Kỷ yếu Hội nghị Khoa học Quốc gia lần thứ IX "Nghiên cứu cơ bản và ứng dụng Công nghệ thông tin (FAIR'9)"; Cần Thơ, ngày 4-5/8/2016
DOI: 10.15625/vap.2016.00059

MỘT THUẬT TOÁN HIỆU QUẢ DỰA TRÊN GIẢI THUẬT TỐI ƯU ĐÀN KIẾN GIẢI BÀI TOÁN $r|p$ TRUNG TÂM

Vũ Đức Quang[1], Hoàng Xuân Huấn[2], Đỗ Thanh Mai[3]

[1] Đại học Sư phạm – Đại học Thái Nguyên
[2] Đại học Công nghệ – Đại học Quốc gia Hà Nội
[3] Bộ môn KHCB – Khoa Ngoại ngữ – Đại học Thái Nguyên
vuducquang@dhsptn.edu.com, huanhx@vnu.edu.vn, dothanhmai.sfl@tnu.edu.vn

TÓM TẮT— *Bài toán $(r|p)$ trung tâm nhằm định vị điểm mở cơ sở cho hai đối thủ Trước và Sau (đối thủ của Trước) để mỗi người thu hút được thị phần lớn nhất cho mình đang là bài toán thời sự. Trong bài toán này, Trước được mở p cơ sở và Sau được mở r cơ sở. Thông thường, các khách hàng sẽ chọn cơ sở gần họ nhất làm nhà cung cấp cho họ. Chúng ta cần tìm cách chọn ra p vị trí đặt cơ sở cho Trước nhằm tối đa hóa thị phần (và đó là lợi nhuận) của mình với lưu ý là Sau cũng luôn tìm cách tối ưu hóa thị phần dựa trên phân bố cơ sở đã biết của Trước. Đây là một bài toán quy hoạch 2 mức thuộc loại NP-khó và đã có nhiều thuật toán được đề xuất. Trong bài báo, chúng tôi đề xuất một thuật toán tối ưu đàn kiến có sử dụng tìm kiếm địa phương. Kết quả thử nghiệm cho thấy thuật toán mới đề xuất của chúng tôi so với 3 phương pháp công bố gần đây có sử dụng phần mềm công cụ CPLEX thì: thuật toán thứ nhất có kết quả ngang bằng với kết quả của chúng tôi, nhưng chạy chậm hơn; thuật toán thứ hai có kết quả kém hơn kết quả của chúng tôi, nhưng chạy nhanh hơn; còn thuật toán thứ 3 cho kết quả ngang bằng và chạy nhanh hơn thuật toán của chúng tôi.*

Từ khóa— *thuật toán ACO, tìm kiếm địa phương, (r|p)-trung tâm.*

I. GIỚI THIỆU

Bài toán $(r|p)$-trung tâm $(r|p)$-centroid) lần đầu tiên được Hakimi [11] nghiên cứu dưới dạng bài toán rời rạc, có thể phát biểu như sau. Cho một tập I hữu hạn các địa điểm có thể chọn để đặt các cơ sở dịch vụ và một tập J hữu hạn của các vị trí của khách hàng, ma trận (d_{ij}) là khoảng cách từ khách hàng $j \in J$ tới cơ sở $i \in I$, các giá trị w_j xác định lợi nhuận của một cơ sở thu được trong việc phục vụ khách hàng j. Hai công ty/người chơi *Trước* và *Sau* sẽ mở các cơ sở kinh doanh tại các điểm của tập I. Đầu tiên, người chơi *Trước* mở p cơ sở. Biết được quyết định của *Trước*, *Sau* sẽ chọn để mở ra r cơ sở. Mỗi khách hàng sẽ chọn ra cơ sở gần họ nhất trong số $p + r$ cơ sở của cả hai người chơi đã mở ra như là nhà cung cấp cho mình. Kết quả là tập khách hàng sẽ được chia thành hai phần: tập khách hàng lựa chọn *Trước* và tập khách hàng lựa chọn *Sau*. Bài toán đặt ra là tìm ra p vị trí đặt cơ sở cho *Trước* để đạt tối đa nhất lợi nhuận dưới sự phản ứng mạnh mẽ nhất có thể của *Sau*.

Hiện nay, các nghiên cứu cơ bản của bài toán này có thể được phát triển thêm nhiều biến thể [8], [15], [17] dưới dạng các bài toán trên đồ thị [15], [17] và trong không gian Euclide [8]. Tuy nhiên, dạng bài toán rời rạc vẫn được nhiều người quan tâm nhất và người ta đã chỉ ra rằng bài toán của *Trước* thuộc loại $\sum_2^P - khó$ [7], [15], [18] còn khi đã biết các cơ sở của *Trước* thì bài toán của *Sau* thuộc loại NP-khó [7], [11], [18].

Đã có nhiều thuật toán đề xuất cho bài toán này [1], [3-5], [7], [9], [13], [18]. Đặc biệt, Davydov cùng các cộng sự [9] theo tiếp cận metaheuristics đã đề xuất 2 thuật toán VNS (Variable Neighborhood Search) và STS (Stochastic Tabu Search) giải gần đúng nhanh bài toán của *Trước*, trong đó họ dùng phần mềm CPLEX (một phần mềm của IBM cung cấp nhằm giải các bài toán quy hoạch tuyến tính) để tìm lời giải tối ưu cho *Sau* mỗi khi biết các cơ sở của *Trước*; Alekseeva cùng các cộng sự [2] phát triển thuật toán IM giải đúng bài toán *Trước*, trong đó cũng sử dụng phần mềm CPLEX cho toán *Sau*. Kết quả thực nghiệm cho thấy ưu điểm của các thuật toán này so với các thuật toán đã biết trước đó.

Trong bài báo này, chúng tôi đề xuất thuật toán tìm kiếm địa phương dựa trên giải thuật tối ưu đàn kiến (ACO) để giải bài toán $(r|p)$ trung tâm đồng thời cho cả *Trước* và *Sau* mà không dùng CPLEX. Kết quả thực nghiệm so sánh với các thuật toán VNS và STS [9], IM [2] trên bộ dữ liệu *Euclidean* và *Uniform* [19] cho thấy ưu điểm nổi trội của thuật toán mới.

Ngoài phần kết luận, phần còn lại của bài báo được trình bày như sau: mục II phát biểu bài toán $(r|p)$-trung tâm và một số vấn đề liên quan; thuật toán đề xuất được giới thiệu trong mục III; mục IV, chúng tôi trình bày kết quả các thử nghiệm so sánh với các thuật toán đã nêu với các bộ dữ liệu khác nhau (theo các công bố tương ứng) lấy từ thư viện [19].

II. BÀI TOÁN $(r|p)$ TRUNG TÂM VÀ MỘT SỐ VẤN ĐỀ LIÊN QUAN

2.1. Phát biểu bài toán

Bài toán dưới dạng đồ thị. Bài toán $(r|p)$-trung tâm rời rạc được phát biểu như sau (xem [2], [9], [18]).

Xét một đồ thị hai phía đầy đủ có trọng số $G = (I, J, E)$, trong đó tập đỉnh $I = \{1, \ldots, m\}$ biểu diễn tập hợp các địa điểm cơ sở tiềm năng mà 2 người chơi *Trước* và *Sau* có thể lựa chọn để mở cơ sở, tập đỉnh $J = \{1, \ldots, n\}$ biểu diễn tập khách hàng, $E = I \times J$ là tập các cạnh có độ đo khoảng cách tương ứng d_{ij} $\forall (i,j) \in E$, mỗi $j \in J$ có trọng số w_j $(w_j > 0)$ ứng với doanh thu mà cơ sở nhận được nếu khách hàng này chọn cơ sở làm nhà cung cấp. Biết rằng

mỗi khách hàng sẽ chọn cơ sở phục vụ gần nó nhất, trong trường hợp khoảng cách tới *Trước* bằng khoảng cách tới *Sau* thì khách hàng sẽ chọn *Trước*.

Ta cần tìm p vị trí trong tập I cho *Trước* sao cho tối đa hóa doanh thu của *Trước* với lưu ý rằng *Sau* sẽ chọn r cơ sở từ các địa điểm còn lại cũng nhằm tối đa hóa doanh thu của họ khi đã biết vị trí dịch vụ của *Trước*.

Gọi (X,Y) là lời giải cho bài toán $(r|p)$-trung tâm, trong đó $X \subseteq I$, $|X| = p$ là tập các cơ sở được *Trước* chọn, và $Y \subseteq \{I \setminus X\}$, $|Y| = r$ là tập các cơ sở được *Sau* lựa chọn. Với mỗi tập $V \subseteq I$ và $\forall j \in J$, ký hiệu $D(j,V) = min\{d_{ji} | \ i \in V\}$ cho khoảng cách tối thiểu từ khách hàng j đến tất cả các cơ sở trong tập V. Khi đó tập khách hàng sẽ được chia thành hai phần: Tập khách hàng lựa chọn *Trước* $U^T = \{ j \in J | D(j,Y) \geq D(j,X)\}$ và tập khách hàng lựa chọn *Sau* $U^S = \{J \setminus U^T\}$. Doanh thu của *Trước* sẽ là $p^T = \sum_{j \in U^T} w_j = \sum_{j \in J} w_j - p^S$ còn doanh thu của *Sau* sẽ là $p^S = \sum_{j \in U^S} w_j = \sum_{j \in J} w_j - p^T$. Yêu cầu bài toán là tìm ra tập cơ sở X cho *Trước* sao cho lợi nhuận của nó nhận được nhiều nhất cho dù *Sau* có lựa chọn cơ sở Y nào đi nữa. Bài toán tìm tập cơ sở Y tối ưu cho *Sau* khi biết trước X được gọi là bài toán $(r|Xp)$-trung vị ($(r|Xp) - medianoid$) và nó đã được Hakimi chứng minh là NP-khó [11]. Noltemeier cùng các cộng sự [15] đã chứng minh bài toán tập cơ sở X cho *Trước* có độ phức tạp là $\sum_2^P -khó$ ngay cả khi ma trận (d_{ij}) là ma trận khoảng cách Euclide trên mặt phẳng.

Bài toán dưới dạng quy hoạch hai mức. Bài toán $(r|p)$-trung tâm có thể phát biểu dưới dạng bài toán tìm minimax trong bài toán quy hoạch hai mức. Ký hiệu:

$$x_i = \begin{cases} 1 \ nếu \ Trước \ mở \ cơ \ sở \ i \\ 0 \ nếu \ ngược \ lại \end{cases} \tag{1}$$

$$y_i = \begin{cases} 1 \ nếu \ Sau \ mở \ cơ \ sở \ i \\ 0 \ nếu \ ngược \ lại \end{cases} \tag{2}$$

$$z_j = \begin{cases} 1 \ nếu \ khách \ hàng \ j \ được \ Trước \ phục \ vụ \\ 0 \ nếu \ khách \ hàng \ j \ được \ Sau \ phục \ vụ \end{cases} \tag{3}$$

Khi đó $X = \{ i \in I \mid x_i = 1\}$, $Y = \{ i \in I \mid y_i = 1\}$. Với mỗi khách hàng j, chúng ta định nghĩa tập cơ sở $I_j(X)$ cho phép Sau "thu hút" khách hàng j.

$$I_j(X) = \left\{ i \in I \mid d_{ij} < \min_{l \in I} \{ d_{lj} \mid x_l = 1 \} \right\} \tag{4}$$

Với các ký hiệu như trên, bài toán $(r|p)$-trung tâm được biểu diễn như sau:

$$\begin{aligned} &\max_x \sum_{j \in J} w_j z_j^*(X), \\ &\sum_{i \in I} x_i = p, \\ &x_i \in \{0,1\}, i \in I, \end{aligned} \tag{5}$$

Với $z_j^*(X), y_i^*(X)$ là phương án tối ưu thì bài toán *Sau* sẽ là:

$$\begin{aligned} &\max_{y,z} \sum_{j \in J} w_j (1 - z_j), \\ &\sum_{i \in I} y_i = r, \\ &1 - z_j \leq \sum_{i \in I_j(X)} y_i, \ i \in I, \\ &y_i, z_j \in \{0,1\}, \ i \in I, j \in J. \end{aligned} \tag{6}$$

Lưu ý rằng, hàm mục tiêu $W^*(X) = \sum_{j \in J} w_j z_j^*(X)$ là tổng lợi nhuận của *Trước* khi nó mở đúng p cơ sở, giá trị này phụ thuộc vào lời giải tối ưu của *Sau*.

2.2. Một số vấn đề liên quan

Đã có nhiều thuật toán đúng, heuristics và metaheuristics được đề xuất cho bài toán này. Gần đây, Davydov cùng các cộng sự [9] đã đề xuất 2 thuật toán metaheuristics VNS và STS giải gần đúng nhanh bài toán *Trước*; Alekseeva cùng các cộng sự [1] đã đề xuất 2 thuật toán metaheuristics IEM và MEM giải đúng bài toán cho *Trước*, sau đó cách phát triển thuật toán IM [2] hiệu quả hơn các phương pháp đã biết trước đó. Các thuật toán này đều giải bài toán quy hoạch 2 mức nhờ dùng phần mềm CPLEX để tìm lời giải tối ưu cho *Sau* mỗi khi biết các cơ sở của *Trước*. Trước khi giới thiệu thuật toán mới, chúng tôi giới thiệu tóm tắt phương pháp tối ưu hóa đàn kiến.

Phương pháp tối ưu hóa đàn kiến (ACO)

Phương pháp này được Dorigo đề xuất (1991), là một phương pháp metaheuristics, mô phỏng các tìm đường đi của kiến tự nhiên nhờ sử dụng vết mùi như thông tin học tăng cường, thực nghiệm cho thấy một công cụ mạnh mẽ để giải nhiều bài toán tối ưu tổ hợp khó và đang được sử dụng rộng rãi [10]. Trong ACO, bài toán gốc được đưa về bài toán tìm lời giải trên đồ thị cấu trúc $G = (V, E, \Omega, \eta, T)$, trong đó V tập đỉnh, E là tập cạnh, η và T tương ứng là các thông tin heuristics và vết mùi (thể hiện thông tin học tăng cường) có thể để ở các đỉnh hoặc các cạnh. Mỗi lời giải chấp nhận được là một đường đi thỏa mãn điều kiện Ω, bắt đầu từ một đỉnh trong tập C_0 của V, rồi mở rộng nhờ thủ tục bước ngẫu nhiên dựa trên thông tin heuristics và vết mùi. Các thuật toán ACO dùng N_{ant} kiến nhân tạo, trong mỗi bước lặp, mỗi con kiến tìm một lời giải nhờ thủ tục bước ngẫu nhiên trên đồ thị cấu trúc, các lời giải được đánh giá và áp dụng thủ tục tìm kiếm địa phương. Sau đó các lời giải lại được đánh giá và cập nhật mùi như là thông tin học tăng cường cho các con kiến tìm lời giải trong lần lặp tiếp theo. Lược đồ của thủ tục ACO áp dụng trong bài báo này được đặc tả trong Hình 1.

```
Procedure Thuật toán-ACO
Begin
    Khởi tạo ma trận vết mùi và N_ant kiến;
    Repeat
            Xây dựng lời giải; // Mỗi kiến sẽ xây dựng lời giải cho mình
            Tìm kiếm địa phương; // Áp dụng cho lời giải tốt nhất
            Cập nhật vết mùi;
    Until gặp điều kiện dừng;
End;
```

Hình 1. Thuật toán ACO tổng quát

Khi áp dụng phương pháp ACO, có 4 yếu tố quan trọng ảnh hưởng nhiều tới hiệu suất thuật toán: 1) Đồ thị cấu trúc và thủ tục bước ngẫu nhiên để tìm lời giải, 2) thông tin heuristics, 3) quy tắc cập nhật mùi, 4) kỹ thuật tìm kiếm địa phương.

III. THUẬT TOÁN $r|p$-ACO

3.1. Lược đồ tổng quát

Trong thuật toán này, *Trước* và *Sau* thực hiện quá trình lặp tuần tự việc tìm lời giải gần đúng cho mỗi người chơi. Ký hiệu $n_{ant}T$ và $n_{ant}S$ tương ứng là số kiến được dùng để tìm lời giải gần đúng cho người chơi *Trước* và *Sau* trong mỗi vòng lặp, N_{total} là số vòng lặp tuần tự tìm lời giải của thuật toán. Khi đó với r và p đã cho thuật toán $r|p$-ACO thực hiện theo lược đồ như Hình 2.

```
Bước 1. Khởi tạo ma trận vết mùi cho Trước và n_antT, n_antS, N_total;
Bước 2. Thực hiện lặp:
            2.1. ACO- Trước;  //Với mỗi kiến k tìm lời giải X_k cho Trước
            2.2. ACO- Sau;    //Tìm lời giải Y_k cho Sau với lời giải của Trước là X_k;
            2.3. Chọn ra X* là lời giải tốt nhất trong số các lời giải X_k;
            2.4. LS(X*);      //Tìm kiếm địa phương cho lời giải X* tốt nhất
            2.5. Cập nhật X* và trở lại 2.1 nếu chưa kết thúc.
Bước 3. Trích lời giải cho trước và sau.
```

Hình 2. Thuật toán $r|p$-ACO

Trong đó các thủ tục ACO- *Trước* và ACO- *Sau* thực hiện như sau.

3.2. Thủ tục ACO

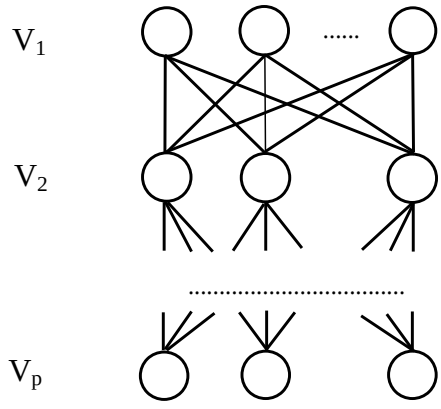
Các thủ tục ACO- *Trước* và ACO- *Sau* thực hiện như mô tả trong Hình 1, chỉ khác nhau ở tập vị trí được chọn trong mỗi bước lặp. Trước khi mô tả cụ thể từng thủ tục, chúng tôi giới thiệu các thành phần chính của thuật toán: đồ thị cấu trúc và thủ tục bước ngẫu nhiên, thông tin heuristics, quy tắc cập nhật mùi, kỹ thuật tìm kiếm địa phương.

Đồ thị cấu trúc và thủ tục bước ngẫu nhiên.

Đồ thị cấu trúc cho ACO- *Trước* (hoặc ACO- *Sau*) là một đồ thị $G(V, E)$ tương ứng gồm p (hoặc r tầng), các đỉnh ở mỗi tầng có cấu trúc như nhau ($V_i = I$) là tập vị trí có thể đặt cơ sở, các đỉnh ở tầng trước có cạnh kết nối với các đỉnh ở tầng liền sau nó như được mô tả trong Hình 3. Khi xây dựng lời giải theo thủ tục bước ngẫu nhiên, kiến chọn ngẫu nhiên một đỉnh thuộc tập ứng cử *allow* ở tầng hiện tại dựa trên vết mùi và thông tin heuristics sau đó đỉnh này được loại khỏi tập ứng cử cho việc chọn đỉnh kế tiếp ở tầng sau. Ở tầng thứ nhất, tập *allow* của *Trước* bằng $\{I\}$, còn tập *allow* của *Sau*$= \{I - X\}$. Nếu kiến k ở đỉnh y nào đó của tầng i thì xác suất nó chọn đỉnh x trong tập *allow*(y) ở tầng sau được cho bởi công thức (7).

$$p_x^k = \frac{\tau_x^\alpha n_x^\beta}{\sum_{z \in allow(y)} \tau_z^\alpha n_z^\beta} \quad (7)$$

Lưu ý rằng khi một đỉnh (vị trí) đã được kiến đi qua thì đỉnh tương ứng bị loại khỏi tập ứng cử cho mỗi tầng sau khi xây dựng lời giải. Việc xây dựng lời giải của kiến kết thúc khi qua hết các tầng.



Hình 3. Đồ thị cấu trúc

Thông tin heuristic

Thông tin heuristics η_x của đỉnh x được tính bằng tổng lợi nhuận của l khách hàng gần đỉnh x nhất chia cho tổng độ dài từ l khách hàng này tới x như trong công thức (8)

$$n_x = \frac{\sum_{i=1}^{l} w_i}{\sum_{i=1}^{l} dists[x,i]}, \tag{8}$$

trong đó $l = \frac{|I|}{p}$ là tỷ số giữa số lượng cơ sở trong I và p, τ_x là giá trị vết mùi trên đỉnh x ($x \in V_i$).

Quy tắc cập nhật vết mùi

Sau mỗi vòng lặp của thủ tục ACO- *Trước* / ACO- *Sau*, cường độ vết mùi trên mỗi đỉnh sẽ được cập nhật theo quy tắc SMMAS (xem [19]) cho bởi công thức (9)

$$\tau_i \leftarrow (1-\rho)\tau_i + \Delta\tau_i \tag{9}$$

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

w(t) là lời giải tốt nhất ở bước lặp thứ t sau khi đã thực hiện tìm kiếm địa phương, τ_{max} và τ_{min} là 2 tham số được xác định trước.

ACO-Trước

Thủ tục ACO- *Trước* thực hiện quá trình tìm lời giải cho người chơi *Trước* được đặc tả trong Hình 4, trong đó mỗi kiến sẽ lần lượt xây dựng lời giải cho riêng mình.

```
Procedure ACO- Trước
Begin
    for mỗi kiến k ∈ n_ant T do
        Xây dựng lời giải X_k cho kiến thứ k;
    Return X;
End;
```

Hình 4. Thủ tục ACO- *Trước*

ACO- Sau

Để đánh giá được lợi nhuận mà *Trước* nhận được khi chọn phương án X thì bài toán *Sau* phải được giải quyết. Như đã được trình bày ở trên, bài toán *Sau* hay còn được gọi là bài toán $(r|Xp) - medianoid$ đã được chứng minh là NP-khó. Trong phần này, chúng tôi xây dựng thủ tục ACO- *Sau* nhằm tìm ra r cơ sở tối ưu cho *Sau* khi biết được p cơ sở của phương án X do *Trước* chọn như trong Hình 5. Với mỗi phương án X, chúng tôi sử dụng số lượng kiến là $n_{ant}S$, mỗi kiến sẽ xây dựng phương án Y cho *Sau* cho riêng mình. Lời giải tốt nhất của phương án Y sẽ được sử dụng để tìm kiếm địa phương nhằm tăng chất lượng lời giải tại mỗi bước lặp. Kết thúc quá trình lặp sẽ trả về lời giải Y tốt nhất (Y*) cho người chơi *Sau*.

Cấu trúc đồ thị, thông tin heuristic và quy tắc cập nhật vết mùi trong thuật toán ACO- *Sau* tương tự như trong thuật toán ACO- *Trước*.

Để đánh giá chất lượng lời giải của thuật toán ACO- *Sau*, chúng tôi tiến hành chạy thử nghiệm thuật toán ACO- *Sau* so sánh kết quả của thuật toán với kết quả của phần mềm thương mại CPLEX của IBM được công bố trong [9] và nhận thấy rằng thuật toán ACO- *Sau* cho kết quả chính xác như phần mềm CPLEX với sai số xấp xỉ bằng 0.

```
Procedure ACO- Sau(X)
Begin
    Khởi tạo ma trận vết mùi cho Sau và n_ant S kiến;
    repeat
          for mỗi kiến k ∈ n_ant S do
              Xây dựng lời giải Y_k cho kiến thứ k;
          Chọn ra Y* là lời giải tốt nhất trong các Y_k;
          LS(Y*); //Tìm kiếm địa phương cho phương án Y*
           Cập nhật Y*;
     until gặp điều kiện dừng
     return Y*;
End;
```

Hình 5. Thuật toán ACO-Sau

Kỹ thuật tìm kiếm địa phương

Kỹ thuật tìm kiếm địa phương thực hiện đối với phương án X như sau: Mỗi phần tử trong phương án X sẽ được thay thế bởi một phần tử trong tập ứng cử U, nếu hàm mục tiêu thu được là cao hơn thì ghi nhận lại phương án X. Quá trình này được lặp lại cho đến khi mọi phần tử trong U được thay thế vào mọi vị trí trong X.

```
Procedure LS(X)
Begin
    U = I – X;
    for mỗi x ∈ X do
       for mỗi u ∈ U do
            Thay thế x bằng u;
            If (lợi nhuận thu được là tốt hơn) then Cập nhật X;
    Return X;
End;
```

Hình 6. Thuật toán tìm kiếm địa phương

Thuật toán tìm kiếm địa phương giúp cho việc cải thiện kết quả được tốt hơn, tuy nhiên độ phức tạp của thuật toán lại khá lớn, vì thế chúng tôi chỉ sử dụng tìm kiếm địa phương với kiến tốt nhất tại mỗi bước lặp nhằm mục đích tìm kiếm được lời giải tối ưu toàn cục.

IV. THỬ NGHIỆM

Trong phần này, chúng tôi cài đặt thuật toán $r|p$-ACO và thử nghiệm trên bộ dữ liệu từ thư viện Discrete Location Problems [19]. Tất cả các bộ thử nghiệm đều có kích thước như nhau $|I| = |J| = 100$. Có 2 bộ dữ liệu là *Eclidean* và *Uniform*. Trong loại *Eclidean*, ma trận (d_{ij}) xác định khoảng cách Eclide giữa các điểm trên một mặt phẳng và tất cả các điểm đó đều thuộc phạm vi 7000 x 7000. Trong loại *Uniform*, mỗi phần tử của ma trận (d_{ij}) có giá trị ngẫu nhiên trong khoảng từ 0 đến 10^4. Trong mỗi bộ dữ liệu đều có hai loại lợi nhuận, trường hợp thứ nhất $w_j = 1$ với mọi $j \in J$, còn trong trường hợp thứ hai thì các giá trị này được lựa chọn ngẫu nhiên trong khoảng từ 0 đến 200. Chúng tôi thử nghiệm với các bộ $p = r = \{10, 15\}$ trên bộ dữ liệu *Eclidean* và $p = r = \{7\}$ trên bộ dữ liệu *Uniform*.

Các thử nghiệm đều được tiến hành trên máy tính Intel Pentium G3220 3.0GHz, RAM 4GB, Window 7 Professional. Mục đích của thử nghiệm là đánh giá hiệu suất của thuật toán đề xuất thông qua so sánh lợi nhuận lớn nhất của *Trước* nhận được và độ phức tạp thời gian (phút) của thuật toán đề xuất với các giá trị tương ứng của thuật toán IM [2] (cài đặt trên máy tính Intel Xeon X5675, 3 GHz, RAM 96 GB, Windows Server 2008 và phần mềm CPLEX 12.3) và VNS, STS [9] (cài đặt trên máy tính Pentium Intel Core Dual PC, 2.66 GHz, 2GB RAM).

Các tham số trong thử nghiệm được thiết đặt như sau:

$|I| = |J| = 100$, $n_{ant}T$ = 50, $n_{ant}S$ = 10, N_{total} =100;

$\tau_{max} = 1.0, \tau_{min} = \frac{\tau_{max}}{2*|I|}\alpha = 1, \beta = 2, \rho = 0.1.$

Các Bảng 1, 2, 3 là các kết quả tính toán tương ứng của các thuật toán $r|p$-ACO, IM, VNS và STS. Trong đó, Bảng 1, 2 là kết quả khi chạy trên bộ dữ liệu *Euclidean*, Bảng 3 là kết quả khi chạy trên bộ dữ liệu *Uniform*. Trong mỗi bảng, cột Bộ test thể hiện mã của mỗi bộ test thử nghiệm trong bộ dữ liệu, cột W*(X) và Time tương ứng là lợi nhuận lớn nhất mà *Trước* thu được và thời gian chạy của mỗi thuật toán được tính theo đơn vị phút.

Bảng 1. Bộ dữ liệu Eclidean, $p = r = 10$

Bộ test	$W_j = 1$				$W_j \in 0\ldots200$							
	W*(X)		Time (phút)		W*(X)				Time (phút)			
	$r\|p$-ACO	IM	$r\|p$-ACO	IM	$r\|p$-ACO	IM	VNS	STS	$r\|p$-ACO	IM	VNS	STS
111	50	50	13.28	13	4,361	4,361	4,361	4,361	5.83	60	0.35	1.07
211	49	49	5.28	20	5,310	5,310	5,310	5,310	4.28	42	0.42	0.4
311	48	48	13.83	195	4,483	4,483	4,483	4,483	1.48	146	0.35	0.35
411	49	49	20.9	135	4,994	4,994	4,994	4,994	1.53	33	3.33	0.33
511	48	48	0.5	270	4,906	4,906	4,906	4,906	1.37	399	1.78	0.47
611	47	47	9.13	900	4,595	4,595	4,595	4,595	0.9	143	1.8	0.75
711	51	51	0.9	12	5,586	5,586	5,586	5,586	7.22	73	0.93	1.7
811	48	48	20.3	145	4,609	4,609	4,609	4,609	3.65	152	3.47	1.48
911	49	49	1.45	102	5,302	5,302	5,302	5,302	2.35	6	0.4	0.33
1011	49	49	5.92	180	5,005	5,005	5,005	5,005	3.28	97	3.57	1.73

Trong Bảng 1, với trường hợp W_j = 1 thì thuật toán $r|p$-ACO cho kết quả tương đồng với thuật toán IM nhưng với thời gian nhỏ hơn nhiều. Còn đối với trường hợp $W_j \in 0\ldots200$ thì thuật toán VNS, STS và $r|p$-ACO hơn nhau không đáng kể về mặt thời gian, bởi vì khi số lượng cơ sở được chọn cho *Trước* và *Sau* là thấp thì độ phức tạp của bài toán còn nhỏ, nhưng với số lượng cơ sở được chọn tăng dần thì độ phức tạp của bài toán tăng lên đáng kể. Trong [3] Ekaterina Alekseeva đã chứng minh độ phức tạp của bài toán là lớn nhất khi p = r = {15, 16, 17}.

Bảng 2. Bộ dữ liệu Eclidean $p = r = 15$

Bộ test	$W_j \in 0\ldots200$							
	W*(X)				Time (phút)			
	$r\|p$-ACO	IM	VNS	STS	$r\|p$-ACO	IM	VNS	STS
111	4,596	4,596	4,596	4,596	20.08	72	4.97	2.9
211	5,373	5,373	5,373	5,373	45.17	3,845	3.35	1.4
311	4,800	4,800	4,800	4,800	7.03	395.00	0.38	1.5
411	5,064	5,064	5,058	5,064	14.47	1,223	1.85	2.03
511	5,131	5,131	5,123	5,131	26.83	2,120	3.24	3.62
611	4,881	4,881	4,881	4,881	18.92	2,293	1.4	1.92
711	5,827	5,827	5,827	5,827	13.6	1,320	4.69	3.52
811	4,675	4,675	4,620	4,675	6.35	4,570	5.03	2.1
911	5,158	5,158	5,157	5,158	41.67	>600	4.23	2.63
1011	5,195	5,195	5,195	5,195	81.73	>600	0.4	0.82

Kết quả trong Bảng 2 cho thấy thuật toán $r|p$-ACO cho kết quả chính xác trong thời gian ngắn hơn nhiều so với thuật toán IM khi số lượng cơ sở được chọn cho *Trước* và *Sau* ngày càng tăng. Với $p = r = 15$, trong một số bộ test, thuật toán IM mặc dù với cấu hình mạnh nhưng phải chạy trong vài nghìn phút mới thu được kết quả (ví dụ bộ test 811 yêu cầu một khoảng thời gian lên đến 4,570 phút), nhưng thuật toán $r|p$-ACO chạy trong khoảng thời gian chấp nhận được (trung bình khoảng 27 phút). So với thuật toán VNS và STS thì thuật toán $r|p$-ACO chạy chậm hơn bởi lý do chính là $r|p$-ACO được dùng để giải cả hai bài toán cho *Trước* và *Sau*, trong khi đó VNS và STS chỉ giải bài toán của *Trước* (còn bài toán của Sau được giải bởi phần mềm CPLEX).

Bảng 3. Bộ dữ liệu Uniform $p = r = 7$

Bộ test	$W_j \in 0\ldots200$					
	W*(X)			Time (phút)		
	$r\|p$-ACO	VNS	STS	$r\|p$-ACO	VNS	STS
123	5009	5009	5009	38.26	5.08	1.1
223	5459	5459	5459	36.92	3.05	1.1
323	5019	5009	5019	79.28	2.42	0.92
423	4908	4908	4908	175.53	4.95	2.44
523	5208	5198	5208	8.4	4.88	0.38
623	5032	5032	5032	11.67	4.95	3.3
723	5055	5055	5055	18.62	4.78	1.05
823	4951	4860	4951	4.2	4.93	1.25
923	5127	5060	5127	14.43	3.63	1.87
1023	5084	5067	5084	59.1	5.38	4.65

Từ kết quả thử nghiệm trong Bảng 3 có thể kết luận rằng thuật toán $r|p$-ACO đề xuất cho kết quả tương đương với thuật toán STS trong khi đó thuật toán VNS chỉ đúng với các bộ test 123, 223, 423, 623, 723. Thời gian chạy của thuật toán không tối ưu bởi lý do tương tự như phân tích ở trên.

V. KẾT LUẬN

Trên đây, chúng tôi đề xuất một giải thuật ACO cho bài toán $(r|p)$-trung tâm rời rạc. Trong thuật toán này, phương pháp ACO được áp dụng dựa trên sự biểu diễn bài toán như một bài toán tối ưu hóa rời rạc hai mức. Thử nghiệm cho thấy thuật toán mới so với ba phương pháp gần đây phải dùng CPLEX để tìm lời giải cho Sau thì thuật toán mới này mặc dù chỉ chạy nhanh hơn một thuật toán khác, nhưng cho kết quả tính toán ngang bằng hoặc tốt hơn kết quả của thuật toán khác. Việc cải thiện tốc độ thực hiện của thuật toán mới thông qua cải tiến tìm kiếm địa phương và/hoặc kết hợp với CPLEX là một hướng nghiên cứu mới của chúng tôi trong tương lai. Chúng tôi cũng hy vọng tiếp cận này có thể phát triển cho các bài toán tương tự về mạng, khi khách hàng nằm ở các đỉnh của đồ thị còn các cơ sở có thể mở tại các điểm tùy ý trên các cạnh của nó.

TÀI LIỆU THAM KHẢO

[1] Alekseeva E. and Kochetov Y., "Metaheuristics and Exact Methods for the Discrete (r|p)-Centroid Problem", Metaheuristics for Bi-Level Optimization, Talbi, E.-G. and Brotcorne, L., Eds., Berlin:Springer, 2013.

[2] Alekseeva E., Kochetov Y., Plyasunov A., "An exact method for the discrete (r|p)-centroid problem", Springer Science+Business Media New York, Vol. 63. p. 445–460, 2015.

[3] Alekseeva E., Kochetova N., Kochetov Y., Plyasunov A., "A Heuristic and Exact Methods for the Discrete(r|p)-Centroid Problem", LNCS, vol. 6022, pp. 11–22, 2010.

[4] Benati, S. and Laporte, G., "Tabu Search Algorithms for the (r|Xp)-Medianoid and (r|p)-Centroid Problems", Locat. Sci., vol. 2, no. 2, pp. 193–204, 1994.

[5] Campos Rodríguez C., Moreno-Pérez J.A., and Santos-Peńate D. R., "Particle Swarm Optimization with Two Swarms for the Discrete (r|p)-Centroid problem", LNCS, vol. 6927, pp. 432–439, 2012.

[6] Campos Rodríguez C., Santos Peńate D. R., and Moreno Perez J. A., "An Exact Procedure and LP Formulations for the Leader–Follower Location Problem", TOP, vol. 18, no. 1, pp. 97–121, 2010.

[7] Davydov I. A., "Local Tabu Search for the Discrete (r|p)-Centroid Problem", Diskret. Anal. Issled. Oper., , vol. 19, no. 2, pp. 19–40, 2012.

[8] Davydov I. A., Kochetov Y., and Carrizosa E., "A Local Search Heuristic for the (r|p)-Centroid Problem" Computers & Operations Research, DOI: 10.1016/j.cor.2013.05.003, 2013.

[9] Davydov I. A., Kochetov Yu .A., Mladenovic N., and Urosevic D., "Fast Metaheuristics for the Discrete (r|p)-Centroid Problem", Automation and Remote Control, Vol. 75 (4). p. 677–687, 2014.

[10] H. Hoang Xuan, T. Nguyen Linh, D. Do Duc, H. Huu Tue, Solving the Traveling Salesman Problem with Ant Colony Optimization: A Revisit and New Efficient Algorithms, REV Journal on Electronics and Communications, Vol. 2, No. 3–4, , p.121-129, 2012.

[11] Hakimi S. L., "Locations with Spatial Interactions: Competitive Locations and Games", Discrete Location Theory, Mirchandani P. B. and Francis R. L., Eds., London: Wiley, pp. 439–478, 1990.

[12] Hotelling H., "Stability in Competition", Economic J., vol. 39, pp. 41–57, 1929.

[13] Kochetov Y. A., Facility Location: "Discrete Models and Local Search Methods, in Combinatorial Optimization. Methods and Applications", Chvatal, V., Ed., Amsterdam: IOS Press, pp. 97–134, 2011.

[14] Kress D. and Pesch E., "Sequential Competitive Location on Networks", Eur. J. Oper. Res.,vol. 217, pp. 483-499, 2012.

[15] Noltemeier H., Spoerhase J., and Wirth H., "Multiple Voting Location and Single Voting Location on Trees", Eur. J. Oper. Res., vol. 181, pp. 654–667, 2007.

[16] Roboredo M. C. and Pessoa A. A., "A Branch-and-Cut Algorithm for the Discrete (r|p)-Centroid Problem", Eur. J. Oper. Res., vol. 224, no. 1, pp. 101–109, 2013.

[17] Spoerhase J. and Wirth H., "(r, p)-Centroid Problems on Paths and Trees", J. Theor. Comput. Sci. Archive, vol. 410, pp. 5128–5137, 2009.

[18] Talbi, E.-G. (ed.) "Metaheuristics for bi-level optimization". Studies in Computational Intelligence, vol. 482, pp. 189–219. Springer, Berlin, 2013.

[19] http://math.nsc.ru/AP/benchmarks/, accessed on 13/05/2016.

AN EFFECTIVE ALGORITHM BASED ACO SOLVE $r|p$ CENTROID PROBLEM

Vu Duc Quang, Hoang Xuan Huan, Do Thanh Mai

ABSTRACT— *The (r|p) centroid problem is to locate the openning facility for two players Leader and Follower (Leader's competitor). It helps the players gain the largest market share for themselves and therefore it is now one of the most attractive proprlems for researchers. In this problems, Leader opens p facilities then Follower opens r facilities. Typically, each client chooses the nearest facility as his supplier. We need to find how to choose p facilities for the Leader in such a way that it maximizes his market share (profit), and then Follower always wants to optimize the market share based on distribution of Truoc's facilities. This problem can be represented as a bilevel programming problem of the NP-hard type and there have been many proposed algorithms for the problem. In this paper, we propose an algorithm based on ant colony optimization algorithm with local search. The test results show that our algorithm in comparison with three algorithms recently published using the CPLEX tool is that the first algorithm's result is equal to ours, but it runs slower than ours; the second algorithm runs faster than ours but the result is not good as ours, the third algorithm's result is equal to ours, and it run faster than ours.*