ĐẠI HỌC THÁI NGUYÊN
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

ĐỖ XUÂN QUYỀN

PHƯƠNG PHÁP TỐI ƯU ĐÀN KIẾN DÓNG HÀNG HAI ĐỒ THỊ

LUẬN VĂN THẠC SĨ KHOA HỌC MÁY TÍNH

THÁI NGUYÊN - 2016

ĐẠI HỌC THÁI NGUYÊN
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

ĐỖ XUÂN QUYỀN

PHƯƠNG PHÁP TỐI ƯUĐÀN KIẾN DÓNG HÀNG HAI ĐỒ THỊ

Chuyên ngành: Khoa học máy tính
Mã số: 60.48.01.01

LUẬN VĂN THẠC SĨ KHOA HỌC MÁY TÍNH

Người hướng dẫn khoa học: TS. Đỗ Đức Đông

THÁI NGUYÊN - 2016

LỜI CAM ĐOAN

Tôi xin cam đoan đây là công trình nghiên cứu của riêng tôi dưới sự hướng dẫn của TS. Đỗ Đức Đông. Các kết quả được viết chung với các tác giả khác đều được sự đồng ý của đồng tác giả trước khi đưa vào luận văn. Các kết quả thực nghiệm nêu trong luận văn đều là chính xác và chưa từng được công bố.

Tôi hoàn toàn chịu trách nhiệm về tính pháp lý của luận văn này.

Thái Nguyên, tháng 7 năm 2016

Học viên thực hiện

Đỗ Xuân Quyền

LỜI CẢM ƠN

Lời đầu tiên, em xin bày tỏ sự biết ơn sâu sắc đến TS.Đỗ Đức Đông người đã tận tình hướng dẫn, chỉ bảo, giúp đỡ em trong suốt quá trình làm luận văn.

Em cũng xin gửi lời cảm ơn đến các thầy cô giáo trường Đại học Công nghệ thông tin và Truyền thông - Đại học Thái Nguyên, các thầy cô Viện Công nghệ thông tin đã truyền đạt những kiến thức và giúp đỡ em trong suốt quá trình học của mình.

Tôi cũng xin gửi lời cảm ơn tới Sở Giáo dục và Đào tạo Hải Phòng, Ban giám hiệu trường THPT Quang Trung Hải Phòng đã tạo điều kiện thuận lợi cho tôi tham gia khóa học và trong suốt quá trình hoàn thành luận văn.

Cuối cùng, tôi xin gửi lời cảm ơn tới các đồng nghiệp, gia đình và bạn bè những người đã ủng hộ, động viên tạo mọi điều kiện giúp đỡ để tôi có được kết quả như ngày hôm nay.

Thái Nguyên, tháng 7 năm 2016

Học viên

Đỗ Xuân Quyền

MỤC LỤC

DANH MỤC CÁC KÍ HIỆU, CHỮ CÁI VIẾT TẮT

τ_{max}	*Cận trên của vết mùi*
τ_{min}	*Cận dưới của vết mùi*
τ_{mid}	*Cận giữa của vết mùi*
τ_0	*Vết mùi được khởi tạo ban đầu*
τ_{ij}	*Vết mùi trên cạnh* (i, j)
N_c	*Số vòng lặp trong thuật toán ACO*
N_a	*Số kiến sử dụng trong thuật toán ACO*
ρ	*Tham số bay hơi*
3-LAS	*Three-Level Ant System (Hệ kiến ba mức)*
ACO	*Ant Colony Optimization (Tối ưu đàn kiến)*
ACS	*Ant Colony System (Hệ đàn kiến)*
AS	*Ant System (Hệ kiến)*
CRM	*Cis-Regulatory Module (Mô-đun điều tiết)*
EC	*Evolutionary Computing (Tính toán tiến hoá)*
GA	*Genetic Algorithm(Thuật toán di truyền)*
G-best	*Global-best (Lời giải tốt nhất tính đến thời điểm hiện tại)*
I-best	*Iteration-best (Lời giải tốt nhất trong bước lặp hiện tại)*
MLAS	*Multi-level Ant System (Hệ kiến đa mức)*
MMAS	*Max-Min Ant System (Hệ kiến Max Min)*
SA	*Simulated Annealing (Thuật toán mô phỏng luyện kim)*
SMMAS	*Smoothed Max-Min Ant System (Hệ kiến Max Min trơn)*
n_{keep}	*Số nút cần giữ lại*
$SeedV_{12}$	*Tập nút đã dóng hàng thuộc đồ thị* G_1

DANH MỤC CÁC HÌNH

DANH MỤC CÁC BẢNG

MỞ ĐẦU

Tin sinh học là một lĩnh vực khoa học mới rất được quan tâm trong hơn thập kỷ gần đây, với sự phát triển của khoa học công nghệ, mối liên kết giữa sinh học và tin học trở lên khăng khít hơn, Tin học hỗ trợ cho việc giải quyết các bài toán với dữ liệu lớn của Sinh học, đồng thời đó cũng là một hướng phát triển mới của ngành Tin học về giải thuật và ứng dụng.

Dóng hàng hai đồ thị là một bài toán quan trọng trong lý thuyết đồ thị, nó giúp chúng ta xác định tính tương đồng của hai đồ thị. Về mặt sinh học nó giúp xác định tính tương đồng giữa các mạng tương tác protein.Hiện nay có nhiều tiêu chí về cách đánh giá cho dóng hàng. Một cách đánh giá thường được sử dụng hiện nay là đánh giá dựa trên lực lượng của tập cạnh (sự tương đồng về cấu trúc) và sự tương đồng giữa các nút.

Dóng hàng hai đồ thị được Aladag và Erten chứng minh là bài toán thuộc lớp NP-khó [4] và có nhiều ứng dụng. Đặc biệt, trong những năm gần đây, với sự phát triển của các kỹ thuật sinh học công nghệ cao đã cho phép các nhà nghiên cứu xây dựng được các mạng tương tác protein (Protein-Protein Interraction Network – PPI Network) tương đối đầy đủ cho nhiều loài sinh vật. Bài toán dóng hàng mạng PPI là một bài toán quan trọng trong phân tích mạng PPI nói chung.***Các mạng tương tác protein được mô tả bằng đồ thị, bài toán dóng hàng mạng được chuyển tải về bài toán dóng hàng đồ thị***.

Phương pháp tối ưu đàn kiến (Ant Colony Optimization - ACO) là cách tiếp cận metaheuristic , được giới thiệu bởi Dorigo năm 1991 đang được nghiên cứu và ứng dụng rộng rãi cho các bài toán tối ưu tổ hợp khó.

Chính vì vậy tác giả chọn đề tài khoa học ***phương pháp tối ưu đàn kiến dóng hàng hai đồ thị***. Thực nghiệm, tác giả sẽ sử dụng bộ dữ liệu vào

(Input) mà Spinal và FastNA đã dùng, từ đó sẽ đánh giá và so sánh kết quả ra để thấy được hiệu quả của phương pháp tác giả đề xuất với những phương pháp có trước, cụ thể là so sánh với hai phương pháp tốt nhất hiện nay Spinal và FastNA.

CHƯƠNG I
DÓNG HÀNG HAI ĐỒ THỊ VÀ CÁC PHƯƠNG PHÁP TIẾP CẬN HIỆN NAY

1.1. Bài toán dóng hàng hai đồ thị

Giả sử $G_1 = (V_1, E_1)$ và $G_2 = (V_2, E_2)$ là hai mạng tương tác protein (đơn đồ thị), trong đó V_1, V_2ký hiệu tập các nút mô tả các protein trong mạng G_1, G_2 tương ứng;E_1, E_2 ký hiệu tập các cạnh mô tả mối quan hệ tương tác giữa các protein trong các mạng G_1, G_2. Không giảm tổng quát, ta xem $|V_1| \leq |V_2|$ trong đó $|V|$ ký hiệu số phần tử của tập V.

Dóng hàng mạng là tìm một đơn ánh từ V_1vào V_2 tốt nhất theo một tiêu chí đánh giá nào đó. Hiện nay chưa có định nghĩa rõ ràng cho tiêu chí này, dưới đây phát biểu toán học cho định nghĩa bài toán dóng hàng theo tiêu chí thông dụng đã được dùng trong [4,5,6,12,27].

Định nghĩa 1. (Dóng hàng mạng) Đồ thị $A_{12} = (V_{12}, E_{12})$ là một mạng dóng hàng của hai đồ thịG_1, G_2 nếu nó thỏa mãn:

i) Mỗi nút của V_{12} được ký hiệu là $< u_i, v_j >$ tương ứng với một cặp đỉnh u_i thuộc V_1 và v_j thuộc V_2.

ii) Hai nút phân biệt$< u_i, v_j >$ và $< u_i^{'}, v_j^{'} >$ thuộc V_{12}thì$u_i \neq u_i^{'}$ và $v_j \neq v_j^{'}$

iii) Cạnh($< u_i, v_j >, < u_i^{'}, v_j^{'} >$) thuộc E_{12} nếu và chỉ nếu $(u_i, u_i^{'}) \in E_1$ và $(v_i, v_i^{'}) \in E_2$.

Định nghĩa 2. (Dóng hàng mạng toàn cục) Một dóng hàng $A_{12} = (V_{12}, E_{12})$ là lời giải của bài toán dóng hàng toàn cục của các mạng proteins G_1, G_2 nếu nó cực đại *global network alignment score* cho bởi Eq(1.1):

$$GNAS(A_{12}) = \alpha|E_{12}| + (1-\alpha)\sum_{\forall <u_i, v_j>} similar(u_i, v_j) \qquad (1.1)$$

trong đó $\alpha \in [0,1]$ là tham số cân bằng giữa sự tương đồng về tô pô mạng và sự tương đồng trình tự giữa các nút, giá trị $Similar(u_i, v_j)$được tính xấp xỉ dựa trên BLAST bit-scores hoặc E-values.

Trong [4] Aladag và Erten đã chứng minh bài toán tìm dóng hàng tối ưu này là NP-hard.

1.2. Một số phương pháp tiếp cận hiện nay

Từ khi bài toán được đề xuất, các kỹ thuật dóng hàng mạng PPI phát triển theo hai hướng: dóng hàng cục bộ và dóng hàng toàn cục. Với dóng hàng cục bộ, mục tiêu sẽ là xác định các mạng con gần nhau về tô pô mạng hoặc tương tự xâu [11,23,27]. Thông thường, kết quả của dóng hàng cục bộ sẽ thể hiện nhiều mạng con chồng lấn nhau, điều này có thể dẫn đến sự nhập nhằng khi một protein có thể được dóng hàng với nhiều protein khác. Để khắc phục nhược điểm đó dóng hàng mạng toàn cục sẽ tìm ra một đơn ánh mà mỗi protein của mạng PPI này chỉ dóng với một protein ở mạng PPI kia.

IsoRank [4,28] được Sing et al. đề xuất năm 2008 là một trong những thuật toán dóng hàng mạng toàn cục đầu tiên, nó được phát triển dựa trên dóng hàng cục bộ. Ý tưởng chính của IsoRank là hai nút được dóng hàng với nhau, nếu các nút kề với chúngtương ứng được dóng hàng.Thuật toán bao gồm hai giai đoạn chính, thứ nhất thực hiện với mỗi nút i ($i \in V_1 | \ i = 1..|V_1|$) phải duyệt tất cả nút j ($j \in V_2 | \ j = 1..|V_2|$) để tìm*điểm sốdóng hàng* $R_{i,j}$ cho từng cặp nút i và j, theo

$$R_{i,j} = \sum_{u \in N(i)} \sum_{v \in N(j)} \frac{1}{|N(u)|.|N(v)|} . R_{u,v} \qquad (1.2)$$

ở công thức trên*N(x)* là tập tất cả các nút láng riềng của *x*. Kết quả của giai đoạn này cho ta một ma trận giá trị (*điểm số dóng hàng*)*R* tương ứng với mọi cặp nút<*i*,*j*> ($i \in V_1, j \in V_2$).Giai đoạn thứ hai, từ ma trận *R* tìm ra một cách sắp xếp (ghép) từng cặp *i*,*j* ($i \in V_1, j \in V_2$) sao cho tổng điểm số dóng hàng là

tốt nhất.Việc này được cho là đơn giản vì nó chỉ là cách sắp xếp hai đồ thị dựa trên mức độ tương tự giữa các nút.

Sau IsoRank, một số thuật toán tương tự đã được đề xuất như PATH và GA [35], PISwap [6,7] nhờ đưa thêm các nới lỏng thích hợp của hàm đánh giá trên tập các ma trận ngẫu nhiên hoặc ứng dụng tìm kiếm cục bộ trên dóng hàng lời giải có sẵn từ một thuật toán khác. MI-GRAAL [12,13] và các biến thể [16,17] dựa trên kết hợp kỹ thuật tham ăn với thông tin heuristics như: graphlet, hệ số phân nhóm, độ lập dị (eccentricities) và độ tương tự (giá trị E-values từ chương trình BLAST). Các thuật toán này đều đưa ra kết quả nhanh và tốt hơn so với các thuật toán trước đó. Tuy nhiên, những thuật toán đã nêu chỉ tối ưu cho độ chính xác (hàm mục tiêu) hoặc tính khả mở (scalability). Vì các mạng PPI có thường số đỉnh lớn nên cả tính chính xác và tính khả mở (thời gian chạy) cần được quan tâm. Gần đây, đã xuất hiện những thuật cho kết quả tốt với độ phức tạp thời gian đa thức điển hình là SPINAL và FastNA sẽ được giới thiệu và tìm hiểunhiều hơn.

1.2.1. SPINAL

Năm 2013 Aladag và Erten đề xuất thuật toán SPINAL [4], thuật toán này cho kết quả tốt nhất theo tiêu chí đánh giá của công thức (1.1) và nhanh nhất tính đến thời điểm nó ra đời. SPINAL là một thuật toán heuristic thời gian đa thức, gồm hai pha:

Pha đầu tính điểm tương đồng cho tất cả cặp protein theo công thức

$$T_{(u_i,v_j)} = \alpha \times \frac{\sum_{(x_i,y_j)\in C} \frac{P(x_i,y_j)}{\deg_{G_1}(x_i)\times \deg_{G_2}(y_j)}}{\sqrt{|C|}} + (1-\alpha)\times seq(u_i,v_j) \qquad (1.3)$$

trong đó$\alpha\in$ [0,1] là tham số cân bằng giữa sự tương đồng về tô pô mạng và sự tương đồng trình tự giữa các nút, $\deg_{G_1}(x_i)$, $\deg_{G_2}(y_j)$ tương ứng là bậc của x_i và y_j trong *G1* và *G2,và seg(u_i,v_j)* được tính xấp xỉ dựa trên BLAST bit-scores hoặc E-values.

Pha sau xây dựng đơn ánh xạ bằng cách cải tiến một cách cục bộ từng tập con của lời giải hiện có.

Algorithm 1:SPINAL global alignment algorithm

1: **Input:** $G_1 = (V_1, E_1), G_2 = (V_2, E_2), seg, \alpha$

2: **Output:** Node set V_{12} of the global alignment network A_{12}

3: // Coarse-grained

4: **for** all $u_i \in V_1, v_j \in V_2$ **do**

5: $P(u_i, v_j) = \alpha \times DegDiff(u_i, v_j) + (1-\alpha) \times seg(u_i, v_j)$

6: **endfor**

7: **repeat**

8: $P' = P$

9: **for** all $u_i \in V_1, v_j \in V_2$ **do**

10: construct $NBG(\{\prec u_i, v_j \succ\}, P')$

11: construct contributors set *C* of *NBG*

12: comput $P(u_i, v_j)$ as in Equation(2)

13:**endfor**

14:**until** enough iterations

15:// Fine-grained

16: $SP = List$ of $\prec u_i, v_j \succ sorted$ w.r.t P, for $u_i \in V_1, v_j \in V_2$

17:**repeat**

18: // Find new connected component in A_{12}

19: pop unaligned $\prec u_i, v_j \succ$ from SP, insert into V12

20:**repeat**

21: Construct $NBG(V_{12}, P)$

22: Construct contributors set *C* of *NBG*

23: Swap improvements for each *NBG* edge not in *C*

24: Insert $\prec x_i, y_j \succ$ into V_{12}, for each $(x_i, y_j) \in C$

25:**until** no contributors

26:**until** no unaligned pair in *SP*

Hình 1.1: Thuật toán SPINAL

theo [4] thuật toán SPINAL được chứng minh có độ phức tạp thời gian đa thức:

SPINAL Complexity = $O(k \times |V_1| \times |V_2| \times \Delta_1 \times \Delta_2 \times \log(\Delta_1 \times \Delta_2))$ (1.4)

trong đó k là số lần chạy của vòng lặp chính (thuật toán hội tụ với số vòng lặp từ 10-15 trong các trường hợp thử nghiệm), Δ_1, Δ_2 lần lượt là bậc lớn nhất của đỉnh trong đồ thị G_1, G_2

Thực nghiệm trên các tập dữ liệu Saccharomyces cerevisiae, Drosophila melanogaster, Caenorhabditis elegans and Homo sapiens cho thấy SPINAL tốt hơn hai thuật toán IsoRank và MI-GRAAL, là hai thuật toán tốt nhất đến lúc đó.

1.2.2. FastNA

Năm 2013, Tiến sĩ Đỗ Đức Đông và một số đồng nghiệp thuộc Viện Công nghệ thông tin, Đại học Quốc giađã đề xuất một thuật toán mới là FastNA đểdóng hàng toàn cục mạng PPI.

Thuật toán gồm hai pha: pha thứ nhất xây dựng dóng hàng ban đầu

Algorithm 1 Procedure of FastNA
Input: Graph 1: $G_1 = (V_1, E_1)$; Graph 2: $G_2 = (V_2, E_2)$;
Similarities of node pairs: $Similar[i][j]$;
Balancing parameterα
Output: Alignment network$G_{12} = (V_{12}, E_{12})$
Begin
$V_{12} = \{< i, j >\}$//The best similar pair $< i, j >$
for$k$=2 **to**$|V_1|$**do**
i = find_next_node(G_1);
j = choose_best_matched_node(i, G_1, G_2);
$V_{12} = V_{12} \cup < i, j >$
Update(E_{12})
end-for
Rebuild(G_{12});
End

Hình 1.2: Thuật toán FastNA

Có thể diễn tả lại thuật toán của pha thứ nhất như sau:

Input gồm: đồ thị G_1, G_2; tham sốα và các độ tương tự của các cặp đỉnh $<i,j>$ tương ứng của V_1, V_2.

Bước 1. Khởi tạo V_{12} là cặp đỉnh $<i,j>$ có độ tương tự lớn nhất

Bước 2. Thực hiện lặp với k= *2* tới $|V_1|$

2.1. Tìm node i trong $V_1 - V_{12}^1$có số cạnh tới các đỉnh trong V_{12}^1 lớn nhất;

2.2. Tìm node j trong $V_2 - V_{12}^2$ mà khi bổ sung $< i, j >$ vào V_{12} thì *GNAS(A_{12})* tính bởi (1.1) lớn nhất, trong đó A_{12} là đồ thị có đỉnh là tập V_{12} và các cạnh sinh bởi G_1, G_2. Khi đó j đượi gọi làbest_matched_node($i, V_{1,2}$);

2.3. Bổ sung $< i, j >$ vào V_{12};

2.4. Update E_{12} dựa trên V_{12}; //bổ sung thêm các cặp cạnh khớp vào E_{12} khi thêm cặp nút <i,j> trong bước 2.3

Bước 3. Thực hiện lặp cải tiến$G_{12} = (V_{12}, E_{12})$ nhờ thủ tục *Rebuild*.

Nhận xét: sau pha thứ nhất FastNa đã thu được một dóng hàng toàn cục ban đầu G_{12}, tuy nhiên nó sẽ chưa tốt vì khi khởi tạo G_{12} chỉ có một cặp đỉnh <*i,j*> và bước 2 khi tìm node *i* trong $V_1 - V_{12}^1$có số cạnh tới các đỉnh trong V_{12}^1 lớn nhất để ghép, lúc này vì số nút ghép được ở những lần ghép đầu tiên rất ít nên việc xác định nút i(có nhiều cạnh nối tới các đỉnh trong đồ thị đã ghép được) để tiếp tục ghép chưa chắc đã cho một kết quả tốt nhất. Đó chính là nhược điểm rất lớn dẫn tới G_{12}ban đầu có chất lượng theo yêu cầu là không tốt. Điều này được khắc phục ở pha thứ hai với mục tiêutối ưu cục bộ với *thủ tục Rebuild*.

Thủ tục Rebuild

Với G_{12}đã được xây dựng trong pha đầu và số n_{keep} đã cho để xác định số lượng nút trong tập SeedV_{12} , thủ tục này được thực hiện như sau:

Bước 1. Xác định tập SeedV_{12} của V_1 gồm n_{keep} đỉnh có điểm số (score) tốt nhất của V1 theo tiêu chí cho bởi (1.5):

$$score(u) = \alpha \times w(u) + (1 - \alpha) \times similar(u, f(u)) \quad (1.5)$$

trong đó *u* thuộc V_1và $f(u)$ là đỉnh thuộc V_2được ghép với *u* trongG_{12},*w(u)* là số lượng nút *v* thuộc V_1 mà (*u,v*) thuộc E_1 và (*f(u),f(v)*) thuộc E_2

Bước 2. Xác định V_{12} khởi tạo nhờ $SeedV_{12_1}$và G_{12}

Bước 3. Thực hiện lặp như bước 2 của phase 1 vơi $k = n_{keep} + 1$ tới $|V_1|$để xác định A_{12}

Sau mỗi lần thực hiện thủ tục Rebuild sẽ có một dóng hàng mới làm input G_{12}cholần lặp tiếp theo, quá trình này lặp lại cho đến khi không cải tiến được GNAS(A_{12}) nữa.

Rebuild được đặc tả như sau

Algorithm 2 Procedure of Rebuild
Input: Alignment networkG_{12};n_{keep}
Output: Better Alignment network$A_{12} = (V_{12}, E_{12})$
Begin
Build $SeedV_{12}$;
Buld V_{12}// based on $SeedV_{12_1}$ and G_{12}
for$k$=n_{keep} +1 **to**$|V_1|$**do**
i = find_next_node(G_1);
j = choose_best_matched_node(i, G_1, G_2);
$V_{12} = V_{12} \cup < i, j >$
Update(E_{12})
end-for
end

Hình 1.3: Thủ tục Rebuild - FastNA

Pha thứ hai với thủ tục Rebuild là một ý tưởng độc đáo, nó trở điểm mạnh của thuật toán, khắc phục nhược điểm của pha thứ nhất và cho một kết quả tốt hơn hẳn về chất lượng dóng hàng và cả thời gian thực hiện.

Theo FastNA, Tiến sĩ Đỗ Đức Đông và các đồng tác giả củaFastNA đã chứng minh độ phức tạp về thời gian của thuật toán mình đề xuất thấp hơn so với thuật toán tốt nhất tại thời điểm này là SPINAL, cụ thể như sau:

Độ phức tạp của phase 1 và mỗi bước lặp trong phase 2 của thuật toán FastNA là:

$$O(|V_1| \times (E_1| + |E_2|)) \quad (1.6)$$

số lần lặp của phase 2 trong thực nghiệm không vượt qúa 10. Bởi vì $|V_1| \times \Delta_1 \geq E_1$liên hệ tới độ phức tạp của SPINAL trong công thức (4) ta có:

$$|V_1| \times |V_2| \times \Delta_1 \times \Delta_2 \geq E_1 \times E_2 > (|V_1| \times (E_1| + |E_2|)) \quad (1.7)$$

Như vậy ta có thể khẳng định lại độ phức tạp của FastNA trong biểu thức (1.6) so với độ phức tạp của SPINAL trong biểu thức (1.4) thấp hơn nhiều.

Cũng theo FastNA thực nghiệm trên cùng 4 bộ dữ liệu đã được thử cho SPINAL trong [4] để so sánh mục tiêu GNAS và số cạnh khớp (Edge Correctness: EC). Kết quả thực nghiệm cho thấy với mọi giá trị α (*α=0.3, α=0.4, α=0.5, α=0.6, α=0.7*) trên cả 6 trường hợp thực nghiệm(*ce-dm, ce-hs, ce-sc, dm-hs, dm-sc, hs-sc*)FastNA đều tìm ra được lời giải mà hàm mục tiêu GNAS và số cạnh khớp EC nhiều hơn và thời gian thực hiện cũng tốt hơn hẳn so vớiSPINAL.

1.3. Kết luận chương

Dóng hàng hai đồ thị (dóng hàng toàn cục các mạng tương tác PPI) là một bài toán quan trọng trong lĩnh vực sinh học cũng như trong lý thuyết đồ thị, qua nhiều năm nghiên cứu rất nhiều tác giả trên thế giới đã đưa ra những thuật toán tốt. Thực tế cho thấy rằng, những thuật toán được công bố trên các tạp chí khoa học toàn thế giới tính theo thời gian dần về sau, đều cho kết quả tốt hơn những thuật toán có trước. Nhưng tất cả các thuật toán đều có một điểm chung đó là dựa trên kỹ thuật tham kết hợp với các thông tin heuristic.Riêng với FastNA có sự khác biệt, khi đề xuất cải tiến chất lượng nhờ dỡ bỏ dóng hàng ban đầu và chỉ giữ lại lượng một phần số nút tốt nhất (nút có điểm số tính theo Eq(1.5) cao nhất) để làm cơ sở cho việc tìm kiếm các nút dóng hàng tiếp theo trong lần thực hiện sau. Trên cơ sở của các thuật toán đã tồn tại tác giả nhận thấy nếu kết hợp thông tin heuristic với cách học tăng cường của phương pháp metaheuristic trong việc tìm nút dóng hàng sẽ cho kết quả tốt hơn.Vì theo các phương pháp heuristic kết quả sẽ giao động lúc gần, lúc xa so với kết quả tốt nhất (mang tính ngẫu nhiêu); nhưng khi tìm kết quả theo cách học tăng cường thì kết quả sau luôn luôn cho kết quả tốt hơn (gần hơn tới kết quả tốt nhất mong muốn).

CHƯƠNG II
PHƯƠNG PHÁP TỐI ƯU ĐÀN KIẾN

Trong lĩnh vực công nghệ thông tin, đặc biệt là thời điểm hiện nay ta thường gặp các bài toán tối ưu tổ hợp, khi đó phải tìm các giá trị cho các biến rời rạc để làm cực trị hàm mục tiêu nào đó. Đa số các bài toán này thuộc lớpNP-khó.Trừ các bài toán cỡ nhỏ có thể tìm lời giải bằng cách tìm kiếm vét cạn, còn lại thì thường không thể tìm được lời giải tối ưu.

Đối với các bài toán cỡ lớn không có phương pháp giải đúng, đến nay hướngtiếp cậnvẫn theo các phương pháp sau:

1) Tìm kiếm heuristic, trong đó dựa trên phân tích toán học, người ta đưa ra các quy tắc định hướng tìm kiếm một lời giải đủ tốt.

2) Sử dụng các kỹ thuật tìm kiếm cục bộ để tìm lời giải tối ưu địa phương.

3) Tìm lời giải gần đúng nhờ các thuật toán mô phỏng tự nhiên như mô phỏng luyện kim (Simulated Annealing - SA), giải thuật di truyền (*Genetic Algorithm - GA*), tối ưu bầy đàn (Particle Swarm Optimization -PSO)…

Hai cách tiếp cận đầu thường cho lời giải nhanh nhưng việc cải thiện lời giải rất hạn chế, nên cách tiếp cận thứ ba đang được sử dụng rộng rãi cho các bài toán cỡ lớn.

Trongcác phương pháp mô phỏng tự nhiên, tối ưu đàn kiến (Ant Colony Optimization - ACO) là cách tiếp cận metaheuristic tương đối mới , được giới thiệu bởi Dorigo năm 1991 (xem [19,20,22]) đang được nghiên cứu và ứng dụng rộng rãi cho các bài toán tối ưu tổ hợp khó.

Các thuật toán ACO mô phỏng cách tìm đường đi của các con kiến thực. Trên đường đi, mỗi con kiến thực để lại một vết hoá chất gọi là vết mùi(pheromone trail) và theo vết mùi của các con kiến khác để tìm đường đi. Đường có nồng độ vết mùi càng cao thì càng có nhiều khả năng được các con kiến chọn. Nhờ cách giao tiếp gián tiếp này, đàn kiếntìm được đường đi ngắn

nhất từ tổ tới nguồn thức ăn. Theo ý tưởng đó, các thuật toán ACO sử dụng kết hợp thông tin kinh nghiệm (heuristic) và học tăng cường qua các vết mùi của các con kiến nhân tạo để giải các bài toán tối ưu tổ hợp bằng cách đưa về bài toán tìm đường đi tối ưu trên đồ thị cấu trúc tương ứng của bài toán.

Với mỗi bài toán tối ưu tổ hợptồn tại bộ(S, f, Ω), trong đó S là tập hữu hạn các phương án chấp nhận được, f là hàm mục tiêu xác định trên S và Ω là để xác định S qua các thành phần của tập hữu hạn C và các liên kết của tập này, mỗi lời giải s trong S sẽ tương ứng với một hoặc một tập hữu hạn các vectơ$x = < u_0, \ldots, u_k > (u_i \in C \; \forall i \leq k \leq h)$ có độ dài bị chặn thỏa mãn các ràng buộc Ω. Như vậy, việc tìm kiếm các lời giải trong S được đưa về xây dựng lời giải trên các vectơ độ dài không quá h thỏa mãn ràng buộc đã cho. Về lý thuyết, quá trình giải có thể thực hiện trên đồ thị đầy đủ có tập đỉnh được gán nhãn bởi tập C với một số thông tin thêm (được gọi là đồ thị cấu trúc). Tùy theo các ràng buộc mà trong nhiều trường hợp ta có thể xét bài toán tìm trên đồ thị cấu trúc đơn giản hơn để thu hẹp miền tìm kiếm.

Trong mỗi lần lặp của các thuật toán, một đàn kiến nhân tạo sẽ xây dựng lời giải theo thủ tục phát triển tuần tự trên đồ thị cấu trúc, sau đó so sánh lời giải tìm được để cập nhật vết mùi như là thông tin học tăng cường dùng cho các vòng lặp sau. Nhờ đó mà lời giải được cải tiến dần.Các thuật toán này được áp dụng rộng rãi để giải nhiều bài toán khó và hiệu quả nổi trội của chúng so với các phương pháp mô phỏng tự nhiên khác đã được chứng tỏ bằng thực nghiệm.

Khi áp dụng phương pháp ACO cho mỗi bài toán cụ thể, có ba yếu tố quyết định hiệu quả thuật toán:

1) Xây dựng đồ thị cấu trúc thích hợp;

2) Chọn thông tin heuristic;

3) Chọn quy tắc cập nhật mùi.

Hai yếu tố đầu phụ thuộc vào đặc điểm của từng bài toán cụ thể, còn quy tắc cập nhật mùi là yếu tố phổ dụng cho các bài toán và nó thường dùng làm tên để phân biệt các thuật toán ACO.

2.1. Từ kiến tự nhiên đến kiến nhân tạo

Khi tìm đường đi, đàn kiến trao đổi thông tin gián tiếp và hoạt động theo phương thức tự tổ chức. Mặc dù đơn giản nhưng phương thức này giúp cho đàn kiến có thể thực hiện được những công việc phức tạp vượt xa khả năng của từng con kiến, đặc biệt là khả năng tìm đường đi ngắn nhất từ tổ đến nguồn thức ăn mặc dù chúng không có khả năng đo độ dài đường đi. Trước hết ta xem cách đàn kiến tìm đường đi như thế nào mà có thể giải quyết được các vấn đề tối ưu hóa.

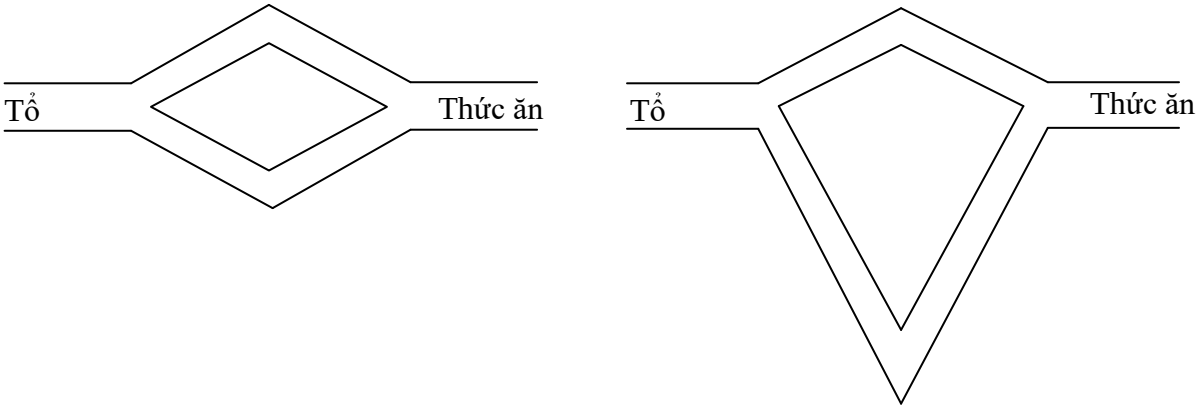
2.1.1.Kiến tự nhiên

Trên đường đi, mỗi con kiến để lại một chất hóa học gọi là vết mùi (pheromone) dùng để đánh dấu đường đi. Bằng cách cảm nhận vết mùi, kiến có thể lần theo đường đi đến nguồn thức ăn được các con kiến kháckhám phá theo phương thức chọn ngẫu nhiên có định hướng theo nồng độ vết mùi. Kiến chịu ảnh hưởng vết mùi của các con kiến khác chính là ý tưởng thiết kế thuật toán ACO.

Thí nghiệm trên cây cầu đôi

Có nhiều thực nghiệm nghiên cứu về hành vi để lại vết mùi và đi theo vết mùi của loài kiến. Thực nghiệm, được thiết kế bởi Deneubourg và các đồng nghiệp [22], dùng một chiếc cầu đôi nối từ tổ kiến tới nguồn thức ăn, như minh họa trong hình 2.1.Họ đã thực nghiệm với tỉ lệ độ dài đường $r = \frac{l_l}{l_s}$ giữa hai nhánh khác nhau của chiếc cầu đôi, trong đó l_l là độ dài của nhánh dài còn l_s là độ dài của nhánh ngắn.

Trong thực nghiệm thứ nhất, chiếc cầu đôi có hai nhánh bằng nhau ($r = 1$, hình 2.1.a). Ban đầu, kiến lựa chọn đường đi một cách tự do từ tổ đến nguồn thức ăn, cả hai nhánh đều có kiến đi, nhưng sau một thời gian các con kiến này tập trung đi theo cùng một nhánh. Kết quả có thể được giải thích như sau: ban đầu không có vết mùi nào trên cả hai nhánh, do đó kiến lựa chọn nhánh bất kỳ với xác suất như nhau. Một cách ngẫu nhiên, sẽ có một nhánh có số lượng kiến lựa chọn nhiều hơn nhánh kia. Do kiến để lại vết mùi trong quá trình di chuyển, nhánh có nhiều kiến lựa chọn sẽ có nồng độ mùi lớn hơn nồng độ mùi của nhánhcòn lại. Nồng độ mùi trên cạnh lớn hơn sẽ ngày càng lớn hơn vì ngày càng có nhiều kiến lựa chọn.Cuối cùng, hầu như tất cả các kiến sẽ tập trung trên cùng một nhánh.Thực nghiệm này cho thấylà sự tương tác cục bộ giữa các con kiến với thông tin gián tiếp là vết mùi để lại cho phépđiều chỉnh hoạt động vĩ mô của đàn kiến.



(a) (b)

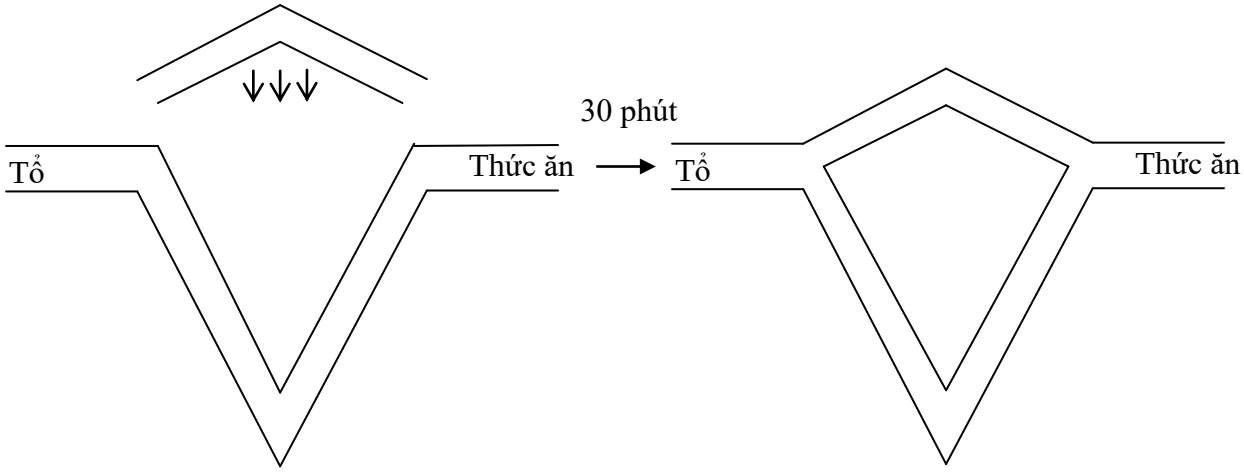
Hình 2.1: Thực nghiệm cây cầu đôi

(a) Hai nhánh có độ dài bằng nhau. (b) Hai nhánh có độ dài khác nhau.

Trong thực nghiệm thứ hai(xem hình 2.1b), độ dài của nhánh dài gấp đôi độ dài nhánh ngắn (tỉ lệ $r = 2$). Trong trường hợp này, sau một thời gian

tất cả các con kiến đều chọn đoạn đường ngắn hơn. Cũng như trong thực nghiệm thứ nhất, ban đầu đàn kiến lựa chọn hai nhánh đi như nhau, một nửa số kiến đi theo nhánh ngắn và một nửa đi theo nhánh dài (mặc dù trên thực tế, do tính ngẫu nhiên có thể một nhánh nào đó được nhiều kiến lựa chọn hơn nhánh kia). Nhưng thực nghiệm này có điểm khác biệt quan trọng với thực nghiệm thứ nhất: Những kiến lựa chọn đi theo nhánh ngắn sẽ nhanh chóng quay trở lại tổ và khi phải lựa chọn giữa nhánh ngắn và nhánh dài, kiến sẽ thấy nồng độ mùi trên nhánh ngắn cao hơn nồng độ mùi trên nhánh dài, do đó sẽ ưu tiên lựa chọn đi theo nhánh ngắn hơn. Tuy nhiên, trong thời gian đầu không phải tất cả các kiến đều đi theo nhánh ngắn hơn.Phải mất một khoảng thời gian tiếp theo nữa bầy kiến mới lựa chọn đi theo nhánh ngắn. Điều này minh chứng bầy kiến đã sử dụng phương thức thăm dò, tìm đường mới.

Một điểm thú vị nữa là quan sát xem sẽ xảy ra điều gì khi quá trình tìm kiếm đang hội tụ, lại xuất hiện một đường mới từ tổ đến nguồn thức ăn. Việc này được thực nghiệm như sau: ban đầu từ tổ đến nguồn thức ăn chỉ có một nhánh dài và sau 30 phút, thêm một nhánh ngắn (xem hình 2.2). Trong trường hợp này, nhánh ngắn thường không được kiến chọn mà chúng tập trung đi trên nhánh dài. Điều này có thể giải thích như sau: nồng độ vết mùi trên cạnh dài cao và sự bay hơi của vết mùi diễn ra chậm nên đại đa số các con kiến vẫn lựa chọn nhánh dài (có nồng độ vết mùi cao).Hành vi này tiếp tục được củng cố vàkiến chọn đi theo nhánh dài, ngay cả khi có một nhánh ngắn xuất hiện. Việc bay hơi vết mùi là cơ chế tiện lợi cho việc tìm đường mới, nghĩa là việc bay hơi có thể giúp kiến quên đi đường đi tối ưu cục bộ đã được tìm thấy trước đây để tìm khám phá đường đi mới, tốt hơn.



Hình 2.2: Thí nghiệm bổ sung

Ban đầu chỉ có một nhánh và sau 30 phút thêm nhánh ngắn hơn

2.1.2.Kiến nhân tạo

Thực nghiệm cây cầu đôi cho thấy đàn kiến tự nhiên có thể sử dụng luật di chuyển theo xác suất, dựa trên thông tin địa phương để tìm được đường đi ngắn nhất giữa hai địa điểm. Vết mùi của đàn kiến cho phép liên tưởng tới cách học tăng cường (reinforcement learning) trong bài toán chọn tác động tối ưu[9], gợi mở mô hình mô phỏng cho bài toán tìm đường đingắn nhất giữa hai nút (tương ứng là tổ và nguồn thức ăn) trên đồ thị, trong đó các tác tử (agent) là đàn kiến nhân tạo.

Tuy nhiên, trong các bài toán ứng dụng các đồ thị thường phức tạp hơn.Từ mỗi đỉnh có thể có nhiều cạnh, nên nếu mô phỏng thực sự hành vi của đàn kiến tự nhiên nhiều con kiến sẽ đi luẩn quẩn và do đó hiệu quả thuật toán sẽ rất kém. Vì vậy, người ta dùng kỹ thuật đa tác tử (multiagent) mô phỏngđàn kiến nhân tạo, trong đó mỗi con kiến nhân tạo có khả năng nhiều hơn so với kiến tự nhiên.Kiến nhân tạo có bộ nhớ riêng, có khả năng ghi nhớ các đỉnh đã thăm trong hành trình và tính được độ dài đường đi nó chọn.

Ngoài ra, kiến có thể trao đổi thông tin với nhau, thực hiện tính toán cần thiết, cập nhật mùi…

Sử dụng mô hình kiến nhân tạo này, Dorigo (1991) [19] đã xây dựng thuật toán *hệ kiến* (AS) giải bài toán người chào hàng.Hiệu quả của thuật toán so với các phương pháp mô phỏng tự nhiên khác như SA và GA đã được kiểm chứng bằng thực nghiệm.Thuật toán này về sau được phát triển và có nhiều ứng dụng phong phú, được gọi chung làphương pháp ACO.

2.2.Phương pháp ACO cho bài toán tối ưu tổ hợp tổng quát

Theo [2], Phần này giới thiệu phương pháp tối ưu đàn kiến.Trước khi mô tả thuật toán tổng quát, ta tìm hiểu đồ thị cấu trúc cho bài toán tối ưu tổ hợp.

2.2.1.Đồ thị cấu trúc

Xét bài toán tối ưu tổ hợp tổng quát được nhắc đến phần đầu chương, dưới dạng bài toán cực tiểu hoá (S, f, Ω), trong đó S là tập hữu hạn trạng thái (lời giải tiềm năng hay phương án), f là hàm mục tiêu xác định trên S, cònΩ là các ràng buộc để xác định tậpS có các thành phần được lấy từ tập hữu hạn C. Các tập S, C và Ωcó các đặc tính sau:

1) Ký hiệu X là tập các vectơ trong C độ dài không quá h: $X = \{< u_0, \ldots, u_k >: u_i \in C,\ \forall i \leq k \leq h\}$.Khi đó, mỗi phương án s trong S được xác địnhbởi ít nhất mộtvectơ trong X như ở điểm 2).

2) Tồn tại tập con X^*của X và ánh xạ φ từ X^* lên S sao cho$\varphi^{-1}(s)$ không rỗng với mọi $s \in S$, trong đó tập X^*có thể xây dựng được từ tập conC_0 của C nhờ mở rộng tuần tự dưới đây.

3) Từ C_0 ta mở rộng tuần tự thành X^*như sau:

i) Ta xem $x_0 = < u_0 >$là mở rộng được với mọi $u_0 \in C_0$.

ii) Giả sử $x_k =< u_0, \ldots, u_k >$ là mở rộng được và chưa thuộc X^*. Từ tập ràng buộc Ω, xác định tập con $J(x_k)$ của C, sao cho với mọi $u_{k+1} \in J(x_k)$ thì $x_{k+1} =< u_0, \ldots, u_k, u_{k+1} >$ là mở rộng được.

iii) Áp dụng thủ tục mở rộng từ các phần tử $u_0 \in C_0$ cho phép ta xây dựng được mọi phần tử củaX^*.

Như đã nói trong phần trước, mỗi bài toán tối ưu tổ hợp được xem như một bài toán tìm kiếm vectơ độ dài không quá h trên đồ thị đầy đủ có các đỉnh được gán nhãn trong tập C. Để tìm các lời giải chấp nhận được, ta xây dựng đồ thị đầy đủ với tập đỉnh V, mỗi đỉnh của nó tương ứng với mỗi thành phần củaC. Các lời giải chấp nhận được sẽ là các vectơ được xác định theo thủ tục mở rộng tuần tự hay mở rộng ngẫu nhiên, như đã được mô tả chi tiếttrong mục2.2.2.

Thông thường, đối với các bài toán thuộc loạiNP-khó, người ta đưa ra các phương pháp heuristic tìm lời giải đủ tốt cho bài toán. Các thuật toán ACO kết hợp thông tin heuristic này với phương pháp học tăng cường, mô phỏng hành vi của đàn kiến, để tìm lời giải tốt hơn.

Mỗi cạnh nối đỉnh $i, j \in C$ có trọng số heuristic $h_{i,j}$ để định hướng chọn thành phần mở rộng là j khi thành phần cuối của trạng thái hiện tại x_k là i(theo thủ tục mở rộng tuần tự đã nêu ở trên). Ký hiệu H là vectơ các trọng số heuristic của cạnh, còn τlà vectơ biểu thị các thông tin học tăng cường $\tau_{i,j}$ (trong luận văn từ nay về sau gọi là vết mùi, ban đầu được khởi tạo giá trị τ_0>0). Trường hợp đặc biệt$h_{i,j}$ và $\tau_{i,j}$ chỉ phụ thuộc vào j, các thông tin này sẽ gắn với các đỉnh. Không làm mất tính tổng quát, ta xét trường hợp các thông tin này gắn vào các cạnh.

Ta gọi đồ thị $G = (V, E, H, \tau)$ là *đồ thị cấu trúc* của bài toán tối ưu tổ hợp, trong đó V là tập đỉnh,E là tập cạnh,H và τlà các thông tin gắn với cạnh. Từ các

cạnh, xây dựng tập X^* nhờ mở rộng tập C_0theo thủ tục tuần tự. Nếu không có thông tin heuristic thì ta xem H có các thành phần như nhau và bằng 1.

Trường hợp tổng quát, G là đồ thị đầy đủ.Tuy nhiên, tùy theo ràng buộc của bài toán, các cạnh có thể lược bớt để giảm miền tìm kiếm lời giải theo thủ tục mở rộng tuần tự.

2.2.2.Mô tả thuật toán ACO tổng quát

Sử dụng điều kiện kết thúc (có thể theo số bước lặp hoặc/và giới hạn thời gian chạy), ta dùng đàn kiến cóm con,tiến hành lặp quá trình xây dựng lời giải trên đồ thị cấu trúc$G = (V, E, H, \tau)$ như sau:Tại mỗi lần lặp, kiến chọn ngẫu nhiên một đỉnh$u_0 \in C_0$ làm thành phần khởi tạo$x_0 = \{u_0\}$ và thực hiện xây dựng lời giải theo thủ tục bước ngẫu nhiên. Dựa trên lời giải tìm được, đàn kiến sẽ thực hiện cập nhật mùi theo cách học tăng cường.

Thủ tục bước ngẫu nhiên:

Giả sử $x_k =< u_0, \ldots, u_k >$ là mở rộng được và chưa thuộc X^*. Từ tập ràng buộc Ω, xác định tập con $J(x_k)$ của C, sao cho với mọi $u_{k+1} \in J(x_k)$ thì$x_{k+1} =< u_0, \ldots, u_k, u_{k+1} >$ là mở rộng được. Đỉnh $j = u_{k+1}$ để mở rộng, được chọn với xác suất $P(j)$ như sau:

$$P(j) = \begin{cases} \frac{[\tau_{ij}]^{\alpha}[h_{ij}]^{\beta}}{\sum_{l \in J(x_k)}[\tau_{il}]^{\alpha}[h_{il}]^{\beta}} & j \in J(x_k) \\ 0 & j \notin J(x_k) \end{cases} \quad (2.1)$$

Trong đó τ_{ij} là thông tin mùi, h_{ij} là thông tin heuristic, α và β là tham số cân bằng giữa thông tin mùi và thông tin heuristic .

Quá trình mở rộng tiếp tục cho tới khi kiến r tìm được lời giải chấp nhận được x^r trongX^* và do đó $s^r = \varphi(x^r) \in S$.

Cập nhật mùi:

Tùy theo chất lượng của lời giải tìm được, vết mùi trên mỗi cạnh sẽ được điều chỉnh tăng hoặc giảm tùy theo đánh giá mức độ ưu tiên tìm kiếm về

sau. Lượng mùi cập nhật theocác quy tắc cập nhật mùi khác nhau sẽ cho các thuật toán khác nhau. Vì vậy,quy tắc cập nhật mùi thường dùng làm tên gọi thuật toán. Các quy tắc thông dụng sẽ được nêu trong mục 2.4, đa số chúng đều có dạng:

$$\tau_{i,j} \leftarrow (1-\rho)\tau_{i,j} + \Delta(i,j) \quad (2.2)$$

đối với các cạnh được cập nhật, trong đó ρ là hằng số thuộc khoảng (0,1) là tỷ lệ lượng mùi bị bay hơi.

Các bước thực hiện của các thuật toán ACO được mô tả trong hình 2.3.

ProcedureThuật toán ACO;
Begin
Khởi tạo tham số, ma trận mùi, khởi tạo m con kiến;
repeat
for$k = 1$**to**m**do**
Kiếnk xây dựng lời giải;
end-for
Cập nhật mùi;
Cập nhật lời giải tốt nhất;
until(Điều kiện kết thúc);
Đưa ra lời giải tốt nhất;
End;

Hình 2.3: Thuật toán ACO

Nhận xét chung về các thuật toán ACO

Nhờ kết hợp thông tin heuristic, thông tin học tăng cường và mô phỏng hoạt động của đàn kiến, các thuật toán ACO có các ưu điểm sau:

1) Việc tìm kiếm ngẫu nhiên dựa trên các thông tin heuristic trở nên linh hoạt và mềm dẻo trên miền rộng hơn so với các phương pháp heuristic đã có.Do đó, cho ta lời giải tốt hơn và có thể tìm được lời giải tối ưu.

2) Học tăng cường thông qua thông tin về cường độ vết mùi cho phép từng bước thu hẹp không gian tìm kiếm, mà vẫn không loại bỏ các lời giải tốt, do đó nâng cao chất lượng thuật toán.

Chú ý. Khi áp dụng phương pháp ACO cho các bài toán cụ thể, bayếu tố sau có ảnh hưởng quyết định đến hiệu quả thuật toán:

1) *Xây dựng đồ thị cấu trúc thích hợp*.Trong mục 2.2.1 đã chỉ ra rằng việc xây dựng đồ thị cấu trúc để tìm được lời giải cho bài toán theo thủ tục tuần tự không khó.Khó khăn chính là với các bài toán cỡ lớn, không gian tìm kiếm quá rộng, đòi hỏi ta sử dụng các ràng buộc Ωmột cách hợp lý để giảm miền tìm kiếm của kiến.

2) *Chọn thông tin heuristic*. Thông tin heuristic tốt sẽtăng hiệu quả thuật toán. Tuy nhiên, trong nhiều bài toán không có thông tin này thì có thể đánh giá chúng như nhau. Khi đó, ban đầu thuật toán chỉ đơn thuần chạy theo phương thức tìm kiếm ngẫu nhiên, vết mùi thể hiện định hướng của học tăng cường và thuật toán vẫn thực hiện được.

3) *Chọn quy tắc cập nhật mùi*. Quy tắc cập nhật mùi thể hiện chiến lược học của thuật toán.Trong khi đồ thị cấu trúc và thông tin heuristic phụ thuộc vào bài toán cụ thể, quy tắc cập nhật mùi lại là yếu tố phổ dụng và thường dùng để đặt tên cho thuật toán. Có nhiều quy tắc cập nhật mùi đã được đề xuất, trong luận văn này tác giả sẽ lựa chọn một thuật toán cập nhật mùi phù hợp với bài toán của mình để cho hiệu quả tốt.

2.3.Một số vấn đề liên quan

2.3.1.Đặc tính hội tụ

Gutjahr [33,34,35] là một trong những người đầu tiên nghiên cứu đặc tính hội tụ của thuật toán MMAS, nhưng chưa xét đến yếu tố có sử dụng thông tin heuristic. Ký hiệu $P(t)$ là xác suất tìm thấy lời giải của thuật toán MMAS trong vòng t phép lặp, $w(t)$ là lời giải tốt nhất ở bước lặp t. Nhờ sử dụng mô hình Markov không thuần nhất, Gutjahr đã chứng minh rằng với xác suất bằng 1 ta có:

1) $\lim_{t\to\infty} w(t) = w^*, \lim_{t\to\infty} P(t) = 1$ (2.3)

2) $\lim_{t\to\infty} \tau_{i,j} = \tau_{max}$ với mọi cạnh (i,j)thuộc lời giải tối ưu (2.4)

Mô hình này của Gutjahr không áp dụng được cho ACS.Trong trường hợp MMASkhông sử dụng thông tin heuristic, Stützle và Dorigo [32] đã chứng minh rằng:

$\forall \varepsilon > 0, \exists\ t$đủ lớn$P(t) > 1 - \ \varepsilon,$ (2.5)

do đó $\lim_{t\to\infty} P(t) = 1.$ (2.6)

Các tác giả cũng suy ra rằng kết quả này cũng đúng cho cả thuật toán ACS. Với giả thiết tìm được lời giải tối ưu sau hữu hạn bước, Stützle và Dorigo suy ra rằng vết mùi của các cạnh thuộc lời giải tối ưu tìm được sẽ hội tụ đến τ_{max} , còn vết mùi trên các cạnh không thuộc lời giải sẽ hội tụ về τ_{min} hoặc τ_0.

Plelegrini và Elloro [25] chỉ ra rằng sau một thời gian chạy, đa số vết mùi trên cạnh trở nên bé và chỉ có số ít cạnh có giá trị vết mùi là lớn vượt trội.

2.3.2.Thực hiện song song

Đặc tính tự nhiên của các thuật toán ACO cho phép thực hiện song song theo dữ liệu hoặc theo quần thể (xem [31]). Trên thực tế, có nhiều mô

hình song song được sử dụng cho các thuật toán dựa trên quần thể, dễ dàng tương thích với ACO.

2.3.3.ACO kết hợp với tìm kiếm cục bộ

Nhiều tài liệu [14,22] chỉ ra rằng với các phương pháp metaheuristic, một cách tiếp cận đầy hứa hẹn cho phép nhận được lời giải có chất lượng cao là kết hợp với thuật toán tìm kiếm cục bộ.

Mô hình ACO có thể bao gồm cả tìm kiếm cục bộ [2].Sau khi kiến xây dựng xong lời giải, có thể áp dụng tìm kiếm cục bộ để nhận được lời giải tối ưu địa phương.Việc cập nhật mùi được thực hiện trên các cạnh thuộc lời giải tối ưu địa phương này.Kết hợp xây dựng lời giải với tìm kiếm cục bộ sẽ là một cách tiếp cận có triển vọng, là do trên thực tế, cách xây dựng lời giải của ACO có sử dụng lân cận khác với tìm kiếm cục bộ. Thực nghiệm cho thấy khả năng kết hợp tìm kiếm cục bộ cải tiến được lời giải là khá cao.

2.3.4. Thông tin heuristic

Ta biết rằng thuật toán ACO mà không sử dụng tìm kiếm cục bộ, thông tin heuristic sẽ rất cần thiết để có được lời giải tốt. Trên thực tế, ở giai đoạn đầu vết mùi được khởi tạo như nhau.Khi đó vết mùi không thể giúp kiến tìm đường đi dẫn tới các lời giải tốt, vì chưa khác nhau nhiều.Vai trò chính của thông tin heuristic là để khắc phục điều này, giúp kiến có thể xây dựng được các hành trình tốt ngay trong giai đoạn đầu.Trong nhiều trường hợp, nhờ sử dụng tìm kiếm cục bộ, kiến vẫn có thể tìm được lời giải tốt ngay trong giai đoạn đầu, không cần sử dụng thông tin heuristic nào cả, mặc dù có làm cho quá trình tìm kiếm chậm hơn.

2.3.5.Số lượng kiến

Như đã nói ở trên, nếu không sử dụng tìm kiếm cục bộ và thông tin heuristic ít (hoặc không có), trong giai đoạn đầu vết mùi không thể giúp kiến tìm đường đi dẫn tới các lời giải tốt.Nếu sử dụng số lượng kiến ít, trong giai đoạn đầu sẽ không tìm được lời giải tốt và như vậy, việc cập nhật mùi được cập nhật dựa trên các lời giải không tốt. Khi đó, sẽ hướng việc tìm kiếm xung quanh lời giải không tốt và do đó thuật toán sẽ không hiệu quả. Có thể khắc phục phần nào nhược điểm này bằng cách tăng số kiến, để tăng khả năng tìm được lời giải tốt ở mỗi vòng lặp.Khi có sử dụng tìm kiếm cục bộ hoặc thông tin heuristic mạnh, sử dụng nhiều kiến là lãng phí.

2.3.6.Tham số bay hơi

Ở mỗi vòng lặp (các kiến cùng thực hiện tìm lời giải), khi xây dựng được lời giải tốt (sử dụng tìm kiếm cục bộ hoặc thông tin heuristic mạnh), tham số bay hơi sẽ được xác lập có giá trị lớn, điều này giúp kiến quên đi những lời giải đã xây dựng, tập trung công việc tìm kiếm xung quanh lời giải tốt mới được xây dựng. Trong trường hợp ngược lại, ở mỗi vòng lặp, khả năng kiến tìm được lời giải tốt không cao thì tham số bay hoiphải được thiết lập với giá trịnhỏ.

2.4. Tính biến thiên của vết mùi và các thuật toán cập nhật mùi

Như đã trình bày trong các phần trước, Gutjahr [33,34,35], Stützle và Dorigo [32] đã xét tính hội tụ theo xác suất tới lời giải tối ưu của MMAS, ACS và sự hội tụ của cường độ vết mùi cho các biến thể của thuật toán MMAS. Các tác giả chưa khảo sát sự hội tụ của cường độ vết mùi đối với ACS.

Tuy nhiên, trong các bài toán tối ưu tổ hợp do số phương án là hữu hạn, nên kết quả xác suất tìm thấy lời giải hội tụ về 1 khi số lần lặp dần ra vô hạn là không có nhiều ý nghĩa. Phần này trong [2] phân tích chi tiết hơn về các

đặc tính biến thiên của vết mùi trong các thuật toán ACO thông dụng và xem xét các quy tắc cập nhật mùi theo cách nhìn *học tăng cường*, trên cơ sở đó vận dụng vào việc cập nhật mùi cho bài toán dóng hàng hai đồ thị đã nêu trong chương I.

2.4.1.Thuật toán tổng quát

Xét một bài toán tối ưu tổ hợp cực tiểu hoá (S, f, Ω)trong mục 2.2 với đồ thị cấu trúc:$G = (V, E, H, \tau)$, trong đó V là tập đỉnh, E là tập các cạnh, H là vectơ các trọng số heuristic của cạnh tương ứng, còn τlà vectơ vết mùi tích luỹ được (ban đầu được khởi tạo bằng τ_0>0),C_0 là tập đỉnh khởi tạo để xây dựng các lời giải chấp nhận được theo thủ tục bước ngẫu nhiên.Thuật toán sử dụng m kiến, thực hiện N_c bước lặp xây dựng lời giải nhờ thủ tục bước ngẫu nhiên.

2.4.1.1.Quy tắc chuyển trạng thái

Giả sử kiến r đã xây dựng $x_k =< u_0, \dots, i >$ là mở rộng được, nó chọn đỉnh $y$thuộc $J(x_k)$ để mở rộng thành $x_{k+1} =< u_0, \dots, i, y >$ với xác suất được cho bởi công thức (2.7) sau đây:

$$P(y/\tau, x_k) = \begin{cases} \dfrac{\tau_{i,y}^{\alpha} h_{i,y}}{\sum_{j \in J(x_k)} \tau_{i,j}^{\alpha} h_{i,j}} & với\ y \in J(x_k) \\ 0 & với\ y \notin J(x_k) \end{cases} \qquad (2.7)$$

Quá trình mở rộng tiếp tục cho tới khi kiến r tìm được lời giải chấp nhận được với độ dài không quá h.

Chú ý. Quy tắc này khác một chút so với quy tắc chuyển trạng thái của thuật toán ACS và công thức (2.1), nhưng không ảnh hưởng tới các kết quả phân tích toán học về sau.

Ký hiệu $w(t)$ là lời giải tốt nhất các kiến tìm được cho tới lần lặp thứ t và $w^i(t)$ là lời giải tốt nhất trong bước lặpthứ t.Nếu $w^i(t)$ không tốt hơn

$w(t-1)$ ta có $w(t) = w(t-1)$. Ta sẽ quan tâm tới các lời giải gần đúng $w(t)$ này.

2.4.1.2. Cập nhật mùi

Luận văn xét hai quy tắc điển hình, được sử dụng phổ biến nhất hiện nay xuất phát từ ACS và MMAS. Giả sử g là một hàm thực, xác định trên S sao cho $\forall s \in S\ 0 < g(s) < \infty$ và $g(s) > g(s')$nếu $f(s) < f(s')$, khi đó ở mỗi bước lặp cường độ vết mùi sẽ thay đổi theo một trong các quy tắc sau đây:

Quy tắc ACS. Quy tắc này phỏng theo ACS, bao gồm cả cập nhật địa phương và toàn cục.

*Cập nhật mùi địa phương.*Nếu kiến k thăm cạnh (i,j),tức là $(i,j) \in s(k)$ thì cạnh này sẽ thay đổi mùi theo công thức sau:

$$\tau_{i,j} \leftarrow (1-\rho)\,\tau_{i,j} + \rho\tau_1 \tag{2.8}$$

*Cập nhật mùi toàn cục.*Cập nhật mùi toàn cục đối với các cạnh thuộc $w(t)$:

$$\tau_{i,j} \leftarrow (1-\rho)\,\tau_{i,j} + \rho g\big(w(t)\big) \tag{2.9}$$

Quy tắc MMAS. Quy tắc này giống như trongMMAS. Sau khi kiến xây dựng xong lời giải ở bước lặp nào đó, vết mùi được thay đổi theo công thức sau:

$$\tau_{i,j} \leftarrow (1-\rho)\,\tau_{i,j} + \Delta\tau_{i,j} \tag{2.10}$$

trong đó

$$\Delta\tau_{i,j} = \begin{cases} \rho g\big(w(t)\big) & (i,j) \in w(t) \\ \max\{\tau_1 - (1-\rho)\tau_{i,j}, 0\} & (i,j) \notin w(t) \end{cases} \tag{2.11}$$

ở đây τ_1>0 là tham số.

Chú ý:

1) Công thức (2.8) trở thành công thức (2.10) trong MMAS khi lấy$\tau_1 = \tau_{min}$ và giả thiết $\tau_{max} \geq g(w(t))$

2) Các quy tắc cập nhật mùi ở trên chính là quy tắc G-best.Nếu trong các công thức (2.9) và (2.11) thay $w(t)$bởi $w^i(t)$, thì ta nói là quy tắc I-best.Trong mục sau ta chỉ xét cho quy tắc G-best.

2.4.2. Đánh giá

Trong cácbài toán tối ưu tổ hợp, về mặt lý thuyết, ta có thể tìm lời giải tối ưu bằng cách vét cạn, nhưng thực tế điều này không khả thi, do không gian tìm kiếm quá rộng.Các quy tắc heuristic cho phép ta dựa trên"*các kinh nghiệm"* có đượcđể tìm nhanh các lời giải đủ tốt trong phạm vi tìm kiếm hẹp và chấp nhận loại bỏ những lời giải tốt hơn. Sự kết hợp học tăng cường thông qua thông tin về cường độ vết mùi cho phép ta từng bước thu hẹp miền tìm kiếm, mà vẫn không loại bỏ các lời giải tốt.Do đó, nâng cao chất lượng thuật toán.

Ta thấy chất lượng của thông tin heuristic tốt sẽ nâng cao hiệu quả thuật toán.Tuy nhiên, các quy tắc này không phải luôn có được và rất khó có thể can thiệp để thay đổi chất lượng. Do vậy, ta sẽ quan tâm tới cách cập nhật mùi để nâng cao chất lượng thuật toán. Dưới đây, sau khi nhận xét chung về đặc tính khai thác và khám phá của các thuật toán, Tiến sĩ Đỗ Đức Đông [2] đưa ra một số đề xuất.

2.4.2.1. Tính khai thác và khám phá

Tính khai thác là việc tập trung tìm kiếm lời giải xung quanh phạm vi của các cạnh thuộc các lời giải tốt nhất đã được biết cho tới thời điểm đang xét, còn tính khám phá là tìm kiếm ở các phạm vi khác. Trong cách cập nhật mùi G-best, ta đã biết $w(t)$ nên việc tìm kiếm quanh nó sẽ hạn chế nhiều tính khám phá, còn khi cập nhật theoI-best sẽ mở rộng miền này hơn. Vì vậy, trong thực nghiệm cập nhật theoI-best sẽ cho kết quả tốt hơn G-best.

Trong các bài toán tối ưu tổ hợp, xác suất một phương án cho trước được kiến tìm rảở mỗi lần lặp là rất bé. Vì vậy, có thể sau một số bước lặp, cường

độ vết mùi trên mỗi cạnh không thuộc $w(t)$sẽ bé,do đó làm giảm khả năng khám phá, mặc dù chúng có thể có triển vọng thuộc lời giải tốt.

Các điểm hạn chế của ACO

1) Mệnh đề trên cho thấy khi thuật toán mới bắt đầu, các vết mùi được khởi tạo như nhau, một cạnh (k,h) "tốt hơn" cạnh (i,j), do nó thuộc chu trình dài hơn có thể bị đảo ngược một cách rất ngẫu nhiên. Khi một cạnh do ngẫu nhiên không được cập nhật mùi, sau một số bước cường độ mùi của nó nhanh chóng bị giảm xuống và do vậy khó được kiến chọn ở bước sau đó, mặc dù "chất lượng" của nó chưa chắc đã là "xấu".

2) Nếu khởi tạo mùi như nhau và không dùng thông tin heuristic, xác suất của mỗi cạnh được kiến đã cho sử dụng trong lần lặp đầu sẽlà $\frac{2}{n-1}$.Xác suất này rất bé khi n lớn. Như vậy, tùy theo từng loại bài toán mà tỷ lệ giữa τ_0 và τ_1sẽ rất có ý nghĩa cho cân bằng giữa tính khám phá và tính khai thác của thuật toán.

3) Các lượng mùi cập nhật theo các công thức từ (2.8) đến (2.11) phụ thuộc vào giá trị hàm mục tiêu của lời giải được kiến xây dựng được trong các bước lặp. Việc xác định các giá trịτ_0 và τ_1 hayτ_{min} và τ_{max} cũng phụ thuộc vào tương quan với các giá trị chưa được xác định trước này của từng bài toán, khi đó thuật toán mới tốt được. Tuy nhiên, điều này rất khó thực hiện.

2.4.2.2. Các thuật toán cập nhật mùi theo quy tắc ACS

Như đã nói ở mục 3.2.2, cách cập nhật mùi toàn cục của ACS [19] thực hiện bay hơi đối với các cạnh không được kiến chọn. Vì vậy, không đảm bảo cường độ vết mùi thỏa mãn điều kiện thuộc khoảng $[\tau_{min}, \tau_{max}]$.Vết mùi của những cạnh, không được kiến sử dụng và không thuộc đường đi tốt, sẽ nhanh chóng dần về 0,làm cho các con kiến sau sẽ có bỏ qua các cạnh này. Tuy vậy, như đã chỉ ra ở trên, cạnh này vẫn có thể là cạnh tham gia vào lời giải "tốt",nhưng bị loại do rủi ro. Hiện tượng này làm giảm tính khám phá của ACS, vì thế hiệu quả của nó kém MMAS (xem [31])

Quy tắc cập nhật mùi toàn cục trong [19] (được giới thiệu ở chương trước) đã khắc phục được hạn chế này nên đến nay nó được thay cho cách cập nhật ở [5].Tuy vậy, việc cập nhật địa phương chưa cho thấy rõ ý nghĩa của học tăng cường.

2.4.2.3. Các thuật toán cập nhật mùi theo quy tắc MMAS

Theo quy tắc này, việc tìm kiếm chỉ tập trung quanh lời giải tốt nhất, còn các cạnh không thuộc lời giải này sẽ có cường độ vết mùi nhanh chóng tụt về τ_{min} Vì vậy, khi τ_{min} nhỏ hơn nhiều so với τ_{max} ,tính khám phá sẽ kém, còn nếu chọn τ_{min} gần với τ_{max} thì thuật toán chủ yếu là tìm kiếm ngẫu nhiên dựa theo thông tin heuristic.

Trong [2] đề xuất quy tắc cải tiến của ACS vàMMAS:

a) Phương pháp cập nhật mùi đa mức: MLAS (Multi-level Ant System)

Dựa vào nhận xét ở mục trước, thay cho việc bay hơi vết mùi ở các thành phần không thuộc các lời giải của mỗi con kiến trong mỗi lần cập nhật mùi ở mỗi bước lặp, ta cho τ_1 và τ_{max} tăng dần. Độ lệch giữa τ_1 và τ_{max} cho phép ta điều khiển tính hội tụ và khám phá.Nếu thấy lời giải tốt ít thay đổi thì cho τ_1 gần τ_{max} để tăng tính khám phá và ngược lại, cho τ_1 dịch xa τ_{max} để cho lời giải tập trung tìm kiếm quanh lời giải tốt nhất tìm được.

Quy tắc này đã thử nghiệm cho các bài toán TSP và JSS, cho kết quả khả quan so với MMAS.Tuy nhiên, việc điều khiển độ lệch giữa τ_1 và τ_{max} rất khó áp dụng cho các bài toán cụ thể, nên chúng tôi thay bởi phương pháp 3-LAS sẽ trình bày ở phần c) dưới đây.

b) Phương pháp Max-Min trơn: SMMAS (Smoothed Max Min Ant System)

Dựa vào nhận xét ở mục trên, ta thấy không nên giảm vết mùi ở các cạnh không thuộc lời giải tốt quá nhanh như trong quy tắc MMAS, mà nên dùng quy tắc Max-Min trơn như sau:

$$\tau_{i,j} \leftarrow (1-\rho)\tau_{i,j} + \Delta\tau_{i,j} \qquad (2.12)$$

với

$$\Delta\tau_{i,j} = \begin{cases} \rho\tau_{min} \ nếu \ (i,j) \notin w(t) \\ \rho\tau_{max} \ nếu \ (i,j) \in w(t) \end{cases}$$

Khi cài đặt, lấy $\tau_0 = \tau_{max}$.

c) Phương pháp 3-LAS (Three-level Ant System)

Đối với các bài toán sử dụng thông tin heuristic, ảnh hưởng nhiều tới chất lượng tìm kiếm lời giải, chẳng hạn như bài toán TSP, phương pháp 3-LAS tương tự ACS, nhưng dễ dùng hơn và hiệu quả tốt hơn. Phương pháp này sử dụng thêm tham số τ_{mid} thuộc khoảng (τ_{min} , τ_{max}) và cập nhật mùi tương tự SMMAS cho các cạnh có kiến sử dụng hoặc thuộc $w(t)$.Cụ thể là:

$$\tau_{i,j} \leftarrow (1-\rho)\tau_{i,j} + \Delta\tau_{i,j} \qquad (2.13)$$

với

$$\Delta\tau_{i,j} = \begin{cases} \rho\tau_{max} & nếu \ (i,j) \in w(t) \\ \rho\tau_{mid} & nếu \ \forall (i,j) \Subset w(t) \text{ và có kiến đi qua} \\ \rho\tau_{min} & \text{cho các cạnh còn lại} \end{cases}$$

2.4.2.4. Ưu điểm khi sử dụng SMMAS và 3-LAS

Ta thấy thuật toán SMMAS và 3-LAS có một số ưu điểm sau so với ACS và MMAS.

1) Với ACS và MMAS, để xác định τ_0 hay τ_{min} và τ_{max} người ta cần tìm một lời giải theo phương pháp heuristic và dựa vào giá trị hàm mục tiêu của nó. Vì giá trị hàm mục tiêu này nhận được ngẫu nhiên, nên khó xác định tốt tham số cho học tăng cường. Quy tắc cập nhật mới cho phép ta xác định các tham số này đơn giản và hợp lý hơn, cụ thể: trong SMMAS và 3-LAS ta không cần xác định chính xác giá trị τ_{min} , τ_{max} mà chỉ cần xác định tỉ lệ giữa τ_{min} , τ_{max} . Trong thực nghiệm,luận án luôn thiết đặt $\tau_{max} = 1.0$ và xác định τ_{min} qua tỉ lệ giữa τ_{min} , τ_{max} . Cần nhấn mạnh rằng, việc chỉ cần lựa chọn tỉ

lệ giữa τ_{min} , τ_{max} đơn giản và mất ít thời gian thực nghiệm hơn rất nhiều so với việc lựa chọn cụ thể hai tham số τ_{min} , τ_{max} .

2)Việc thêm mùi cho các cạnh thuộc lời giải tốt ở mỗi bước lặp trong thuật toán ACS và MMAS, ta phải xây dựng hàmđể tính lượng mùi được thêm dựa trên chất lượng lời giải do kiến xây dựng được. Ví dụ, trong bài toán TSP, ACS và MMAS sử dụng hàm nghịch đảo độ dài đường đi được kiến xác định.Điều này cũng là một trong những khó khăn khi áp dụng ACS (hoặc MMAS) đối với một bài toán mới.Tuy nhiên, trong SMMAS và 3-LAS không cần phải xây dựng hàm này.

3) Dễ dàng kiểm tra được các thuật toán này có cùng độ phức tạp như MMASvà ACS, nhưng ít phép toán hơn MMAS vì không phải tính hàm mục tiêu ở lượng mùi cập nhật và không phải so sánh để giới hạn vết mùi trong khoảngτ_{min} , τ_{max} . Theo cách cập nhật của SMMAS và 3-LAS, vết mùi luôn trong khoảngτ_{min} , τ_{max} .

2.4.3. Tính bất biến

Một số tác giả đã xét tính bất biến của các quy tắc cập nhật mùi khi hàm mục tiêu được biến đổi tuyến tính đơn điệu [10,37]. Để khảo sát tính bất biến, ta cần khái niệm về các thể hiện của bài toán tối ưu tổ hợp và giả thiết về tính lặp của máy tạo số giả ngẫu nhiên.

Định nghĩa 2.2. (Thể hiện của bài toán)

Xét hai bài toán tối ưu tổ hợp(S, f, Ω)và (S, f', Ω).Ta sẽ gọi chúng là hai thể hiện I và I' tương ứng của một bài toán, nếu $f'(s) = g(f(s))$ với mọi s thuộc S trong đó g là hàm đơn điệu tăng chặt.

Thông thường ta sẽ xét g thuộc một nhóm các hàm G nào đó.

Định nghĩa 2.3. (Giả thiết về tính lặp của máy tạo số giả ngẫu nhiên)

Khi chạy một thuật toán tìm kiếm ngẫu nhiên cho hai thể hiện của một bài toán, các quyết định dựa trên các thí nghiệm ngẫu nhiên nhờ một máy tạo số giả ngẫu nhiên. Ta giả thiết các số này được tạo ra cùng một phương pháp (chẳng hạn cùng một giống:seed), như vậy dãy số ngẫu nhiên tạo ra khi giải hai thể hiện là như nhau và ta gọi nó là máy phát lặp.

Bây giờ ta định nghĩa tính bất biến của thuật toán.

Định nghĩa 2.4. (Tính bất biến của thuật toán)

Ta nói thuật toán A bất biến trên nhóm biến đổi đơn điệu G đối với bài toán (S, f, Ω) nếu khi sử dụng nó nhờ một máy phát lặp để giải hai thể hiện của bài toán vẫn cho ta cùng một dãy lời giải và vết mùi.

Birattari [10] và Zang [37] đã xét tính bất biến của một số thuật toán nhờ biến đổi tuyến tính đơn điệu tăng: $G = \{g(s) = af(s) + b\}$, trong đó $a > 0$.

Ta dễ dàng kiểm tra được các thuật toán SMMAS và 3-LAS bất biến đối với nhóm biến đổi đơn điệu tăng của bài toán (S, f, Ω).Định lý sau là đúng.

Định lý 2.5.Giả sử I và I'là hai thể hiện của một bài toán tối ưu tổ hợp tùy ý.Khi giải bằng một trong hai thuật toán SMMAS hoặc 3-LAS với cùng số lần lặp nhờ dùng một máy phát lặp sẽ cho kết quả cùng một dãy lời giải và các vectơ vết mùi.

Kết luận của định lý là hiển nhiên vì quyết định và lượng mùi thay đổi trong mỗi lần lặp sẽ như nhau ở mỗi lần lặp.

2.4.4. Kết luận chương

Tối ưu đàn kiến là phương pháp metaheuristic được áp dụng phổ biến cho các bài toán tối ưu tổ hợp khó, thực nghiệm đã chứng minh điều đó mang lại hiệu quả nổi trội. Xuất phát từ tư tưởng tìm đường đi của kiến tự nhiên, người ta đã mô phỏng các con kiến nhân tạo tìm đường đi trên đồ thị cấu trúc

theo thông tin heuristic và thông tin học tăng cường là vết mùi để lại trên đường đi.

Khi áp dụng thuật toán này có ba yếu tố quan trọng quyết định chất lượng thuật toán:

1) Xây dựng đồ thị cấu trúc
2) Xác định thông tin heuristic
3) Chọn quy tác cập nhật mùi

Hai yếu tố đầu phụ thuộc vào bài toán cụ thể, còn yếu tố thứ ba có nhiều đề xuất cải tiến. Trong luận văn này có trình bày ba thuật toán cải tiến quy tắc cập nhật mùi của Tiến sĩ Đỗ Đức Đông [2], với ưu thế đã trình bày tác giả sẽ chọn quy tắc cập nhật mùi SMMAS để áp dụng trong thực nghiệm.

CHƯƠNG III
PHƯƠNG PHÁP TỐI ƯU ĐÀN KIẾN GIẢI BÀI TOÁN DÓNG HÀNG HAI ĐỒ THỊ

Bài toán dóng hàng hai đồ thị như đã trình bày trong chương I, là bài toán tối ưu tổ hợp được chứng minh thuộc lớp NP-khó [4]. Nhiều thuật toán được nghiên cứu và công bố cho thấy có hiệu quả từ năm 2008 đến nay (xem [1,29]), tuy nhiên,hầu hết đều dựa trên sự kết hợp giữa kỹ thuật tham với thông tin heuristic. Trong chương này tác giả đề xuất một giải pháp mới metaheuristic, phương pháp tối ưu đàn kiến cho bài toán dóng hàng hai đồ thị. Sau này để tiện cho việc thực nghiệm và so sánh kết quả tác giả gọi tên phương pháp này là ACOPPI (Ant Colony Optimiration to Alignment Protein-Protein Interraction Network).

3.1. Thuật toán tối ưu đàn kiến giải bài toán dóng hàng hai đồ thị

Nhắc lại bài toán đã đề cập ở chương I và các kiến thức liên quan trong chương II.

*Bài toán:*Giả sử $G_1 = (V_1, E_1)$ và $G_2 = (V_2, E_2)$ là hai mạng tương tác protein, trong đó V_1, V_2 ký hiệu tập các nút mô tả các protein trong mạng G_1, G_2 tương ứng;E_1, E_2 ký hiệu tập các cạnh mô tả mối quan hệ tương tác giữa các protein trong các mạng G_1, G_2. Không giảm tổng quát, ta xem $|V_1| \leq |V_2|$ trong đó $|V|$ ký hiệu số phần tử của tập V.

Dóng hàng mạng: Đồ thị $A_{12} = (V_{12}, E_{12})$ là một mạng dóng hàng của hai đồ thị G_1, G_2 nếu nó thỏa mãn:

i) Mỗi nút của V_{12} được ký hiệu là $< u_i, v_j >$ tương ứng với một cặp đỉnh u_i thuộc V_1 và v_j thuộc V_2.

ii) Hai nút phân biệt$< u_i, v_j >$ và $< u_i^{'}, v_j^{'} >$ thuộc V_{12}thì$u_i \neq u_i^{'}$ và $v_j \neq v_j^{'}$

iii) Cạnh($< u_i, v_j >, < u_i^{'}, v_j^{'} >$) thuộc E_{12} nếu và chỉ nếu $(u_i, u_i^{'}) \in E_1$ và $(v_i, v_i^{'}) \in E_2$.

Một *dóng hàng mạng*$A_{12} = (V_{12}, E_{12})$ là lời giải của bài toán *dóng hàng toàn cục* của các mạng proteins G_1, G_2 nếu nó cực đại *global network alignment score* cho bởi (1.1):

$$GNAS(A_{12}) = \alpha|E_{12}| + (1-\alpha)\sum_{\forall <u_i, v_j>} similar(u_i, v_j)$$

trong đó $\alpha \in [0,1]$ là tham số cân bằng giữa sự tương đồng về tô pô mạng và sự tương đồng trình tự giữa các nút, giá trị $Similar(u_i, v_j)$được tính xấp xỉ dựa trên BLAST bit-scores hoặc E-values.

Chú ý: như đã nói ở phần mở đầu, Các mạng tương tác protein được mô tả bằng đồ thị, bài toán dóng hàng mạng được chuyển tải về bài toán dóng hàng đồ thị.

** Dữ liệu của bài toán*

Input: 1. hai mạng PPI (đơn đồ thị)$G_1(V_1, E_1)$ và $G_2(V_2, E_2)$

2. tham số cân bằng $\alpha \in [0,1]$

3. similar(u_i,v_j)

Output: Số điểm *GNAS*max tính theo (1.1)

$$GNAS(A_{12}) = \alpha|E_{12}| + (1-\alpha)\sum_{\forall <u_i, v_j>} similar(u_i, v_j)$$

Để giải một bài toán bằng phương pháp tối ưu đàn kiến ta nhắc lại các yếu tố quyết định đến hiệu quả thuật toán được nêu trong [2]

1) Xây dựng đồ thị cấu trúc thích hợp;

2) Chọn thông tin heuristic;

3) Chọn quy tắc cập nhật mùi.

Nhận xét: đối với mỗi bài toán cụ thể, việc áp dụng phương pháp cho lời giải tốt người ta phải tìm cách giải quyết từng khâu cho phù hợp. Tác giả xin trình bày một vài nhận xét sau:

3.1.1. Xây dựng đồ thị cấu trúc thích hợp

Đối với mỗi bài toán cần có ý tưởng tổng quan của thuật toán, từ đó xác định cho kiến hoạt động ra sao để xây dựng cấu trúc của kiến và đồ thị cấu trúc. Theo tư tưởng đã trình bày ở chương II, việc kết hợp ACO với tìm kiếm cục bộ sẽ cho kết quả tốt, điều đó cũng được kiểm chứng qua thực nghiệm[2]. Vì vậy tư tưởng chính của tác giả là cải tiến thuật toán FastNA bằng cách áp dụng thuật toán ACO xây dựng lời giải ban đầu (trong thuật toán FastNA là xây dựng lời giải ban đầu bằng kỹ thuật tham theo thông tin heuristic). Sau đó giữ nguyên pha thứ hai của FastNA, dỡ bỏ kết quả thu được trong pha một chỉ giữ lại phần khung, từ phần khung đó ta sẽ phát triển lời giải nhờ tìm kiếm cục bộ để có được lời giải tốt hơn, việc tìm lời giải kết thúc theo điều kiện thời gian quy định. Lý giải cho việc lựa chọn trên của tác giả, với FastNA việc tìm kiếm dựa trên thông tin heuristic chỉ tốt khi lời giải đã được xây dựng một phần, tuy đã được khắc phục ở pha thứ hai nhưng kết quả ban đầu vẫn ảnh hưởng đến kết quả cuối cùng của thuật toán. Với tư tưởng thuật toán tác giả đề xuất, các kiến sẽ xây dựng lời giải ban đầu bằng cách kết hợp thông tin heuristic với thông tin mùi (với tư tưởng này ta không cần xây dựng đồ thị cấu trúc cho bài toán mà chỉ cần một mảng hai chiều để lưu thông tin mùi khi các kiến dóng hàng mỗi đỉnh $i \in V_1$ với đỉnh $j \in V_2$), các kiến sau sẽ tìm lời giải dựa trên cơ chế học tăng cường ở vết mùi kiến trước để lại. Đó là cơ sở để xây dựng lời giải ban đầu trong pha một tốt hơn so với FastNA, từ đó kết quả tìm kiếm cục bộ ở pha thứ hai cũng sẽ cho kết quả tốt hơn;

3.1.2. Chọn thông tin heuristic;

Đây là bài toán có nhiều cách chọn thông tin heuristic (xem[1,4]). Tác giả chọn thông tin heuristic để dóng hàng hai nút <*i*,*j*> khi chúng có nhiều mối liên kết với thành phần đã ghép được nhất. Nút $i \in V_1$ để dóng hàng tiếp

phụ thuộc vào thông tin số cạnh được ghép nhiều nhất với thành phần đã ghép được trong V_1 có ý nghĩa gì? Có thể đặt câu hỏi:vì sao không chọn thông tin nút $i \in V_1$ có bậc lớn nhất còn lại (các nút chưa được dóng hàng) trong V_1 để dóng hàng tiếp? Ta có thể lý giải xuất phát từ công thức (1.1) để GNAS đạt điểm số lớn nhấtthì không những nút được ghép có điểm số similar lớn mà số cạnh được ghép E_{12} cũng phải lớn. Vì vậy nếu chọn thông tin heuristic theo i có bậc lớn nhất thì chưa chắc số cạnh ghép với thành phần đã ghép được trong V_1 là tốt nhất từ đó việc chọn $j \in V_2$ để dóng sẽ không hiệu quả (mang tính may rủi cao);

3.1.3. Cập nhật mùi

Theo [2] cho thấy quy tắc cập nhật mùi SMMAS (2.11) là một thuật toán hiệu quảnên tác giả sẽ nghiên cứu để áp dụng vào giải thuật trình bày trong luận văn.

Thuật toán tác giả đề xuất gồm hai pha chính, pha thứ nhất tìm lời giải (dóng hàng) ban đầu, pha thứ hai dỡ bỏ dóng hàng ban đầu chỉ giữ lại thành phần tốt nhất (những nút có điểm số cao nhất tính theo 1.5) làm cơ sở cho việc dóng hàng lại để tìm lời giải tốt hơn.

Thuật toán

* *Pha thứ nhất*: xây dựng dóng hàng ban đầu

Khởi tạo lời giải ban đầu $A_{12} = \Phi$, cho tất cả kiến chạy đồng thời tìm lời giải, chọn lời giải của kiến có kết quả tốt nhất để làm dóng hàng ban đầu.

Input: Hai đồ thị G_1, G_2;

Tham sốα

Độ tương tự của các cặp đỉnh <i,j> tương ứng của V_1, V_2. Với mỗi tập con các cặp đỉnh V_{12} của tập $V_1 \times V_2$, ký hiệu $V_{12}^1 = \{i \in V_1 : < i, j > \in V_{12}\}$, $V_{12}^2 = \{j \in V_2 : < i, j > \in V_{12}\}$

<u>*Output*</u>: Một dóng hàng toàn cục ban đầu A_{12}

Bước 1. Khởi tạo: $V_{12} = \emptyset$

$$\tau_{max} = 1; \tau_{\min} = \tau_{max} / |V_2|$$

ma trận mùi $\tau_{[i,j]} = \tau_{max}$ (i=1..$|V_1|$, j=1..$|V_2|$)

Bước 2. Lặp với k= *1* tới $|V_1|$ //mỗi kiến xây dựng lời giải

2.1. Kiến chọnnút*i* trong $V_1 - V_{12}^1$có nhiều cạnh nối với các đỉnh của V_{12}^1

2.2. Nút $j \in V_2$ được lựa chọn ngẫu nhiên với xác suất lựa chọn nút $j \in V_2$ với nút $i \in V_1$ là:

$$P_{i,j} = \begin{cases} \frac{[\tau_{i,j}]^{\alpha}[\eta_{i,j}]^{\beta}}{\sum_{l \in R}[\tau_{i,l}]^{\alpha}[\eta_{i,l}]^{\beta}}, nếu\ j \in R \\ 0 \qquad\qquad ngược\ lại \end{cases} \quad (3.1)$$

Trong đó $\eta_{i,j} = GNAS(V_{12} + (i,j))$ là giá trị thông tin heuristic, α, β là hai tham số quyết định đến sự ảnh hưởng tương quan giữa thông tin mùi và thông tin heuristic, R là tập các đỉnh thuộc V_2 mà chưa được ghép .

2.3. Bổ sung $< i,j >$ vào V_{12};

2.4. Update E_{12} dựa trên V_{12}; //cập nhật các cạnh của E_{12} sau khi bổ sung cặp nút <i,j> vào V_{12}

Bước 3. Chọn lời giải của kiến có kết quả tốt nhất tính theo (1.1);

Bước 4. Cập nhật mùi $\tau_{[i,j]}$

Pha thứ nhất được đặc tả bằng thủ tục ACOPPI như sau:

Algorithm 1 Procedure ACOPPI
Input: Graph 1: $G_1 = (V_1, E_1)$; Graph 2: $G_2 = (V_2, E_2)$;
Similarities of node pairs: $Similar[i][j]$;
Balancing parameterα
Output: Alignment network $A_{12} = (V_{12}, E_{12})$
Begin
$V_{12} = \emptyset$
$\tau_{max} = 1,\ \tau_{\min} = \tau_{max} / |V_2|$
$\tau_{[i,j]} = \tau_{max}$ (i=1..|V_1|, j=1..|V_2|)
for k=1 **to** |V_1| **do**
i = next_node_align(G_1); //kiến chọn nút i có nhiều cạnh ghép với thành V_{12}
j = best_node_align(i, G_1, G_2); // kiến tìm nút j để dóng hàng với i theo (3.1)
UpdateV(V_{12}) // $V_{12} = V_{12} \cup < i, j >$
UpdateE(E_{12}) // cập nhật các cạnh của A_{12}
end-for
UpdateG(GNAS); //chọn kết quả tốt nhất
UpdateT($\tau_{[i,j]}$); // cập nhật mùi
End

Hình 3.1: Thuật toán ACOPPI tạo dóng hàng ban đầu

* *Pha thứ hai*: Local seach

Dỡ bỏ dóng hàng ban đầu chỉ giữ lại những nút (xương sống) có điểm số tốt nhất tính theo (1.5) và thực hiện dóng hàng lại bằng thủ tục Rebuild.

Thủ tục Rebuild.

Với A_{12}đã được xây dựng trong phase1 và số n_{keep} đã cho để xác định số lượng nút trong tập SeedV_{12} , thủ tục này được đặc tả trong hình 3.2 và thực hiện như sau:

Bước 1. Xác định tập $SeedV_{12}$ của V_1 gồm n_{keep} đỉnh có score tốt nhất của V_1 theo tiêu chí cho bởi (1.5):

Bước 2. Xác định V_{12} khởi tạo nhờ $SeedV_{12_1}$ và G_{12}

Bước 3. Thực hiện lặp với$k = n_{keep} + 1$ tới $|V_1|$để xác định A_{12}

3.1. Tìm node i trong $V_1 - V_{12}^1$có số cạnh tới các đỉnh trong V_{12}^1 lớn nhất;

3.2. Tìm node j trong $V_2 - V_{12}^2$ mà khi bổ sung $< i, j >$ vào V_{12} thì *GNAS(A_{12})* tính bởi Eq(1) lớn nhất, trong đó A_{12} là độ thị có đỉnh là tập V_{12} và các cạnh cảm sinh bởi G_1, G_2. Khi đó j đượi gọi là best_matched_node($i, V_{1,2}$);

3.3. Bổ sung $< i, j >$ vào V_{12};

3.4. Update E_{12} dựa trên V_{12};

Pha thứ hai được đặc tả bằng thuật toán Rebuild dưới đây

Algorithm 2 Procedure of Rebuild

Input: Alignment networkA_{12};n_{keep}

Output: Better Alignment network$A_{12} = (V_{12}, E_{12})$

Begin

Build $SeedV_{12}$;

Build V_{12}// Tính điểm của các đỉnh thuộc V_1 theo (1.5) sau đó sắp xếp giảm dần và giữ lại n_{keep} nút của V_1 theo thứ tự điểm giảm dần và n_{keep} nút của V_2 đã dóng hàng tương ứng với các nút giữ lại trong V_1

for$k$=n_{keep} +1 **to**$|V_1|$**do**

i = find_next_node(G_1);

j = choose_best_matched_node(i, G_1, G_2);

$V_{12} = V_{12} \cup < i, j >$

Update(E_{12}) //Cập nhật các cạnh khớp khi bổ sung thêm <i,j>

end-for

end

Hình 3.2: Thuật toán Rebuild xây dựng lại lời giải

Nhận xét: Khi cho hai pha lặp với số lần xác định, sau mỗi lần lặp các kiến tìm lời giải ta sẽ chọn được một lời giải tốt nhất, rồi cập nhật mùi để lại trên đường đi. Những lần lặp sau các kiến đã có thông tin mùi giúp cho việc học tăng cường, từ đó chọn được cách dóng hàng tốt hơn.

Đánh giá độ phức tạp thời gian của thuật toán tác giả đề xuất, ta thấy ở giai đoạn một k kiến đồng thời tìm điểm $j \in V_2$ để dóng hàng với mỗi điểm $i \in V_1$ vậy độ phức tạp là

$$O(k \times |V_1| \times |V_2|) \quad (3.1)$$

Thực hiện n lần lặp, phục vụ cho việc cập nhật vết mùi và học tăng cường khi kiến tìm đường đi. Vậy độ phức tạp về thời gian của toàn bộ pha thứ nhất là

$$O(n \times k \times |V_1| \times |V_2|) \quad (3.2)$$

Ở giai đoạn hai độ phức tạp $O(|V1| \times (|E_1| + |E_2|)$ (3.3)

Vậy cả hai pha sẽ có độ phức tạp là:

$$\boxed{O(n \times k \times |V_1| \times (|E_1| + |E_2|))} \quad (3.4)$$

3.2. Thực nghiệm, so sánh kết quả với phướng pháp SPINAL và FastNA

Tác giả tiến hành thực nghiệm chương trình với các bộ dữ liệu mà SPINAL và FastNA dùng thực nghiệm, từ đó làm cơ sở để so sánh hiệu quả với hai phương pháp này.

3.2.1. Thực nghiệm

Bảng 3.1:Thông tin về dữ liệu

dataset	No. of protein	No. of interaction
ce	2805	4495
dm	7518	25635
sc	5499	31261
hs	9633	34327

Với 4 bộ dữ liệu mạng PPI: Saccharomyces cerevisiae (sc), Drosophila melanogaster (dm), Caenorhabditis elegans (ce), and Homo sapiens (hs). Các dữ liệu này lấy từ [24] với cỡ mạng (số protein-số nút và tương tác-số cạnh) được cho trong Bảng 3.1. Như vậy có 6 cặp mạng khác nhau (*ce-dm* , *ce-hs, ce-sc, dm-hs, dm-sc, hs-sc*) để tiến hành dóng hàng. Tham số α nhận năm giá trị lần lượt bằng *0.3, 0.4, 0.5, 0.6, 0.7* như trong [4]. Với mỗi cặp mạng PPI và một tham số αtác giả thực hiện chương trình 20 lần (kiểm tra tính ổn định của

thuật toán), mỗi lần sử dụng 10 kiến (đồng thời tìm đường đi) và số vòng lặp là 100 (thể hiện cách học tăng cường của kiến, sau mỗi vòng lặp sẽ cập nhật mùi trên đường đi của kiến, lần lặp sau các kiến sẽ có thông tin mùi để lại trên đường đi).

Thực nghiệm cho kết quả là dữ liệu trong Bảng 3.2 và Bảng 3.3, với mỗi bộ dữ liệu (một hàng) tác giả lấy giá trị tốt nhất, tồi nhất, trung bình theo hai tiêu chí GNAS (số điểm dóng hàng theo 1.5) và EC (số cạnh khớp), với tiêu chí GNAS tác giả tính thêm độ lệch chuẩn để kiểm tra tính ổn định của thuật toán. Các số liệu cho thấy độ lệch chuẩn bình quân của tất cả các lần thực nghiệm là 36.6 tương ứng với số điểm dóng hàng bình quân 2629.47 chiếm tỉ lệ 1.39%. Trong đó độ lệch tồi nhất là 3.33% (của bộ dữ liệu ce-dm, α=0.3, có độ lệch chuẩn là 28.2 và số điểm dóng hàng bình quân là 793.27), thuật toán thể hiện tính ổn định nhất với độ lệch chuẩn là 0.74% (bộ dữ liệu dm-hs, α=0.6 có độ lệch chuẩn là 34.3 và số điểm dóng hàng bình quân 4614.06). Nếu xem xét kỹ ta thấy chỉ có 02 bộ dữ liệu có độ lệch chuẩn trên 3% (ce-dm, α=0.3; ce-sc, α=0.5), còn lại đều dưới 3%. Điều đó cho thấy tuy là một thuật toán mang tính tự nhiên nhưng có học tăng cường sau mỗi lần thực hiện đã cho kết quả có tính ổn định, ít biến động. Theo tiêu chí EC kết quả thể hiện ở bảng III, kết quả này cho thấy số cạnh khớp được EC tỉ lệ thuận với kết quả GNAS.

Bảng 3.2: Kết quả thực nghiệm của ACOPPI theo tiêu chí GNAS

Dữ liệu	*α*	*GNAS*			*Độ lệch chuẩn*	*Tỉ lệ %*
		Tốt nhất	*Tồi nhất*	*Trung bình*		
ce-dm	**0.3**	**871.90**	**832.52**	**845.87**	**28.2**	**3.33%**
	0.4	1132.28	1098.48	1117.88	17.1	1.53%
	0.5	1423.58	1370.75	1398.01	30.9	2.21%
	0.6	1695.37	1640.24	1675.85	25.5	1.52%
	0.7	1988.85	1882.64	1948.93	49.8	2.55%
ce-sc	0.3	907.83	879.68	897.11	14.0	1.56%
	0.4	1204.98	1151.3	1182.04	28.1	2.37%
	0.5	**1533.92**	**1464.01**	**1490.97**	**47.1**	**3.16%**
	0.6	1814.15	1777.82	1794.66	22.0	1.23%
	0.7	2125.83	2038.63	2091.70	43.1	2.06%
ce-hs	0.3	947.17	921.03	933.16	16.2	1.74%
	0.4	1253.58	1216.44	1233.45	23.4	1.89%
	0.5	1584.46	1526.57	1552.57	37.1	2.39%
	0.6	1881.86	1827.45	1856.31	29.8	1.60%
	0.7	2209.16	2093.65	2161.71	56.4	2.61%
dm-hs	0.3	2338.23	2279.92	2311.71	31.4	1.36%
	0.4	3100.49	3065.58	3078.24	24.2	0.79%
	0.5	3903.09	3807.47	3843.51	67.1	1.75%
	0.6	4643.84	4592.64	4614.06	34.3	0.74%
	0.7	5418.28	5237.49	5355.05	83.6	1.56%
dm-sc	0.3	2064.87	2016.55	2045.85	22.6	1.10%
	0.4	2745.92	2709.92	2728.14	21.5	0.79%
	0.5	3440.56	3387.25	3408.96	35.7	1.05%
	0.6	4146.21	4072.89	4107.10	46.6	1.13%
	0.7	4823.11	4734.98	4782.83	45.8	0.96%
hs-sc	0.3	2470.01	2430.87	2448.04	25.8	1.05%
	0.4	3292.55	3224.92	3262.86	37.5	1.15%
	0.5	4121.73	4047.1	4090.46	39.5	0.97%
	0.6	4948.06	4823.59	4893.83	64.3	1.31%
	0.7	5770.44	5678.02	5733.31	50.1	0.87%
Độ lệch chuẩn bình quân				2629.47	36.6	1.39%

Bảng 3.3: Kết quả thực nghiệm của ACOPPI theo tiêu chí EC

Dữ liệu	*α*	*EC*		
		Tốt nhất	*Tồi nhất*	*Trung bình*
ce-dm	0.3	2880	2749	2792.6
	0.4	2814	2728	2777.7
	0.5	2839	2732	2786.4
	0.6	2819	2727	2786.5
	0.7	2838	2685	2780.0
ce-sc	0.3	3009	2914	2973.4
	0.4	3003	2866	2944.6
	0.5	3061	2920	2974.6
	0.6	3020	2958	2986.5
	0.7	3034	2909	2985.3
ce-hs	0.3	3132	3042	3084.5
	0.4	3120	3024	3067.1
	0.5	3160	3041	3094.6
	0.6	3129	3038	3086.6
	0.7	3152	2985	3083.5
dm-hs	0.3	7740	7548	7652.0
	0.4	7716	7631	7661.2
	0.5	7783	7592	7664.4
	0.6	7725	7640	7675.2
	0.7	7731	7472	7640.3
dm-sc	0.3	6862	6700	6797.6
	0.4	6850	6760	6806.1
	0.5	6871	6765	6808.3
	0.6	6905	6782	6838.7
	0.7	6886	6760	6828.6
hs-sc	0.3	8205	8075	8132.5
	0.4	8215	8045	8141.5
	0.5	8232	8085	8170.5
	0.6	8239	8032	8149.6
	0.7	8239	8108	8186.2
Trung bình				5245.2

3.2.2. So sánh

Từ dữ liệu trong bảng 3.2 và bảng 3.3, tác giả tiến hành lấy kết quả trung bình so sánh với kết quả của hai thuật toán SPINAL và FastNA, thể hiện trong bảng 3.4 và bảng 3.5.

Với mỗi bộ dữ liệu là một cặp mạng PPI và một giá trị tham số α tác giả so sánh kết quả của thuật toán ACOPPI với SPINAL và FastNA theo hai tiêu chí GNAS và EC. Bảng dữ liệu 3.4 và 3.5 cho thấy, toàn bộ kết quả của ACOPPI đều vượt trội so với SPINAL và hơn đáng kể so với FastNA. Đặc biệt, kết quả tồi nhất trong 20 lần chạy của ACOPPI cũng đều tốt hơn FastNA và SPINAL.Theo tiêu chí GNAS thì kết tốt nhất của ACOPPI so với SPINAL hơn 1667.09 điểm (chiếm 41% kết quả của SPINAL đưa ra), so với FastNA hơn 453.43 điểm (8.6%); kết quả thấp nhất của ACOPPI so với SPINAL vẫn hơn 127.88 (18%) điểm, so với FastNA hơn 51.4 (2.3%) điểm. Tính theo tiêu chí EC thì kết quả tốt nhất của ACOPPI so với SPINAL là 2548.5 (46%) cạnh khớp, kết quả thấp nhất hơn 449.6 (19%) cạnh khớp, so với FastNA tốt nhất hơn 648.1 (8.6%) cạnh khớp, thấp nhất hơn 161.5 (2.1%) cạnh khớp.

Bảng 3.4: So sánh kết quả thực nghiệm của ACOPPI với SPINAL và FastNA theo tiêu chí GNAS

Dữ liệu	Thuật toán	α=0.3	α=0.4	α=0.5	α=0.6	α=0.7
ce-dm	SPINAL	717.99	941.19	1159.93	1350.59	1586.87
	FastNA	778.46	1034.20	1290.11	1545.86	1801.24
	ACOPPI	**845.87**	**1117.88**	**1398.01**	**1675.85**	**1948.93**
ce-hs	SPINAL	728.26	993.07	1229.95	1501.61	1764.93
	FastNA	863.46	1144.17	1429.89	1708.81	1994.87
	ACOPPI	**933.16**	**1233.45**	**1552.57**	**1856.31**	**2161.71**
ce-sc	SPINAL	709.12	963.28	1168.95	1422.74	1683.13
	FastNA	834.79	1109.93	1389.21	1663.39	1936.83
	ACOPPI	**897.11**	**1182.04**	**1490.97**	**1794.66**	**2091.70**
dm-hs	SPINAL	1883.22	2517.23	3160.48	3790.79	4451.60
	FastNA	2260.31	3007.11	3755.36	4496.45	5242.32
	ACOPPI	**2311.71**	**3078.24**	**3843.51**	**4614.06**	**5355.05**
dm-sc	SPINAL	1579.06	2075.14	2668.65	3180.27	3759.07
	FastNA	1977.82	2631.85	3290.03	3950.16	4603.41
	ACOPPI	**2045.85**	**2728.14**	**3408.96**	**4107.10**	**4782.83**
hs-sc	SPINAL	1731.81	2253.66	2839.00	3434.54	4066.22
	FastNA	2268.21	3017.96	3772.96	4520.51	5279.88
	ACOPPI	**2448.04**	**3262.86**	**4090.46**	**4893.83**	**5733.31**

Bảng 3.5: So sánh kết quả thực nghiệm của ACOPPI với SPINAL và FastNA theo tiêu chí EC

Dữ liệu	Thuật toán	α=0.3	α=0.4	α=0.5	α=0.6	α=0.7
ce-dm	SPINAL	2343.0	2320.0	2300.0	2237.0	2258.0
	FastNA	2560.7	2564.6	2567.2	2567.7	2567.6
	ACOPPI	**2792.6**	**2777.7**	**2786.4**	**2786.5**	**2780.0**
ce-hs	SPINAL	2370.0	2446.0	2437.0	2487.0	2512.0
	FastNA	2842.8	2838.1	2844.9	2838.0	2843.4
	ACOPPI	**3084.5**	**3067.1**	**3094.6**	**3086.6**	**3083.5**
ce-sc	SPINAL	2326.0	2384.0	2323.0	2361.0	2398.0
	FastNA	2761.1	2761.2	2769.7	2766.5	2763.1
	ACOPPI	**2973.4**	**2944.6**	**2974.6**	**2986.5**	**2985.3**
dm-hs	SPINAL	6189.0	6235.0	6282.0	6291.0	6344.0
	FastNA	7478.3	7481.9	7429.0	7478.2	7478.8
	ACOPPI	**7652.0**	**7661.2**	**7664.4**	**7675.2**	**7640.3**
dm-sc	SPINAL	5203.0	5150.0	5311.0	5283.0	5360.0
	FastNA	6569.7	6565.5	6570.7	6577.4	6572.3
	ACOPPI	**6797.6**	**6806.1**	**6808.3**	**6838.7**	**6828.6**
hs-sc	SPINAL	5703.0	5593.0	5651.0	5706.0	5798.0
	FastNA	7531.8	7528.5	7535.2	7527.0	7538.1
	ACOPPI	**8132.5**	**8141.5**	**8170.5**	**8149.6**	**8186.2**

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

ACOPPI là phương pháp metaheuristic cho bài toán dóng hàng hai đồ thị, có ý nghĩa trong sinh học là cung cấp thông tin giúp phát hiện chức năng của các protein. Ngoài ra còn bổ sung thêm vào lý thuyết đồ thị một phương pháp mới cho bài toán dóng hàng đồ thị.

Thực nghiệm cho thấy so với các phương pháp heuristic trước đây, thấy thuật toán đề xuất có tính ổn định và có điểm dóng hàng, số cạnh khớp vượt trội so với SPINAL và tốt hơn đáng kể so với FastNA.

Trong [4], các tác giả có đề xuất một phiên bản của SPINAL cho tiêu chí GOC, đó cũng là hướng phát triển luận văn.

TÀI LIỆU THAM KHẢO

Tài liệu tiếng Việt

[1]. Đỗ Đức Đôngvà Hoàng Xuân Huấn (2011), "Về biến thiên của vết mùi trong phương pháp ACO và các thuật toán mới", *Tạp chí Tin học và điều khiển học*, Tập 27, tr. 263-275.

[2]. Đỗ Đức Đông, *Phương pháp tối ưu đàn kiến và ứng dụng*- Luận án tiến sỹ tin học Đại học Công nghệ thông tin - Đại học quốc gia Hà Nội, 2012.

[3]. Lê Sỹ Vinh, *Giáo trình Tin sinh học* – Trường Đại học Công nghệ - Đại học Quốc gia Hà Nội. 2014

Tài liệu tiếng Anh

[4]. Aladag,A.E. and Erten,C. (2013), SPINAL: scalable protein interaction network alignment. *Bioinformatics*, Vol. 29 no 7, 917–924

[5]. B. Doerr, F. Neumann, D. Sudholdt, and C. Witt (2007), On the influence of pheromone updates in ACO algorithms, Technical Report CI-223/07, University of Dortmund,SFB 531.

[6]. Chindelevitch,L. et al. (2010), Local optimization for global alignment of protein interaction networks. In: *Pacific Symposium on Biocomputing*,Hawaii,USA, pp. 123–132

[7]. Chindelevitch L. et al. (2013), Optimizing a global alignment of protein interaction networks, *Bioinformatics* ,Vol. 29 no. 21, 2765–2773

[8]. Do Duc, Dong, Huy Q. Dinh, and Huan Hoang Xuan. "On the pheromone update rules of ant colony optimization approaches for the job shop scheduling problem." *Intelligent Agents and Multi-Agent Systems*. Springer Berlin Heidelberg, 2008. 153-160.

[9]. E. Alpaydın (2010), Introduction to Machine Learning, Massachusetts Institute of Technology, Second Edition.

[10]. Kelley,B.P. et al. (2003), Conserved pathways within bacteria and yeast as revealed by global protein network alignment. *Proc. Natl Acad. Sci. USA*, 100, 11394–11399.

[11]. Kelley,B.P. et al. (2004), Pathblast: a tool for alignment of protein interaction networks. *Nucleic Acids Res.*, 32,83–88.

[12]. Kuchaiev,O. et al. (2010), Topological network alignment uncovers biological function and phylogeny. *J. R. Soc. Interface.*, 7, 1341–1354.

[13]. Kuchaiev,O. and Przulj,N. (2011) Integrative network alignment reveals large regions of global network similarity in yeast and human. *Bioinformatics*, 27, 1390–1396.

[14]. K.Socha, M. Sampels and M. Manfrin (2003). "Ant Algorithms for the Univerrsity Course Timetabling Problem with Regard to the State-of-the-Art", *Applications of Evolutionary Computing*, Proceedings of the EvoWorkshops 2003, pp. 334-345.

[15]. Liao,C.S. et al. (2009) IsoRankN: spectral methods for global alignment of multiple protein networks. *Bioinformatics*, 25, i253–i258.

[16]. Memisevic,V. and Przulj,N. (2012), C-graal: common-neighbors-based global graph alignment of biological networks. Integr. Biol., 4, 734–743.

[17]. Milenkovic,T. et al. (2010), Optimal network alignment with graphlet degree vectors. *Cancer Inform.*, Vol.9, 121–137.

[18]. M. Birattari, P. Pellegrini, and M. Dorigo (2007), "On the invariance of ant colony optimization", *IEEE Transactions on Evolutionary Computation*, Vol11 (6), pp. 732-742.

[19]. M. Dorigo, V. Maniezzo and A. Colorni (1991), The Ant System: An autocatalytic optimizing process, Technical Report 91-016 Revised, *Dipartimento di Elettronica*, Politecnico di Milano, Milano, Italy.

[20]. M. Dorigo (1992), Optimization, learning and natural algorithms, *PhD. dissertation*, Milan Polytechnique, Italy.

[21]. M. Dorigo and L.M. Gambardella (1997), "Ant colony system: A cooperative learning approach to the traveling salesman problem", *IEEE Trans. on evolutionary computation*, Vol 1 (1), pp. 53-66.

[22]. M. Dorigo, and T. Stützle (2004), *Ant Colony Optimization*, The MIT Press, Cambridge, Masachusetts.

[23]. Narayanan,M. and Karp,R.M. (2007), Comparing protein interaction networks via a graph match-and-split algorithm. J. Comput. *Biol.*, Vol. 14, 892–907.

[24]. Park,D. et al. (2011) IsoBase: a database of functionally related proteins across PPI networks. *Nucleic Acids Res.*, 39, 295–300

[25]. P. Pellegrini and A. Ellero (2008), "The Small World of Pheromone Trails", Proc. of the 6th international conference on Ant Colony Optimization and Swarm

[26]. Remm,M. et al. (2001), Automatic clustering of orthologs and in-paralogs from pairwise species comparisons. J. Mol. *Biol.*, 314, 1041–1052.

[27]. Sharan,R. et al. (2005), Conserved patterns of protein interaction in multiple species. *Proc. Natl Acad. Sci. USA*, 102, 1974–1979.

[28]. Singh,R. et al. (2008), Global alignment of multiple protein interaction networks. In: Pacific Symposium on Biocomputing. pp. 303–314.

[29]. Supervisors: Prof. Dr. Ulf Leser, André Koschmieder, "Survey on the Graph Alignment Problem and a Benchmark of Suitable Algorithms", *Christoph Döpmann*, 19 July 2013.

[30]. Tran Ngoc Ha, Do Duc Dong and Hoang Xuan Huan, "An Efficient Ant Colony Optimization Algorithm for Multiple Graph Alignment", *International Conference on Computing*, Management and Telecommunications, pp.386-391, 2013.

[31]. T. Stützle and H. H. Hoos (2000), "Max-Min ant system", *Future Gene. Comput. Syst.*, Vol 26 (8), pp. 889-914.

[32]. T. Stützle and M. Dorigo (2002), "A short convergence proof for a class of ACO algorithms", *IEEE-EC*, Vol6 (4), pp. 358-365.

[33]. W.J. Gutjahr (2000), "An Ant based System and its convergence", *future generation Comput. Systems*, Vol16, pp. 873-888.

[34]. W.J. Gutjahr (2002), "ACO algorithms with guaranteed convergence to the optimal solution", *Info.Proc. Lett.*, Vol 83 (3), pp. 145-153.

[35]. W. J. Gutjahr (2007), "Mathematical runtime analysis of ACO algorithms: survey on an emerging issue", *Swarm Intelligence*, Vol 1 (1), pp. 59-79.

[36]. Zaslavskiy,M. et al. (2009) Global alignment of protein-protein interaction networks by graph matching methods. *Bioinformatics*, Vol.25, 259–267.

[37]. Z. Zang and Z. Feng (2012), "Two-stage updating pheromone forinvariant ant colony optimization algorithm", *Expert System with applications*, Vol 39 (1), pp. 706-712.