HNUE JOURNAL OF SCIENCE DOI: 10.18173/2354-1059.2023-0007
Natural Sciences 2023, Volume 68, Issue 1, pp. 77-90
This paper is available online at http://stdb.hnue.edu.vn

GIẢI THUẬT TIẾN HÓA CHO BÀI TOÁN LẬP LỊCH VỚI TÀI NGUYÊN GIỚI HẠN MỚI

Nguyễn Thế Lộc[1], Đặng Quốc Hữu[2] và Vũ Thái Giang[1]
[1]*Khoa Công nghệ thông tin, Trường Đại học Sư phạm Hà Nội*
[2]*Khoa Hệ thống thông tin Kinh tế và Thương mại điện tử, Trường Đại học Thương mại*

Tóm tắt. Bài báo này đề xuất và phát biểu dưới hình thức toán học một bài toán mới được đặt tên là TDOS-RCPSP. Là bài toán thuộc họ RCPSP đã biết, TDOS-RCPSP có nhiều ứng dụng thiết thực trong các lĩnh vực khác nhau, đặc biệt là trong việc tổ chức các dây chuyền sản xuất công nghiệp. Bài báo này sẽ đề xuất một thuật toán tiến hóa dựa trên chiến lược Cuckoo Search để tìm lời giải gần đúng cho bài toán TDOS-RCPSP. Bài báo sẽ giới thiệu những kết quả thực nghiệm được tiến hành nhằm kiểm chứng hiệu quả của thuật toán đề xuất dựa trên hai bộ dữ liệu thực nghiệm. Bộ dữ liệu thứ nhất, iMOPSE, đã được công bố trong các công trình nghiên cứu cùng lĩnh vực trước đây. Bộ dữ liệu còn lại được tác giả thu thập từ các dây chuyền dệt may công nghiệp của tập đoàn TNG. Những kết quả thu được đã khẳng định sự vượt trội của thuật toán tiến hóa đề xuất so với những thuật toán trong cùng lĩnh vực đã được công bố trước đó.

Từ khóa: lập lịch với tài nguyên giới hạn, thuật toán tiến hóa, thuật toán Cuckoo Seach.

1. Mở đầu

RCPSP [1] là lớp các bài toán lập lịch có ứng dụng trong nhiều lĩnh vực thực tế mà điển hình là việc tổ chức các dây chuyền sản xuất công nghiệp [2-5]. Một số bài toán trong lớp RCPSP đã được chứng minh là NP-Khó, ví dụ như bài toán MS-RCPSP (Multi Skill - Resource-Constrained Project Scheduling Problem: Bài toán lập lịch đa kĩ năng với tài nguyên giới hạn) [6-9].

Đã có nhiều nhóm nghiên cứu công bố giải pháp cho bài toán MS-RCPSP hướng tới việc tìm lịch biểu có thời gian thực hiện (từ đây gọi là *makespan*) tối thiểu trong một số điều kiện ràng buộc về tài nguyên và giả thiết chúng có những mức kĩ năng khác nhau. Tuy nhiên sự phát triển của công nghệ trong những năm gần đây đã khiến cho giả thiết của bài toán MS-RCPSP - rằng thời gian một tài nguyên xử lí một tác vụ chỉ phụ thuộc vào bản thân tác vụ đó - trở nên không phù hợp. Trong thực tế, tài nguyên có mức kĩ năng cao hơn thường thực hiện công việc trong thời gian ngắn hơn tài nguyên với mức kĩ năng thấp. Ví dụ, để hoàn thành cùng một tác vụ, người thợ bậc cao hay robot phiên bản mới sẽ tốn thời gian ít hơn người thợ bậc thấp hay robot phiên bản cũ.

Ngày nhận bài: 15/3/2023. Ngày sửa bài: 24/3/2023. Ngày nhận đăng: 31/3/2023.
Tác giả liên hệ: Nguyễn Thế Lộc. Địa chỉ e-mail: locnt@hnue.edu.vn

Để khắc phục hạn chế nêu trên của bài toán MS-RCPSP, bài báo này đề xuất và định nghĩa một bài toán mới thuộc họ RCPSP tên là TDOS-RCPSP (Time Depends On Skill - RCPSP). Khác biệt quan trọng nhất của bài toán đề xuất so với bài toán MS-RCPSP trước đây là những tài nguyên với mức kĩ năng khác nhau sẽ có thời gian xử lí khác nhau đối với mỗi loại tác vụ. Khác biệt đó cũng là nguyên nhân khiến TDOS-RCPSP phù hợp với thực tế hơn so với bài toán MS-RCPSP trước đây.

Phần còn lại của bài báo gồm những nội dung chính sau đây: Mục 2.1 giới thiệu họ bài toán RCPSP và những công trình nghiên cứu có liên quan; Mục 2.2 phát biểu bài toán mới TDOS-RCPSP dưới dạng mô hình toán học; Mục 2.3 trình bày thuật toán đề xuất có tên là GCS được phát triển dựa trên ý tưởng của chiến lược tìm kiếm Cuckoo Search [10]. Cải tiến quan trọng của GCS mang lại hiệu năng mạnh mẽ cho thuật toán này là hàm Improve cũng được giới thiệu trong mục này. Những kết quả thực nghiệm nhằm kiểm chứng hiệu năng của thuật toán đề xuất được trình bày ở Mục 2.4 dựa trên hai bộ dữ liệu: bộ dữ liệu iMOPSE [7, 11] và bộ dữ liệu TNG [12, 13]. Cuối cùng, Mục 3 tóm tắt những kết quả chính trong bài và phác thảo những ý tưởng nghiên cứu tác giả dự định tiến hành để mở rộng các nội dung trong bài.

2. Nội dung nghiên cứu

2.1. Họ bài toán RCPSP và những nghiên cứu liên quan

RCPSP là họ bài toán lập lịch trong điều kiện tài nguyên hạn chế, chẳng hạn như các ràng buộc sau đây:

- Tập hợp tài nguyên là có hạn và cho trước;
- Tại mỗi thời điểm, một tài nguyên chỉ có thể xử lí được duy nhất một tác vụ. Nói cách khác tài nguyên không thể đồng thời xử lí nhiều tác vụ đồng thời;
- Với một tác vụ cho trước, không phải mọi tài nguyên đều có khả năng xử lí mà chỉ những tài nguyên thuộc tập con ứng với tác vụ đó mà thôi.

Trong họ RCPSP, bài toán MS-RCPSP được nhiều nhóm nghiên cứu quan tâm giải quyết. Đầu tiên MS-RCPSP được H. Najafzad chứng minh là thuộc lớp bài toán NP-Khó [6]. Sau đó nhóm nghiên cứu của Myszkowski đã đề xuất các giải pháp heuristic [8] như: sắp xếp lại tập tác vụ dựa trên thời gian xử lí của chúng theo thứ tự giảm dần rồi gán cho các tài nguyên để thực hiện. Sau đó Myszkowski và cộng sự lần lượt đề xuất những thuật toán hiệu quả hơn dựa trên các cơ chế Tabu Search, Ant colony và GA [7, 8, 11]. Bên cạnh những thuật toán đó, một đóng góp quan trọng của nhóm Myszkowski là xây dựng được bộ dữ liệu thực nghiệm iMOPSE [7]. Bộ dữ liệu này được công nhận rộng rãi như bộ dữ liệu chuẩn để tiến hành kiểm chứng những giải pháp mà các nhóm nghiên cứu tìm ra cho lớp bài toán RCPSP.

Ngoài MS-RCPSP, một số bài toán khác trong họ RCPSP cũng đã được nghiên cứu, chẳng hạn như bài toán MMSRCPSP (Multi-mode Multi-skilled Resource-Constrained Project Scheduling Problem) đã được Hosseinian và cộng sự đề xuất giải pháp [2]. MMSRCPSP là một phiên bản khác của bài toán MS-RCPSP với ràng buộc bổ sung là mỗi tác vụ chỉ có thể được thực hiện trong một số chế độ cho trước. Sau thời điểm bắt đầu quá trình thực thi tác vụ thì chế độ thực thi không thể thay đổi được nữa mà phải giữ nguyên cho tới khi tác vụ được thực hiện xong.

Để giải quyết bài toán MMSRCPSP, Hosseinian và cộng sự đã sử dụng thuật toán GA truyền thống kết hợp với phương pháp ra quyết định dựa trên độ đo thông tin Shanon-entropy để tìm kiếm các cá thể phù hợp cho thế hệ lời giải kế tiếp [3]. Trong một công bố khác [4], nhóm nghiên cứu của Hosseinian đã sử dụng thuật toán Dandelion Algorithm để giải quyết bài toán MS-RCPSP và sử dụng bộ dữ liệu thực nghiệm iMOPSE. Trong nghiên cứu này, Hosseinian hướng đến hai mục tiêu là tối thiểu hóa thời gian và chi phí thực hiện dự án.

Tương tự như Hosseinian, H. Davari-Ardakani và đồng nghiệp [6] cũng nghiên cứu một biến thể đa mục tiêu của bài toán MS-RCPSP hướng đến mục tiêu giảm thiểu thời gian và chi phí của dự án. Bài toán đề xuất, được H. Davari-Ardakani gọi là MSPSP, tập trung nghiên cứu các dự án có hai đặc điểm sau:

- Thời gian thực hiện dự án là tùy ý. Ví dụ như dự án có thể được thực hiện vào buổi tối hoặc cuối tuần;

- Chi phí năng lượng cho quá trình thực hiện dự án là rất cao, chẳng hạn như lương trả cho nhân viên theo chế độ làm việc ngoài giờ.

Trong bài báo của mình, H. Davari-Ardakani chỉ mới phát biểu bài toán MSPSP mà chưa đề xuất bất kỳ giải pháp nào nên đến nay bài toán được coi là mở cho những nhà nghiên cứu đi sau.

Các nghiên cứu kể trên đều có một hạn chế, đó là giả thiết rằng với một tác vụ cho trước, dù tài nguyên nào xử lí thì thời gian thực hiện cũng như nhau. Nói cách khác, thời gian xử lí của một tác vụ chỉ phụ thuộc vào bản thân tác vụ chứ không phụ thuộc vào trình độ kĩ năng của tài nguyên. Giả thiết này không phù hợp với thực tế, ví dụ trong nhà máy công nhân càng lành nghề thì thời gian hoàn thành sản phẩm càng ngắn, hay robot càng hiện đại thì thời gian chế tạo sản phẩm càng nhanh.

Phần tiếp theo mô tả và phát biểu một bài toán mới được chúng tôi đặt tên là TDOS-RCPSP, được xây dựng dựa trên các giả thiết nhằm khắc phục nhược điểm vừa nêu.

2.2. Mô hình toán học của bài toán TDOS-RCPSP

Trong bài toán TDOS-RCPSP, chúng tôi giả thiết rằng mỗi dự án bao gồm một tập hợp các tác vụ (công việc) được thực hiện, xử lí bởi một tập hợp các tài nguyên (công nhân, robot). Mỗi tác vụ yêu cầu mức kĩ năng riêng đối với tài nguyên để có thể thực hiện được tác vụ đó. Nói cách khác, tài nguyên cần phải có mức kĩ năng bằng hoặc cao hơn mức kĩ năng mà tác vụ yêu cầu thì mới thực hiện được tác vụ đó.

✓ Tài nguyên có khả năng xử lí

× Tài nguyên không đủ khả năng xử lí

Tài nguyên

L_1	$S_{1.3}$, $S_{2.2}$
L_2	$S_{2.1}$, $S_{3.2}$
L_3	$S_{1.2}$, $S_{2.1}$
L_4	$S_{2.2}$, $S_{3.3}$

Tác vụ

W_1	W_2	W_3	W_4
$S_{2.2}$	$S_{3.1}$	$S_{2.2}$	$S_{1.1}$
✓	×	✓	✓
×	✓	×	×
×	×	×	✓
✓	✓	✓	×

Hình 1. Yêu cầu của tác vụ đối với tài nguyên về mức kĩ năng

Trong Hình 1, ký hiệu $S_{i,j}$ có nghĩa là kĩ năng cần thiết là S_i và trình độ kĩ năng tối thiểu phải là j;

Ví dụ: cột đầu tiên của bảng con Tác vụ (W_1 , $S_{2,2}$) cho thấy rằng tác vụ W_1 yêu cầu tài nguyên phải có kĩ năng S_2 với mức độ kĩ năng lớn hơn hoặc bằng 2. Bảng con Tài nguyên cho thấy tài nguyên L_1 có kĩ năng S_2 và mức độ kĩ năng của S_2 là 2 nên tài nguyên L_1 có thể thực hiện được tác vụ W_1.

Bài toán TDOS-RCPSP được định nghĩa như sau với kí hiệu:

- C_j là tập hợp các tác vụ cha của tác vụ Wj, những tác vụ này cần được hoàn thành trước khi bắt đầu thực hiện tác vụ Wj;
- S: tập các kĩ năng; S^i: tập con kĩ năng mà tài nguyên L_i sở hữu. Dĩ nhiên ta có $S^i \subseteq S$;
- S_i: kĩ năng thứ i;
- t_{jk}: thời gian để tài nguyên thứ k thực hiện tác vụ thứ j;
- L: tập hợp tất cả các tài nguyên; L_i: tài nguyên thứ i;
- L^k: tập hợp các tài nguyên có đủ khả năng thực hiện tác vụ k; Dĩ nhiên ta có $L^k \subseteq L$;
- W: tập tất cả các tác vụ của dự án; W_i: tác vụ thứ i;
- W^k: tập hợp tất cả các tác vụ mà tài nguyên k có đủ khả năng thực hiện; Ta có $W^k \subseteq W$;
- r^i: tập hợp kĩ năng đòi hỏi bởi tác vụ i;
- $B(W_k)$ and $E(W_k)$: thời điểm tác vụ k được bắt đầu thực hiện và thời điểm mà nó được hoàn thành;
- $A_{u,v}^{\ i}$: biến logic xác định tài nguyên v đang thực hiện tác vụ u tại thời điểm i hay không, $A_{u,v}^{\ i} = 1$ nghĩa là tác vụ u đang được thực hiện bởi tài nguyên v tại thời điểm i;
- h_i: mức kĩ năng của kĩ năng S_i; g_i: loại kĩ năng của kĩ năng S_i;
- P: một lịch biểu; P_{all}: tập hợp tất cả các lịch biểu;
- n: số lượng tác vụ; z: số lượng tài nguyên;
- $W^P_{\ first}$: tác vụ đầu tiên được thực hiện dưới sự bố trí theo lịch biểu P;
- $W^P_{\ last}$: tác vụ cuối cùng được hoàn thành dưới sự bố trí theo lịch biểu P;
- $f(P)$: hàm mục tiêu, giá trị của hàm này chính là thời gian thực hiện (makespan) của lịch biểu P:

$$f(P) = E(W^P_{\ last}) - B(W^P_{\ first}) \quad (1)$$

Bài toán TDOS-RCPSP được phát biểu như sau. Hãy tìm lịch biểu sao cho:

$$f(P) = E(W^P_{\ last}) - B(W^P_{\ first}) \rightarrow \min$$

Công thức (1) xác định giá trị của hàm mục tiêu chính là khoảng thời gian từ thời điểm bắt đầu thực hiện tác vụ đầu tiên đến thời điểm tác vụ cuối cùng được hoàn thành.

Một lịch biểu P phải thỏa mãn các ràng buộc sau đây mới được coi là hợp lệ:

$$S^k \neq \varnothing \quad \forall L_k \in L \quad (2)$$

$$t_{jk} \geq 0 \quad \forall W_j \in W, \quad \forall L_k \in L \quad (3)$$

$$E(W_j) \geq 0 \qquad \forall W_j \in W \quad (4)$$

$$E(W_i) \leq B(W_j) \quad \forall W_j \in W, j \neq 1, W_i \in C_j \quad (5)$$

$$\forall W_i \in W^k \; \exists S_q \in S^k \; g_{S_q} = g_{r^i} \text{ and } h_{S_q} \geq h_{r^i} \quad (6)$$

$$\forall L_k \in L, \forall q \in \left[B\left(W^P_{first}\right), E\left(W^P_{last}\right)\right] : \sum_{i=1}^{n} A^q_{i,k} \leq 1 \quad (7)$$

$$\forall W_j \in W, \forall q \in \left[B\left(W^P_{first}\right), E\left(W^P_{last}\right)\right], \exists!\ L_k \in L: A^q_{j,k} = 1 \quad (8)$$

$$if\ h_k \le h_l \text{ then } t_{ik} \le t_{il}\ \forall\ (r^k, r^l) \in \{S^k \times S^l\} \quad (9)$$

Trong đó:

- Ràng buộc (2) qui định rằng mỗi tài nguyên phải sở hữu ít nhất một kĩ năng;

- Ràng buộc (3,4) đảm bảo rằng thời gian thực hiện tác vụ phải là một giá trị dương;

- Ràng buộc (5) ngụ ý rằng tác vụ cha (W_i) phải được hoàn thành trước khi tác vụ con (W_j) bắt đầu;

- Ràng buộc (6) đảm bảo rằng với một tác vụ bất kỳ $W_i \in W^k$ (tập hợp tất cả các tác vụ mà tài nguyên *k* có đủ khả năng thực hiện), luôn có một kĩ năng $S \in S^k$ có cùng loại kĩ năng mà nhiệm vụ W_i yêu cầu và mức kĩ năng cao hơn hoặc bằng mức kĩ năng mà W_i yêu cầu;

- Ràng buộc (7) qui định rằng tại mỗi thời điểm (q) trong quá trình thực hiện lịch biểu P, mỗi tài nguyên chỉ thực hiện nhiều nhất một tác vụ. Nếu $\sum_{i=1}^{n} A_{i,k}^{q} = 0$ thì tài nguyên *k* không được phân công thực hiện nhiệm vụ nào cả. Trong trường hợp $\sum_{i=1}^{n} A_{i,k}^{q} = 1$ thì tài nguyên *k* được phân công thực hiện một tác vụ;

- Ràng buộc (8) đảm bảo rằng mỗi tác vụ chỉ được thực hiện bởi một tài nguyên duy nhất;

- Ràng buộc (9) mô tả rằng mức kĩ năng của tài nguyên càng cao thì thời gian thực hiện tác vụ càng ngắn.

So sánh với những bài toán trước đây như MS-RCPSP [6], khác biệt cơ bản và quan trọng của bài toán TDOS-RCPSP nằm ở ràng buộc (9). Ràng buộc này mô tả rằng thời gian thực hiện tác vụ phụ thuộc vào trình độ kĩ năng của tài nguyên, do đó mức kĩ năng của tài nguyên càng cao thì thời gian để tài nguyên thực hiện nhiệm vụ càng ngắn. Điều này làm cho tính thực tiễn của bài toán TDOS-RCPSP cao hơn so với những bài toán đã được phát biểu trước đó như MS-RCPSP. Chẳng hạn trong dây chuyền sản xuất của nhà máy, các công nhân lành nghề hơn hay các máy móc và rô-bốt phiên bản tiên tiến hơn luôn có thời gian thực hiện ngắn hơn.

2.3. Thuật toán đề xuất

Cuckoo Search là một cơ chế tiến hóa hiệu quả được công bố bởi Xin-She Yang và Suash Deb vào năm 2009 [10]. Để giải quyết bài toán TDOS-RCPSP nêu trên, bài báo này đề xuất thuật toán mới tên là GCS (Greedy Cuckoo Search) dựa trên cơ chế Cuckoo Search. Để nâng cao hiệu quả của Cuckoo Search truyền thống, chúng tôi xây dựng thêm hàm Improve để nâng cao chất lượng lời giải cho những lịch biểu tốt nhất trong mỗi thế hệ.

2.3.1. Hàm Improve

Quan sát việc thực hiện các dự án thực tế, chẳng hạn một dây chuyền sản xuất công nghiệp, ta thấy có thể xảy ra hai trường hợp sau:

- Tác vụ cha và tác vụ con được thực hiện bởi cùng một tài nguyên, khi đó các tài nguyên sẽ được sử dụng một cách hiệu quả, liên tục và không có thời gian rỗi.

- Tác vụ cha và tác vụ con được thực hiện bởi hai tài nguyên khác nhau. Trong trường hợp này, đôi khi tài nguyên cho tác vụ con đã sẵn sàng nhưng phải chờ tác vụ cha vì nó chưa kết thúc, hậu quả là tài nguyên bị rơi vào tình trạng nhàn rỗi và bị lãng phí.

- Để khắc phục tình trạng lãng phí tài nguyên trên, chúng tôi xây dựng hàm Improve để giảm thiểu thời gian nhàn rỗi của tài nguyên từ đó giảm thời gian thực hiện dự án. Các bước cụ thể như sau:

- *Bước 1*: Tìm tài nguyên trong dự án kết thúc công việc muộn nhất (ký hiệu là L_b).

- *Bước 2*: Kiểm tra các tác vụ do L_b thực hiện theo thứ tự ngược (tương tự như thứ tự LIFO trong ngăn xếp), nghĩa là tác vụ thực hiện sau sẽ được xem xét trước. Đối với mỗi tác vụ, hàm Improve thực hiện các bước sau:

+ Tìm khoảng thời gian nhàn rỗi của tài nguyên

+ Sửa đổi thứ tự thực hiện các tác vụ bằng cách di chuyển nó đến vị trí tương ứng với thời gian nhàn rỗi của tài nguyên. Kiểm tra tính hợp lệ của lịch biểu mới, nhất là ràng buộc về mức độ ưu tiên giữa các tác vụ.

+ Nếu lịch biểu hợp lệ thì cập nhật thời gian bắt đầu và kết thúc của các tác vụ liên quan. Tính toán lại thời gian thực hiện (makespan) theo sự sắp xếp của lịch biểu mới. Nếu makespan nhỏ hơn (tốt hơn) lịch biểu cũ thì cập nhật lại lịch biểu. Nếu không thì khôi phục lại lịch biểu cũ.

Nguyên tắc của hàm Improve là phân công lại tác vụ cho các tài nguyên để giảm thiểu thời gian nhàn rỗi của chúng, nhờ đó giảm được tổng thời gian thực hiện của dự án.

2.3.2. Thuật toán GCS

Là cơ chế tiến hóa hiệu quả thường được áp dụng cho các bài toán NP-Hard, Cuckoo Search [10] cập nhật quần thể bằng cách sử dụng bước dịch chuyển ngẫu nhiên Lévy Flight [14] với độ dịch chuyển tuân theo phân phối Levy và kết hợp với hai kĩ thuật tìm kiếm là Tìm kiếm cục bộ và Tìm kiếm toàn cục. Hai kĩ thuật tìm kiếm này giúp Cuckoo Search mở rộng không gian tìm kiếm và tránh được bẫy cực trị địa phương. Chúng tôi đã xây dựng thuật toán GCS dựa trên ý tưởng Cuckoo Search kết hợp với hàm Improve để cải thiện lịch biểu tốt nhất cho mỗi thế hệ. Cụ thể GCS thực hiện các bước sau:

- *Bước 1*: Khởi tạo quần thể ban đầu gồm n cá thể (tức là lịch biểu).

- *Bước 2*: Tính các tham số đặc trưng của quần thể như makespan của cá thể tốt nhất (*bestnest*), makespan tốt nhất của mỗi cá thể trong quần thể qua các thế hệ (*fitness*).

- *Bước 3*: Thực hiện các bước tiến hóa cho quần thể trong từng thế hệ.

- *Bước 4*: Với mỗi thế hệ:

+ *Bước 4.1*: Xây dựng lịch biểu mới P_{new} bằng cơ chế Tìm kiếm toàn cục của Cuckoo Search.

+ *Bước 4.2*: chọn một cá thể bất kỳ P_i population từ quần thể:

$$P_i = \begin{cases} P_{new} & if\ f(P_{new}) < f(P_i) \\ P_i & if\ f(P_{new}) \geq f(P_i) \end{cases}$$

+ *Bước 4.3*: Tính makespan của cá thể tốt nhất (*bestnest*), makespan tốt nhất của mỗi cá thể trong quần thể qua các thế hệ (*fitness*).

+ *Bước 4.4*: Thay thế *pa%* cá thể tồi nhất trong quần thể bằng những cá thể mới tìm thấy thông qua cơ chế Tìm kiếm cục bộ của Cuckoo Search.

+ *Bước 4.5*: gọi hàm Improve để cải thiện chất lượng cho cá thể tốt nhất trong quần thể. Đây là cải tiến quan trọng của GCS so với Cuckoo Search truyền thống.

- *Bước 5*: quay lại Bước 2 nếu điều kiện kết thúc lặp chưa được thỏa mãn.

Algorithm GCS
Input: dataset, t_{max} (maximum number of evolution generations)
Output: the best schedule and its makespan

1. Begin
2. Load and Valid dataset // Nạp bộ dữ liệu thực nghiệm vào
3. t ← 0
4. P_{all} ← Initial population // Khởi tạo quần thể ban đầu
5. f ← Calculate the fitness, makespan, bestnest // Tính các tham số
6. while($t<t_{max}$)
7. P_{new} ← Create a new schedule by using Lévy Flight and Global search
   // Tạo cá thể mới bằng kĩ thuật dịch chuyển Levy Flight và tìm kiếm toàn cục
8. P_i ← Take an arbitrary schedule from P_{all}
9. $P_i = \begin{cases} P_{new} & if\ f(P_{new}) < f(P_i) \\ P_i & if\ f(P_{new}) \geq f(P_i) \end{cases}$
10. P^{pa} ← pa% worst schedules in the P_{all} // thay thế pa% cá thể tồi nhất
11. For j = 1 to size(P^{pa})
12. P_{new} ← Create a new schedule by using Lévy Flight and Local search
    // thay thế pa% cá thể tồi nhất bằng cá thể mới nhờ kĩ thuật Lévy Flight and Local search
13. $P_j^{pa} = \begin{cases} P_{new} & if\ f(P_{new}) < f\left(P_j^{pa}\right) \\ P_j^{pa} & if\ f(P_{new}) \geq f\left(P_j^{pa}\right) \end{cases}$
14. End For
15. P_{all} ← P^{pa}
16. f ← Calculate the fitness, makespan, bestnest // Tính lại các tham số
17. bestnest = Improve (bestnest) // gọi hàm cải thiện chất lượng lời giải
18. makespan = f(bestnest)// f(): hàm tính makespan của một lịch biểu cho trước
19. t = t+1
20. End while
21. return {bestnest, makespan}
22. End

2.4. Thực nghiệm

2.4.1. Hai bộ dữ liệu thực nghiệm iMOPSE và TNG

Để kiểm chứng hiệu quả của thuật toán đề xuất GCS, các thực nghiệm được tiến hành trên hai bộ dữ liệu:

- Bộ dữ liệu iMOPSE [7]: đây là bộ dữ liệu đã được công bố và sử dụng rộng rãi trong các nghiên cứu trước đây [4, 8, 11] về họ bài toán RCPSP.

- Bộ dữ liệu TNG: đây là bộ dữ liệu được chúng tôi thu thập từ các dây chuyền may công nghiệp của tập đoàn dệt may TNG [12]. Bộ dữ liệu TNG chứa thông tin từ hai hợp đồng dệt may như trong Bảng 1.

Bảng 1. Hợp đồng về sản phẩm dệt may

Stt	Mã hợp đồng	Loại sản phẩm	Số lượng sản phẩm	Số lượng tác vụ/sản phẩm
1	WE1190/1698402 Liner Buy Mar 14-F19	Áo sơ mi	33,693	71
2	FM4013/ 1536181 buy Nov 08- F19	Quần bơi	83,340	137

Bộ dữ liệu TNG bao gồm 8 tập con được lưu trữ tại [13] với các trường thông tin như sau:

- Số lượng tác vụ/sản phẩm: 71 (với các tập con TNG 1,..., TNG 4); 137 (TNG 5,..., TNG 8);

- Số lượng sản phẩm: từ 37 (với các tập con TNG 1, TNG 5) đến 45 (TNG 4, TNG 8). Trong trường hợp này, tài nguyên là công nhân của nhà máy;

- Số lượng liên hệ giữa các tác vụ (quy định thứ tự thực hiện của chúng): 1026 (TNG 1,..., TNG 4); 1894 (TNG 5,..., TNG 8);

- Số lượng kĩ năng của tài nguyên (trong trường hợp này là số bậc thợ): 6;

- Các kĩ năng và mức kĩ năng tương ứng của từng tài nguyên. Một tài nguyên (công nhân) có thể có nhiều kĩ năng;

- Thời gian nhà máy sản xuất ra sản phẩm được đo tại dây chuyền sản xuất của nhà máy: từ 296 giờ (với tập con TNG 3) đến 1174 giờ (TNG 5).

Các thực nghiệm được tiến hành với các tham số cấu hình như sau: Bộ dữ liệu: 30 tập con (Bảng 2) của tập dữ liệu iMOPSE và 8 tập con (Bảng 3) của bộ dữ liệu TNG; Số lần tiến hành thực nghiệm trên mỗi tập con dữ liệu: 50.

Thực nghiệm được tiến hành trên CPU Core i5 2.2 GHz, RAM 6GB, Windows 10 với công cụ phần mềm Matlab 2014.

2.4.2. Kết quả thực nghiệm

Để đánh giá và kiểm chứng hiệu quả, thuật toán đề xuất GCS sẽ được so sánh với thuật toán sau:

- Thuật toán GA-M [7], một trong những thuật toán hiệu quả nhất đã được đề xuất trước đây;

- Thuật toán R-CSM (Reallocate Cuckoo Search): được xây dựng từ thuật toán GCS nhưng loại bỏ hàm Improve để đánh giá những tác động tích cực hoặc tiêu cực của hàm này đối với tốc độ hội tụ và chất lượng lời giải.

Các tiêu chí so sánh bao gồm:

- Thời gian thực hiện (makespan) của lịch biểu tốt nhất mà thuật toán tìm được trong 50 lần thực hiện chương trình (cột Best trong Bảng 2 và Bảng 3).

- Giá trị trung bình của makespan tính trong toàn bộ 50 lần thực hiện chương trình (cột Avg trong Bảng 2 và Bảng 3).

- Giá trị độ lệch trung bình của makespan trong các lần thực hiện chương trình (cột Std trong Bảng 2 và Bảng 3).

Kết quả thực nghiệm trên bộ dữ liệu iMOPSE và bộ dữ liệu TNG được trình bày lần lượt trong Bảng 2 và Bảng 3.

Bảng 2. Kết quả thực nghiệm trên bộ dữ liệu iMOPSE

Tập dữ liệu con iMOPSE	GA-M			R-CSM			GCS		
	Best	***Avg***	***Std***	***Best***	***Avg***	***Std***	***Best***	***Avg***	***Std***
iMOPSE-01	463	467	3.5	454	455	1.0	450	452	1.9
iMOPSE-02	520	526	5.5	506	509	2.1	502	504	0.7
iMOPSE-03	480	485	4.3	471	473	1.5	464	467	2.5
iMOPSE-04	477	487	8.1	461	464	2.3	452	460	5.4
iMOPSE-05	449	455	3.4	430	432	1.4	429	431	1.1
iMOPSE-06	220	224	2.4	208	210	1.3	203	207	2.9
iMOPSE-07	245	253	6.3	238	240	1.6	235	237	1.5
iMOPSE-08	235	243	7.7	237	238	0.1	232	234	1.7
iMOPSE-09	240	251	9.4	222	225	2.1	214	220	5.3
iMOPSE-10	242	247	3.2	230	232	1.8	226	230	3.2
iMOPSE-11	110	115	4.8	95	101	5.2	92	97	3.5
iMOPSE-12	148	153	4.0	144	152	7.8	143	148	4.1
iMOPSE-13	119	125	3.7	119	126	6.1	117	120	2.7
iMOPSE-14	193	200	6.8	185	189	3.6	183	185	1.8
iMOPSE-15	124	131	6.5	101	110	8.9	100	101	0.3
iMOPSE-16	440	446	5.7	422	428	5.1	421	422	0.6
iMOPSE-17	468	471	2.9	466	467	0.9	463	466	2.3
iMOPSE-18	481	484	2.6	459	461	1.2	454	459	4.3
iMOPSE-19	487	498	10.9	484	485	0.4	479	484	4.8
iMOPSE-20	465	469	4.0	456	457	0.3	453	456	2.7
iMOPSE-21	222	229	6.1	175	185	9.2	170	175	4.1
iMOPSE-22	249	255	5.1	237	240	2.7	234	237	2.2
iMOPSE-23	248	254	5.7	228	232	3.1	227	228	0.8
iMOPSE-24	317	321	3.3	288	293	4.9	283	288	4.8
iMOPSE-25	234	241	6.2	232	234	1.5	228	232	3.9
iMOPSE-26	126	134	7.4	110	119	8.1	106	110	3.6
iMOPSE-27	160	164	3.8	123	129	5.7	122	123	0.3
iMOPSE-28	133	146	12.1	121	132	10.3	117	121	3.4
iMOPSE-29	130	133	3.0	112	125	12.6	107	112	4.9
iMOPSE-30	133	139	5.8	117	127	9.1	114	117	2.0

Bảng 3. Kết quả thực nghiệm trên bộ dữ liệu TNG

Tập dữ liệu con TNG	GA-M			R-CSM			GCS		
	Best	**Avg**	**Std**	**Best**	**Avg**	**Std**	**Best**	**Avg**	**Std**
TNG1	201	203	3.5	131	136	4.5	129	133	3.3
TNG2	198	205	8.3	133	138	5.2	132	137	4.5
TNG3	212	218	7.1	132	135	1.6	130	132	1.7
TNG4	176	187	11.3	127	134	7.6	126	133	6.9
TNG5	751	757	6.2	572	577	4.1	563	568	4.3
TNG6	791	796	5.5	626	636	8.5	607	613	6.2
TNG7	810	820	10.7	569	573	5.7	556	560	3.8
TNG8	720	728	9.4	560	564	3.6	543	545	1.6

Các kết quả thực nghiệm được tóm tắt và tổng kết như sau:

- Với bộ dữ liệu iMOPSE:

+ Xét về tiêu chí "Thời gian thực hiện của lịch biểu tốt nhất (Best)": thuật toán đề xuất GCS vượt trội so với GA-M từ 1,07% đến 30,89% và vượt R-SCM từ 0,02% đến 4,46%. So sánh chi tiết được minh họa trong Hình 3. Kết quả này cũng cho thấy hiệu quả mà hàm Improve mang lại là đáng kể.

+ Xét về tiêu chí "Giá trị trung bình (Avg)": thuật toán đề xuất GCS vượt trội so với GA-M từ 1,0% đến 33,34% và vượt R-SCM từ 0,02% đến 10,40%. So sánh chi tiết được minh họa trong Hình 4.

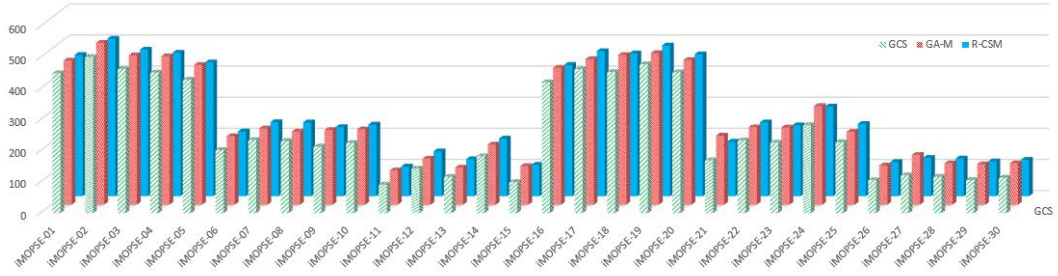
+ So sánh về tiêu chí "Độ lệch chuẩn (Std)": độ lệch chuẩn của GA-M, R-CSM và GCS lần lượt là 164,2, 121,9 và 83,3. Kết quả này cho thấy tính ổn định của thuật toán GCS tốt hơn hai thuật toán đối chứng là R-CSM và GA-M (Hình 5).

- Với bộ dữ liệu TNG:

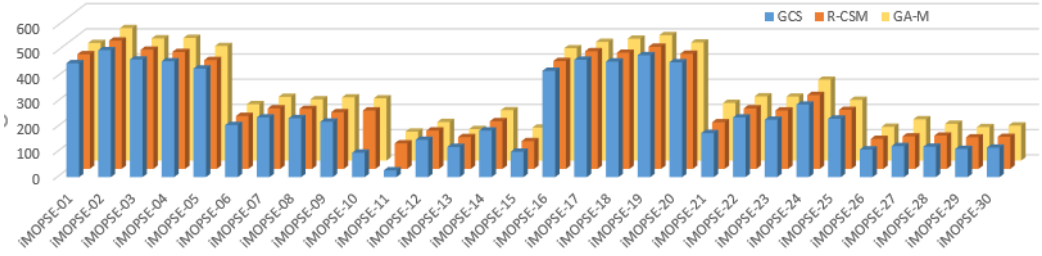
+ Xét về tiêu chí "Thời gian thực hiện của lịch biểu tốt nhất (Best)": GCS vượt trội hơn GA-M từ 23,26% đến 38,68%, tốt hơn R-CSM từ 0,5 % đến 2,4%. So sánh chi tiết được minh họa trong Hình 6.

+ Xét về tiêu chí "Giá trị trung bình (Avg)": thuật toán đề xuất GCS vượt trội so với GA-M từ 23,14% đến 40,56% và vượt R-SCM từ 0.5% đến 2.9%. So sánh chi tiết được minh họa trong Hình 7.

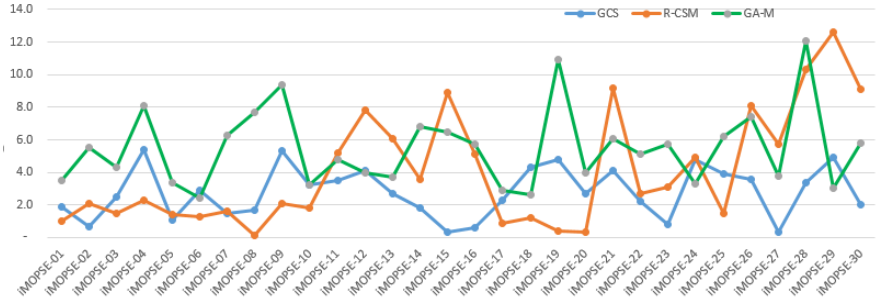
+ So sánh về tiêu chí "Độ lệch chuẩn (Std)": độ lệch chuẩn của GA-M, R-CSM và GCS lần lượt là 62.0, 40.8 và 32.3. Kết quả này vẫn cho thấy tính ổn định của thuật toán GCS tốt hơn hai thuật toán đối chứng là R-CSM và GA-M (Hình 8).



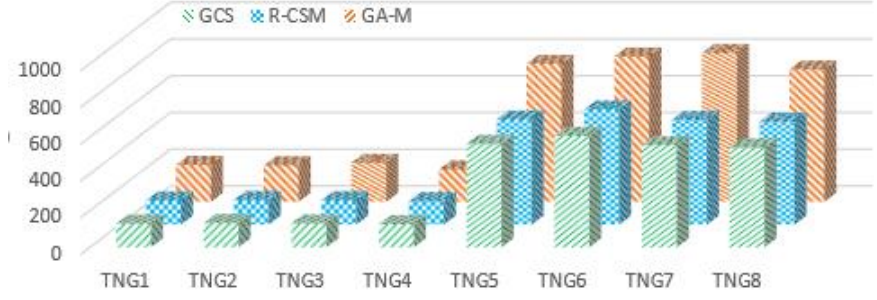
Hình 3. So sánh về tiêu chí "Thời gian thực hiện của lịch biểu tốt nhất" (bộ dữ liệu iMOPSE)



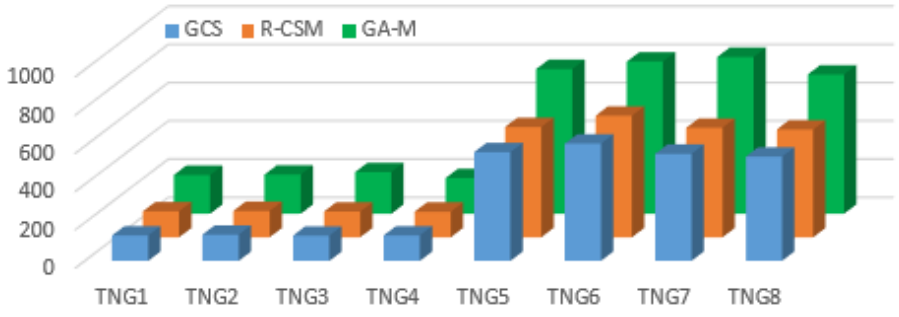
Hình 4. So sánh về tiêu chí "Giá trị trung bình (Avg)" (bộ dữ liệu iMOPSE)



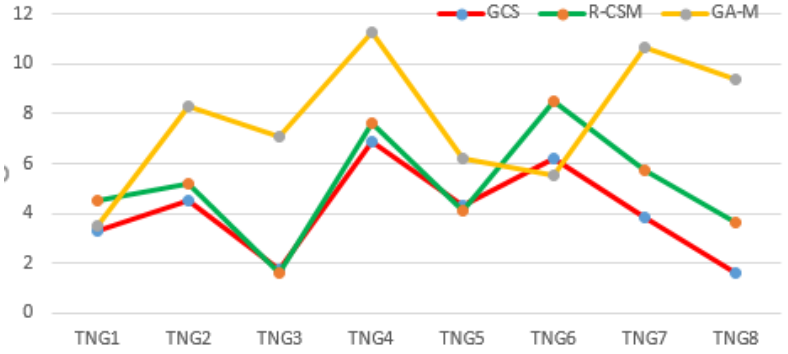
Hình 5. So sánh về tiêu chí "Độ lệch chuẩn (Std)" (bộ dữ liệu iMOPSE)



Hình 6. So sánh về tiêu chí "Thời gian thực hiện của lịch biểu tốt nhất (Best)" (bộ TNG)



Hình 7. So sánh về tiêu chí "Giá trị trung bình (Avg)" (bộ dữ liệu TNG)



Hình 8. So sánh về tiêu chí "Độ lệch chuẩn (Std)" (bộ dữ liệu TNG)

3. Kết luận

Bài báo này trình bày bài toán mới TDOS-RCPSP, một bài toán tối ưu tổ hợp được ứng dụng rộng rãi trong khoa học và thực tiễn. Bài báo đã phát biểu và trình bày bài toán TDOS-RCPSP dưới dạng mô hình toán học, sau đó đề xuất thuật toán tiến hóa GCS để tìm lời giải gần đúng. Được xây dựng dựa trên cơ chế tiến hóa Cuckoo Search, chúng tôi đã bổ sung thêm hàm Improve nhằm cải thiện chất lượng lời giải cho cá thể tốt nhất của mỗi thế hệ.

Để đánh giá hiệu quả của thuật toán đề xuất GCS, chúng tôi đã tiến hành thực nghiệm trên hai bộ dữ liệu iMOPSE và TNG. Kết quả thực nghiệm cho thấy hiệu năng của thuật toán đề xuất vượt trội so với thuật toán đối chứng GA-M về thời gian thực hiện của lịch biểu tốt nhất và giá trị trung bình. Điều này cho thấy chất lượng lời giải của thuật toán đề xuất tốt hơn so với thuật toán đối chứng. Về tiêu chí độ lệch chuẩn, thuật toán đề xuất cũng cho thấy hoạt động ổn định hơn so với thuật toán đối chứng trên cả hai bộ dữ liệu.

Trong thời gian tới, tác giả sẽ tiếp tục thu thập các bộ dữ liệu thực tế trong những lĩnh vực sản xuất có mô hình dữ liệu khác, chẳng hạn như mô hình dữ liệu Khả phân (Divisble Workload), từ đó hy vọng bổ sung thêm những tính năng giúp thuật toán đề xuất có thể thích nghi và hoạt động hiệu quả trên nhiều mô hình dữ liệu khác nhau.

TÀI LIỆU THAM KHẢO

[1] R.V. Eynde, M. Vanhoucke, 2022. A Theoretical Framework for Instance Complexity of the Resource-Constrained Project Scheduling Problem, *Mathematics of Operation Research*, DOI: https://doi.org/10.1287/moor.2021.1237.

[2] A.H. Hosseinian, V. Baradaran, 2019. An Evolutionary Algorithm Based on a Hybrid Multi-Attribute Decision Making Method for the Multi-Mode Multi-Skilled Resource-Constrained Project Scheduling Problem. *Journal of Optimization in Industrial Engineering*, Vol. 12.2, pp. 155-178.

[3] A.H. Hosseinian, V. Baradaran, 2019. Detecting communities of workforces for the multi-skill resource-constrained project scheduling problem: A dandelion solution approach. *Journal of Industrial and Systems Engineering*, pp. 72-99, Vol. 12.

[4] A.H. Hosseinian, V. Baradaran, 2020. P-GWO and MOFA: two new algorithms for the MSRCPSP with the deterioration effect and financial constraints (case study of a gas treating company). *Applied Intelligence*, Vol. 50, pp. 2151-2176.

[5] H. Li, K. Womer, 2016. Stochastic Resource-Constrained Project Scheduling and Its Military Applications. *IEEE Trans Computer*, Vol. 65(12), pp. 3702-3712.

[6] H. Najafzad, H. Davari-Ardakani, R. Nemati-Lafmejani, 2019. Multi-skill project scheduling problem under time-of-use electricity tariffs and shift differential payments. *Energy Journal*, Elsevier, Vol. 168, pp. 619-636.

[7] P.B. Myszkowski, M. Laszczyk, I. Nikulin, M. Skowro, 2019. iMOPSE: a library for bicriteria optimization in Multi-Skill Resource-Constrained Project Scheduling Problem. *Soft Computing Journal*, Vol. 23: 32397.

[8] P.B. Myszkowski, M. Skowroński, L. Olech, K. Oślizło, 2015. Hybrid Ant Colony Optimization in solving Multi–Skill Resource–Constrained Project Scheduling Problem. *Soft Computing Journal*, Volume 19, Issue 12, pp. 3599-3619.

[9] S. Javanmard, B. Afshar-Nadjafi, S.T. Niaki, 2017, Preemptive multi-skilled resource investment project scheduling problem: Mathematical modeling and solution approaches. *Computers & Chemical Engineering*, Vol. 96, pp. 55-68.

[10] X. Yang, 2020, Nature-Inspired Computation and Swarm Intelligence: Algorithms, Theory and Applications, *Elsevier*, ISBN 978-0-12-819714-1, DOI: https://doi.org/10.1016/ C2019-0-00628-0.

[11] P.B.Myszkowski, M.Laszczyk, 2022, Investigation of benchmark dataset for many-objective Multi-Skill Resource Constrained Project Scheduling Problem, *Applied Soft Computing*, Vol. 127, https://doi.org/10.1016/j.asoc.2022.109253.

[12] TNG Investment and Trading Joint Stock Company, 434/1 Bac Kan street - Thai Nguyen city, Viet Nam; https://tng.vn/trang-chu#.

[13] TNG dataset: https://github.com/huudqtmu/dataset/blob/main/TNGDEF.zip.

[14] L.M. Verburgt, 2020. The First Random Walk: A Note on John Venn's Graph. *The Mathematical Intelligencer*, Vol 15.

ABSTRACT

Evolutionary Algorithm for the Novel RCPSP Scheduling Problem

Nguyen The Loc [1], Dang Quoc Huu [2] and Vu Thai Giang[1]

[1]*Faculty of Information Technology, Hanoi National University of Education*

[2]*Faculty of Economic Information System and E-Commerce, Thuongmai University*

This paper proposes and states a novel problem called TDOS-RCPSP, a general case of the classical RCPSP problem. TDOS-RCPSP problem has many applications in science, especially in industrial production lines such as car assembly and textiles. In this study, the TDOS-RCPSP problem is proved to be NP-Hard. Besides, like other scheduling problems, TDOS-RCPSP will be classified and represented by Graham notation. Since the TDOS-RCPSP is NP-hard, a new evolutionary algorithm based on the Cuckoo Search strategy is proposed to find near-optimal schedules in a reasonable computation time. To evaluate the proposed algorithm, we perform an evaluation study using two datasets containing the iMOPSE dataset, the datasets of previous algorithms, and the TNG industrial sewing dataset. The experimental results show that the proposed algorithm has better performance than previous algorithms.

Keywords: resource constrained project scheduling, evolutionary algorithm, Cuckoo Seach algorithm.