnh 1 (hình 4.1(a)). Thuật toán N-EA sẽ tìm được lời giải như trong hình 4.1(b). Nếu cạnh (1, 10) của lời giải trong hình 4.1(b) được thay thế bởi cạnh (4, 10) thì lời giải mới nhận được (như trong hình 4.1(c)) sẽ tốt hơn lời giải ban đầu.
- 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.
- Khi đồ thị đầu vào là đồ thị thưa (*sparse graph*), 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.

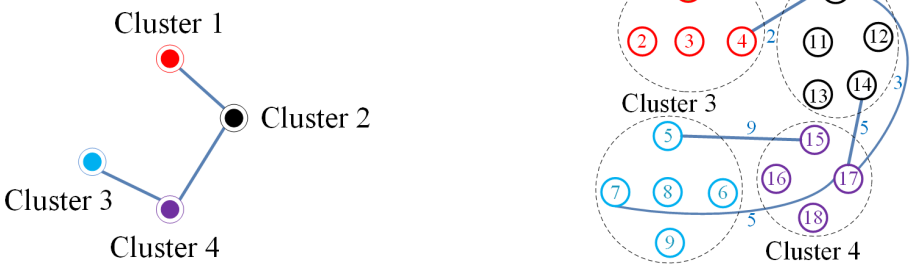


Hình 4.1: Ví dụ về các hạn chế của thuật toán N-EA

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. Mỗi cá thể trong thuật toán G-MFEA sẽ một cây khung gồm các thông tin về liên kết giữa các cụm, thông tin về các đỉnh trong mỗi cụm có cạnh nối tới các cụm khác và thông tin về đỉnh gốc của mỗi cụm. 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.

Tuy nhiên, để thực sự cải thiện chất lượng lời giải tìm được, cách mã hóa mới này cần phải xác định được cạnh tốt nhất nối giữa hai cụm trong lời giải của bài toán H-Problem khi có nhiều hơn một cạnh nối giữa hai cụm này. Xét ví dụ với đồ thị đầu vào và lời giải của bài toán H-Problem lần lượt được minh họa trong hình 4.1(a) và hình 4.2(a). Hình 4.2(b) minh họa tập cạnh nối giữa các cụm tương ứng với đồ thị đầu vào được minh họa trong hình 4.1(a) và lời giải bài toán H-Problem được minh họa trong hình 4.2(a). Trong hình 4.2(b), cả hai cạnh (5, 15) và (7, 17) đều có thể được chọn làm cạnh nối giữa cụm 3 và cụm 4. Do đó, thuật toán HB-RGA [77] (trình bày trong phần 2.2) được áp dụng để tìm lời giải tốt nhất

của bài toán CluSPT từ lời giải của bài toán H-Problem.



(a) Lời giải bài toán H-Problem (b) Tập cạnh nối giữa các cụm

Hình 4.2: Ví dụ về hạn chế của biểu diễn lời giải trong thuật toán N-EA

Từ các nhận xét trên, thuật toán G-MFEA gồm hai tác vụ: một tác vụ hướng tới việc khai phá (*exploration*) không gian tìm kiếm USS thông qua sử dụng thuật toán GA để xây dựng lời giải của bài toán H-Problem, trong khi tác vụ còn lại sẽ sử dụng thuật toán HB-RGA để khai thác (*exploitation*) không gian tìm kiếm USS.

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:
  - Thuật toán thứ nhất sẽ xây dựng lời giải bài toán CluSPT bằng cách sử dụng thuật toán Dijkstra để tìm cây khung nhỏ nhất trong mỗi cụm.
  - Thuật toán thứ hai sẽ xây dựng lời giải bài toán CluSPT từ lời giải bài toán H-Problem thông qua sử dụng thuật toán HB-RGA.

Thuật toán 4.1 minh họa các bước chính của thuật toán G-MFEA. Điểm chú ý trong thuật toán 4.1 là mỗi lời giải của bài toán CluSPT được xây dựng từ một cá thể trong không gian USS bằng thuật toán dựa trên thuật

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

1 **begin**
2 $t \leftarrow 0$;
3 $P_t \leftarrow$ Khởi tạo ngẫu nhiên N cá thể của bài toán H-Problem;
4 **foreach** *cá thể* $ind_j \in P_t$ **do**
5 Gán ngẫu nhiên chỉ số kỹ năng phù hợp nhất τ_j cho cá thể;
6 Tạo lời giải s_j của bài toán CluSPT dựa trên ind_j và τ_j ▷ Chi tiết trong thuật toán 3.5;
7 Tính chi phí đối với mỗi tác vụ của cá thể ind_j dựa trên chi phí của lời giải s_j;
8 Tính xếp hạng đối với mỗi tác vụ, giá trị thích nghi vô hướng của cá thể ind_j;
9 **end**
10 **while** *điều kiện dừng chưa thỏa mãn* **do**
11 $P'_t \leftarrow$ Tournament Selection(P_t) ▷ Chọn các cá thể cha mẹ;
12 $O_t \leftarrow$ Thực hiện lai ghép và đột biến (P'_t) ▷ Chi tiết trong thuật toán 3.7 và 3.8;
13 **foreach** *cá thể* $c_j \in O_t$ **do**
14 Tạo lời giải s'_j của bài toán CluSPT dựa trên c_j và chỉ số kỹ năng phù hợp nhất của c_j ▷ Chi tiết trong thuật toán 3.5;
15 Đánh giá chi phí đối với mỗi tác vụ của c_j dựa trên chi phí của lời giải s'_j;
16 **end**
17 $R_t \leftarrow O_t \cup P_t$;
18 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;
19 $P_{t+1} \leftarrow$ Chọn N cá thể tốt nhất trong tập R_t;
20 $t \leftarrow t + 1$;
21 **end**
22 **return** Lời giải tốt hơn của bài toán CluSPT trong hai tác vụ;
23 **end**

toán HB-RGA hoặc bằng thuật toán dựa trên thuật toán Dijkstra. Việc sử dụng thuật toán nào để xây dựng lời giải phụ thuộc vào chỉ số kỹ năng phù hợp nhất của cá thể. Điểm chú ý khác trong thuật toán 4.1 là chi phí đối với mỗi tác vụ của cá thể trong không gian USS được tính dựa vào giá trị hàm mục tiêu của lời giải bài toán CluSPT tương ứng với chỉ số kỹ năng phù hợp nhất (tại dòng 7 và dòng 15 trong thuật toán 4.1). Điều đó có nghĩa là, để tính chi phí đối với mỗi tác vụ của một cá thể ind_j (là lời giải bài toán H-Problem), thuật toán G-MFEA trước tiên sẽ xây dựng và

tính giá trị hàm mục tiêu của lời giải bài toán CluSPT s_j tương ứng với chỉ số kỹ năng phù hợp nhất τ_i, sau đó mới xác định chi phí đối với mỗi tác vụ của cá thể ind_j. Do thuật toán G-MFEA được đề xuất theo hướng: các cá thể được tạo ra không vi phạm các rằng buộc của bài toán CluSPT nên để đơn giản, luận án tính:

$$\Psi_i^j = f(s_j) \tag{4.1}$$

trong đó $f(s_j)$ là hàm mục tiêu của bài toán CluSPT tính theo công thức (1.1).

Do cả hai tác vụ của thuật toán G-MFEA đều trả về lời giải của bài toán CluSPT nên lời giải tốt hơn sẽ được chọn làm đầu ra của thuật toán 4.1.

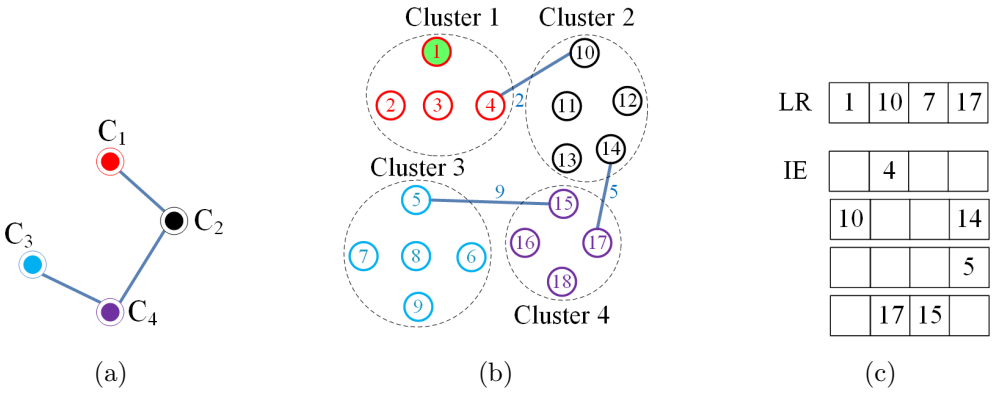
Phương pháp mã hóa trong thuật toán G-MFEA sẽ mã hóa một cá thể gồm ba thành phần: thành phần đầu tiên được sử dụng để tạo các cạnh nối giữa các cụm, trong khi thành phần thứ hai và thứ ba chủ yếu được sử dụng để xây dựng cây khung trong mỗi cụm.

4.3. Biểu diễn cá thể

Để xây dựng lời giải bài toán CluSPT từ lời giải bài toán H-Problem, một cá thể trong không gian USS cần phải lưu các thông tin: thông tin về các cạnh nối giữa các cụm (lưu vào thuộc tính ES); thông tin về các 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). Do đó, cấu trúc của một cá trong không gian USS sẽ gồm các thành phần:

- Một tập các cạnh ES sẽ lưu thông tin về tất cả các cạnh nối giữa các cụm với nhau.
- Một mảng hai chiều IE sẽ lưu các đỉnh, các đỉnh này được được sử dụng để xác định cạnh nối giữa hai cụm. Nghĩa là, giá trị IE[i][j] là một đỉnh thuộc cụm i nối với cụm j.
- Một mảng LR lưu các đỉnh gốc của tất cả các cụm.

Hình 4.3 minh họa ví dụ về mã hóa trong thuật toán G-MFEA trong đó



(a) (b) (c)

Hình 4.3: Ví dụ về mã hóa lời giải trong thuật toán G-MFEA

cụm i được ký hiệu là C_i. Xét trường hợp lời giải của bài toán H-Problem như trong hình 4.3(a) thì tập ES của cá thể trong không gian USS là ES = $\{(C_1, C_2), (C_2, C_4), (C_4, C_3)\}$. Nếu các đỉnh được sử dụng để xác định cạnh nối giữa các cụm được minh họa như trong hình 4.3(b) thì mảng IE sẽ là IE[1][2] = 4; IE[2][1] = 10; IE[2][4]=14; IE[4][2]=17; IE[3][4]=5; IE[4][3]=15. Nếu tập đỉnh gốc của các cụm lần lượt là {1, 10, 17} thì LR[1] = 1; LR[2] = 10; LR[3] = 7; LR[4] = 17. Các thuộc tính IE và LR trong ví dụ này được minh họa trong hình 4.3(b).

Giả sử mỗi con trỏ đòi hỏi a đơn vị bộ nhớ và mỗi đỉnh của đồ thị đòi hỏi b đơn vị bộ nhớ. Với cách mã hóa cá thể được đề xuất, yêu cầu bộ nhớ cần thiết để thuật toán G-MFEA lưu trữ một cá thể là:

- Do tập ES lưu thông tin của $h-1$ cạnh nối giữa h đỉnh của lời giải bài toán H-Problem nên khi biểu diễn đồ thị bởi ma trận kề đòi hỏi $ah + 2b(h-1)$ đơn vị bộ nhớ.
- Mảng hai chiều IE đòi hỏi bh^2 đơn vị bộ nhớ.
- Mảng LR đòi hỏi bh đơn vị bộ nhớ.

Như vậy, cần $ah + 2b(h-1) + bh^2 + bh = bh^2 + (3b+a)h - 2b$ đơn vị bộ nhớ để lưu trữ môt cá thể trong thuật toán G-MFEA.

Điểm cần lưu ý trong cách mã hóa này là đỉnh gốc của một cụm có thể khác với đỉnh có cạnh nối từ cụm này tới một cụm khác.

Để tạo lời giải bài toán CluSPT từ cá thể trong không gian USS, thuật

toán G-MFEA hiệu chỉnh phương thức xây dựng tập cạnh (thuật toán 3.5) được giới thiệu trong phần 3.2.3. Các bước của thuật toán mới này được trình bày trong thuật toán 4.2.

Thuật toán 4.2: Phương pháp xây dựng cạnh cho lời giải bài toán CluSPT từ cá thể trong không gian USS

Input: Đồ thị phân cụm $G = (V, E, C)$; Đỉnh nguồn s;
Cá thể *I=(ES, IE, LR)*
Output: Cây khung $T' = (V', E')$

```
1  begin
2  |  V' ← {1, 2, ..., |V|};
3  |  E' ← ∅;
4  |  foreach cạnh e = (C_i, C_j) ∈ ES do
5  |  |  E' ← E' ∪ e' = (IE[i][j], IE[j][i]);
6  |  end
7  |  foreach cụm C_j do
8  |  |  T_j ← Sử dụng thuật toán Dijkstra để xác định cây đường đi ngắn nhất của
   |  |   G[C_j] với đỉnh bắt đầu LR[j];
9  |  end
10 |  E' ← (∪_{i=1}^{h} E(T_i)) ∪ E';
11 |  return T'
12 end
```

4.4. Phương pháp khởi tạo cá thể

Mỗi cá thể trong không gian USS được khởi tạo ngẫu nhiên dựa trên một trong các thuật toán tạo cây khung ngẫu nhiên được giới thiệu trong nghiên cứu [71] (ký hiệu chung là thuật toán M4CRST). Cá thể được khởi tạo ngẫu nhiên thông qua ba bước chính như sau:

1. Tạo đồ thị $G - Graph$ G' = (V', E') dựa trên đồ thị đầu vào G = (V, E).
2. Áp dụng thuật toán M4CRST để tạo cây khung ngẫu nhiên 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.

Chi tiết các bước của phương thức khởi tạo cá thể được trình bày thuật toán 4.3.

Thuật toán 4.3: Phương pháp tạo cá thể ngẫu nhiên trong thuật toán G-MFEA

Input: Đồ thị phân cụm $G = (V, E, C)$; Đỉnh nguồn s
Output: Cá thể *I=(ES, IE, LR)*

```
1  begin
2      Tạo đồ thị G' = (V', E') từ đồ thị đầu vào G;
3      T' ← M4CRST(V', E');
       /* Xác định các đỉnh trong mỗi cụm có cạnh nối với các đỉnh thuộc
          cụm khác                                                    */
4      foreach cạnh e=(v'_i, v'_j) ∈ T' do
5          ES ← ES ∪ e;
6          e = (α, β) ← Chọn ngẫu nhiên cạnh nối α ∈ C_i với đỉnh β ∈ C_j;
7          IE[i][j] ← α;
8          IE[j][i] ← β;
9      end
       /* Xác định gốc của các cụm                                    */
10     C_m ← Xác định cụm chứa đỉnh nguồn s;
11     foreach cụm C_i do
12         if C_i ≠ C_m then
13             LR[i] ← Chọn ngẫu nhiên một đỉnh thuộc cụm C_i;
14         end
15         else
16             LR[i] ← s;
17         end
18     end
19     return I=(ES, IE, LR);
20 end
```

4.5. Toán tử lai ghép

Toán tử lai ghép sử dụng phương pháp tạo cây khung ngẫu nhiên M4CRST để tạo tập cạnh cho các cá thể trong không gian USS. Tuy nhiên để xây dựng được lời giải bài toán CluSPT, cá thể trong không gian USS phải lưu trữ thông tin về đỉnh gốc của mỗi cụm và các đỉnh trong mỗi cụm có cạnh nối ra cụm khác, nên 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 ở trên một cá thể cha mẹ thì thuộc tính IE, LR của các thể con được kế thừa từ cá thể cha mẹ đó.

Các bước của toán tử lai ghép được trình bày trong thuật toán 4.4.

Hình 4.4 minh họa các bước cơ bản của toán tử lai ghép, trong đó đồ thị đầu vào được minh họa trong hình 4.4(a); hai cá thể cha mẹ được minh họa trong hình 4.4(b) và hình 4.4(c). Giả sử tập cạnh của cá thể con sau khi áp dụng phương thức M4CRST (tại dòng 3 trong thuật toán 4.4) được minh họa trong hình 4.4(d). Các thuộc tính ES, LR của các thể con được xác định như sau:

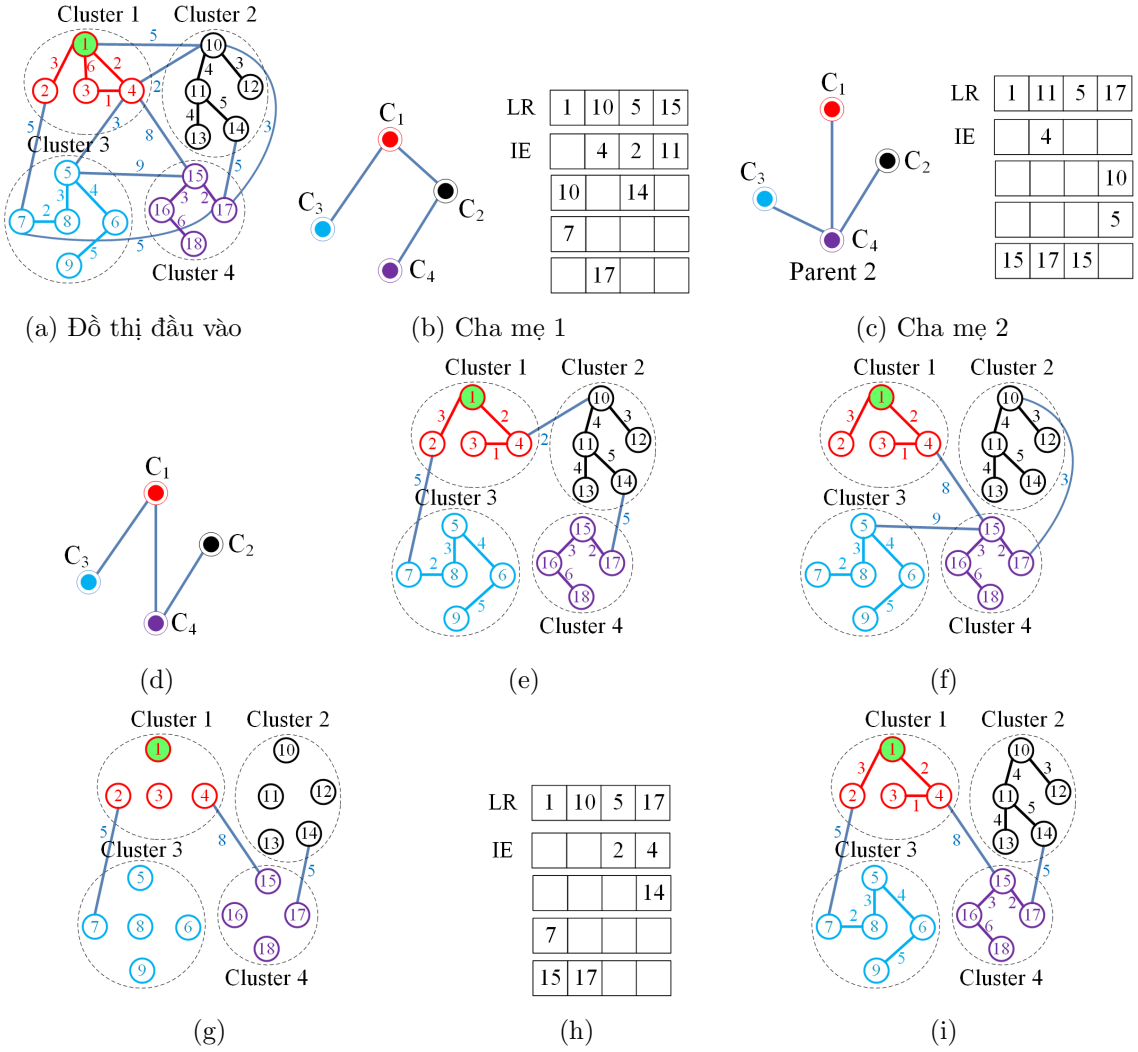
Đối với thuộc tính ES: Giả sử lời giải của bài toán CluSPT xây dựng từ hai cá thể cha mẹ trong hình 4.4(b) và hình 4.4(c) được minh họa lần lượt trong hình 4.4(e) và hình 4.4(f). Xét cá thể con trong hình 4.4(d), do chỉ có một cạnh (2, 7) nối cụm 1 và cụm 3 trong cá thể cha mẹ 1 nên cá thể con sẽ kế thừa cạnh này từ cá thể cha mẹ 1. Tương tự với cạnh nối cụm 2 và cụm 4, cá thể con sẽ kế thừa cạnh (4, 15) từ cá thể cha mẹ 2. Đối với cạnh nối cụm 2 và cụm 4, do cả hai cá thể cha mẹ đều có cạnh nối giữa hai cụm này (cạnh (14, 17) của cha mẹ 1 và cạnh (10, 17) của cha mẹ 2) nên cá thể con sẽ chọn ngẫu nhiên một cạnh từ các cá thể cha mẹ để kế thừa. Hình 4.4(g) minh họa cá thể con sau khi cạnh (14, 17) của cá thể cha mẹ 1 được chọn để kế thừa.

Thuật toán 4.4: Toán tử lai ghép trong thuật toán G-MFEA

Input: Đồ thị phân cụm $G = (V, E, C)$;
Hai cá thể cha mẹ $P_i = (ES_i, IE_i, LR_i), i = 1, 2$;
Output: Cá thể con $O = (ES_o, IE_o, LR_o)$;

begin
  /* Tìm cây khung của đồ thị G-Graph */
  $V' \leftarrow \{v'_i\}_{i=1,\ldots,h}$ where a vertex v'_i represents a cluster C_i;
  $T' \leftarrow$ M4CRST(V', $ES_1 \cup ES_2$);
  $ES_o \leftarrow \emptyset$;
  foreach *cạnh* $e=(v'_m, v'_j) \in T'$ **do**
    $ES_o \leftarrow ES_o \cup e$;
    if *cạnh e có trong cả hai cá thể* P_1 *và* P_2 **then**
      Tạo ngẫu nhiên số *rnd* nhận giá trị trong khoảng (0, 1);
      if *rnd* < 0.5 **then**
        $IE_o[m, j] \leftarrow IE_1[m, j]$;
        if $LR_o[m]$ *chưa được gán* **then**
          $LR_o[m] \leftarrow LR_1[m]$;
        end
        if $LR_o[j]$ *chưa được gán* **then**
          $LR_o[j] \leftarrow LR_1[j]$;
        end
      end
      else
        $IE_o[m, j] \leftarrow IE_2[m, j]$;
        if $LR_o[m]$ *chưa được gán* **then**
          $LR_o[m] \leftarrow LR_2[m]$;
        end
        if $LR_o[j]$ *chưa được gán* **then**
          $LR_o[j] \leftarrow LR_2[j]$;
        end
      end
    end
    else
      if *cạnh e thuộc cá thể* P_i ($i = 1$ *hoặc* $i = 2$) **then**
        $IE_o[m, j] \leftarrow IE_i[m, j]$;
        if $LR_o[m]$ *chưa được gán* **then**
          $LR_o[m] \leftarrow LR_i[m]$;
        end
        if $LR_o[j]$ *chưa được gán* **then**
          $LR_o[j] \leftarrow LR_i[j]$;
        end
      end
    end
  end
end
return $I=(ES_o, IE_o, LR_o)$;

Đối với thuộc tính LR: Các đỉnh gốc của các cụm trong các thể con được chọn từ các đỉnh gốc của các cụm của một trong hai cá thể cha mẹ. Việc cá thể con kế thừa đỉnh gốc của cụm từ cá thể cha mẹ nào phụ thuộc vào cạnh chứa đỉnh gốc có ở cả hai cá thể cha mẹ hay chỉ có ở một cá thể cha mẹ. Giả sử đỉnh gốc của cụm 1 và cụm 2 được kế thừa từ cha mẹ 1, còn các đỉnh gốc của cụm khác được kế thừa từ cha mẹ 2. Chi tiết về các thuộc tính IE và LR của cá thể con được minh họa trong hình 4.4(h). Lời giải bài toán CluSPT được tạo thành từ cá thể con của phép lai ghép được minh họa trong hình 4.4(i).



Hình 4.4: Ví dụ minh họa các bước của toán tử lai ghép trong thuật toán G-MFEA

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 nhật thông tin các thuộc tính IE và LR của cá thể mới.

Các bước của toán tử đột biến sử dụng trong thuật toán G-MFEA được trình bày trong thuật toán 4.5.

Thuật toán 4.5: Toán tử đột biến trong thuật toán G-MFEA

Input: Đồ thị phân cụm $G = (V, E, C)$;
Cá thể cha mẹ: $I = (ES, IE, LR)$;
Output: Cá thể con $I^* = (ES^*, IE^*, LR^*)$;

begin
  Tạo đồ thị $G' = (V', E')$ từ đồ thị G;
  $e' = (v'_i, v'_j) \leftarrow$ Chọn ngẫu nhiên một cạnh từ tập $E' \backslash ES$;
  $ES^* \leftarrow ES$;
  Thêm cạnh e' vào E^* để tạo thành chu trình;
  Xác định chu trình Δ trong E^*;
  Chọn ngẫu nhiên cạnh $e^* = (v'_m, v'_n) \in \Delta$ sao cho $e^* \neq e'$;
  Xóa cạnh e^* từ tập E^*;
  /* Cập nhật đỉnh gốc cục bộ của các cụm và cạnh nối giữa các đỉnh của các cụm */
  $u = (u_h, u_p) \leftarrow$ Chọn ngẫu nhiên từ tập E một cạnh nối giữa đỉnh $u_h \in C_i$ và đỉnh $u_p \in C_j$;
  /* Cập nhật thuộc tính IE của cá thể con đối với cạnh bị xóa */
  $IE^*[m][n] \leftarrow null$;
  $IE^*[n][m] \leftarrow null$;
  /* Cập nhật thuộc tính IE của cá thể con đối với cạnh mới thêm */
  $IE^*[i][j] \leftarrow u_h$;
  $IE^*[j][i] \leftarrow u_p$;
  /* Cập nhật đỉnh gốc cục bộ của các cụm C_m và C_n */
  if *(đỉnh gốc cục bộ của cụm $C_t (t = m, n)$ thuộc cạnh bị xóa e^*) và (đỉnh gốc cục bộ của cụm C_t không còn cạnh nào nối tới một cụm khác)* **then**
    $LR^*[t] \leftarrow$ Chọn ngẫu nhiên từ cụm C_t một đỉnh có cạnh nối với một cụm khác;
  end
  return (ES^*, IE^*, LR^*);
end

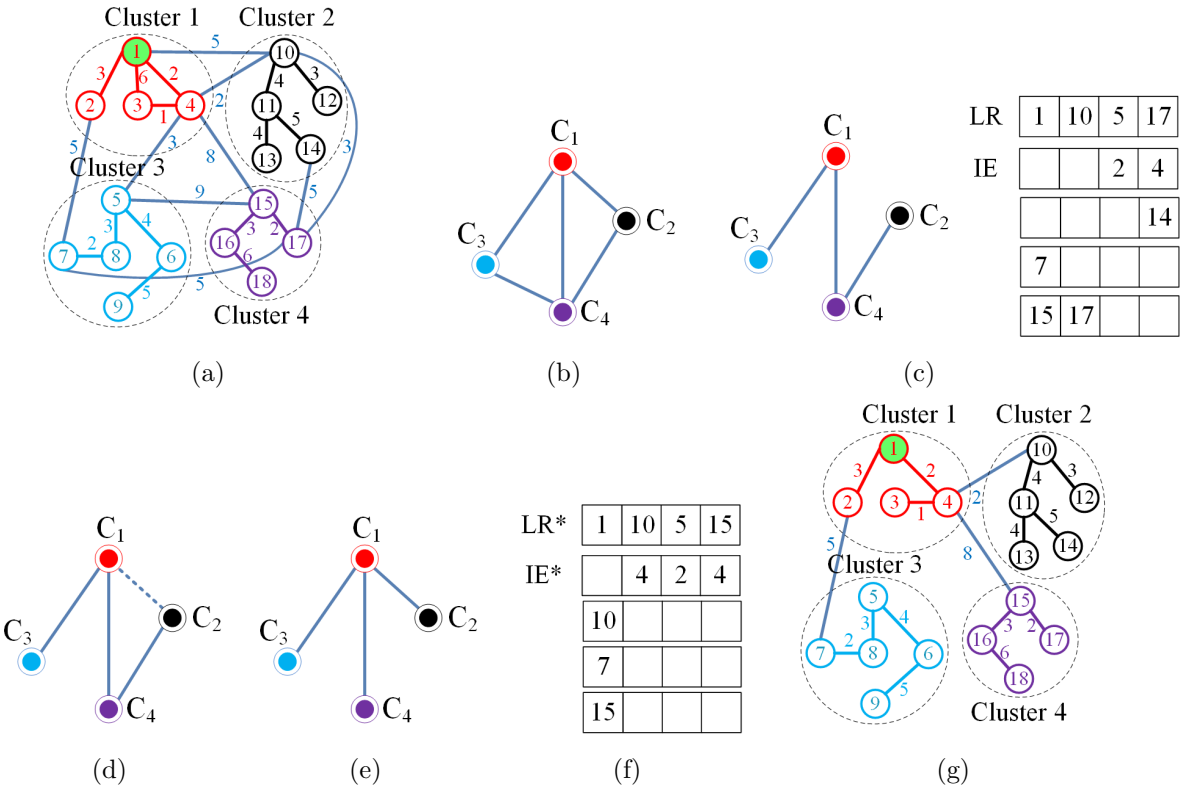
Ví dụ về các bước chính của thuật toán 4.5 được minh họa trong

hình 4.5. Xét đồ thị đầu vào được minh họa như trong hình 4.5(a). Đồ thị $G - Graph$ được xây dựng từ đồ thị đầu vào được minh họa trong hình 4.5(b), cá thể trong không gian USS được minh họa trong hình 4.5(c). Hình 4.5(d) minh họa cá thể con nhận được sau khi kết thúc bước thứ nhất của toán tử đột biến, trong đó (C_1, C_2) là cạnh được giả sử mới thêm vào cá thể để tạo thành chu trình (C_1, C_2, C_4, C_1); còn (C_2, C_4) là cạnh được xóa bỏ từ chu trình (C_1, C_2, C_4, C_1) để tạo thành cá thể mới. Trong bước thứ hai, toán tử đột biến sẽ cập nhật các thuộc tính IE và LR của cá thể con mới tạo ra trong bước thứ nhất. Do cạnh (4, 10) được chọn để nối giữa hai cụm C_1 và C_2 nên tập IE* của cá thể con được cập nhật như trong hình 4.5(f). Sau khi xóa cạnh (14, 17) nối giữa hai cụm C_2 và C_4 ở trong bước thứ nhất, đỉnh 17 trong cụm 4 không thuộc bất cứ cạnh nào nối cụm 4 tới các cụm khác. Do đó, một đỉnh khác có cạnh nối ra cụm khác được chọn làm đỉnh gốc của cụm 4. Trong cụm 4 chỉ có duy nhất đỉnh 15 thỏa mãn điều kiện nên đỉnh này được làm đỉnh gốc của cụm. Hình 4.5(g) minh họa lời giải bài toán CluSPT được xây dựng từ các thể con mới.

4.7. Phương pháp giải mã

Phương pháp giải mã giúp tìm lời giải của các tác vụ từ các thể trong không gian USS của thuật toán G-MFEA gồm các bước:

- Đối với tác vụ thứ nhất, lời giải bài toán CluSPT được xây dựng từ cá thể trong không gian USS bằng cách sử dụng các thuộc tính LR và IE. Trong đó, tập cạnh nối các đỉnh của hai cụm khác nhau được tạo dựa trên thông in từ tập IE. Trong khi đó cây khung của mỗi cụm được tạo bằng cách áp dụng thuật toán Dijkstra với đỉnh bắt đầu là đỉnh gốc của cụm.
- Phương pháp giải mã của tác vụ thứ hai được xây dựng dựa trên thuật toán HB-RGA. Thuật toán HB-RGA được sử dụng để tìm cạnh tốt nhất nối giữa các đỉnh của hai cụm khác nhau. Tuy nhiên, với mỗi cá thể khác nhau trong không gian USS, đồ thị đầu vào của thuật toán



Hình 4.5: Ví dụ minh họa các bước của toán tử đột biến

HB-RGA sử dụng trong thuật toán G-MFEA sẽ khác nhau. Với mỗi cá thể *IND*, đồ thị đầu vào G_{IND} của thuật toán HB-RGA được tạo từ đồ thị đầu vào G và tập cạnh ES của *IND* như sau:

- Tập đỉnh của đồ thị G_{IND} được giữ giống như tập đỉnh của đồ thị G.
- Các cạnh trong mỗi cụm của đồ thị G_{IND} được giữ giống các cạnh trong các cụm của đồ thị G.
- Với mỗi cạnh nối giữa hai cụm của cá thể G_{IND}, các cạnh nối các đỉnh của hai cụm trong đồ thị G_{IND} là các cạnh nối giữa hai cụm tương ứng của đồ thị G.

Thuật toán 4.6 minh họa các bước của phương pháp giải mã được sử dụng dụng trong thuật toán G-MFEA.

Các bước chính của phương pháp giải mã được minh họa trong hình 4.6 với đỉnh nguồn là đỉnh 1 và đồ thị đầu vào như trong hình 4.6(a). Giả sử,

Thuật toán 4.6: Phương pháp giải mã trong thuật toán G-MFEA

Input: Đồ thị phân cụm $G = (V, E, C)$; Đỉnh nguồn s;
Cá thể trong không gian USS $I = (ES,\ IE,\ LR)$.
Output: Lời giải $T(V, E)$ của bài toán CluSPT .

```
1  begin
       /* Tạo đồ thị G' từ đồ thị G                                    */
2      Tạo đồ thị G − Graph G' = (V', E') từ đồ thị G;
3      E' ← ∅;
4      V' ← V;
5      C' ← C;
6      foreach cạnh e = (v'_i, v'_j ∈ ES do
7          foreach đỉnh u ∈ C_i, đỉnh v ∈ C_j do
8              E' ← E' ∪ (u, v);
9          end
10     end
11     foreach cạnh e ∈ C_i do
12         E' ← E' ∪ e;
13     end
14     T ← HB-RGA(G'=(V', E', C'), s)        ▷ Trình bày trong phần 2.2;
15     return T;
16 end
```

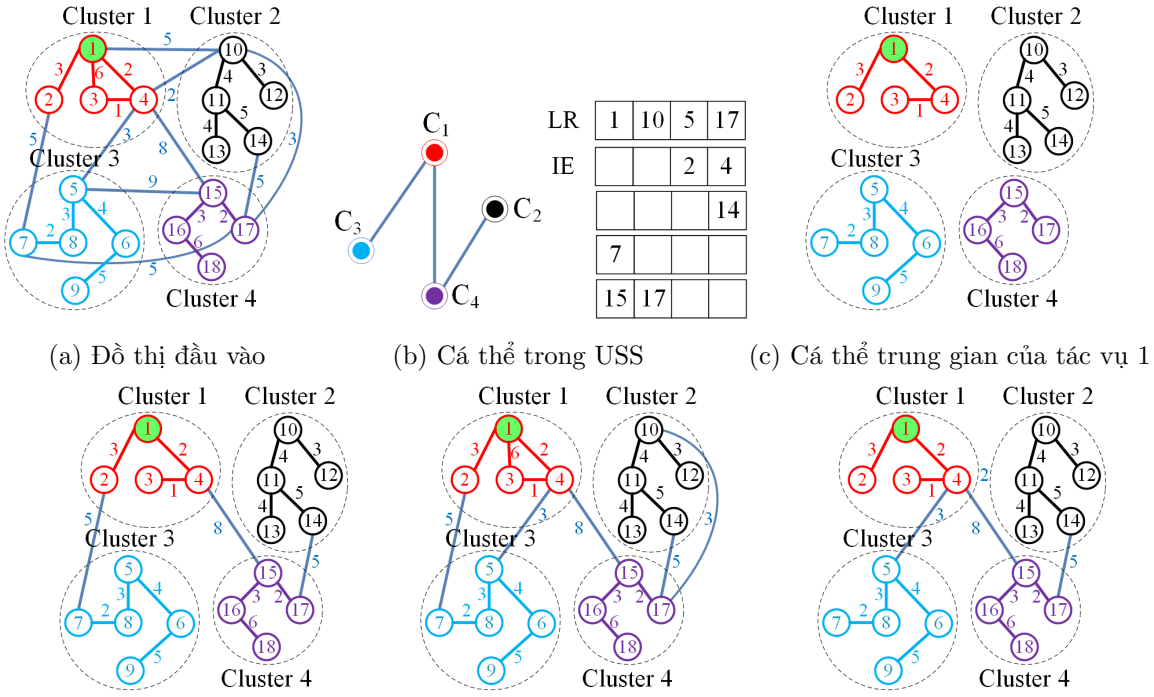
cá thể trong không gian USS như trong hình 4.6(b), khi đó:

Đối với tác vụ thứ nhất: Hình 4.6(c) minh họa một phần lời giải nhận được từ cá thể trong hình 4.6(b) sau khi áp dụng thuật toán Dijkstra. Hình 4.6(d) minh họa lời giải bài toán CluSPT được tạo từ các thể trung gian trong hình 4.6(c) sau khi thêm các cạnh nối giữa các cụm. Các cạnh này được xây dựng dựa trên thuộc tính IE của cá thể trong không gian USS (xem hình 4.6(b)).

Đối với tác vụ thứ hai: Hình 4.6(e) minh họa đồ thị đầu vào của thuật toán HB-RGA. Đồ thị này được tạo từ đồ thị đầu vào trong hình 4.6(a) và tập cạnh của cá thể trong hình 4.6(b). Hình 4.6(f) minh họa lời giải bài toán CluSPT nhận được sau khi áp dụng thuật toán HB-RGA cho đồ thị đầu vào trong hình 4.6(e).

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

Mỗi cá thể trong không gian USS gồm có 3 thành phần: lời giải của bài toán H-Problem, tập cạnh IE và tập đỉnh gốc LR. Để xây dựng lời



(a) Đồ thị đầu vào (b) Cá thể trong USS (c) Cá thể trung gian của tác vụ 1

(d) Lời giải tạo bởi tác vụ 1 (e) Đầu vào của HB-RGA (f) Lời giải tạo bởi tác vụ 2

Hình 4.6: Ví dụ minh họa về phương pháp giải mã trong thuật toán G-MFEA

giải bài toán CluSPT, tác vụ thứ nhất sử dụng thông tin của cả ba thành phần, trong khi tác vụ thứ hai chỉ sử dụng thông tin về lời giải bài toán H-Problem.

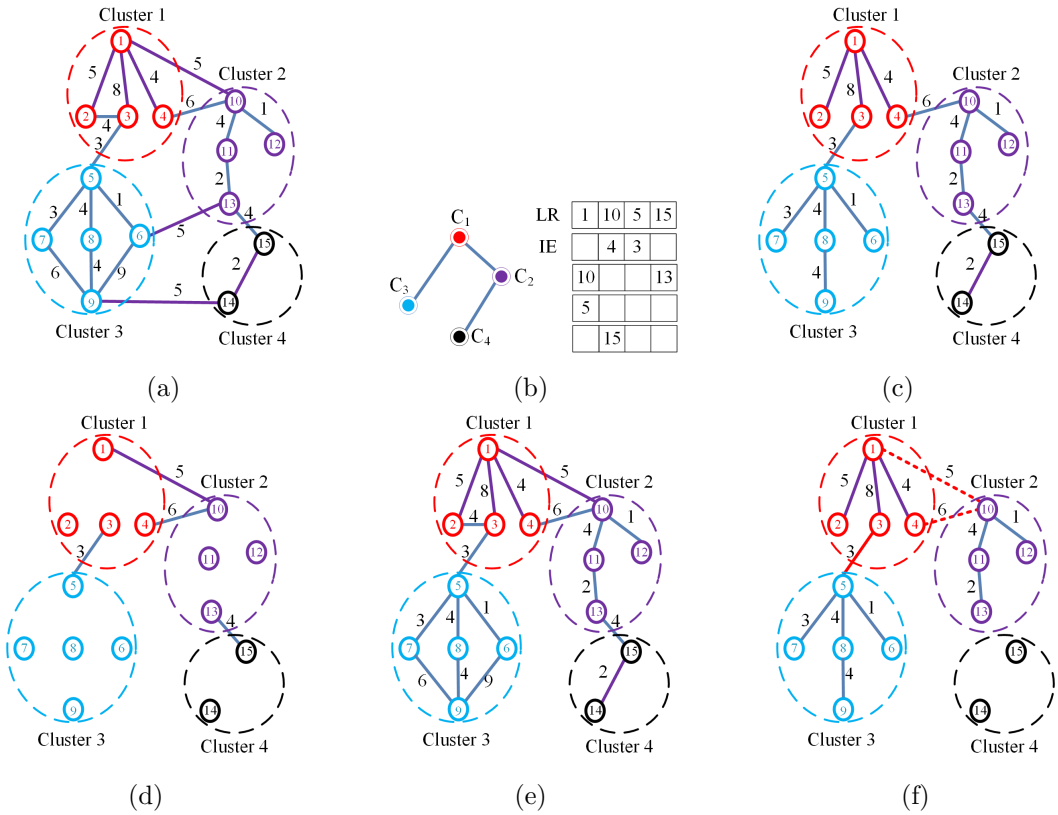
Hình 4.7 minh họa hoạt động của tác vụ thứ hai với đồ thị đầu vào như trong hình 4.7(a) và đỉnh nguồn là đỉnh 1. Hình 4.7(b) minh họa một cá thể trong không gian USS với nửa bên trái là lời giải bài toán H-Problem, còn nửa bên phải là các thuộc tính LR, IE.

Đối với lời giải bài toán CluSPT được tạo ra bởi tác vụ thứ nhất, các cạnh nối giữa các cụm được tạo ra bởi các toán tử lai ghép và đột biến, nên các cạnh này có thể chưa phải là các cạnh tốt nhất. Xét lời giải là đầu ra của tác vụ thứ nhất (trong hình 4.7(c)) nhận được sau khi áp dụng phương pháp giải mã cho cá thể trong hình 4.7(b). Trong hình 4.7(c), cạnh (4, 10) nối giữa cụm 1 và cụm 2 không phải là cạnh tốt nhất nối giữa hai cụm này do nếu thay cạnh (4, 10) bởi cạnh (1, 10) thì sẽ tạo ra lời giải mới tốt hơn lời giải trong hình 4.7(c).

Từ các nhận xét trên, phương pháp giải mã của tác vụ thứ hai sử dụng thuật toán HB-RGA để xác định cạnh tốt nhất nối giữa các cụm. Đầu tiên, phương pháp mã hóa sẽ xây dựng đồ thị đầu vào G_{RGA} của thuật toán HB-RGA. Do cả đồ thị G_{RGA} và đầu ra của phương thức giải mã của tác vụ thứ nhất đều được xây dựng từ cùng một cá thể trong không gian USS nên cạnh nối giữa cụm C_i và cụm C_j (một cạnh hoặc nhiều cạnh) trong đồ thị G_{RGA} chỉ tồn tại khi và chỉ khi hai cụm này được nối bởi một cạnh duy nhất trong lời giải nhận được từ phương thức giải mã của tác vụ thứ nhất. Điều đó có nghĩa rằng, với mỗi cá thể trong không gian USS, thuật toán HB-RGA chỉ tối ưu các cạnh nối giữa hai cụm khi hai cụm đó cũng được nối bởi một cạnh trong lời giải nhận được từ phương thức giải mã của tác vụ thứ nhất.

Xét ví dụ trong hình 4.7(d) về tập các cạnh nối giữa các cụm nhận được từ đồ thị đầu vào trong hình 4.7(a) và lời giải bài toán H-Problem trong Hình 4.7(b). Hình 4.7(e) minh họa đồ thị G_{RGA} nhận được từ đồ thị đầu vào trong hình 4.7(a) và lời giải bài toán H-Problem trong hình 4.7(b). Khi so sánh lời giải bài toán CluSPT được tạo ra bởi tác vụ thứ nhất trong hình 4.7(c) và đồ thị G_{RGA} trong hình 4.7(f) ta có thể thấy rằng nếu hai cụm trong lời giải bài toán CluSPT được tạo bởi tác vụ thứ nhất có cạnh nối với nhau thì hai cụm đó cũng được nối với nhau trong đồ thị G_{RGA} và ngược lại. Ví dụ, cụm 1 và cụm 2 trong hình 4.7(c) được nối với nhau bởi cạnh (4, 10), còn trong đồ thị G_{RGA}, hai cụm này được nối với nhau bởi hai cạnh (4, 10) và cạnh (1, 10) (như trong hình 4.7(e)); hoặc không có cạnh nào nối giữa cụm 3 và cụm 4 (như trong cả hai hình 4.7(c) và hình 4.7(e)).

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. Xét ví dụ với đồ thị đầu vào của thuật toán HB-RGA như trong hình 4.7(e), khi đó thuật toán HB-RGA sẽ tính chi phí của các cạnh như sau:



Hình 4.7: Ví dụ minh họa cách cải thiện chất lượng lời giải của tác vụ thứ hai

- Chi phí của cạnh $e = (1, 10)$: $C_e = d[1, 1] + w[1, 10] + costSPT(10)$
- Chi phí của cạnh $e' = (4, 10)$: $C_{e'} = d[1, 4] + w[4, 10] + costSPT(10)$ trong đó $costSPT(10)$ là tổng chi phí của đường đi ngắn nhất từ đỉnh 10 tới các đỉnh khác trong cụm 2; $d(1, 1)$ là $d(1, 4)$ lần lượt là chi phí đi từ đỉnh 1 tới đỉnh 1 và từ đỉnh 1 tới đỉnh 4; $w[u, v]$ là chi phí của cạnh nối giữa đỉnh u tới đỉnh v.

  Do $C_e = 0 + 5 + costSPT(10) < C_{e'} = 4 + 6 + costSPT(10)$, nên cạnh e = (1, 10) được chọn để nối giữa cụm 1 và cụm 2.

4.9. Đánh giá thuật toán

Để phân tích hiệu quả của thuật toán G-MFEA luận án đã tiến hành thực nghiệm trên 141 bộ dữ liệu thuộc 6 tập đồ thị metric và 75 bộ dữ liệu thuộc 3 tập đồ thị đầy đủ phi metric. Kết quả nhận được của thuật toán G-MFEA được so sánh với lời giải tối ưu và bốn thuật toán xấp xỉ khác.

Bên cạnh đó, luận án cũng tiến hành phân tích ảnh hưởng của các tham

số của bộ dữ liệu đầu vào (số cụm và số đỉnh của đồ thị đầu vào) tới hiệu quả của thuật toán G-MFEA khi so sánh với các thuật toán đề xuất khác.

Để đánh giá các thuật toán đề xuất được chi tiết, trên nhiều khía cạnh và trên đa dạng các tiêu chí so sánh, đặc biệt là để thực hiện các phân tích thống kê và phân tích sự ảnh hưởng của các tham số dữ liệu đầu vào tới hiệu quả so sánh giữa nhiều thuật toán. Vì vậy, luận án phân tích kết quả các thuật toán, cũng như mô tả thực nghiệm chi tiết tại chương 5.

4.10. Kết luận chương

Chương này trình bày đề xuất dựa trên sự kết hợp giữa thuật toán MFEA và thuật toán HB-RGA để giải bài toán CluSPT. 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 một số điểm mạnh và hạn chế nhất định như:

- **Điểm mạnh**:
  - Thuật toán G-MFEA khai thác được các đặc trưng của thuật toán MFEA và 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 được cải thiện.
  - Thuật toán cũng khắc phục được các hạn chế của các thuật toán khác như: sử dụng được trên cả đồ thị thưa, giảm chi phí tính toán để xác định thành phần liên thông của đồ thị.
- **Hạn chế**:

– Thuật toán G-MFEA sử dụng ý tưởng dựa trên thuật toán HB-RGA nên chỉ áp dụng được để giải bài toán CluSPT.
– Do sử dụng kết hợp giữa hai thuật toán MFEA và HB-RGA nên khó để cài đặt thực nghiệm thuật toán G-MFEA.

Chương 5

KẾT QUẢ THỰC NGHIỆM

Chương này đánh giá hiệu quả của thuật toán HB-RGA (trình bày trong chương 2), thuật toán C-MFEA và N-EA (trình bày trong chương 3), thuật toán G-MFEA (trình bày trong chương 4) thông qua phân tích lời giải tìm được của các thuật toán trên và so sánh với lời giải tối ưu tìm được bằng thuật toán SLA-M. Hiệu quả của các thuật toán đề xuất được đánh giá dựa trên kết quả phân tích thống kê, phân tích chi tiết và phân tích các nhân tố ảnh hưởng. Trong phần này, luận án cũng trình bày phương pháp tiến hành thực nghiệm như: dữ liệu thực nghiệm, tiêu chí đánh giá các thuật toán, tham số và môi trường thực nghiệm.

5.1. Dữ liệu thực nghiệm và đánh giá lời giải

Do bài toán CluSPT mới được quan tâm nghiên cứu, nên các công trình nghiên cứu trước đây tập trung vào khía cạnh lý thuyết, trong đó đặc biệt tập trung chứng minh bài toán CluSPT là bài toán NP-Khó.

5.1.1. Đồ thị metric

Trong các bộ dữ liệu để thực nghiệm bài toán đồ thị có các đỉnh được phân cụm thì tập dữ liệu MOM-Lib được đề xuất trong công trình của tác giả M. Mestria và các cộng sự [40, 56] được sử dụng nhiều. Bộ dữ liệu MOM-Lib gồm 6 kiểu, mỗi kiểu dữ liệu được tạo từ một thuật toán khác nhau. Dựa theo số chiều của dữ liệu, các dữ liệu thuộc tập MOM-Lib được phân thành hai nhóm: các dữ liệu nhỏ - bao gồm các đồ thị có số đỉnh trong khoảng từ 30 tới 120 và các dữ liệu lớn - bao gồm các đồ thị có số đỉnh lớn hơn 262. Mặc dù ban đầu được thiết kết để đánh giá bài toán CluTSP, song tập dữ liệu MOM-Lib cũng được xác định là phù hợp để đánh giá các bài toán được phân cụm [56].

Do tính đa dạng về kiểu dữ liệu và nhận được đánh giá cao từ nhiều nhà nghiên cứu nên luận án lựa chọn tập dữ liệu MOM-Lib để xây dựng tập dữ liệu đánh giá bài toán CluSPT (gọi tắt là CluSPT-Lib).

Do đầu vào của bài toán CluSPT cần thêm thông tin về đỉnh nguồn, các bộ dữ liệu trong tập MOM-Lib đã được điều chỉnh để phù hợp với bài toán. Cụ thể, một đỉnh được chọn ngẫu nhiên trong các đỉnh có sẵn để làm đỉnh nguồn và thêm vào trong mỗi bộ dữ liệu. Thông tin tóm tắt về các bộ dữ liệu được trình bày trong bảng 5.1 và được lưu tại [76].

Bảng 5.1: Tóm tắt thông tin về các bộ dữ liệu đồ thị metric

	Số bộ dữ liệu	**Số đỉnh**	**Số cụm**
Type 1 small	27	51 . . . 105	5 . . . 75
Type 1 large	28	262 . . . 1397	10 . . . 150
Type 2	10	1000	10
Type 3 large	14	300 . . . 1000	6 . . . 30
Type 4 large	41	200 . . . 3000	4 . . . 20
Type 5 small	21	30 . . . 120	5 . . . 10
Type 5 large	40	300 . . . 3000	5 . . . 200
Type 6 small	37	51 . . . 105	4 . . . 41
Type 6 large	30	262 . . . 1379	9 . . . 144

Do có tới 248 bộ dữ liệu thuộc 6 tập dữ liệu (có tên là Type 1, .., Type 6) nên luận án chỉ tiến hành thực nghiệm với các bộ dữ liệu thuộc tập dữ liệu nhỏ và tập dữ liệu lớn có số đỉnh nhỏ hơn 500. Thông tin chi tiết về các bộ dữ liệu được sử dụng để thực nghiệm đánh giá các đề xuất được trình bày trong các bảng 15 – 18 của phần phụ lục.

5.1.2. Đồ thị đầy đủ phi metric

Để đánh giá các thuật toán đề xuất, luận án còn sử dụng các bộ dữ liệu đồ thị đầy đủ, nhưng không thỏa mãn điều kiện bất đẳng thức tam giác trong định nghĩa 1.12 về đồ thị metric (*gọi là các bộ dữ liệu đồ thị đầy đủ phi metric*). Mỗi bộ dữ liệu đầy đủ phi metric được xây dựng từ một bộ dữ liệu thuộc CluSPT-Lib bằng cách:

- Giữ nguyên các thông tin về đỉnh nguồn, danh sách các đỉnh trong mỗi cụm.
- Trọng số các cạnh nối hai đỉnh trong đồ thị được sinh ngẫu nhiên trong đoạn $[1, 100]$ (do được sinh ngẫu nhiên nên trọng số cạnh $e = (u, v)$

có thể khác trong số cạnh $e' = (v, u)$).

- Tên các bộ dữ liệu đầy đủ phi metric bắt đầu bằng ký tự "N-" sau đó là tên bộ dữ liệu tương ứng trong CluSPT-Lib. Ví dụ, bộ dữ liệu đầy đủ phi metric được tạo từ bộ dữ liệu "5Eil51" có tên là "N-5Eil51".

Thông tin thêm về các bộ dữ liệu đầy đủ phi metric được cập nhật tại [76].

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.

Đánh giá độ phức tạp của thuật toán, nói đơn giản, là đánh giá lượng tài nguyên các loại mà thuật toán đòi hỏi sử dụng. Có hai loại tài nguyên quan trọng nhất là bộ nhớ và thời gian, mà hiện nay, khi giải quyết nhiều vấn đề ứng dụng thực tiễn quan trọng người sử dụng quan tâm nhiều hơn đến yếu tố thời gian.

Do thuật toán gần đúng không đảm bảo đưa ra lời giải tối ưu nên cần có tiêu chí đánh giá hiệu quả của thuật toán. Hiện nay có rất nhiều tiêu chí đánh hiệu năng của các thuật toán, tuy nhiên các tiêu chí liên quan tới chất lượng lời giải mà thuật toán tìm được và thời gian thực hiện của thuật toán thường được sử dụng để đánh giá các thuật toán. Do đó, luận án tập trung phân tích các tiêu chí về thời gian thực hiện và chất lượng lời giải của các thuật toán. Chi tiết về các tiêu chí đánh giá được sử dụng trong luận án được miêu tả trong bảng 5.2.

Bảng 5.2: Các tiêu chí đánh giá thuật toán

Average (Avg)	Trung bình cộng giá trị hàm mục tiêu của các lần thực hiện.
Best-found (BF)	Giá trị tốt nhất tìm được trong tất cả các lần thực hiện.
Time	Thời gian thực hiện trung bình của thuật toán (đơn vị tính: phút).
RPD	tỉ lệ phần trăm chênh lệch tương đối (*Relative Percentage Differences - RPD*).

Để so sánh chất lượng lời giải, luận án sử dụng công thức tính RPD [1, 65] giữa lời giải tìm được của các thuật toán. Tỉ lệ RPD được tính theo

công thức:

$$RDP_{ar}^{i} = \frac{S_{ar}^{i} - B^{i}}{B^{i}} * 100 \tag{5.1}$$

trong đó S_{ar}^{i} là lời giải nhận được của thuật toán a tại lần thực hiện thứ r đối với bộ dữ liệu i; B^{i} là lời giải tốt nhất của tất cả các thuật toán đối với bộ dữ liệu i.

5.1.4. Môi trường, tham số thực nghiệm

Để đánh giá hiệu quả của các thuật toán đề xuất, luận án tiến hành hai nhóm thực nghiệm chính:

- Trong nhóm thực nghiệm đầu tiên, luận án 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.
- Trong nhóm thực nghiệm thứ hai, luận án 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 cài đặt hệ điều hành Microsoft Windows 8 với cấu hình: CPU - Intel Core i7 (4790M - 3.60GHz), RAM - 16GB. Thuật toán HB-RGA được thực nghiệm với các tham số $\gamma = 50$; các thuật toán tiến hóa và tiến hóa đa nhân tố được thực nghiệm với tham số: số lần đánh giá là 5000 lần, kích thước quần thể $P = 100$, tỉ lệ lai ghép $p_c = 0.5$, tỉ lệ đột biến $p_m = 0.05$, xác suất ghép cặp ngẫu nhiên $rmp = 0.5$.

5.2. Kết quả thực nghiệm

Để chứng minh hiệu quả của các thuật toán đề xuất cũng như để so sánh giữa các đề xuất, luận án tiến hành phân tích kết quả thực nghiệm theo ba hình thức:

- **Phân tích thống kê:** do sử dụng phân tích thông kê để đánh giá hiệu quả của thuật toán ngày càng phổ biến và trở thành kỹ thuật

quan trọng trong tính toán thông minh (computational intelligence). Vì vậy, luận án sử dụng phân tích thống kê để phân tích hiệu quả của các đề xuất.

- **Phân tích chi tiết**: luận án so sánh chi tiết kết quả tìm được của các thuật toán đề xuất theo từng bộ dữ liệu thuộc các thuật toán khác nhau.
- **Phân tích nhân tố ảnh hưởng:** luận án phân tích sự ảnh hưởng 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

Để chỉ ra hiệu quả của các thuật toán đề xuất dựa trên chất lượng lời giải tìm được, 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 một số kiểm định như: Friedman, Aligned Friedman, Quade,... [10] để kiểm tra sự khác biệt giữa kết quả của các thuật toán có ý nghĩa thống kê hay không.
- Bước 2 sẽ được thực hiện nếu bước 1 đã bác bỏ được các giả thuyết về sự tương đương của kết quả nhận được từ các thuật toán. Luận án sử dụng các phân tích thống kê hậu kiểm (*Post-hoc statistical*) [10] để xác định chi tiết sự khác biệt giữa kết quả của các thuật toán, cũng như xác định thuật toán tốt nhất (theo phương diện chất lượng lời giải tìm được) trong các thuật toán đề xuất.

Chi tiết kết quả tìm được từ các thuật toán trên các tập dữ liệu được trình trong các bảng 19 – bảng 21 (trong phần phục lục). Để tiện cho theo dõi và đánh giá kết quả, trong mỗi dòng của các bảng này, giá trị tương ứng với lời giải tốt nhất sẽ được in nghiêng và có mầu đỏ, còn giá trị trung bình của lời giải tìm được bằng giá trị tối ưu sẽ được in đậm.

Bảng 5.3 trình bày kết quả thống kê của các kiểm định Friedman và Iman-Davenport. Kết quả trong bảng cho thấy rằng các giá trị Friedman

Bảng 5.3: Bảng kết quả phân tích kiểm định Friedman và Iman-Davenport (α=0.05)

Friedman			**Iman-Davenport**		
Giá trị Friedman	Giá trị X^2	Trị số p	Giá trị Iman-Davenport	Giá trị F_F	Trị số p
478.58	9.49	$1.77 * 10^{-10}$	**816.98**	2.39	$3.723 * 10^{-231}$

và Iman-Davenp lớn nhiều lần so với các giá trị tới hạn (*critical value*) và trị số p nhỏ hơn nhiều lần so với ngưỡng $\alpha = 0.05$, điều đó có nghĩa là có sự khác biệt rõ ràng giữa kết quả thực nghiệm nhận được của các thuật toán với sai số nhỏ hơn 5%. Hay nói cách khác sự khác biệt giữa kết quả các thuật toán có ý nghĩa thống kê với ngưỡng $\alpha = 0.05$.

Bảng 5.4: Xếp hạng trung bình của các thuật toán nhận được từ được đánh giá bởi các kiểm định Friedman, Friednman Aligned và Quade

Thuật toán	Friedman	Friednman Aligned	Quade
AAL	4.742	576.964	4.641
HB-RGA	1.793	212.121	1.713
N-EA	2.761	215.254	2.772
C-EA	4.257	536.5	4.359
G-MFEA	1.446	211.660	1.514

Bảng 5.4 trình bày giá trị xếp hạng trung bình của các thuật toán được đánh giá bởi các kiểm định Friedman, Friednman Aligned và Quade. Kết quả trong bảng củng cố thêm nhận định về sự khác nhau về chất lượng lời giải tìm được giữa các thuật toán được so sánh. Trong bảng 5.4, 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.

Để so sánh thuật toán G-MFEA với các thuật toán khác, luận án sử dụng các phương thức như Holm, Holland,... Bảng 5.5 trình bày kết quả đánh giá các giả thuyết khi so sánh giữa thuật toán điều khiển G-MFEA với các thuật toán còn lại.

Bảng 5.6 trình bày các giá trị đã hiệu chỉnh của trị số p thu được từ các kiểm định Friedman và Quade đối với các so sánh giữa thuật toán G-MFEA và các thuật toán khác. Trong bảng giá trị đã hiệu chỉnh p, ký

Bảng 5.5: Các trị số z và p của các kiểm định Friedman, Quade (G-MFEA là thuật toán điều khiển)

FRIEDMAN								
i	Thuật toán	z	p	Holm	Holland	Rom	Finner	Li
4	AAL	17.443	$3.887*10^{-68}$	0.0125	0.0127	0.0131	0.0127	0.0491
3	C-EA	14.8729	$4.9423*10^{-50}$	0.01667	0.0170	0.01667	0.0253	0.0491
2	N-EA	6.9545	$3.5369*10^{-12}$	0.025	0.0253	0.025	0.0377	0.0491
1	HB-RGA	1.8331	0.0667	0.05	0.050	0.05	0.050	0.05
QUADE								
4	AAL	10.1487	$3.3587*10^{-24}$	0.0125	0.0127	0.0131	0.0127	0.0253
3	C-EA	9.2357	$2.5663*10^{-20}$	0.01667	0.01695	0.01667	0.0253	0.0254
2	N-EA	4.0841	$4.4240*10^{-5}$	0.025	0.0253	0.025	0.0377	0.0253
1	HB-RGA	0.6464	0.5180	0.05	0.0500	0.05	0.050	0.05

Bảng 5.6: Bảng giá trị đã hiệu chỉnh của trị số p của các kiểm định Friedman và Quade (G-MFEA là thuật toán điều khiển)

FRIEDMAN						
i	Thuật toán	Unadjusted p	p_{Bonf}	p_{Holm}	p_{Holl}	p_{Li}
1	**AAL**	$3.89 * 10^{-68}$	$1.55 * 10^{-67}$	$1.55 * 10^{-67}$	0.0	$4.17 * 10^{-68}$
2	**C-EA**	$4.94 * 10^{-50}$	$1.98 * 10^{-49}$	$1.48 * 10^{-49}$	0.0	$5.30 * 10^{-49}$
3	**N-EA**	$3.54 * 10^{-12}$	$1.41 * 10^{-11}$	$7.07 * 10^{-12}$	$7.07 * 10^{-12}$	$3.79 * 10^{-12}$
4	HB-RGA	0.0667	0.267	0.0667	0.0667	0.0667
QUADE						
i	Thuật toán	Unadjusted p	p_{Bonf}	p_{Holm}	p_{Holl}	p_{Li}
1	**AAL**	$3.36 * 10^{-24}$	$1.34 * 10^{-23}$	$1.34 * 10^{-23}$	0.0	$6.96 * 10^{-24}$
2	**C-EA**	$2.56 * 10^{-20}$	$1.02 * 10^{-19}$	$7.699 * 10^{-20}$	0.0	$5.32 * 10^{-20}$
3	**N-EA**	$4.42 * 10^{-5}$	$1.76 * 10^{-4}$	$8.84 * 10^{-5}$	$8.848 * 10^{-5}$	$9.18 * 10^{-5}$
4	HB-RGA	0.51	2.07	0.518	0.518	0.518

hiệu p_{Bonf}, p_{Holm}, p_{Holl}, p_{Li} lần lượt là giá trị sau khi hiệu chỉnh của trị số p đối với các phương thức Bonferroni-Dunn, Holm, Holland và Li (cách ký hiệu này sẽ được sử dụng cho cả các bảng giá trị đã hiệu chỉnh của trị số p trong phần sau). Trong bảng 5.6, các thuật toán AAL, C-EA và N-EA (các thuật toán có tên được in đậm) có giá trị của trị số p sau khi hiệu chỉnh nhỏ hơn 0.05 nên kết quả so sánh giữa các thuật toán này với thuật toán G-MFEA có ý nghĩa thống kê với ngưỡng $\alpha = 0.05$. Nghĩa là thuật toán 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

Phần này trình bày các so sánh chi tiết về chất lượng lời giải tìm được của các thuật toán đối với mỗi bộ dữ liệu và mỗi tập dữ liệu.

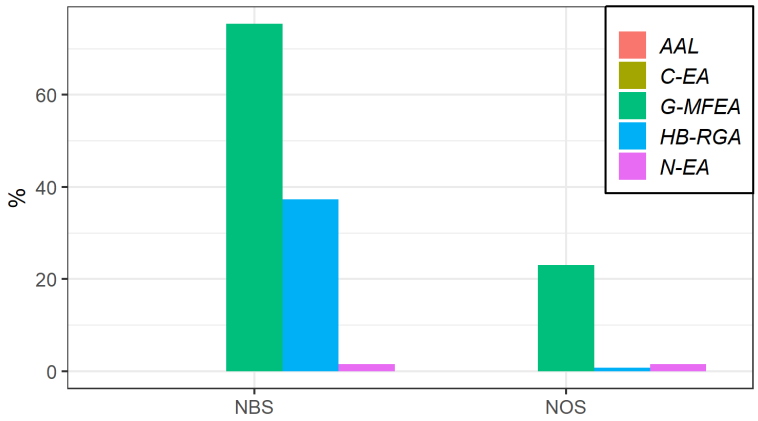
Bảng 5.7: Bảng tóm tắt số lần tìm được lời giải tốt nhất và số lần tìm được lời giải tối ưu của các thuật toán

Tập	**LỜI GIẢI TỐT NHẤT**					**LỜI GIẢI TỐI ƯU**					**TBDL**
	AAL	HB-RGA	N-EA	C-EA	G-MFEA	AAL	HB-RGA	N-EA	C-EA	G-MFEA	
1 Nhỏ		9			17					8	26
Lớn		10			5						15
5 Nhỏ		1	1		21		1	1		10	21
Lớn		4			11						15
6 Nhỏ		14	1		22			1		7	36
Lớn		6			7						13
3					5						5
4		3			7					4	10
Tổng:		47	2	0	95	0	1	2	0	29	141

TBDL: Tổng số bộ dữ liệu.

Bảng 5.7 trình bày tóm tắt về thống kê số lần tìm được lời giải tối ưu và số lần tìm được lời giải tốt nhất của các thuật toán. Để cho dễ theo dõi các ô tương ứng với thuật toán không tìm được lời giải tối ưu hoặc không có lần nào tìm được được lời giải tốt nhất được để trống. 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. Chi tiết hơn về tỉ lệ số lần tìm được lời giải tối ưu và số lần tìm được lời giải tốt nhất so với tổng số bộ dữ liệu của mỗi tập được minh họa trong hình 5.1.

Từ bảng 5.7 và hình 5.1 cho thấy rằng thuật toán G-MFEA tìm được lời giải có chất lượng trội hơn so với các thuật toán còn lại.



Hình 5.1: Tỉ lệ phần trăm số lời giải tối ưu và số lời giải tốt nhất tìm được của các thuật toán (NBS: số lời giải tốt nhất tìm được; NOS: số lời giải tối ưu tìm được)

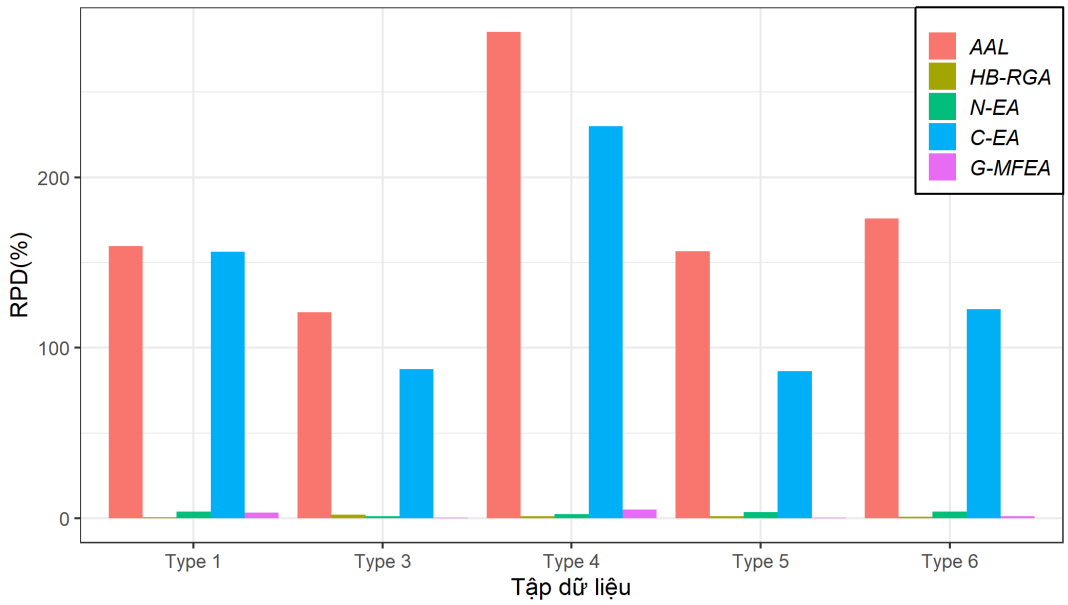
Kết quả trong các bảng 19 – bảng 21 cho thấy các thuật toán đề xuất đều tốt hơn thuật toán AAL trên đa số bộ dữ liệu. Bên cạnh đó, bảng 5.4 cũng cho thấy chất lượng lời giải tìm của các thuật toán kém nhất là thuật toán AAL, tiếp đến thuật toán C-EA và N-EA. Tuy nhiên, giá trị xếp hạng trung bình của thuật toán N-EA chênh lệch đáng kể so với hai thuật toán AAL và C-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.

Bảng 5.8: Bảng giá trị trung bình RPD của các thuật toán trên các tập dữ liệu

	AAL	HB-RGA	N-EA	C-EA	G-MFEA
Type 1	159.70	0.79	3.83	156.21	3.27
Type 5	156.67	1.40	3.65	86.17	0.48
Type 6	175.93	0.88	4.04	122.78	1.30
Type 3	120.80	2.26	1.42	87.66	0.28
Type 4	285.33	1.13	2.62	229.92	5.18

Để phân tích sâu hơn về chất lượng lời giải tìm được, luận án so sánh giá trị RPD của các thuật toán. Giá trị trung bình RPD trên các tập dữ liệu của các thuật toán được trình bày trong bảng 5.8 cho thấy giá trị RPD của thuật toán AAL lớn hơn các thuật toán khác trên tất cả các bộ dữ liệu. Điều đó có nghĩa rằng sai lệch trung bình của kết quả nhận được của thuật toán AAL so với giá trị tối ưu lớn hơn các thuật toán khác. Hay

nói các khác, 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, thuật toán C-EA có giá trị RPD trung bình lớn hơn so với ba thuật toán còn lại, nên 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. Tuy nhiên giá trị RPD trung bình của thuật toán N-EA lớn hơn hai thuật toán còn lại đối với 4/5 tập dữ liệu (trừ tập dữ liệu Type 4). Điểm đáng chú ý khi so sánh giá trị RPD trung bình của hai thuật toán HB-RGA và G-MFEA là các giá trị này rất nhỏ đối với một số tập dữ liệu như 0.28% và 0.48% đối với Type 3 và Type 4 (thuật toán G-MFEA) và 0.79% đối với Type 1 (thuật toán HB-RGA). Điều đó có nghĩa là trên các tập dữ liệu này, thuật toán G-MFEA và thuật toán HB-RGA tìm được lời giải gần với lời giải tối ưu. Hình 5.2 minh họa trực quan hơn về so sánh giữa giá trị RPD của các thuật toán.



Hình 5.2: Giá trị trung bình RPD của các thuật toán trên các tập dữ liệu

Bảng 5.9 tóm tắt so sánh kết quả nhận được của các thuật toán đối với các bộ dữ liệu thuộc các tập. Trong bảng tóm tắt này, số liệu tại mỗi ô cho

giao của hàng thuộc "Type X" của thuật toán "A" và cột thuộc thuật toán "B" cho biết số lượng bộ dữ liệu thuộc "Type X" mà thuật toán "A" kém hơn thuật toán "B". Ví dụ, ô trên cùng bên trái giao hàng thuộc Type 1 của thuật toán AAL và cột thuộc thuật toán C-EA có giá trị 22. Nghĩa là: thuật toán AAL kém hơn thuật toán C-EA trên 22 bộ dữ liệu thuộc Type 1. Giá trị trong ngoặc đơn "(.)" trong mỗi ô cho biết số bộ dữ liệu mà hai thuật toán tương ứng với hàng và cột của ô đó có kết quả bằng nhau. Ví dụ, ô có giá trị 39(9) thuộc hàng của tập Type 6 của thuật toán AAL và cột thuộc thuật toán C-EA cho biết hai thuật toán này tìm được kết quả bằng nhau trên 9 bộ dữ liệu. Các ô để trống trong bảng 5.9 là tương ứng với các thuật toán đã so sánh trước đó hoặc so sánh với chính thuật toán đó.

Bảng 5.9: Bảng tóm tắt so sánh kết quả dựa trên số bộ dữ liệu của các tập dữ liệu

		C-EA	**N-EA**	**HB-RGA**	**G-MFEA**	**Số bộ dữ liệu**
AAL	Type 1	22	41	41	41	41
	Type 5	31	36	36	36	36
	Type 6	39(9)	48	48	48	48
	Type 3	5	5	5	5	5
	Type 4	7(3)	10	10	10	10
C-EA	Type 1		41	41	41	41
	Type 5		36	36	36	36
	Type 6		48	48	48	48
	Type 3		5	5	5	5
	Type 4		10	10	10	10
N-EA	Type 1			38	31	41
	Type 5			30(1)	35	36
	Type 6			45	44(1)	48
	Type 3			0	***5***	5
	Type 4			10	7	10
HB-RGA	Type 1				22	41
	Type 5				31(1)	36
	Type 6				29	48
	Type 3				***5***	5
	Type 4				7	10

Kết quả trong bảng 5.9 cho thấy:

- Thuật toán AAL kém hơn ba thuật toán: N-EA, HB-RGA và G-MFEA

trên tất cả các bộ dữ liệu thuộc các tập. Thuật toán AAL kém hơn thuật toán C-EA trong hầu hết các bộ dữ liệu thuộc Type 3 – Type 6 và trong phần lớn bộ dữ liệu thuộc Type 1 (với 22/41 bộ dữ liệu).

- Thuật toán C-EA kém các thuật toán N-EA, HB-RGA và G-MFEA trên tất cả các bộ dữ liệu thuộc các tập.
- Thuật toán N-EA kém hơn thuật toán HB-RGA trên hầu hết các bộ dữ liệu thuộc Type 1, Type 5 và Type 6; trên tất cả các bộ dữ liệu thuộc Type 4. Tuy nhiên, thuật toán N-EA lại tốt hơn thuật toán HB-RGA trên tất cả các bộ dữ liệu thuộc Type 3.
- Thuật toán HB-RGA kém hơn thuật toán G-MFEA khoảng 2/3 số bộ dữ liệu của mỗi tập. Đặc biệt đối với Type 3, thuật toán G-MFEA tốt hơn thuật toán HB-RGA trên tất cả các bộ dữ liệu đã thực nghiệm.

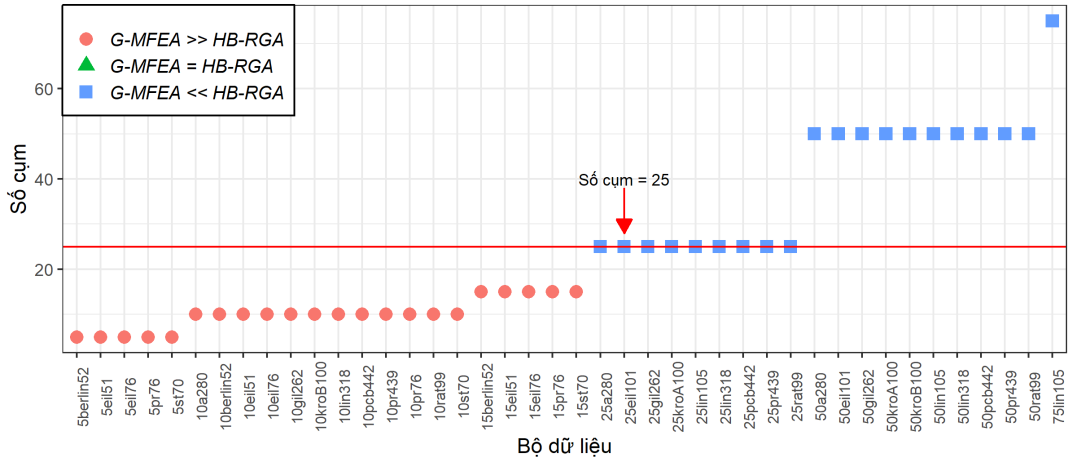
Qua các phân tích trên có thể thấy rằng, chất lượng lời giải tìm được bởi các thuật toán xếp theo thứ tự giảm dần là: G-MFEA $\rightarrow$ HB-RGA $\rightarrow$ N-EA $\rightarrow$ C-EA $\rightarrow$ AAL.

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

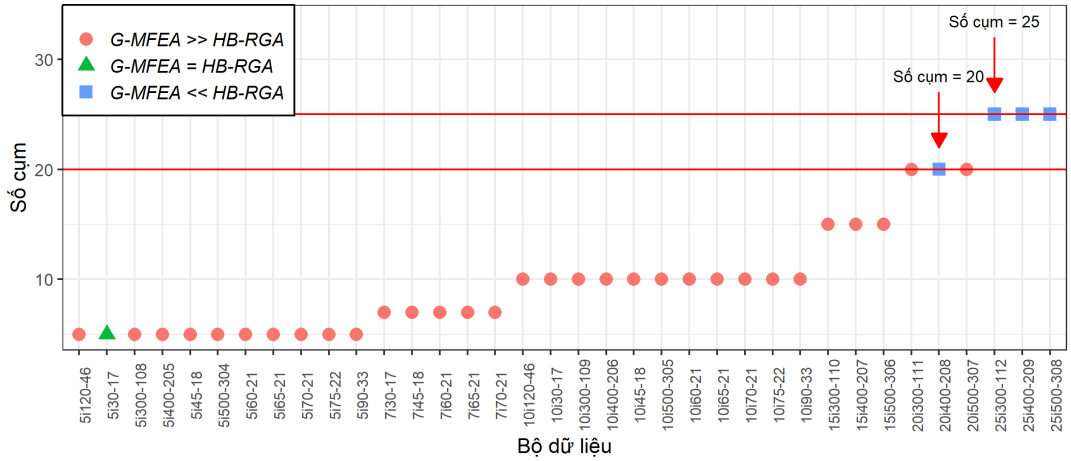
Trong phần a) và b) luận án đã chỉ ra: hai thuật toán AAL và C-EA kém hơn ba thuật toán N-EA, HB-RGA và G-MFEA trên tất cả các bộ dữ liệu. Do đó trong phần này, luận án sẽ phân tích sự ảnh hưởng của số cụm và số đỉnh của đồ thị đầu với tới hiệu quả của ba thuật toán có kết quả tốt hơn N-EA, HB-RGA và G-MFEA. Cũng do thuật toán G-MFEA tìm được lời giải có chất lượng trội hơn hai thuật toán N-EA và HB-RGA nên phần so sánh này tập trung vào phân tích sự liên hệ giữa số cụm và số đỉnh của đồ thị tới kết quả so sánh giữa thuật toán G-MFEA và hai thuật toán N-EA và HB-RGA.

Do thuật toán G-MFEA tốt hơn tốt hơn cả hai thuật toán N-EA và HB-RGA trên tất cả các bộ dữ liệu của tập Type 3 (ô in nghiêng, màu đỏ trong bảng 5.9) và số bộ dữ liệu của tập Type 4 nhỏ nên luận án chỉ tập trung đánh giá trên các bộ dữ liệu thuộc tập Type 1, Type 5 và Type 6.

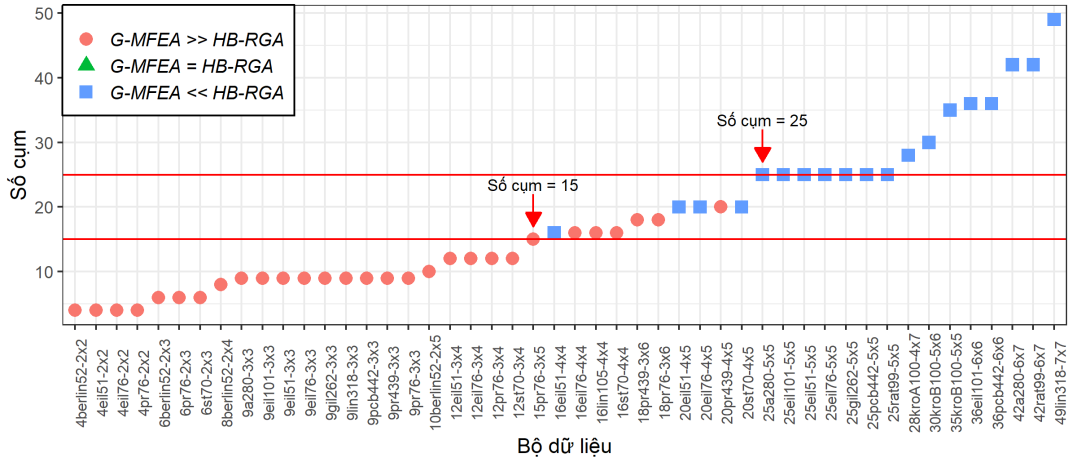
Để xác định mối liên hệ giữa số cụm, số đỉnh của đồ thị và kết quả



(a) Type 1



(b) Type 5



(c) Type 6

Hình 5.3: Biểu đồ phân tán của kết quả so sánh hai thuật toán G-MFEA và HB-RGA với số cụm của đồ thị.

so sánh giữa các thuật toán (trường hợp khi thuật toán G-MFEA tốt hơn thuật toán N-EA, HB-RGA được ký hiệu lần lượt là O_{N-EA} và O_{HB-RGA}), với mỗi tập dữ liệu, luận án phân tích các biểu đồ phân tán đối với số cụm, số đỉnh của đồ thị và O_{N-EA}, O_{HB-RGA}).

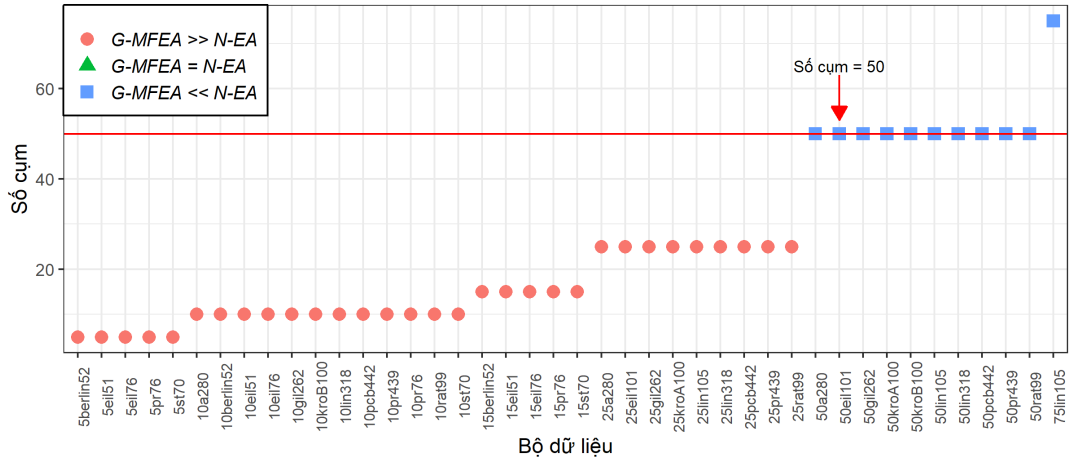
Trong các biểu đồ phân tán, ký hiệu "$A >> B$", "$A = B$", "$A << B$" lần lượt có nghĩa là thuật toán A tìm được lời giải tốt hơn (hình chấm tròn màu đỏ) , bằng nhau (hình tam giác màu xanh lá cây) và kém hơn thuật toán (hình vuông màu xanh da trời) B. Ví dụ, trong hình 5.3(a), ký hiệu kết quả so sánh tại bộ dữ liệu 5Eil51 là "G-MFEA >> HB-RGA" nghĩa là tại bộ dữ liệu này thuật toán G-MFEA tìm được lời giải tốt hơn thuật toán HB-RGA.

Hình 5.3 và 5.4 minh họa mối quan hệ giữa số cụm của đồ thị và kết quả so sánh giữa các thuật toán N-EA, HB-RGA và G-MFEA. Hình 5.3(a), 5.3(b) và 5.3(c) ta thấy tất cả các ký hiệu ở phía dưới đường thẳng nằm ngang thứ nhất đều là hình tròn. Nghĩa là các bộ dữ liệu có số lượng cụm nhỏ hơn 25 (đối với Type 1), 20 (đối với Type 5), 15 (đối với Type 6) thì thuật toán G-MFEA cho kết quả tốt hơn thuật toán HB-RGA. Các hình trên cũng cho thấy rằng 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.

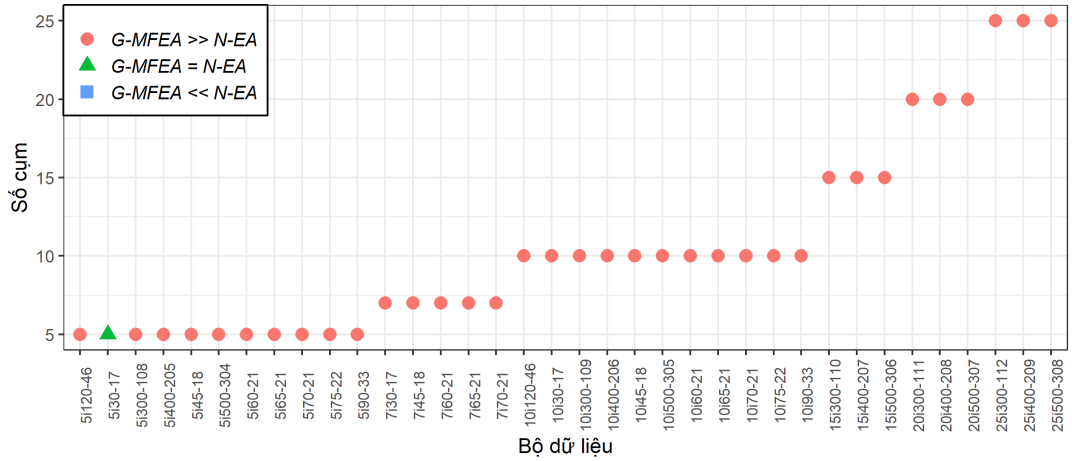
Đối với so sánh giữa thuật toán N-EA và thuật toán G-MFEA, hình 5.4 cho thấy thuật toán N-EA tốt hơn thuật toán G-MFEA trong một số ít bộ dữ liệu khi số cụm lớn hơn 50 (đối với Type 1) và 42 (đối với Type 6). Hay nói cách khác, thuật toán G-MFEA có hiệu quả cao hơn đối với các bộ dữ liệu có số cụm bé.

Hình 5.5 và hình 5.6 minh họa mối quan hệ giữa số lượng đỉnh của đồ thị và kết quả so sánh của các thuật toán N-EA, HB-RGA và G-MFEA.

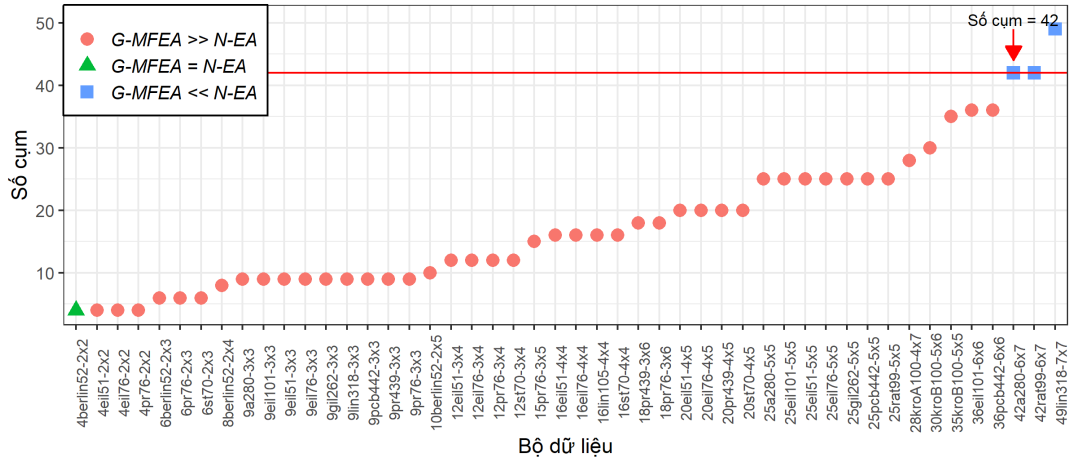
Hình 5.5(a) và 5.5(b) cho thấy các ký hiệu phía dưới đường thẳng nằm ngang đề là hình tròn. Nghĩa là, với các bộ dữ liệu có số đỉnh nhỏ hơn 99 (đối với Type 1), 300 (đối với Type 5) thuật toán G-MFEA tìm được kết quả tốt hơn thuật toán HB-RGA. Tuy nhiên, xu hướng của tương quan



(a) Type 1

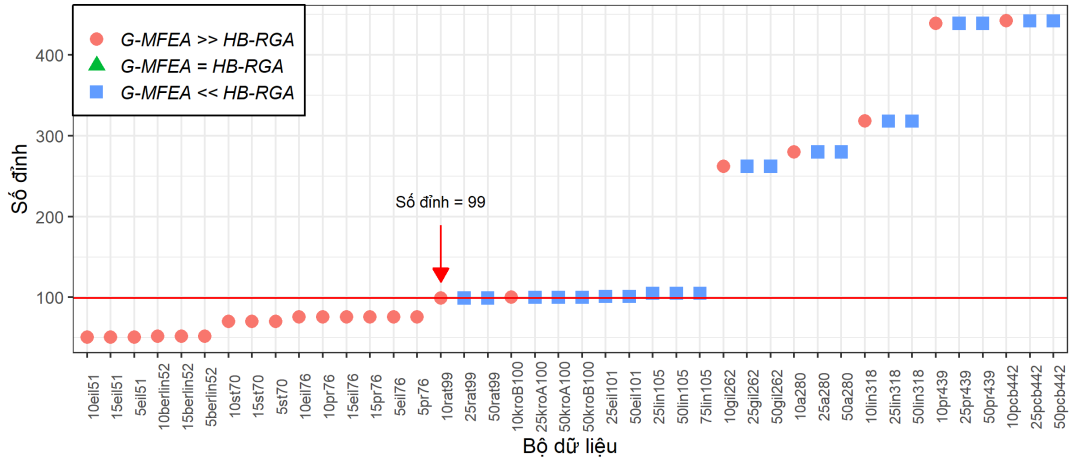


(b) Type 5

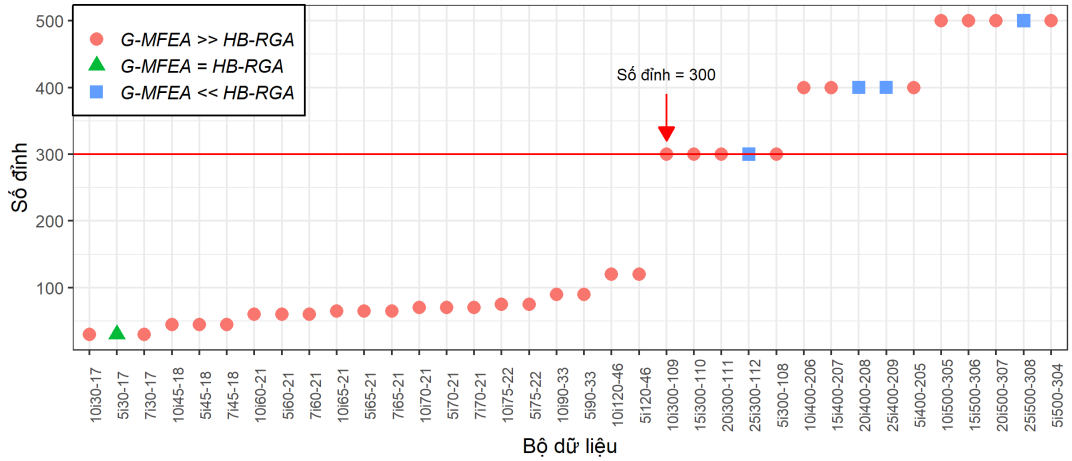


(c) Type 6

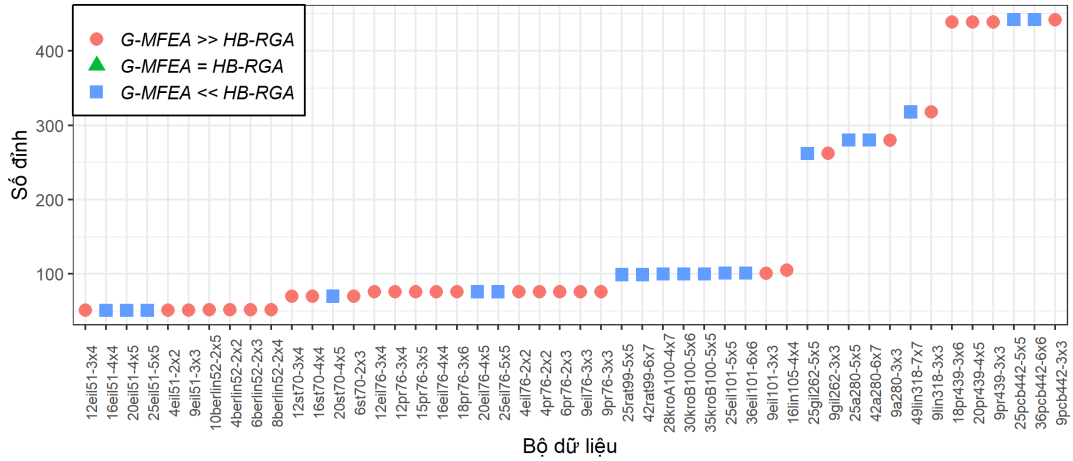
Hình 5.4: Biểu đồ phân tán của kết quả so sánh hai thuật toán G-MFEA và N-EA với số cụm của đồ thị.



(a) Type 1



(b) Type 5



(c) Type 6

Hình 5.5: Biểu đồ phân tán của kết quả so sánh hai thuật toán G-MFEA và HB-RGA với số đỉnh của đồ thị.

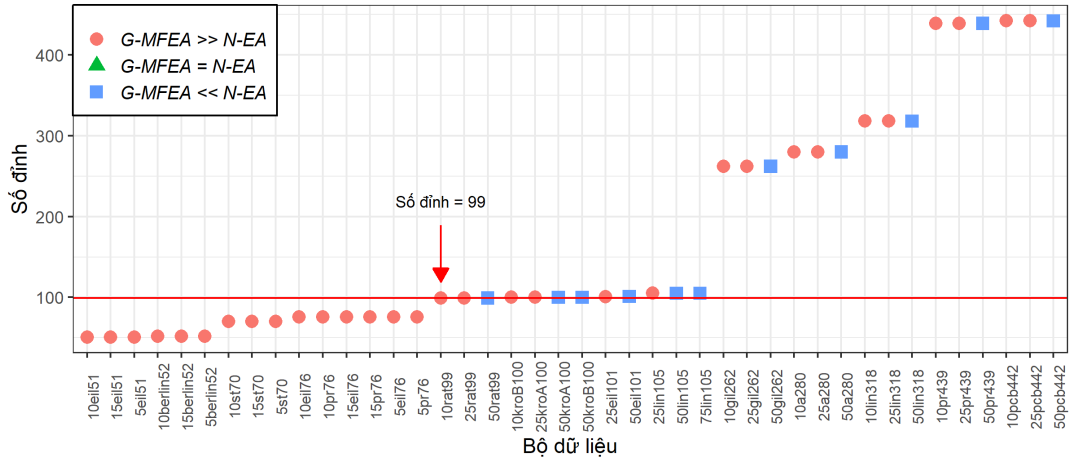
giữa O_{HB-RGA} và số đỉnh của đồ thị không giống nhau giữa các tập dữ liệu. Hình 5.5(c) cho thấy xu hướng không rõ ràng của tương quan này.

Xu hướng của tương quan giữa trong hình 5.6 có điểm tương đồng với xu hướng trong hình 5.5 khi thuật toán G-MFEA cho kết quả trội hơn thuật toán N-EA khi số đỉnh của đồ thị nhỏ hơn 99 (đối với Type 1), 76 (đối với Type 6). Khác với hai tập dữ liệu trên, đối với các bộ dữ liệu thuộc Type 5, thuật toán N-EA không tốt hơn thuật toán G-MFEA trong bất kỳ bộ dữ liệu nào.

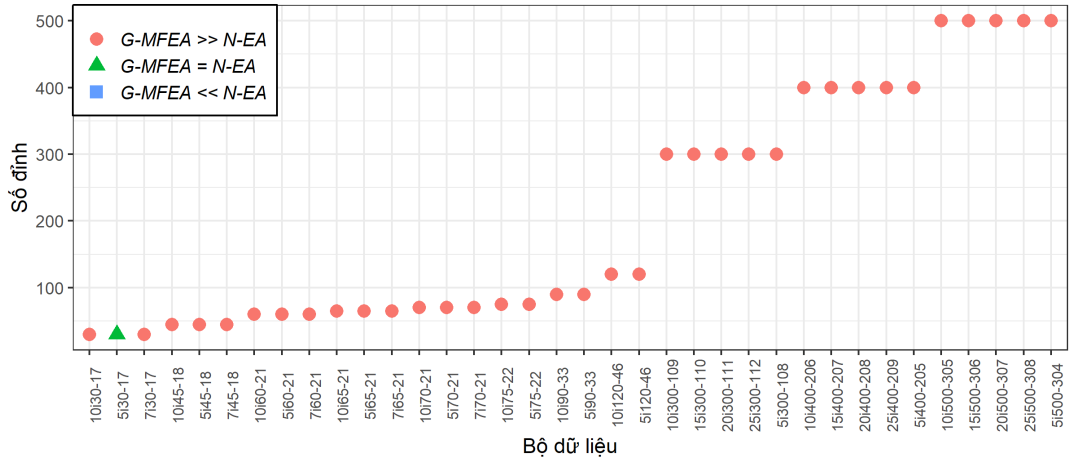
Các phân tích trên cho thấy thuật toán N-EA và HB-RGA không hiệu quả bằng thuật toán G-MFEA khi số cụm hoặc số đỉnh của đồ thị đầu vào nhỏ. Nguyên nhân chính dẫn tới kết quả trên là do thuật toán G-MFEA sử dụng đồng thời cả hai tác vụ để tìm lời giải cho một bộ dữ liệu nên có sự trao đổi thông tin và 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 không gian tìm kiếm lời giải mới của thuật toán EA và khai thác 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

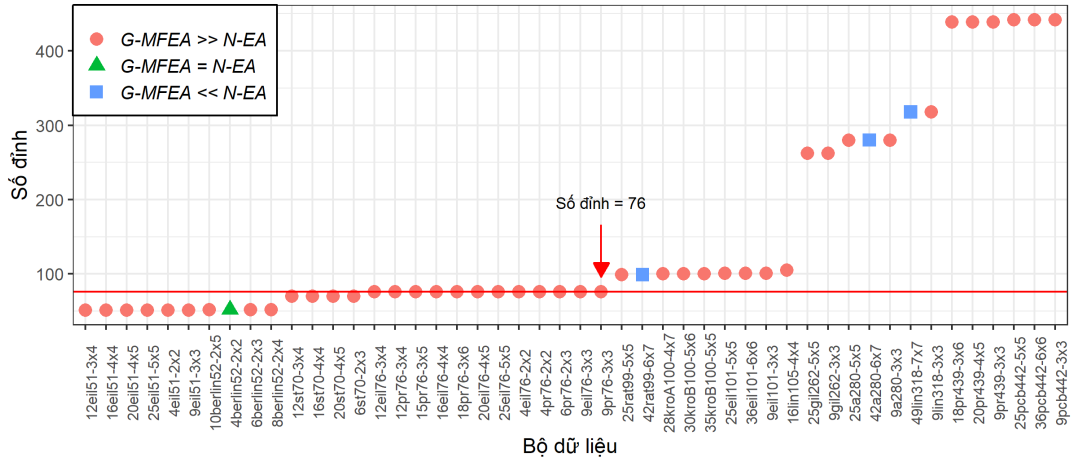
Bảng 23 – bảng 26 trình bày thời gian tính toán (đơn vị tính là phút) của các thuật toán đề xuất. Dữ liệu trong bảng cho thấy thuật toán N-EA có thời gian tính toán thấp hơn các thuật toán còn lại, trong khi thuật toán G-MFEA có thời gian tính toán lớn nhất. Thuật toán N-EA có thời gian thực hiện thấp 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. Do số lượng cụm nhỏ hơn nhiều so với số đỉnh của đồ thị nên thời gian thực hiện các toán tử tiến hóa của thuật



(a) Type 1



(b) Type 5



(c) Type 6

Hình 5.6: Biểu đồ phân tán của kết quả so sánh hai thuật toán G-MFEA và N-EA với số đỉnh của đồ thị.

toán N-EA giảm nhiều so với mã hóa theo tập đỉnh của đồ thị. Cũng do thuật toán N-EA lưu trữ chi phí của mỗi cây khung nhỏ nhất tương ứng với mỗi đỉnh gốc cục bộ trong các cụm để tránh phải tính lại nên cũng làm giảm chi phí tính toán của thuật toán N-EA. Thời gian tính toán của thuật toán G-MFEA lớn do bên cạnh trữ thông tin về các cụm, các cá thể trong thuật toán G-MFEA còn lữu trữ thông tin về các cạnh nối giữa các cụm và thông tin về các đỉnh gốc cục bộ nên các toán tử tiến hóa cần thời gian để xác định các thuộc tính này cho các cá thể con. Bên cạnh đó, mỗi khi tác vụ sử dụng thuật toán HB-RGA thực hiện, thuật toán G-MFEA sẽ xây dựng đồ thị đầu vào cho thuật toán HB-RGA từ cá thể con trong không gian USS, 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, trong đó thời gian tính toán của thuật toán C-EA có xu hướng nhỏ hơn thuật toán HB-RGA trên các bộ dữ liệu có số đỉnh nhỏ hoặc có đỉnh lớn nhưng số cụm bé. Thời gian thực hiện của thuật toán HB-RGA có xu hướng bé hơn so với thuật toán C-EA đối với các bộ dữ liệu có số đỉnh và số cụm lớn.

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

Chi tiết kết quả tìm được từ các thuật toán trên các tập dữ liệu được trình trong các bảng 27 – bảng 29 (phần Phụ lục). Trong các bảng này, giá trị tương ứng với lời giải tốt nhất của mỗi dòng sẽ được in nghiêng, đậm và có mầu đỏ.

Bảng 5.10 trình bày kết quả thống kê của các kiểm định Friedman và Iman-Davenport. Kết quả trong bảng cho thấy rằng các giá trị Friedman

và Iman-Davenport lớn nhiều lần so với các giá trị tới hạn và trị số p nhỏ hơn nhiều lần so với ngưỡng $\alpha = 0.05$, điều đó có nghĩa là có sự khác biệt rõ ràng giữa kết quả thực nghiệm nhận được của các thuật toán với sai số nhỏ hơn 5%. Hay nói cách khác sự khác biệt giữa kết quả các thuật toán trên tập dữ liệu đầy đủ phi metric có ý nghĩa thống kê với ngưỡng $\alpha = 0.05$.

Bảng 5.10: Bảng kết quả kiểm định Friedman và Iman-Davenport (α=0.05)

Friedman			**Iman-Davenport**		
Giá trị Friedman	Giá trị X^2	Trị số p	Giá trị Iman-Davenport	Giá trị F_F	Trị số p
268.3920	9.4877	$0.385 * 10^{-10}$	**628.3538**	2.4021	$3.028 * 10^{-143}$

Bảng 5.11 trình bày giá trị xếp hạng trung bình của các thuật toán được đánh giá bởi các kiểm định Friedman, Friednman Aligned và Quade. Kết quả trong bảng củng cố thêm nhận định về sự khác nhau về chất lượng lời giải tìm được giữa các thuật toán được so sánh. Trong bảng 5.11, thuật toán G-MFEA có thứ hạng nhỏ nhất nên thuật toán G-MFEA có hiệu năng tốt nhất.

Bảng 5.11: Giá trị xếp hạng trung bình của các thuật toán nhận được từ được đánh giá bởi các kiểm định Friedman, Friednman Aligned và Quade

Thuật toán	Friedman	Friedman Aligned	Quade
AAL	5.00	331.400	4.999
HB-RGA	1.820	104.847	1.747
N-EA	2.760	132.080	2.649
C-EA	3.999	269.573	4.000
G-MFEA	1.420	102.099	1.604

Bảng 5.12 trình bày chi tiết hơn về ước lượng kết quả so sánh đối kháng giữa các thuật toán. Tất cả các giá trị tại dòng tương ứng với thuật toán G-MFEA đều giá trị dương khi so sánh với các thuật toán khác. Điều đó có nghĩa là thuật toán G-MFEA có hiệu quả tốt nhất. Dòng tương ứng với thuật toán AAL đều có giá trị âm nên thuật toán này kém hiệu quả nhất. Bảng 5.12 cũng cho thấy hiệu quả của 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.

Bảng 5.12: Ước lượng đối kháng giữa các thuật toán

	AAL	**HB-RGA**	**N-EA**	**C-EA**	**G-MFEA**
AAL	0	$-8.306 * 10^4$	$-7.831 * 10^4$	$-4.275 * 10^{04}$	$-8.379 * 10^4$
HB-RGA	$8.306 * 10^4$	0	4749	$4.031 * 10^4$	-736.6
N-EA	$7.831 * 10^4$	-4749	0	$3.556 * 10^4$	-5486
C-EA	$4.275 * 10^4$	$-4.031 * 10^4$	$-3.556 * 10^4$	0	$-4.104 * 10^4$
G-MFEA	$8.379 * 10^4$	736.6	5486	$4.104 * 10^4$	0

Từ các nhận định dựa trên kết quả trong bảng 5.11 và bảng 5.12, thuật toán điều khiển được chọn là thuật toán điều khiển trong các phân tích thống kê hậu kiểm.

Luận án sử dụng các kiểm định Holm, Holland,... đánh giá kết quả so sánh giữa thuật toán điều khiển G-MFEA với các thuật toán khác. Bảng 5.5 trình bày kết quả đánh giá các giả thuyết khi so sánh giữa thuật toán điều khiển G-MFEA với các thuật toán còn lại.

Bảng 5.13: Bảng giá trị đã hiệu chỉnh của trị số p của các kiểm định Friedman và Quade (G-MFEA là thuật toán điều khiển)

FRIEDMAN						
i	Thuật toán	Unadjusted p	p_{Bonf}	p_{Holm}	p_{Holl}	p_{Li}
1	**AAL**	$1.03 * 10^{-43}$	$4.11 * 10^{-43}$	$4.11 * 10^{-67}$	0.0	$1.17 * 10^{-43}$
2	**C-EA**	$1.65 * 10^{-23}$	$6.59 * 10^{-23}$	$4.94 * 10^{-23}$	0.0	$1.87 * 10^{-23}$
3	**N-EA**	$2.11 * 10^{-7}$	$8.42 * 10^{-7}$	$4.21 * 10^{-7}$	$4.21 * 10^{-7}$	$2.39 * 10^{-7}$
4	HB-RGA	0.121335	0.4853410	0.121335	0.121335	0.121335
QUADE						
i	Thuật toán	Unadjusted p	p_{Bonf}	p_{Holm}	p_{Holl}	p_{Li}
1	**AAL**	$6.37 * 10^{-16}$	$2.55 * 10^{-15}$	$2.55 * 10^{-23}$	0.0	$2.38 * 10^{-15}$
2	**C-EA**	$1.18 * 10^{-8}$	$4.72 * 10^{-8}$	$3.54 * 10^{-8}$	0.0	$4.41 * 10^{-8}$
3	*N-EA*	0.012843	*0.051373*	0.025686	0.025521	0.045748
4	HB-RGA	0.732109	2.928437	0.732109	0.732109	0.732109

Bảng 5.13 trình bày các giá trị đã hiệu chỉnh của trị số p thu được từ các kiểm định Friedman và Quade đối với các so sánh giữa thuật toán G-MFEA và các thuật toán khác. Kết quả trong bảng 5.13 cho thấy:

- Đối với kiểm định Friedman, các thuật toán AAL, C-EA và N-EA (các thuật toán có tên được in đậm) có giá trị của trị số p sau khi

hiệu chỉnh nhỏ hơn 0.05 nên kết quả so sánh giữa các thuật toán này với thuật toán G-MFEA có ý nghĩa thống kê với ngưỡng $\alpha = 0.05$. Nghĩa là thuật toán G-MFEA tốt hơn ba thuật toán AAL, C-EA và N-EA ở ngưỡng được xem xét $\alpha = 0.05$.

- Đối với kiểm định Quade:
  - Giá trị sau khi hiệu chỉnh các thuật toán AAL và C-EA (các thuật toán có tên được in đậm) có giá trị của trị số p sau khi hiệu chỉnh nhỏ hơn 0.05 nên kết quả so sánh giữa các thuật toán này với thuật toán G-MFEA có ý nghĩa thống kê với ngưỡng $\alpha = 0.05$. Hay nói cách khác AAL và C-EA cho kết quả kém hơn thuật toán G-MFEA ở ngưỡng được xem xét $\alpha = 0.05$.
  - Trị số p sau khi hiệu chỉnh của thuật toán N-EA đối với các kiểm định Holm, Holland và Li đều nhỏ hơn 0.05. Nghĩa là đối với các kiểm định trên, thuật toán N-EA kém hơn thuật toán G-MFEA ở ngưỡng được xem xét $\alpha = 0.05$. Trị số p sau khi hiệu chỉnh đối với kiểm định Bonferroni-Dunn (giá trị in nghiêng màu đỏ) là 0.051375. Nghĩa là đối với phương thức Bonferroni-Dunn, kết quả so sánh giữa thuật toán G-MFEA với thuật toán N-EA không có ý nghĩa thống kê ở ngưỡng $\alpha = 0.05$, chỉ có ý nghĩa thống kê ở ngưỡng $\alpha = 0.1$.

b) So sánh kết quả trong từng tập dữ liệu

Bảng tóm tắt kết quả so sánh giữa các thuật toán đối với các bộ dữ liệu đầy đủ phi metric được trình bày trong bảng 5.14. Khác với kết quả trên bộ dữ liệu metric, số lời giải có chi phí bằng nhau tìm được từ các thuật toán trên bộ dữ liệu đầy đủ phi metric rất ít, chỉ có một trường hợp trên bộ dữ liệu N-5I75-22 thuộc tập Type 5 khi hai thuật toán HB-RGA và G-MFEA tìm được lời giải có chi phí bằng nhau. Điều đó có nghĩa là 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:

Bảng 5.14: Bảng tóm tắt so sánh kết quả dựa trên số bộ dữ liệu phi metric

		C-EA	**N-EA**	**HB-RGA**	**G-MFEA**	**Số bộ dữ liệu**
AAL	Type 1	23	23	23	23	**23**
	Type 5	18	18	18	18	**18**
	Type 6	34	34	34	34	**34**
C-EA	Type 1		23	23	23	**23**
	Type 5		18	18	18	**18**
	Type 6		34	34	34	**34**
N-EA	Type 1			20	18	**23**
	Type 5			18	18	**18**
	Type 6			30	28	**34**
HB-RGA	Type 1				17	**23**
	Type 5				15(1)	**18**
	Type 6				22	**34**

- 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.
- Thuật toán N-EA kém hơn thuật toán HB-RGA và thuật toán G-MFEA lần lượt trên 20/23 và 18/23 bộ dữ liệu của tập Type 1, tất cả các bộ dữ liệu của tập Type 5, 30/34 và 28/34 bộ dữ liệu của tập Type 6.
- Thuật toán HB-RGA kém hơn thuật toán G-MFEA trên 17/23 bộ dữ liệu của tập Type 1, 15/18 bộ dữ liệu của tập Type 5 và 22/34 bộ dữ liệu thuộc tập Type 6.

Qua các phân tích trên có thể thấy rằng, chất lượng lời giải tìm được bởi các thuật toán trên các bộ dữ liệu đầy đủ phi metric sắp xếp theo thứ tự giảm dần là: G-MFEA → HB-RGA → N-EA → C-EA → AAL.

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

Kết quả phân tích trong các phần trước đã chỉ ra thuật toán G-MFEA có hiệu quả tốt nhất, hai thuật toán AAL và C-EA kém hơn ba thuật toán còn lại trên tất cả các bộ dữ liệu đã thực nghiệm nên luận án chỉ tập trung phân tích kết quả nhận được của ba thuật toán G-MFEA, N-EA và

HB-RGA theo số đỉnh và số cụm của đồ thị đầu vào.

Hình 5.7 và hình 5.8 minh họa mối quan hệ giữa kết quả so sánh của các thuật toán và số cụm, số đỉnh của đồ thị đầu vào. Trong các hình này, ký hiệu ">>" ("<<") nghĩa là "tốt hơn" ("kém hơn"); ký hiệu "&" nghĩa là "và" hay "đồng thời". Ví dụ, "G-MFEA >> HB-RGA & N-EA" nghĩa là thuật toán G-MFEA tốt hơn cả hai thuật toán HB-RGA và N-EA.

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, ngược lại thuật toán G-MFEA kém hơn cả hai thuật toán còn lại khi số cụm lớn hơn 25. Biểu đồ cũng cho thấy 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) chỉ có ký hiệu hình tròn và hình vuông, điều đó chứng tỏ rằng trên tập dữ liệu Type 5, không có trường hợp nào thuật toán G-MFEA có kết quả kém hơn thuật toán N-EA. Kết quả so sánh giữa các thuật toán trên tập dữ liệu Type 6 (hình 5.7(c)) có nhiều điểm tương đồng với kết quả so sánh trên tập dữ liệu Type 1. 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); số cụm nhỏ nhất để thuật toán G-MFEA kém hơn cả hai thuật toán còn lại là 28 (tại bộ dữ liệu N-28KroA100-4x7).

Kết quả so sánh của các thuật toán trong hai hình 5.8(a) và 5.8(b) có nhiều điểm tương đồng khi số đỉnh của đồ thị đầu vào nhỏ hơn 70 (đối với Type 1), 45 (đối với Type 5) hoặc nằm trong khoảng 70 – 99 (đối với Type 1), 45 – 75 (đối với Type 5) thì thuật toán G-MFEA tốt hơn cả hai thuật toán HB-RGA và N-EA. Bên cạnh đó, hình 5.8(a) cho thấy rằng thuật toán HB-RGA chỉ tìm được kết quả tốt hơn thuật toán G-MFEA khi số đỉnh không nhỏ hơn 99, tương tự đối với thuật toán N-EA là 70. Đối với các bộ dữ liệu thuộc tập Type 5, thuật toán HB-RGA chỉ tốt hơn thuật toán G-MFEA khi số đỉnh không nhỏ hơn 45.

Kết quả so sánh giữa các thuật toán trong hình 5.8(c) không rõ ràng như hai hình 5.8(a) và 5.8(b) khi cùng số đỉnh kết quả so sánh giữa các thuật toán có cả trường hợp hợp tốt hơn và không tốt hơn. Tuy nhiên hình 5.8(c) cũng cho thấy thuật toán N-EA chỉ tốt hơn thuật toán G-MFEA khi số đỉnh của đồ thị không nhỏ hơn 70 và thuật toán G-MFEA chỉ kém hơn cả hai thuật toán HB-RGA và N-EA khi số đỉnh của đồ thị lớn hơn hoặc bằng 99.

5.3. Kết luận chương

Chương này trình bày các phân tích kết quả thực nghiệm của các thuật toán được đề xuất trong luận án và trong các nghiên cứu trước đây. 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.

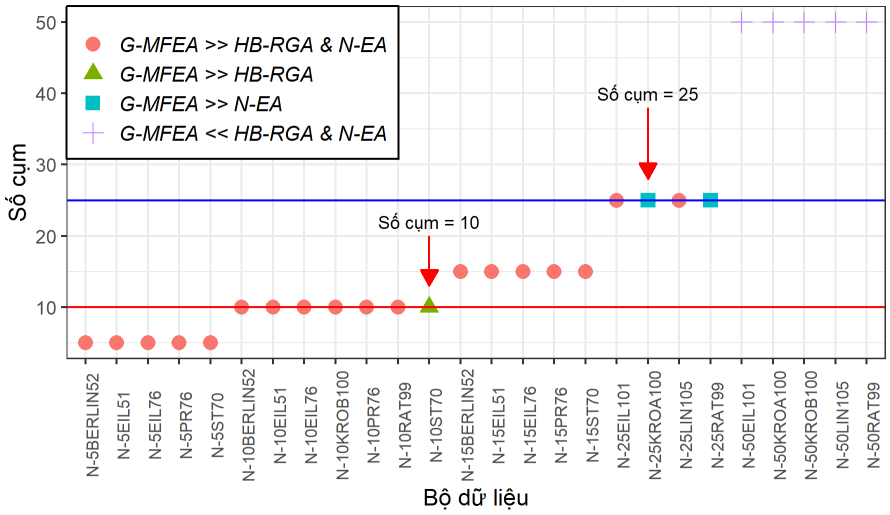
Với mỗi dạng đồ thị, các phân tích được tiến hành theo ba bước. Đầu tiên luận án sử dụng các phân tích thống kê để kiểm chứng ý nghĩa thống kê của các kết quả thực nghiệm thu được và xác định thuật toán có hiệu quả tốt nhất trong các đề xuất. Dựa trên cơ sở của kết phân tích thông kê, luận án sẽ so sánh chi tiết kết quả của các thuật toán đối với mỗi bộ dữ liệu thuộc các tập dữ liệu được lựa chọn thực nghiệm. Tại bước cuối, luận án phân tích ảnh hưởng của một số thuộc tính của đồ thị đầu vào tới kết quả so sánh giữa các thuật toán.

Kết quả phân tích thống kê trên cả đồ thị metric và đồ thị đầy đủ phi metric đã chỉ ra rằng sự khác biệt giữa kết quả của các thuật toán là có ý nghĩa thống kê. Bên cạnh đó, kết quả phân tích đã chỉ ra rằng các thuật toán đề xuất đều cho kết quả tốt hơn thuật toán được công bố trong nghiên cứu trước đây trên cả hai dạng đồ thị được xem xét.

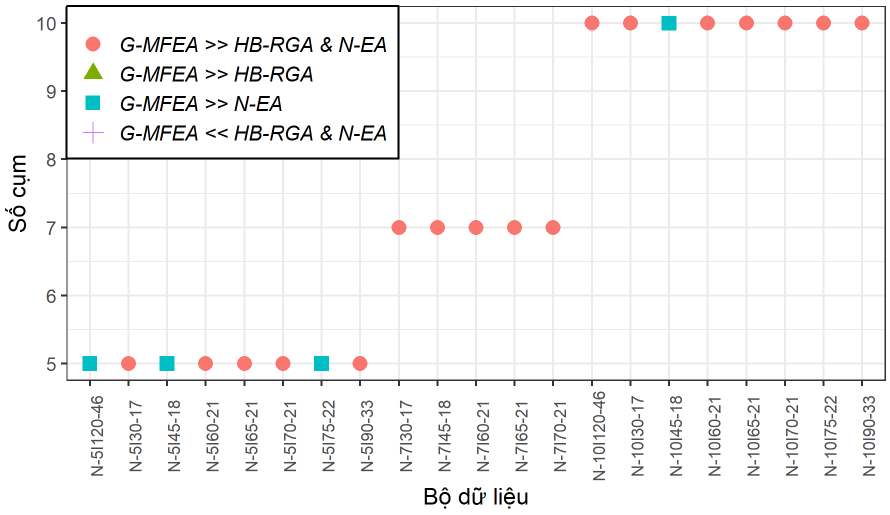
Các phân tích kết quả thực nghiệm cũng chỉ ra rằng thuật toán G-MFEA tìm được lời giải tốt nhất, trong khi thuật toán C-EA tìm được lời giải kém nhất. Lời giải tìm được bởi thuật toán HB-RGA và N-EA có

chất lượng gần nhau, tuy nhiên thuật toán HB-RGA có xu hướng tốt hơn thuật toán N-EA.

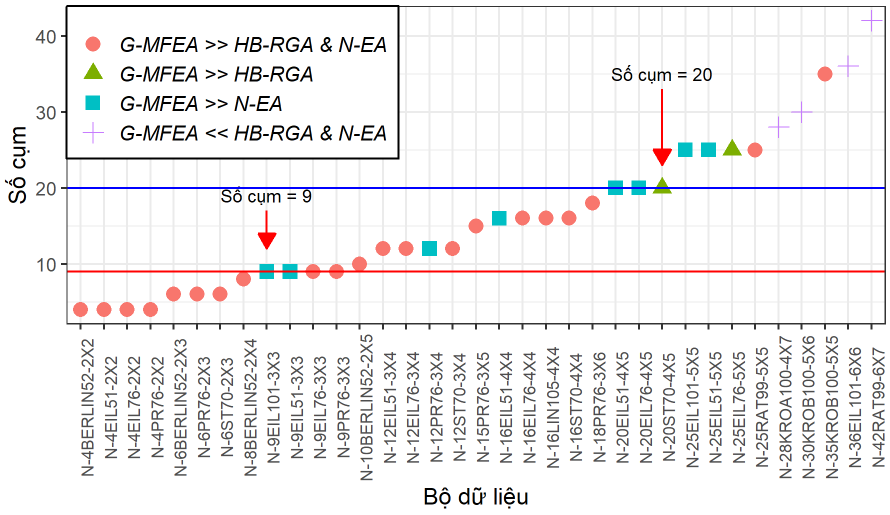
Thông qua phân tích biểu đồ phân tán của kết quả so sánh giữa các thuật toán và số đỉnh, số cụm của đồ thị đầu vào, luận án đã chỉ ra sự ảnh hưởng của số cụm, số đỉnh của đồ thị đầu vào tới kết quả của các thuật toán. Từ đó có thể một phần dự đoán kết quả nhận được của các thuật toán theo hai thuộc tính trên của đồ thị đầu vào.



(a) Type 1

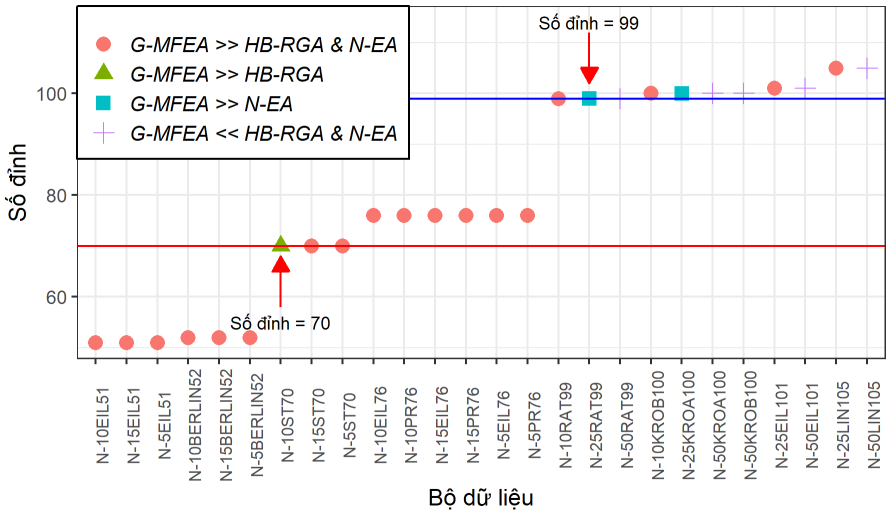


(b) Type 5

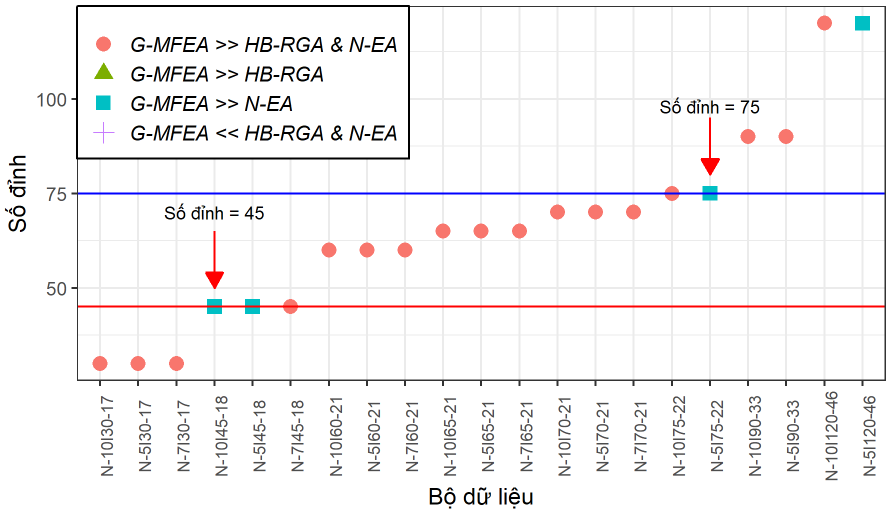


(c) Type 6

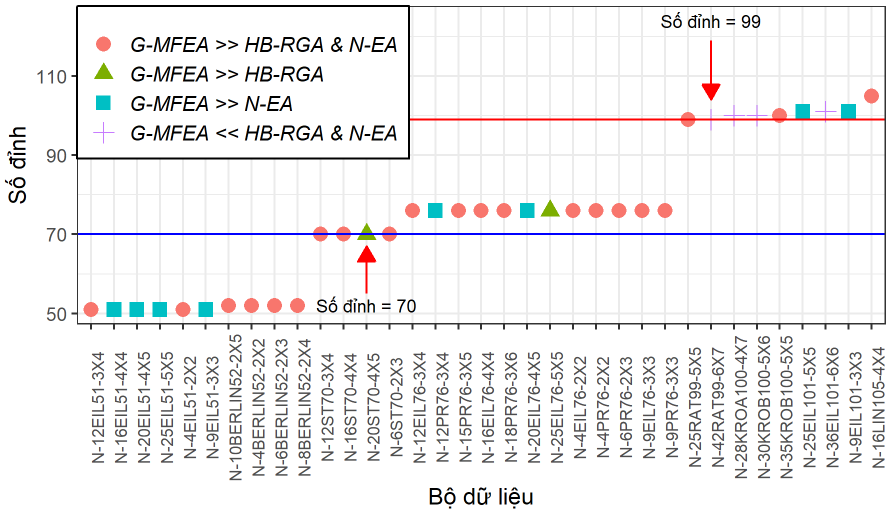
Hình 5.7: Biểu đồ phân tán của kết quả so sánh các thuật toán và số cụm của đồ thị đối với các bộ dữ liệu đầy đủ phi metric.



(a) Type 1



(b) Type 5



(c) Type 6

Hình 5.8: Biểu đồ phân tán của kết quả so sánh các thuật toán và số đỉnh của đồ thị đối với các bộ dữ liệu đầy đủ phi metric.

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. Thuật toán HB-RGA phù hợp khi cần tìm nhanh lời giải của bài toán CluSPT.
- 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. Thuật toán C-EA phù hợp khi cần tìm đồng thời lời giải bài toán CluSPT và lời giải một bài toán cây khung phân cụm khác.
  - 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 N-EA phù hợp khi cần tìm lời giải có chất lượng tốt trên đồ thị đầy đủ với giới hạn thời gian trung bình.

– 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. Thuật toán G-MFEA phù hợp khi cần tìm lời giải có chất lượng tốt và không giới hạn về thời gian.

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 lớn.

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. H. T. Thanh Binh, P. Dinh Thanh, T. Ba Trung and L. Phuong Thao, "Effective Multifactorial Evolutionary Algorithm for Solving the Cluster Shortest Path Tree Problem," 2018 IEEE Congress on Evolutionary Computation (CEC), Rio de Janeiro, 2018, pp. 819-826.

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)*

TÀI LIỆU THAM KHẢO

[1] Aalvanger, G. H., Luong, N. H., Bosman, P. A. N., and Thierens, D. (2018). Heuristics in Permutation GOMEA for Solving the Permutation Flowshop Scheduling Problem. In Auger, A., Fonseca, C. M., Lourenço, N., Machado, P., Paquete, L., and Whitley, D., editors, *Parallel Problem Solving from Nature – PPSN XV*, pages 146–157. Springer International Publishing.

[2] Addario-Berry, L., Broutin, N., Goldschmidt, C., and Miermont, G. (2017). The scaling limit of the minimum spanning tree of the complete graph. *The Annals of Probability*, 45(5):3075–3144. Publisher: Institute of Mathematical Statistics.

[3] Agoston, E. E. (2003). *Introduction to Evolutionary Computing*. Berlin, Springer-Verlag.

[4] Back, T. (1996). *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*. Oxford university press.

[5] Bao, X. and Liu, Z. (2012). An improved approximation algorithm for the clustered traveling salesman problem. *Information Processing Letters*, 112(23):908–910.

[6] Bäck, T., Fogel, D. B., and Michalewicz, Z. (1997). *Handbook of evolutionary computation*. CRC Press.

[7] Bäck, T., Fogel, D. B., and Michalewicz, Z. (2000). *Evolutionary computation 2: Advanced Algorithms and Operators*. CRC press.

[8] Bäck, T., Fogel, D. B., and Michalewicz, Z. (2018). *Evolutionary computation 1: Basic algorithms and operators*. CRC press.

[9] Brabazon, A., O'Neill, M., and McGarraghy, S. (2015). *Natural computing algorithms*. Springer.

[10] Carrasco, J., García, S., Rueda, M. M., Das, S., and Herrera, F. (2020). Recent trends in the use of statistical tests for comparing swarm and evolutionary computing algorithms: Practical guidelines and a critical review. *Swarm and Evolutionary Computation*, page 100665.

[11] Cavalli-Sforza, L. L. and Feldman, M. W. (1973). Cultural versus biological inheritance: phenotypic transmission from parents to children.(A theory of the effect of parental phenotypes on children's phenotypes). *American journal of human genetics*, 25(6):618.

[12] Chambers, L. D. (2019). *Practical handbook of genetic algorithms: complex coding systems*, volume 3. CRC press.

[13] Chen, Y. H. (2017). The clustered and bottleneck clustered selected-internal steiner tree problems. In *The Second Malta Conference in Graph Theory and Combinatorics*.

[14] Cloninger, C. R., Rice, J., and Reich, T. (1979). Multifactorial inheritance with cultural transmission and assortative mating. II. a general model of combined polygenic and cultural inheritance. *American journal of human genetics*, 31(2):176.

[15] Cosma, O., Pop, P. C., and Zelina, I. (2020). A novel genetic algorithm for solving the clustered shortest-path tree problem. *Carpathian Journal of Mathematics*, 36(3):401–414. Publisher: JSTOR.

[16] Cosma, O., Pop, P. C., and Zelina, I. (2021). An Effective Genetic Algorithm for Solving the Clustered Shortest-Path Tree Problem. *IEEE Access*, 9:15570–15591. Publisher: IEEE.

[17] Da, B., Gupta, A., Ong, Y.-S., and Feng, L. (2016). Evolutionary multitasking across single and multi-objective formulations for improved

problem solving. In *Evolutionary Computation (CEC), 2016 IEEE Congress on*, pages 1695–1701. IEEE.

[18] De Jong, K. (2020). Evolutionary computation: a unified approach. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, pages 327–342.

[19] Deb, K. and Sinha, A. (2014). Evolutionary bilevel optimization (ebo). In *Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation*, pages 857–876. ACM.

[20] Del Ser, J., Osaba, E., Molina, D., Yang, X.-S., Salcedo-Sanz, S., Camacho, D., Das, S., Suganthan, P. N., Coello, C. A. C., and Herrera, F. (2019). Bio-inspired computation: Where we stand and what's next. *Swarm and Evolutionary Computation*, 48:220–250. Publisher: Elsevier.

[21] D'Emidio, M., Forlizzi, L., Frigioni, D., Leucci, S., and Proietti, G. (2016). On the Clustered Shortest-Path Tree Problem. In *ICTCS*, pages 263–268.

[22] D'Emidio, M., Forlizzi, L., Frigioni, D., Leucci, S., and Proietti, G. (2019). Hardness, approximability, and fixed-parameter tractability of the clustered shortest-path tree problem. *Journal of Combinatorial Optimization*, pages 1–20.

[23] Di Battista, G., Drovandi, G., and Frati, F. (2009). How to draw a clustered tree. *Journal of Discrete Algorithms*, 7(4):479–499.

[24] Dial, R., Glover, F., Karney, D., and Klingman, D. (1979). A computational analysis of alternative algorithms and labeling techniques for finding shortest path trees. *Networks*, 9(3):215–248.

[25] Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271.

[26] Ding, C., Cheng, Y., and He, M. (2007). Two-level genetic algorithm for clustered traveling salesman problem with application in large-scale tsps. *Tsinghua Science and technology*, 12(4):459–465.

[27] Dror, M., Haouari, M., and Chaouachi, J. (2000). Generalized spanning trees. *European Journal of Operational Research*, 120(3):583–592.

[28] Fampa, M. (2019). Insight into the computation of steiner minimal trees in euclidean space of general dimension. *Discrete Applied Mathematics*.

[29] Feng, L., Zhou, W., Zhou, L., Jiang, S., Zhong, J., Da, B., Zhu, Z., and Wang, Y. (2017). An empirical study of multifactorial pso and multifactorial de. In *Evolutionary Computation (CEC), 2017 IEEE Congress on*, pages 921–928. IEEE.

[30] Feremans, C., Labbé, M., and Laporte, G. (2003). Generalized network design problems. *European Journal of Operational Research*, 148(1):1–13.

[31] Festa, P. (2008). Shortest path tree algorithms. In *Encyclopedia of Optimization*, pages 3507–3519. Springer.

[32] Gao, W., Friedrich, T., Neumann, F., and Hercher, C. (2018). Randomized greedy algorithms for covering problems. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 309–315.

[33] Gerla, M. and Fratta, L. (1988). Tree structured fiber optics MANs. *IEEE journal on selected areas in communications*, 6(6):934–943.

[34] Guo, S.-M. and Yang, C.-C. (2014). Enhancing differential evolution utilizing eigenvector-based crossover operator. *IEEE Transactions on Evolutionary Computation*, 19(1):31–49.

[35] Gupta, A., Mańdziuk, J., and Ong, Y.-S. (2015). Evolutionary multitasking in bi-level optimization. *Complex & Intelligent Systems*, 1(1-4):83–95.

[36] Gupta, A., Ong, Y. S., Da, B., Feng, L., and Handoko, S. D. (2016a). Landscape synergy in evolutionary multitasking. In *Evolutionary Computation (CEC), 2016 IEEE Congress on*, pages 3076–3083. IEEE.

[37] Gupta, A., Ong, Y.-S., and Feng, L. (2016b). Multifactorial evolution: toward evolutionary multitasking. *IEEE Transactions on Evolutionary Computation*, 20(3):343–357.

[38] Gupta, A., Ong, Y.-S., Feng, L., and Tan, K. C. (2016c). Multiobjective multifactorial optimization in evolutionary multitasking. *IEEE transactions on cybernetics*.

[39] Hao, X., Qu, R., and Liu, J. (2020). A Unified Framework of Graph-based Evolutionary Multitasking Hyper-heuristic. *IEEE Transactions on Evolutionary Computation*. Publisher: IEEE.

[40] Helsgaun, K. (2011). Solving the bottleneck traveling salesman problem using the lin-kernighan-helsgaun algorithm. *Computer Science Research Report*, (143):1–45.

[41] Holland, J. H. (1975). Adaptation in natural and artificial systems. an introductory analysis with application to biology, control, and artificial intelligence. *Ann Arbor, MI: University of Michigan Press*, pages 439–444.

[42] Ihler, E., Reich, G., and Widmayer, P. (1999). Class Steiner trees and VLSI-design. *Discrete Applied Mathematics*, 90(1-3):173–194.

[43] Johnson, D. B. (1973). A note on dijkstra's shortest path algorithm. *Journal of the ACM (JACM)*, 20(3):385–388.

[44] Julstrom, B. A. (2005). The blob code is competitive with edge-sets in genetic algorithms for the minimum routing cost spanning tree problem. In *Proceedings of the 7th annual conference on Genetic and evolutionary computation*, pages 585–590. ACM.

[45] Khachay, M. and Neznakhina, K. (2020). Complexity and approximability of the Euclidean generalized traveling salesman problem in grid clusters. *Annals of Mathematics and Artificial Intelligence*, 88(1-3):53–69. ISBN: 1012-2443 Publisher: Springer.

[46] Khuller, S., Raghavachari, B., and Young, N. (1995). Balancing minimum spanning trees and shortest-path trees. *Algorithmica*, 14(4):305–321.

[47] Koziel, S. and Yang, X.-S. (2011). *Computational optimization, methods and algorithms*, volume 356. Springer Science & Business Media.

[48] Kruskal, J. B. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical society*, 7(1):48–50. Publisher: JSTOR.

[49] Li, G., Lin, Q., and Gao, W. (2019). Multifactorial Optimization via Explicit Multipopulation Evolutionary Framework. *Information Sciences*.

[50] Liao, P., Sun, C., Zhang, G., and Jin, Y. (2020). Multi-surrogate multitasking optimization of expensive problems. *Knowledge-Based Systems*, 205:106262.

[51] Lima, T. W. d., Delbem, A. C. B., Soares, A. d. S., Federson, F. M., Junior, J. B. A. L., and Baalen, J. V. (2016). Node-depth phylogenetic-based encoding, a spanning-tree representation for evolutionary algorithms. part I: Proposal and properties analysis. *Swarm and Evolutionary Computation*, 31:1–10.

[52] Lin, C.-W. and Wu, B. Y. (2015). A 2-approximation algorithm for the clustered minimum routing cost tree problem. In *Intelligent Systems and Applications: Proceedings of the International Computer Symposium (ICS) Held at Taichung, Taiwan, December 12–14, 2014*, volume 274, page 3. IOS Press.

[53] Lin, C.-W. and Wu, B. Y. (2016). On the minimum routing cost clustered tree problem. *Journal of Combinatorial Optimization*, pages 1–16.

[54] Liu, J. and Tan, K. C. (2017). *Evolutionary Computation and Complex Networks*. Springer.

[55] Masone, A., Nenni, M. E., Sforza, A., and Sterle, C. (2019). The minimum routing cost tree problem. *Soft Computing*, 23(9):2947–2957.

[56] Mestria, M., Ochi, L. S., and de Lima Martins, S. (2013). GRASP with path relinking for the symmetric euclidean clustered traveling salesman problem. *Computers & Operations Research*, 40(12):3218–3229.

[57] Myung, Y.-S., Lee, C.-H., and Tcha, D.-W. (1995). On the generalized minimum spanning tree problem. *Networks*, 26(4):231–241.

[58] Ong, Y. S. (2015). Towards Evolutionary Multitasking: A New Paradigm. In *Proceedings of the Sixth International Symposium on Information and Communication Technology*, pages 2–2, Hue City, Viet Nam. ACM.

[59] Palmer, C. C. and Kershenbaum, A. (1994). Representing trees in genetic algorithms. In *Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence*, pages 379–384. IEEE.

[60] Paulden, T. and Smith, D. K. (2006). Recent advances in the study of the dandelion code, happy code, and blob code spanning tree rep-

resentations. In *2006 IEEE International Conference on Evolutionary Computation*, pages 2111–2118. IEEE.

[61] Perfecto, C., Bilbao, M. N., Del Ser, J., and Ferro, A. (2016a). A simulation-based quantitative analysis on the topological heritability of Dandelion-encoded meta-heuristics for tree optimization problems. *Soft Computing*, pages 1–14.

[62] Perfecto, C., Bilbao, M. N., Del Ser, J., Ferro, A., and Salcedo-Sanz, S. (2016b). Dandelion-encoded harmony search heuristics for opportunistic traffic offloading in synthetically modeled mobile networks. In *Harmony Search Algorithm*, pages 133–145. Springer.

[63] Poloczek, M. and Szegedy, M. (2012). Randomized greedy algorithms for the maximum matching problem with new analysis. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 708–717. IEEE.

[64] Pop, P., Matei, O., and Pintea, C. (2018a). A two-level diploid genetic based algorithm for solving the family traveling salesman problem. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 340–346. ACM.

[65] Pop, P., Matei, O., and Pintea, C. (2018b). A two-level diploid genetic based algorithm for solving the family traveling salesman problem. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 340–346. ACM.

[66] Pop, P., Oliviu, M., and Sabo, C. (2017). A hybrid diploid genetic based algorithm for solving the generalized traveling salesman problem. In *International Conference on Hybrid Artificial Intelligence Systems*, pages 149–160. Springer.

[67] Pop, P. C., Fuksz, L., Marc, A. H., and Sabo, C. (2018c). A novel

two-level optimization approach for clustered vehicle routing problem. *Computers & Industrial Engineering*, 115:304–318.

[68] Pop, P. C., Matei, O., Sabo, C., and Petrovan, A. (2018d). A two-level solution approach for solving the generalized minimum spanning tree problem. *European Journal of Operational Research*, 265(2):478–487.

[69] Prim, R. C. (1957). Shortest connection networks and some generalizations. *Bell Labs Technical Journal*, 36(6):1389–1401.

[70] Prisco, J. (1986). Fiber optic regional area networks in New York and Dallas. *IEEE journal on selected areas in communications*, 4(5):750–757.

[71] Raidl, G. R. and Julstrom, B. A. (2003). Edge sets: an effective evolutionary coding of spanning trees. *IEEE Transactions on evolutionary computation*, 7(3):225–239.

[72] Reinelt, G. (1994). *The traveling salesman: computational solutions for TSP applications*. Springer-Verlag.

[73] Rice, J., Cloninger, C. R., and Reich, T. (1978). Multifactorial inheritance with cultural transmission and assortative mating. I. Description and basic properties of the unitary models. *American journal of human genetics*, 30(6):618.

[74] Shu-Xi, W. (2012). The improved dijkstra's shortest path algorithm and its application. *Procedia Engineering*, 29:1186–1190.

[75] Sinha, A., Malo, P., and Deb, K. (2017). Evolutionary bilevel optimization: An introduction and recent advances. In *Recent Advances in Evolutionary Multi-objective Optimization*, pages 71–103. Springer.

[76] Thanh, P. D. (2018). CluSPT instances. Mendeley Data v3, http://dx.doi.org/10.17632/b4gcgybvt6.3.

[77] Thanh, P. D., Binh, H. T. T., Long, N. B., et al. (2019). A heuristic based on randomized greedy algorithms for the clustered shortest-path tree problem. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pages 2915–2922. IEEE.

[78] Trivedi, A., Srinivasan, D., Sanyal, K., and Ghosh, A. (2016). A survey of multiobjective evolutionary algorithms based on decomposition. *IEEE Transactions on Evolutionary Computation*, 21(3):440–462. Publisher: IEEE.

[79] Wang, N., Xu, Q., Fei, R., Yang, J., and Wang, L. (2019). Rigorous Analysis of Multi-Factorial Evolutionary Algorithm as Multi-Population Evolution Model. *International Journal of Computational Intelligence Systems*.

[80] Wen, Y.-W. and Ting, C.-K. (2017). Parting ways and reallocating resources in evolutionary multitasking. In *Evolutionary Computation (CEC), 2017 IEEE Congress on*, pages 2404–2411. IEEE.

[81] Whitley, D. (1994). A genetic algorithm tutorial. *Statistics and computing*, 4(2):65–85. Publisher: Springer.

[82] Winter, P. and Zachariasen, M. (1997). Euclidean steiner minimum trees: An improved exact algorithm. *Networks: An International Journal*, 30(3):149–166.

[83] Wu, B. Y., Lancia, G., Bafna, V., Chao, K.-M., Ravi, R., and Tang, C. Y. (2000). A polynomial-time approximation scheme for minimum routing cost spanning trees. *SIAM Journal on Computing*, 29(3):761–778.

[84] Wu, B. Y. and Lin, C.-W. (2014a). Clustered trees with minimum inter-cluster distance. In *Computational Science and Engineering (CSE), 2014 IEEE 17th International Conference on*, pages 1138–1141. IEEE.

[85] Wu, B. Y. and Lin, C.-W. (2014b). Clustered trees with minimum inter-cluster distance. In *Computational Science and Engineering (CSE), 2014 IEEE 17th International Conference on*, pages 1138–1141. IEEE.

[86] Wu, B. Y. and Lin, C.-W. (2015). On the clustered Steiner tree problem. *Journal of Combinatorial Optimization*, 30(2):370–386.

[87] Wu, B. Y., Lin, C.-W., and Chen, L.-H. (2014). The steiner ratio of the clustered steiner tree problem is three. *Unpublished manuscript*.

[88] Xu, H., Zeng, W., Zhang, D., and Zeng, X. (2017). MOEA/HD: A multiobjective evolutionary algorithm based on hierarchical decomposition. *IEEE transactions on cybernetics*, 49(2):517–526. Publisher: IEEE.

[89] Xu, M., Liu, Y., Huang, Q., Zhang, Y., and Luan, G. (2007). An improved dijkstra's shortest path algorithm for sparse network. *Applied Mathematics and Computation*, 185(1):247–254.

[90] Yang, C., Ding, J., Jin, Y., Wang, C., and Chai, T. (2018). Multitasking Multi-objective Evolutionary Operational Indices Optimization of Beneficiation Processes. *IEEE Transactions on Automation Science and Engineering*.

[91] Yang, X.-S. (2010). *Nature-inspired metaheuristic algorithms*. Luniver press.

[92] Yang, X.-S. (2017). *Nature-Inspired Algorithms and Applied Optimization*. Springer.

[93] Yang, Z., Xiao, M.-Q., Ge, Y.-W., Feng, D.-L., Zhang, L., Song, H.-F., and Tang, X.-L. (2017). A double-loop hybrid algorithm for the traveling salesman problem with arbitrary neighbourhoods. *European Journal of Operational Research*.

[94] Yao, S., Dong, Z., Wang, X., and Ren, L. (2020). A Multiobjective multifactorial optimization algorithm based on decomposition and dynamic resource allocation strategy. *Information Sciences*, 511:18–35. ISBN: 0020-0255 Publisher: Elsevier.

[95] Zhang, X.-D. (2020). Evolutionary computation. In *A Matrix Algebra Approach to Artificial Intelligence*, pages 681–803. Springer.

[96] Zhong, Y., Wang, L., Lin, M., and Zhang, H. (2019). Discrete pigeon-inspired optimization algorithm with Metropolis acceptance criterion for large-scale traveling salesman problem. *Swarm and Evolutionary Computation*, 48:134–144. Publisher: Elsevier.

[97] Zhou, L., Feng, L., Liu, K., Chen, C., Deng, S., Xiang, T., and Jiang, S. (2019). Towards Effective Mutation for Knowledge Transfer in Multifactorial Differential Evolution. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pages 1541–1547. IEEE.

[98] Zhou, L., Feng, L., Tan, K. C., Zhong, J., Zhu, Z., Liu, K., and Chen, C. (2020). Toward Adaptive Knowledge Transfer in Multifactorial Evolutionary Computation. *IEEE Transactions on Cybernetics*. Publisher: IEEE.

[99] Zhou, L., Feng, L., Zhong, J., Ong, Y.-S., Zhu, Z., and Sha, E. (2016). Evolutionary multitasking in combinatorial search spaces: A case study in capacitated vehicle routing problem. In *Computational Intelligence (SSCI), 2016 IEEE Symposium Series on*, pages 1–8. IEEE.

PHỤ LỤC

Bảng 15: Thông tin về các bộ dữ liệu đồ thị metric thuộc Type 3 và Type 4

	Tên bộ dữ liệu	**Số đỉnh**	**Số cụm**	**Số đỉnh trung bình**	**Thứ tự cụm lớn nhất**	**Số đỉnh trong cụm lớn nhất**
Type 3	6i300	300	6	50.0	5	73
	6i350	350	6	58.3	5	80
	6i400	400	6	66.7	5	92
	6i450	450	6	75.0	5	100
	6i500	500	6	83.3	5	112
Type 4	4i200a	200	4	50	0	50
	4i200h	200	4	50	0	50
	4i200x1	200	4	50	0	50
	4i200x2	200	4	50	0	50
	4i200z	200	4	50	0	50
	4i400a	400	4	100	0	100
	4i400h	400	4	100	0	100
	4i400x1	400	4	100	0	100
	4i400x2	400	4	100	0	100
	4i400z	400	4	100	0	100

Bảng 16: Thông tin về các bộ dữ liệu đồ thị metric thuộc Type 1

	Tên bộ dữ liệu	Số đỉnh	Số cụm	Số đỉnh trung bình	Thứ tự cụm lớn nhất	Số đỉnh trong cụm lớn nhất
Kích thước nhỏ	10berlin52	52	10	5.2	4	11
	10eil51	51	10	5.1	0	6
	10eil76	76	10	7.6	7	13
	10kroB100	100	10	10.0	7	17
	10pr76	76	10	7.6	0	11
	10rat99	99	10	9.9	8	13
	10st70	70	10	7.0	4	11
	15berlin52	52	15	3.5	8	10
	15eil51	51	15	3.4	4	5
	15eil76	76	15	5.1	4	9
	15pr76	76	15	5.1	12	9
	15st70	70	15	4.7	9	8
	25eil101	101	25	4.0	1	7
	25kroA100	100	25	4.0	10	7
	25lin105	105	25	4.2	13	12
	25rat99	99	25	4.0	4	7
	50eil101	101	50	2.0	13	7
	50kroA100	100	50	2.0	35	5
	50kroB100	100	50	2.0	2	4
	50lin105	105	50	2.1	15	5
	50rat99	99	50	2.0	4	4
	5berlin52	52	5	10.4	1	15
	5eil51	51	5	10.2	0	14
	5eil76	76	5	15.2	2	17
	5pr76	76	5	15.2	4	22
	5st70	70	5	14.0	3	19
	75lin105	105	75	1.4	45	3
Kích thước lớn	10a280	280	10	28.0	3	36
	10gil262	262	10	26.2	3	33
	10lin318	318	10	31.8	1	40
	10pcb442	442	10	44.2	4	54
	10pr439	439	10	43.9	0	76
	25a280	280	25	11.2	9	15
	25gil262	262	25	10.5	5	18
	25lin318	318	25	12.7	5	24
	25pcb442	442	25	17.7	3	32
	25pr439	439	25	17.6	9	47
	50a280	280	50	5.6	2	10
	50gil262	262	50	5.2	31	11
	50lin318	318	50	6.4	40	13
	50pcb442	442	50	8.8	43	17
	50pr439	439	50	8.8	11	21

Bảng 17: Thông tin về các bộ dữ liệu đồ thị metric thuộc Type 5

	Tên bộ dữ liệu	Số đỉnh	Số cụm	Số đỉnh trung bình	Thứ tự cụm lớn nhất	Số đỉnh trong cụm lớn nhất
Kích thước nhỏ	10i120-46	120	10	12.0	1	17
	10i30-17	30	10	3.0	5	5
	10i45-18	45	10	4.5	4	6
	10i60-21	60	10	6.0	8	10
	10i65-21	65	10	6.5	8	10
	10i70-21	70	10	7.0	8	12
	10i75-22	75	10	7.5	3	12
	10i90-33	90	10	9.0	4	12
	5i120-46	120	5	24.0	0	29
	5i30-17	30	5	6.0	4	10
	5i45-18	45	5	9.0	2	12
	5i60-21	60	5	12.0	4	16
	5i65-21	65	5	13.0	4	18
	5i70-21	70	5	14.0	4	20
	5i75-22	75	5	15.0	4	22
	5i90-33	90	5	18.0	1	20
	7i30-17	30	7	4.3	6	7
	7i45-18	45	7	6.4	6	11
	7i60-21	60	7	8.6	6	13
	7i65-21	65	7	9.3	6	14
	7i70-21	70	7	10.0	6	15
Kích thước lớn	10i300-109	300	10	30.0	5	40
	10i400-206	400	10	40.0	0	46
	10i500-305	500	10	50.0	1	63
	15i300-110	300	15	20.0	0	28
	15i400-207	400	15	26.7	13	34
	15i500-306	500	15	33.3	2	38
	20i300-111	300	20	15.0	14	23
	20i400-208	400	20	20.0	7	26
	20i500-307	500	20	25.0	15	33
	25i300-112	300	25	12.0	7	18
	25i400-209	400	25	16.0	2	25
	25i500-308	500	25	20.0	15	34
	5i300-108	300	5	60.0	4	67
	5i400-205	400	5	80.0	4	85
	5i500-304	500	5	100.0	2	105

Bảng 18: Thông tin về các bộ dữ liệu đồ thị metric thuộc Type 6

	Tên bộ dữ liệu	Số đỉnh	Số cụm	Số đỉnh trung bình	Thứ tự cụm lớn nhất	Số đỉnh trong cụm lớn nhất
Kích thước nhỏ	10berlin52-2x5	52	10	5.2	4	21
	12eil51-3x4	51	12	4.2	1	6
	12eil76-3x4	76	12	6.3	5	9
	12pr76-3x4	76	12	6.3	1	11
	12st70-3x4	70	12	5.8	1	8
	15pr76-3x5	76	15	5.1	1	11
	16eil51-4x4	51	16	3.2	9	5
	16eil76-4x4	76	16	4.8	10	10
	16lin105-4x4	105	16	6.6	12	18
	16st70-4x4	70	16	4.4	3	7
	18pr76-3x6	76	18	4.2	1	9
	20eil51-4x5	51	20	2.5	11	5
	20eil76-4x5	76	20	3.8	10	10
	20st70-4x5	70	20	3.5	2	6
	25eil101-5x5	101	25	4.0	16	14
	25eil51-5x5	51	25	2.0	13	4
	25eil76-5x5	76	25	3.0	13	7
	25rat99-5x5	99	25	4.0	8	7
	28kroA100-4x7	100	28	3.6	6	7
	2lin105-2x1	105	2	52.5	0	57
	30kroB100-5x6	100	30	3.3	4	9
	35kroB100-5x5	100	35	2.9	4	9
	36eil101-6x6	101	36	2.8	25	8
	42rat99-6x7	99	42	2.4	0	4
	4berlin52-2x2	52	4	13.0	0	23
	4eil51-2x2	51	4	12.8	1	14
	4eil76-2x2	76	4	19.0	3	22
	4pr76-2x2	76	4	19.0	1	27
	6berlin52-2x3	52	6	8.7	2	24
	6pr76-2x3	76	6	12.7	1	18
	6st70-2x3	70	6	11.7	1	16
	8berlin52-2x4	52	8	6.5	2	18
	9eil101-3x3	101	9	11.2	6	17
	9eil51-3x3	51	9	5.7	1	8
	9eil76-3x3	76	9	8.4	5	12
	9pr76-3x3	76	9	8.4	1	13
Kích thước lớn	18pr439-3x6	439	18	24.4	8	122
	20pr439-4x5	439	20	21.9	7	105
	25a280-5x5	280	25	11.2	10	20
	25gil262-5x5	262	25	10.5	7	17
	25pcb442-5x5	442	25	17.7	19	28
	36pcb442-6x6	442	36	12.3	17	23
	42a280-6x7	280	42	6.7	8	12
	49gil262-7x7	262	49	5.3	3	10
	49lin318-7x7	318	49	6.5	10	16
	9a280-3x3	280	9	31.1	6	39
	9gil262-3x3	262	9	29.1	1	45
	9lin318-3x3	318	9	35.3	2	39
	9pcb442-3x3	442	9	49.1	3	64
	9pr439-3x3	439	9	48.8	5	139

Bảng 19: Kết quả thực nghiệm của các thuật toán trên bộ dữ liệu đồ thị metric thuộc Type 1

		AAL	**SLA-M**	**HB-RGA**			**N-EA**			**C-EA**			**G-MFEA**		
	Instances	Avg	Cost	BF	Avg	RPD	BF	Avg	RPD	BF	Avg	RPD	BF	Avg	RPD
Kích thước nhỏ	10berlin52	82619.1	43724.1	43738.6	43971	0.6	43954.0	44237.6	1.2	50221.7	59277.1	35.6	43724.1	***43724.1***	0.0
	10eil51	4489.1	1713.2	1713.2	1723.2	0.6	1741.5	1770.6	3.4	1976.4	2492.4	45.5	1713.2	***1713.2***	0.0
	10eil76	7220.5	2203.3	2203.3	2208.4	0.2	2264.5	2315.6	5.1	2653.3	3428	55.6	2203.3	***2203.3***	0.0
	10kroB100	384046.7	140522.2	140635.1	141951.4	1.0	143108.6	147539.7	5.0	196536.0	244850.8	74.2	140551.2	*140597.9*	0.1
	10pr76	1042986.0	522213.8	522572.2	525733.1	0.7	531536.7	544954.5	4.4	663067.8	829115	58.8	522213.8	*522340.4*	0.0
	10rat99	22402.3	7520.2	7520.2	7562.1	0.6	7697.8	7899.4	5.0	10689.9	13301.7	76.9	7520.2	*7524*	0.1
	10st70	8057.9	3095.2	3099.5	3131.7	1.2	3098.7	3191.1	3.1	4357.3	5106.6	65.0	3095.2	*3095.7*	0.0
	15berlin52	55735.5	26312.0	26312.0	26437.7	0.5	26463.1	26867.8	2.1	35785.4	51857.5	97.1	26315.5	*26351.7*	0.2
	15eil51	2833.5	1306.4	1306.7	1313.8	0.6	1313.4	1336.5	2.3	2059.3	2692	106.1	1306.8	*1309.1*	0.2
	15eil76	8147.0	2909.1	2911.3	2921.8	0.4	2955.3	3047.8	4.8	3801.3	4965.9	70.7	2909.1	*2913.1*	0.1
	15pr76	1627671.1	704600.6	705017.3	708944.9	0.6	714652.2	728128	3.3	947662.8	1160032	64.6	705226.1	*706505.5*	0.3
	15st70	10074.1	4120.1	4129.9	4147.3	0.7	4145.8	4230.1	2.7	5215.4	6770.2	64.3	4126.7	*4135.5*	0.4
	25eil101	12327.8	4679.0	4680.8	*4686.1*	0.2	4826.6	4885.5	4.4	6998.9	8940.9	91.1	4700.4	4727.9	1.0
	25kroA100	324160.0	147195.0	147239.0	*147716.8*	0.4	150157.7	153155.6	4.0	262653.6	361542.8	145.6	148767.9	149708.1	1.7
	25lin105	211355.4	97944.7	98087.5	*98502.9*	0.6	98991.8	100615.8	2.7	179859.9	280182.5	186.1	98941.4	100585.3	2.7
	25rat99	16747.0	6841.5	6846.3	*6867.8*	0.4	7056.0	7162.3	4.7	12351.0	18127.8	165.0	6930.9	7022.3	2.6
	50eil101	9195.2	3825.3	3827.3	*3828.1*	0.1	3890.7	3919.7	2.5	9907.4	13339	248.7	4034.7	4178.1	9.2
	50kroA100	319270.0	159647.2	159815.2	*160029.9*	0.2	160547.4	161889.6	1.4	442339.9	648645.6	306.3	173113.3	179506.1	12.4
	50kroB100	311384.2	133104.5	133135.4	*133325.8*	0.2	134077.5	135332.2	1.7	472233.0	587388.7	341.3	149465.6	157831.1	18.6
	50lin105	273675.6	145829.1	145869.9	*145951.8*	0.1	146367.1	147175.4	0.9	336318.3	490065.3	236.1	151901.5	154680.7	6.1
	50rat99	19909.6	8007.4	8010.6	*8016.8*	0.1	8104.5	8132.4	1.6	23111.6	32554.7	306.6	8728.0	9002.1	12.4
	5berlin52	48738.1	22746.4	22746.4	23106.9	1.6	22746.4	22938.2	0.8	30894.6	35044.3	54.1	22746.4	***22746.4***	0.0

	5eil51	4397.7	1769.4	1770.5	1792.3	1.3	1769.4	1775.3	0.3	1946.7	2140.4	21.0	1769.4	***1769.4***	0.0
	5eil76	6539.8	2630.8	2630.8	2658.4	1.0	2630.8	2693.1	2.4	3410.7	3757.3	42.8	2630.8	***2630.8***	0.0
	5pr76	1688544.9	585008.0	585008.0	589778.1	0.8	585008.0	591547	1.1	756088.9	829628.1	41.8	585008.0	***585008***	0.0
	5st70	10270.5	4520.1	4520.1	4562.8	0.9	4520.1	4544.9	0.5	5138.6	5580.7	23.5	4520.1	***4520.1***	0.0
Kích thước lớn	10a280	98309.2	27925.2	28193.6	28515.8	2.1	28690.9	29664.8	6.2	59713.8	74656.6	167.3	27936.1	*28079.4*	0.6
	10gil262	97124.4	27637.5	27653.6	27788	0.5	29075.0	29568.4	7.0	47222.9	59201.8	114.2	27637.5	*27645.7*	0.0
	10lin318	1903339.7	809750.0	815294.7	825808.2	2.0	832299.5	841893.2	4.0	1316596.5	1529009	88.8	812744.1	*814264.5*	0.6
	10pcb442	2176527.6	741195.8	745264.4	750362.1	1.2	765561.0	796960.4	7.5	1785678.6	2213061.7	198.6	742112.4	*742678.9*	0.2
	10pr439	4850717.0	1904690.2	1917984.5	1930766.4	1.4	1971633.0	2022257.4	6.2	4506278.5	5400779.5	183.6	1907568.9	*1911815.3*	0.4
	25a280	102680.6	29902.4	30058.6	*30228.5*	1.1	31481.2	32020.2	7.1	76157.0	103640.2	246.6	30373.2	30654.1	2.5
	25gil262	81311.5	30325.7	30535.7	*30711.1*	1.3	31579.5	31949.7	5.4	57331.1	79015.1	160.6	30695.8	30953.2	2.1
	25lin318	1788199.9	584554.0	592166.4	*595489*	1.9	607029.0	617399.9	5.6	1393249.2	1790655.3	206.3	593017.4	601118.8	2.8
	25pcb442	2184230.2	740892.6	744734.7	*748031.6*	1.0	794217.4	805896.7	8.8	1890250.6	2549800.1	244.2	757524.1	762707.5	2.9
	25pr439	4219473.4	1511168.9	1519322.1	*1531976.6*	1.4	1585283.0	1612334.7	6.7	4366256.4	6793182.6	349.5	1531948.9	1556475.7	3.0
	50a280	108612.2	36266.9	36406.5	*36471*	0.6	37458.4	37828.6	4.3	112427.8	134884.2	271.9	38596.8	39872.5	9.9
	50gil262	75824.3	26523.3	26634.0	*26680.8*	0.6	27647.5	27836.2	4.9	87390.4	106922.9	303.1	28780.1	30004	13.1
	50lin318	1522940.0	688724.6	690743.8	*691897.5*	0.5	706854.9	713744.5	3.6	1674054.9	2672878.9	288.1	730023.7	748203.2	8.6
	50pcb442	2826608.2	910478.7	913395.0	*914652.8*	0.5	949830.8	954169	4.8	2381861.7	3235893.9	255.4	962101.7	990436	8.8
	50pr439	5944214.3	2152986.6	2162274.5	*2167713.8*	0.7	2213598.5	2232892.2	3.7	5846421.8	8658899.7	302.2	2302262.5	2375177.6	10.3

Bảng 20: Kết quả thực nghiệm của các thuật toán trên bộ dữ liệu đồ thị metric thuộc Type 5

	AAL	**SLA-M**	**HB-RGA**			**N-EA**			**C-EA**			**G-MFEA**		
Instances	Avg	Cost	BF	Avg	RPD	BF	Avg	RPD	BF	Avg	RPD	BF	Avg	RPD
10i120-46	227078.7	93925.0	94055.2	94596.7	0.7	96168.2	97752.1	4.1	113658.0	139893.8	48.9	93956.9	*94034.3*	0.1
10i30-17	31380.1	13276.6	13276.6	13289.2	0.1	13276.6	13290	0.1	14179.9	17064.8	28.5	13276.6	***13276.6***	0.0

Kích thước nhỏ

10i45-18	59797.3	22890.4	23267.6	23344.8	2.0	23227.3	23985.5	4.8	28489.5	34414.2	50.3	22890.4	*22892.2*	0.0
10i60-21	89193.4	33694.8	33744.5	35002	3.9	34147.0	35233.3	4.6	40178.9	49036	45.5	33694.8	*33702.8*	0.0
10i65-21	102156.3	37353.1	37386.7	37677	0.9	38318.8	39578.8	6.0	44274.5	52774.5	41.3	37353.1	*37353.6*	0.0
10i70-21	103674.7	38059.5	38543.8	38855	2.1	38816.6	39687.3	4.3	46923.9	57188.7	50.3	38066.7	*38187.3*	0.3
10i75-22	139170.4	65361.9	65411.9	65783.1	0.6	65923.2	66485.1	1.7	72954.4	85273.9	30.5	65362.0	*65397.3*	0.1
10i90-33	128438.3	51931.2	52091.2	52617.6	1.3	53076.0	54636.2	5.2	61170.3	73027.2	40.6	51943.2	*51975.6*	0.1
5i120-46	150008.8	61451.5	61776.0	62393.2	1.5	61695.7	62620.1	1.9	81884.4	89593.1	45.8	61451.5	*61495.3*	0.1
5i30-17	48826.7	14399.9	14399.9	***14399.9***	0.0	14399.9	***14399.9***	0.0	14747.0	16207.7	12.6	14399.9	***14399.9***	0.0
5i45-18	59161.2	14884.3	14884.3	14893	0.1	14884.3	14925.6	0.3	16965.3	19689	32.3	14884.3	***14884.3***	0.0
5i60-21	63252.8	28422.7	28422.7	28584.2	0.6	28422.7	28769.6	1.2	33631.3	38772.4	36.4	28422.7	***28422.7***	0.0
5i65-21	71729.1	30907.8	31244.3	31684.7	2.5	30907.8	31254.4	1.1	34282.4	39388.2	27.4	30907.8	*30911.7*	0.0
5i70-21	85093.3	35052.8	35052.8	35384.4	0.9	35052.8	35298.8	0.7	41726.2	46451.9	32.5	35052.8	***35052.8***	0.0
5i75-22	73544.5	34692.5	34811.1	34993.8	0.9	34692.5	35098.7	1.2	41307.9	48845.4	40.8	34692.5	***34692.5***	0.0
5i90-33	114467.8	51977.0	52128.9	52916	1.8	51977.0	52533.8	1.1	62868.1	67725	30.3	51977.0	*51977.3*	0.0
7i30-17	48826.7	20438.9	20438.9	20450	0.1	20438.9	20454.2	0.1	21262.1	23774.5	16.3	20438.9	***20438.9***	0.0
7i45-18	59161.2	20512.0	20512.0	20973.8	2.3	20512.0	20700.8	0.9	22610.9	28426.5	38.6	20512.0	***20512***	0.0
7i60-21	85486.3	36263.9	36263.9	36339.5	0.2	36295.4	37780.7	4.2	41411.0	45776.8	26.2	36263.9	***36263.9***	0.0
7i65-21	82949.6	34847.6	34847.6	34881.9	0.1	35201.2	36136.4	3.7	37632.2	43707.3	25.4	34847.6	***34847.6***	0.0
7i70-21	103983.6	39487.6	39757.8	39970.3	1.2	39613.4	40819.5	3.4	43285.3	53597.5	35.7	39487.6	*39491.1*	0.0
10i300-109	276642.2	112681.0	113292.8	114274.1	1.4	117421.2	119952.9	6.5	192347.4	240378.6	113.3	112876.2	*113017.1*	0.3
10i400-206	484586.8	207521.7	209409.3	211253.4	1.8	214604.4	217399	4.8	375263.4	471201.2	127.1	207778.5	*208087.4*	0.3
10i500-305	897388.3	349675.2	352151.4	356961.1	2.1	355952.4	359614.5	2.8	655241.8	710889.6	103.3	350897.4	*351929.6*	0.6
15i300-110	341743.8	112096.7	114092.2	114531.8	2.2	119922.2	122146.2	9.0	232277.9	295072.3	163.2	112935.9	*113358.9*	1.1
15i400-207	482329.8	164117.8	165565.7	166796.4	1.6	171349.6	175081.7	6.7	372256.9	477146.3	190.7	165328.2	*165854.9*	1.1
15i500-306	914942.4	300734.1	305034.5	306929.7	2.1	310122.7	313184.9	4.1	593109.3	798361.5	165.5	304128.7	*304949.4*	1.4
20i300-111	376885.7	156347.7	157526.1	158563.7	1.4	163927.8	167104.4	6.9	281039.7	409119	161.7	157371.4	*157990.3*	1.1

Kích thước lớn														
20i400-208	584471.5	224012.5	226173.0	*226963.7*	1.3	231753.3	236373.9	5.5	547322.7	684259.3	205.5	226383.9	227069.5	1.4
20i500-307	513104.5	200328.7	203339.4	204679.4	2.2	212306.3	215644.7	7.6	490979.3	612983.1	206.0	202938.5	*203910.7*	1.8
25i300-112	331103.7	116193.6	117661.9	*118121.6*	1.7	125392.7	127466.5	9.7	292327.6	430266.8	270.3	118310.1	119401.3	2.8
25i400-209	738504.4	229913.6	231886.7	*233015.6*	1.3	241529.0	243994.1	6.1	492799.3	698011.4	203.6	233552.7	236456.6	2.8
25i500-308	720399.8	299498.2	300838.0	*301906.8*	0.8	312805.6	316116.8	5.5	579415.8	888926.7	196.8	302497.1	304046.5	1.5
5i300-108	332681.5	177185.9	177698.0	179796.8	1.5	178628.1	180397.2	1.8	249796.7	287635.8	62.3	177185.9	*177220.7*	0.0
5i400-205	405863.5	209389.8	210229.5	216014.7	3.2	211603.0	213125.6	1.8	328640.3	350401.9	67.3	209488.0	*209970.9*	0.3
5i500-304	419228.9	182024.0	183130.1	185809.5	2.1	183656.4	185924.4	2.1	347200.1	417651.8	129.4	182206.2	*182416.4*	0.2

Bảng 21: Kết quả thực nghiệm của các thuật toán trên bộ dữ liệu đồ thị metric thuộc Type 6

	AAL	**SLA-M**	**HB-RGA**			**N-EA**			**C-EA**			**G-MFEA**		
Instances	Avg	Cost	BF	Avg	RPD	BF	Avg	RPD	BF	Avg	RPD	BF	Avg	RPD
10berlin52-2x5	61720.0	27471.4	27472.4	27723.2	0.9	27471.4	27805.3	1.2	35555.7	43563.2	58.6	27471.40	*27473*	0.0
12eil51-3x4	4209.1	1699.0	1699.1	1702.3	0.2	1720.1	1762.7	3.7	2265.0	2722.4	60.2	1699.00	*1699.1*	0.0
12eil76-3x4	7665.9	2650.8	2650.8	2653.2	0.1	2738.6	2802	5.7	3355.6	4213.9	59.0	2650.80	***2650.8***	0.0
12pr76-3x4	1299696.3	600008.6	600597.6	603474.4	0.6	604837.0	621228.6	3.5	811776.4	942316.1	57.1	600430.90	*600818.7*	0.1
12st70-3x4	10072.3	4106.5	4128.1	4144.6	0.9	4148.4	4219.2	2.7	5101.7	6139.6	49.5	4106.50	*4110.1*	0.1
15pr76-3x5	1316199.6	524335.2	526596.7	532896.8	1.6	534613.0	544174	3.8	805909.3	1014229.4	93.4	525170.30	*526166.2*	0.3
16eil51-4x4	3123.2	1301.4	1302.4	*1304*	0.2	1323.8	1351.1	3.8	2196.0	2834	117.8	1302.70	1305.6	0.3
16eil76-4x4	6625.7	2036.0	2042.4	2053.8	0.9	2088.1	2163	6.2	3096.0	4205.5	106.6	2040.00	*2052.2*	0.8
16lin105-4x4	309572.4	125052.2	125052.2	125685.3	0.5	128713.1	130815.3	4.6	164946.6	235828.8	88.6	125052.20	*125289.8*	0.2
16st70-4x4	8073.9	2932.6	2939.3	2966.1	1.1	2963.8	3050.4	4.0	5125.4	6567.4	123.9	2935.40	*2949.2*	0.6
18pr76-3x6	1809111.1	638164.5	642733.0	646399.1	1.3	641209.6	657524.3	3.0	923120.7	1073384.4	68.2	639723.30	*641700.1*	0.6
20eil51-4x5	4836.6	2283.7	2284.2	*2287.9*	0.2	2286.4	2331.3	2.1	3256.3	3942	72.6	2288.70	2295.2	0.5
20eil76-4x5	7923.9	2385.9	2385.9	*2392.6*	0.3	2478.2	2520.3	5.6	4128.4	5377.1	125.4	2390.50	2402.2	0.7
20st70-4x5	8574.0	2934.8	2939.4	*2945.7*	0.4	2976.9	3032.7	3.3	5569.5	7063.9	140.7	2942.80	2967	1.1

Kích thước nhỏ

25eil101-5x5	11580.1	3603.5	3609.1	*3622*	0.5	3711.7	3780.9	4.9	6331.0	8213.3	127.9	3649.20	3670.5	1.9
25eil51-5x5	3697.8	1474.6	1474.6	*1476.2*	0.1	1483.7	1512	2.5	2711.9	3689.2	150.2	1487.40	1507.3	2.2
25eil76-5x5	7512.2	2193.1	2193.1	*2194.8*	0.1	2264.3	2312.1	5.4	4146.2	6241.1	184.6	2219.10	2245.2	2.4
25rat99-5x5	28158.5	11395.8	11400.3	*11418.9*	0.2	11754.1	11869.1	4.2	16441.7	20993.6	84.2	11434.90	11485.9	0.8
28kroA100-4x7	391859.3	133101.6	134129.0	*134532.3*	1.1	138682.6	141334.2	6.2	259023.4	339719.8	155.2	136501.10	138342.8	3.9
30kroB100-5x6	435707.1	197934.6	198976.7	*199205.7*	0.6	201813.7	204967.3	3.6	325726.7	438362.4	121.5	200596.80	202209.8	2.2
35kroB100-5x5	297029.1	129078.7	129122.6	*129832.3*	0.6	133662.2	137003.4	6.1	242049.9	298633.2	131.4	130935.10	132840.4	2.9
36eil101-6x6	10294.1	3850.7	3850.7	*3852.1*	0.0	3977.6	4028.6	4.6	8446.8	12391.4	221.8	3929.20	3981.6	3.4
42rat99-6x7	19472.0	8902.1	8902.5	*8906*	0.0	9093.5	9182.7	3.2	20172.8	27860.2	213.0	9187.00	9393.5	5.5
4berlin52-2x2	70649.9	23287.9	23287.9	23395.9	0.5	23287.9	***23287.9***	0.0	31829.5	38526.9	65.4	23287.90	***23287.9***	0.0
4eil51-2x2	5187.5	1898.5	1898.5	1915.6	0.9	1898.5	1901.3	0.1	2272.3	2422.7	27.6	1898.50	***1898.5***	0.0
4eil76-2x2	6882.8	2948.7	2948.8	2974.6	0.9	2948.7	2955.8	0.2	3610.2	4052.6	37.4	2948.70	***2948.7***	0.0
4pr76-2x2	1607169.6	442693.0	442693.0	445997.3	0.7	442693.0	446682.1	0.9	638744.6	728555.4	64.6	442693.00	***442693***	0.0
6berlin52-2x3	80429.4	32128.6	32130.8	32295.7	0.5	32128.6	32354.7	0.7	37246.6	43918.3	36.7	32128.60	***32128.6***	0.0
6pr76-2x3	1269147.1	648275.7	648884.9	656228.6	1.2	648713.1	659658.8	1.8	728666.8	917852.9	41.6	648275.70	*648507.9*	0.0
6st70-2x3	8147.4	3476.7	3476.7	3503.5	0.8	3478.2	3525.3	1.4	3980.8	5064.2	45.7	3476.70	***3476.7***	0.0
8berlin52-2x4	58925.6	26783.2	26854.4	26969.8	0.7	26783.2	27060.4	1.0	33909.9	41646.8	55.5	26783.20	*26795.4*	0.0
9eil101-3x3	9560.8	3117.6	3135.4	3154.3	1.2	3184.4	3274.5	5.0	4517.9	5472.3	75.5	3117.60	*3120.2*	0.1
9eil51-3x3	4642.4	1907.7	1912.8	1921.3	0.7	1916.0	1963.8	2.9	2323.6	2626.8	37.7	1908.00	*1909.9*	0.1
9eil76-3x3	6659.6	2937.4	2938.4	2956.6	0.7	2999.4	3057.2	4.1	3551.5	4039.2	37.5	2937.40	*2938.6*	0.0
9pr76-3x3	1344518.1	553400.6	554995.8	560230.3	1.2	558349.3	567987.4	2.6	720920.1	835295.8	50.9	553685.60	*553849.7*	0.1
18pr439-3x6	4827227.1	1471788.7	1475993.8	1498114.4	1.8	1525370.2	1553940.8	5.6	5113367.5	6325195.4	329.8	1483618.33	*1488675.8*	1.1
20pr439-4x5	5384342.3	1978001.3	1993816.3	2008663.3	1.6	2035939.4	2075349.7	4.9	4568690.6	6543749.5	230.8	1993350.75	*2000099.7*	1.1
25a280-5x5	115284.3	41690.3	42016.8	*42150.6*	1.1	43408.0	44268.6	6.2	64023.7	104115.3	149.7	42123.08	42388.8	1.7
25gil262-5x5	95797.6	30649.5	30825.3	*31079.8*	1.4	32172.6	32674.3	6.6	52965.5	78711.3	156.8	31116.21	31372.9	2.4
25pcb442-5x5	2210900.7	740883.3	748160.1	*752959.6*	1.6	786167.2	802854.1	8.4	1970190.2	2542620.4	243.2	752573.88	762391.2	2.9

Kích thước lớn														
	36pcb442-6x6	2765556.9	860978.2	866373.4	*869007.5*	0.9	899354.2	913260.7	6.1	2019587.8	2453869.6	185.0	887178.15	901697.8	4.7
	42a280-6x7	115920.5	43896.8	44080.5	*44159.4*	0.6	45163.4	45660.2	4.0	107127.0	139416.9	217.6	45030.02	46000.9	4.8
	49lin318-7x7	1466128.7	569746.3	572106.1	*574040.2*	0.8	591374.9	595249.9	4.5	2060347.8	2552714.1	348.0	609127.76	633218.4	11.1
	9a280-3x3	106871.9	28947.5	29257.3	29497.3	1.9	30443.2	31011.6	7.1	56542.3	71435	146.8	29045.12	*29105.0*	0.5
	9gil262-3x3	71901.7	20935.9	21248.9	21453.8	2.5	22158.9	23059.2	10.1	44033.8	55262.4	164.0	20937.79	*20993.1*	0.3
	9lin318-3x3	1831854.2	716850.2	720268.4	725327.5	1.2	730038.2	740588.6	3.3	1157786.8	1398630	95.1	718479.40	*719450.3*	0.4
	9pcb442-3x3	3104446.7	760238.3	763118.7	769413.2	1.2	803179.2	821884.9	8.1	1801603.5	2214981.5	191.4	760484.37	*761269.1*	0.1
	9pr439-3x3	6659610.2	1800753.9	1824280.6	1857520.1	3.2	1820176.1	1881943	4.5	5012090.6	6263084.4	247.8	1803288.19	*1809146.2*	0.5

Bảng 22: Kết quả thực nghiệm của các thuật toán trên bộ dữ liệu đồ thị metric thuộc Type 3 và Type 4

		AAL	**SLA-M**	**HB-RGA**			**N-EA**			**C-EA**			**G-MFEA**		
	Instances	Avg	Cost	BF	Avg	RPD	BF	Avg	RPD	BF	Avg	RPD	BF	Avg	RPD
Type 3	6i300	48578.0	19264.5	19461.1	19836.6	3.0	19358.8	19467	1.1	28521.3	33277	72.7	19286.3	*19320.3*	0.3
	6i350	48897.6	21217.2	21385.6	21713.4	2.3	21472.8	21702.2	2.3	38582.3	45908.2	116.4	21218.5	*21261.3*	0.2
	6i400	60892.1	29348.2	29513.4	29913.7	1.9	29506.9	29677.7	1.1	46255.5	51863.4	76.7	29389.5	*29437.5*	0.3
	6i450	72207.6	35681.5	35899.2	36463.5	2.2	35866.3	36124.5	1.2	54943.1	61085.5	71.2	35715.7	*35795.9*	0.3
	6i500	79282.6	37510.1	37799.5	38225.9	1.9	37711.6	38045.9	1.4	66632.9	75519.8	101.3	37567.6	*37631.4*	0.3
	4i200a	499297.8	97959.6	97959.6	97974.1	0.0	97959.6	102256.3	4.4	297074.5	358378.4	265.8	97959.6	***97959.6***	0.0
	4i200h	344698.6	87675.3	87675.3	88285.1	0.7	87675.3	89628.9	2.2	196688.0	220612	151.6	87675.3	***87675.3***	0.0
	4i200x1	342335.9	123669.7	123811.4	124825.9	0.9	123669.7	125782.7	1.7	231913.2	267677	116.4	123669.7	*123670.2*	0.0
	4i200x2	400686.5	114012.3	114059.5	115432	1.2	114012.3	116256.5	2.0	218431.5	255622.3	124.2	114012.3	***114012.3***	0.0
	4i200z	399014.8	131683.5	131807.4	133697.8	1.5	131683.5	133873.8	1.7	271353.4	298761.6	126.9	131683.5	*131685.2*	0.0
	4i400a	1169245.8	214115.3	214115.3	214230.5	0.1	217171.4	227530.6	6.3	1048587.2	1244021.3	481.0	214115.3	***214115.3***	0.0
	4i400h	780790.0	256200.5	257054.0	260183.1	1.6	257954.4	260916	1.8	699391.5	786763.8	207.1	256200.5	*256291.2*	0.0
	4i400x1	723488.8	188196.7	188840.1	*191145.3*	1.6	188786.2	191694.7	1.9	653318.1	716077.9	280.5	199389.3	222805.4	18.4
	4i400x2	753930.4	159254.8	159254.8	*162685.6*	2.2	159254.8	163222.8	2.5	567509.5	663607.5	316.7	176188.2	195580.8	22.8

4i400z	687615.8	221423.9	221460.6	*224677.4*	1.5	221460.6	225096.3	1.7	638368.7	728536	229.0	234203.3	244894.6	10.6

Bảng 23: Thời gian tính trung bình của các thuật toán trên bộ dữ liệu đồ thị metric thuộc Type 1

	Instances	HB-RGA	N-EA	C-EA	G-MFEA
Kích thước nhỏ	10berlin52	0.04	0.02	0.02	0.67
	10eil51	0.04	0.02	0.02	0.92
	10eil76	0.05	0.02	0.03	1.60
	10kroB100	0.06	0.02	0.03	2.35
	10pr76	0.04	0.02	0.05	1.67
	10rat99	0.05	0.02	0.05	2.35
	10st70	0.04	0.02	0.03	1.40
	15berlin52	0.03	0.03	0.03	1.07
	15eil51	0.03	0.03	0.03	1.00
	15eil76	0.04	0.03	0.03	1.62
	15pr76	0.04	0.03	0.05	1.63
	15st70	0.03	0.03	0.05	1.40
	25eil101	0.05	0.03	0.08	2.38
	25kroA100	0.05	0.03	0.08	2.37
	25lin105	0.05	0.03	0.08	2.50
	25rat99	0.05	0.03	0.08	2.33
	50eil101	0.05	0.07	0.07	3.15
	50kroA100	0.05	0.07	0.07	3.13
	50kroB100	0.06	0.07	0.07	3.13
	50lin105	0.06	0.07	0.07	3.33
	50rat99	0.05	0.08	0.03	3.15
	5berlin52	0.03	0.01	0.03	1.02
	5eil51	0.03	0.01	0.03	0.95
	5eil76	0.05	0.01	0.03	1.83
	5pr76	0.06	0.01	0.03	1.82
	5st70	0.04	0.01	0.03	1.57
Kích thước lớn	10a280	0.31	0.02	0.43	14.72
	10gil262	0.27	0.02	0.43	12.78
	10lin318	0.37	0.02	0.67	18.55
	10pcb442	0.87	0.02	0.67	39.38
	10pr439	1.32	0.02	0.83	42.50
	25a280	0.23	0.03	0.83	11.17
	25gil262	0.22	0.03	0.48	11.67
	25lin318	0.31	0.03	0.48	16.65
	25pcb442	0.54	0.03	0.68	31.28
	25pr439	0.89	0.03	0.68	29.62
	50a280	0.22	0.10	0.38	12.15
	50gil262	0.22	0.10	0.30	12.12
	50lin318	0.31	0.10	0.38	16.80
	50pcb442	0.57	0.10	0.68	31.00
	50pr439	0.93	0.10	0.68	25.13

Bảng 24: Thời gian tính trung bình của các thuật toán trên bộ dữ liệu đồ thị metric thuộc Type 5

	Instances	HB-RGA	N-EA	C-EA	G-MFEA
Kích thước nhỏ	10i120-46	0.08	0.02	0.05	2.77
	10i30-17	0.02	0.02	0.02	0.58
	10i45-18	0.03	0.02	0.05	0.85
	10i60-21	0.03	0.02	0.03	1.20
	10i65-21	0.04	0.02	0.03	1.32
	10i70-21	0.03	0.02	0.05	1.43
	10i75-22	0.05	0.02	0.02	1.67
	10i90-33	0.05	0.02	0.05	2.03
	5i120-46	0.11	0.02	0.07	3.65
	5i30-17	0.02	0.01	0.07	0.62
	5i45-18	0.02	0.01	0.03	0.88
	5i60-21	0.03	0.01	0.03	1.18
	5i65-21	0.03	0.01	0.03	1.40
	5i70-21	0.04	0.01	0.03	1.57
	5i75-22	0.05	0.01	0.05	1.83
	5i90-33	0.06	0.01	0.05	2.28
	7i30-17	0.02	0.02	0.02	0.57
	7i45-18	0.02	0.02	0.02	0.80
	7i60-21	0.03	0.02	0.03	1.13
	7i65-21	0.04	0.02	0.03	1.33
	7i70-21	0.04	0.02	0.02	1.50
Kích thước lớn	10i300-109	0.33	0.02	0.68	16.98
	10i400-206	0.60	0.02	0.68	31.85
	10i500-305	0.99	0.02	1.07	52.50
	15i300-110	0.28	0.02	1.07	15.83
	15i400-207	0.47	0.02	0.87	28.27
	15i500-306	0.80	0.02	0.87	42.77
	20i300-111	0.24	0.03	0.52	15.30
	20i400-208	0.39	0.03	0.52	26.78
	20i500-307	0.64	0.03	0.83	39.17
	25i300-112	0.22	0.05	0.83	14.38
	25i400-209	0.38	0.05	0.85	23.27
	25i500-308	0.61	0.05	0.85	24.60
	5i300-108	0.67	0.02	0.60	26.78
	5i400-205	1.17	0.02	0.83	52.22
	5i500-304	2.11	0.02	0.83	90.62

Bảng 25: Thời gian tính trung bình của các thuật toán trên bộ dữ liệu đồ thị metric thuộc Type 6

	Instances	HB-RGA	N-EA	C-EA	G-MFEA
Kích thước nhỏ	10berlin52-2x5	0.04	0.02	0.02	0.83
	12eil51-3x4	0.03	0.02	0.02	0.93
	12eil76-3x4	0.04	0.02	0.05	1.57
	12pr76-3x4	0.05	0.02	0.05	1.65
	12st70-3x4	0.03	0.02	0.05	1.43
	15pr76-3x5	0.06	0.03	0.05	1.63
	16eil51-4x4	0.02	0.03	0.03	1.00
	16eil76-4x4	0.04	0.03	0.03	1.67
	16lin105-4x4	0.06	0.03	0.07	2.40
	16st70-4x4	0.03	0.03	0.07	1.45
	18pr76-3x6	0.04	0.03	0.03	1.68
	20eil51-4x5	0.03	0.03	0.03	1.05
	20eil76-4x5	0.03	0.02	0.05	1.68
	20st70-4x5	0.03	0.03	0.05	1.47
	25eil101-5x5	0.05	0.03	0.05	2.50
	25eil51-5x5	0.02	0.03	0.05	1.18
	25eil76-5x5	0.03	0.03	0.05	1.73
	25rat99-5x5	0.05	0.03	0.05	2.33
	28kroA100-4x7	0.05	0.03	0.07	2.38
	30kroB100-5x6	0.05	0.03	0.07	2.50
	35kroB100-5x5	0.05	0.03	0.07	2.48
	36eil101-6x6	0.04	0.05	0.07	2.68
	42rat99-6x7	0.05	0.07	0.03	2.75
	4berlin52-2x2	0.03	0.01	0.03	1.10

	Instances	HB-RGA	N-EA	C-MFEA	G-Graph
Kích thước nhỏ	4eil51-2x2	0.03	0.01	0.03	0.97
	4eil76-2x2	0.05	0.01	0.03	1.98
	4pr76-2x2	0.06	0.01	0.03	2.02
	6berlin52-2x3	0.03	0.01	0.03	1.13
	6pr76-2x3	0.05	0.01	0.03	1.73
	6st70-2x3	0.04	0.01	0.03	1.40
	8berlin52-2x4	0.03	0.02	0.05	0.93
	9eil101-3x3	0.06	0.02	0.05	2.32
	9eil51-3x3	0.03	0.02	0.03	0.92
	9eil76-3x3	0.04	0.02	0.03	1.57
	9pr76-3x3	0.04	0.02	0.03	1.63
Kích thước lớn	18pr439-3x6	2.19	0.03	0.93	57.18
	20pr439-4x5	1.87	0.05	0.93	50.65
	25a280-5x5	0.24	0.05	0.37	13.10
	25gil262-5x5	0.20	0.05	0.37	11.60
	25pcb442-5x5	0.59	0.05	0.62	28.68
	36pcb442-6x6	0.53	0.08	0.93	30.92
	42a280-6x7	0.20	0.12	0.93	12.95
	49lin318-7x7	0.28	0.15	0.47	15.55
	9a280-3x3	0.36	0.02	0.35	14.55
	9gil262-3x3	0.32	0.02	0.35	14.27
	9lin318-3x3	0.47	0.02	0.67	20.27
	9pcb442-3x3	1.05	0.02	0.65	41.25
	9pr439-3x3	2.86	0.03	0.65	56.98

Bảng 26: Thời gian tính trung bình của các thuật toán trên bộ dữ liệu đồ thị metric thuộc Type 3 and Type 4

	Instances	**HB-RGA**	**N-EA**	**C-EA**	**G-MFEA**
Type 3	6i300	0.55	0.02	0.47	23.42
	6i350	0.73	0.02	0.47	33.20
	6i400	1.05	0.02	0.63	46.60
	6i450	1.34	0.02	0.88	60.45
	6i500	1.95	0.02	0.88	82.30
Type 4	4i200a	0.35	0.01	0.22	11.43
	4i200h	0.36	0.01	0.17	11.63
	4i200x1	0.35	0.01	0.22	11.67
	4i200x2	0.37	0.01	0.17	12.03
	4i200z	0.38	0.01	0.53	11.57
	4i400a	2.53	0.02	0.53	64.53
	4i400h	2.41	0.02	1.07	62.68
	4i400x1	2.32	0.02	1.07	36.93
	4i400x2	2.35	0.02	0.65	64.03
	4i400z	2.22	0.02	0.65	36.95

Bảng 27: Kết quả thực nghiệm của các thuật toán trên tập đồ thị đầy đủ phi metric thuộc Type 1

	AAL		**HB-RGA**			**N-EA**			**C-EA**			**G-Graph**		
Instances	BF	Avg	BF	Avg	Time	BF	Avg	Time	BF	Avg	Time	BF	Avg	Time
N-10BERLIN52	35255	66440	17007	16929	0.12	13304	15649	0.02	29210	39828	0.02	13951	***13965***	0.98
N-10EIL51	41381	87557	18656	18570	0.10	19162	22047	0.03	29382	36740	0.02	17522	***17567***	0.95
N-10EIL76	49701	85070	19564	19564	0.20	17543	23723	0.03	47126	58882	0.05	18868	***18870***	1.67
N-10KROB100	77291	122421	29079	28486	0.31	30694	35940	0.03	80493	94388	0.05	26640	***26807***	2.33
N-10PR76	69447	110928	23404	23382	0.17	24676	29754	0.03	47468	58901	0.05	21909	***21920***	1.58
N-10RAT99	69749	131756	20292	20797	0.30	33020	36888	0.03	56603	82053	0.05	19770	***19798***	2.40
N-10ST70	62219	112714	28034	27693	0.16	21451	***24857***	0.05	47154	61538	0.03	24793	24875	1.65
N-15BERLIN52	37863	87875	17572	17571	0.09	17309	19146	0.05	29497	41852	0.03	16064	***16274***	1.03
N-15EIL51	44972	77143	16047	16530	0.09	17145	19478	0.07	27471	40349	0.03	15320	***15490***	1.02
N-15EIL76	67616	125386	28044	28022	0.19	27981	30856	0.07	49741	67972	0.03	25290	***25788***	1.65
N-15PR76	60959	132344	30914	30914	0.22	27597	30307	0.07	41146	65666	0.03	28902	***29750***	1.60
N-15ST70	71696	120153	25060	24519	0.18	25674	28213	0.07	45162	60863	0.03	22710	***23254***	1.53
N-25EIL101	132452	222632	36059	36411	0.33	37599	39654	0.15	81399	114062	0.05	35323	***35924***	2.42
N-25KROA100	105807	176783	34721	***34721***	0.32	37192	39496	0.17	76610	107555	0.07	34747	35450	2.42
N-25LIN105	91959	170056	32004	32632	0.36	31989	34613	0.17	79759	116983	0.07	31002	***31792***	2.45
N-25RAT99	103068	170348	34439	***34408***	0.31	32417	35897	0.17	76073	101082	0.03	33917	34832	2.33
N-50KROA100	92186	181134	28080	***28080***	0.36	28909	30493	0.37	131755	166475	0.05	36559	38389	2.95
N-50KROB100	118308	211411	31185	***31157***	0.36	29759	31213	0.33	121347	172312	0.05	38000	41700	3.10
N-50LIN105	98998	189699	26873	***26873***	0.39	27420	28416	0.25	120302	172680	0.05	34079	38677	3.32

N-50RAT99	74313	202102	28677	***28677***	0.36	29013	29611	0.28	135917	185059	0.03	37737	40932	3.13
N-5EIL51	34825	64779	9513	9503	0.12	17192	19925	0.03	20391	29522	0.02	9083	***9083***	0.98
N-5EIL76	38320	81696	14873	14899	0.22	17794	24890	0.03	43805	57895	0.02	14866	***14866***	1.82
N-5PR76	37486	79343	15741	15942	0.22	17229	24933	0.03	43445	59806	0.03	15704	***15704***	1.87

Bảng 28: Kết quả thực nghiệm của các thuật toán trên tập đồ thị đầy đủ phi metric thuộc Type 5

	AAL		**HB-RGA**			**N-EA**			**C-EA**			**G-Graph**		
Instances	BF	Avg	BF	Avg	Time	BF	Avg	Time	BF	Avg	Time	BF	Avg	Time
N-10I120-46	93781	153708	28044	26812	0.47	32438	38202	0.03	88322	115577	0.05	23784	***23795***	2.93
N-10I30-17	15012	32111	7523	7565	0.05	11145	12267	0.03	11797	16838	0.02	7386	***7386***	0.58
N-10I45-18	29296	59633	13255	***13255***	0.09	12451	14825	0.03	21648	28931	0.05	13768	13769	0.88
N-10I60-21	53232	84570	17365	17072	0.13	19575	22633	0.03	28909	43449	0.03	16627	***16629***	1.17
N-10I65-21	68159	108013	22243	20926	0.15	25966	29881	0.05	41256	50290	0.03	18250	***18281***	1.37
N-10I70-21	61228	101288	25600	25672	0.17	22515	26803	0.05	43554	55795	0.03	23866	***24097***	1.48
N-10I75-22	56621	114394	18995	18797	0.21	24528	27876	0.03	49562	63323	0.02	17241	***17252***	1.67
N-10I90-33	63226	116464	22577	22577	0.29	23656	30710	0.03	60524	74269	0.03	22310	***22325***	2.23
N-5I30-17	14636	30857	8221	7773	0.05	11622	12841	0.03	11480	15297	0.05	7686	***7686***	0.58
N-5I45-18	36539	63124	9260	***9289***	0.09	13709	17369	0.03	17639	22250	0.02	9920	9920	0.87
N-5I60-21	39009	63316	15669	15818	0.14	16002	21399	0.03	32206	39760	0.02	14797	***14813***	1.23
N-5I65-21	41371	63281	9904	10301	0.16	17500	21884	0.03	32898	41264	0.03	10051	***10051***	1.42
N-5I70-21	29893	55564	14180	14093	0.20	14144	19644	0.03	36806	51644	0.03	13756	***13756***	1.60
N-5I75-22	35859	70106	16209	***16209***	0.22	18140	22800	0.03	48841	61194	0.05	16209	***16209***	1.82

N-5I90-33	56650	79529	18137	16686	0.31	21782	28485	0.03	61868	76702	0.05	15806	***15806***	2.40
N-7I30-17	25254	47815	10746	10754	0.05	9692	11881	0.03	14031	17509	0.02	10417	***10417***	0.62
N-7I45-18	28771	48843	14825	15642	0.09	13783	15665	0.05	20718	29364	0.02	14825	***14825***	0.85
N-7I70-21	42903	74797	18834	19466	0.18	19636	23150	0.05	45010	53586	0.03	18994	***19113***	1.52

Bảng 29: Kết quả thực nghiệm của các thuật toán trên tập đồ thị đầy đủ phi metric thuộc Type 6

	AAL		**HB-RGA**			**N-EA**			**C-EA**			**G-Graph**		
Instances	BF	Avg	BF	Avg	Time	BF	Avg	Time	BF	Avg	Time	BF	Avg	Time
N-10BERLIN52-2X5	27046	67860	14066	14067	0.12	13566	16471	0.03	29210	39828	0.02	13222	***13228***	1.00
N-12EIL51-3X4	46729	79172	16938	16897	0.12	18839	20586	0.03	25887	38913	0.02	15583	***15771***	0.95
N-12EIL76-3X4	67314	116362	25160	25161	0.22	26457	30338	0.03	48554	63699	0.03	23652	***23758***	1.60
N-12PR76-3X4	50460	104763	16836	***16836***	0.20	22488	24919	0.05	39357	56960	0.03	18075	18099	1.65
N-12ST70-3X4	65602	127531	21451	21336	0.20	29516	31700	0.05	44432	56507	0.03	19938	***20257***	1.37
N-15PR76-3X5	77022	145160	21335	21122	0.23	26872	30294	0.07	41146	65666	0.03	19708	***20531***	1.70
N-16EIL51-4X4	59847	93480	15581	***15452***	0.11	20191	21749	0.07	24681	37461	0.03	15340	15563	1.17
N-16EIL76-4X4	54744	121785	23703	23684	0.22	25366	27458	0.07	47947	59082	0.03	21214	***21655***	1.82
N-16LIN105-4X4	85885	162854	29109	29263	0.38	36455	38707	0.07	86052	117346	0.05	27257	***28130***	2.52
N-16ST70-4X4	68142	113161	23953	23953	0.19	23535	26269	0.08	51238	64526	0.05	23663	***23881***	1.43
N-18PR76-3X6	72725	124643	27899	28691	0.22	26593	29018	0.08	54849	73283	0.03	25361	***25947***	1.63
N-20EIL51-4X5	42304	89404	19920	***19920***	0.11	19918	21436	0.12	38194	53057	0.03	19937	20840	1.07
N-20EIL76-4X5	58355	131014	20237	***20237***	0.23	23788	25882	0.13	57805	72517	0.03	21289	21691	1.68
N-20ST70-4X5	71385	116034	28438	28438	0.19	22647	***24986***	0.12	61336	72615	0.03	27325	27927	1.55

N-25EIL101-5X5	104454	224353	31602	***32342***	0.30	35626	39322	0.17	81399	114062	0.05	31445	32607	2.47
N-25EIL51-5X5	39138	90954	16214	***16196***	0.10	18173	19068	0.13	38055	54778	0.05	16356	16954	1.18
N-25EIL76-5X5	39611	100308	26088	26080	0.19	20471	***22782***	0.13	58212	81423	0.03	25129	25720	1.75
N-25RAT99-5X5	73608	177278	34705	34556	0.29	37737	39542	0.13	76073	101082	0.03	31570	***32526***	2.35
N-28KROA100-4X7	88347	164469	29874	***29319***	0.30	30548	32809	0.13	90585	109969	0.05	31637	32887	2.48
N-30KROB100-5X6	113688	227689	33189	33350	0.29	29997	***32335***	0.15	79468	120532	0.05	33303	34583	2.38
N-35KROB100-5X5	82115	186825	29606	29365	0.31	32242	36290	0.12	91986	108683	0.05	26240	***26834***	2.43
N-36EIL101-6X6	73573	203318	33797	33788	0.28	30404	***32008***	0.18	101698	138413	0.05	36739	38820	2.73
N-42RAT99-6X7	104602	250279	29876	***29843***	0.27	32373	34210	0.17	129336	158201	0.03	34621	39066	2.92
N-4BERLIN52-2X2	35208	65418	9952	9853	0.09	11356	16059	0.02	36899	51619	0.03	9376	***9376***	1.12
N-4EIL51-2X2	28885	49394	10372	10138	0.09	11681	14972	0.02	26532	32143	0.02	9766	***9766***	0.97
N-4EIL76-2X2	33491	71672	12066	11856	0.17	15480	21808	0.02	50587	59587	0.02	10562	***10562***	1.93
N-4PR76-2X2	36538	72146	12334	12234	0.19	16877	21097	0.03	43748	63615	0.02	12094	***12094***	2.03
N-6BERLIN52-2X3	23216	61151	11918	11913	0.09	10757	14911	0.03	32407	39830	0.02	11583	***11583***	1.10
N-6PR76-2X3	36911	71529	17016	17016	0.17	15636	20814	0.03	43188	54589	0.03	16700	***16700***	1.73
N-6ST70-2X3	38467	73767	16254	16247	0.15	17753	22243	0.03	35087	51390	0.03	16241	***16241***	1.45
N-8BERLIN52-2X4	29495	53001	14630	14570	0.09	14540	17248	0.03	27703	38394	0.03	14332	***14385***	0.78
N-9EIL101-3X3	54392	117879	22628	***22721***	0.37	26838	31318	0.03	69805	92816	0.03	22883	22913	1.83
N-9EIL51-3X3	42803	73038	13207	***13207***	0.10	18860	20893	0.03	26325	33705	0.02	14229	14229	0.68
N-9EIL76-3X3	61006	106921	23194	23194	0.17	24193	28421	0.03	45532	60840	0.02	21135	***21135***	1.20