VẬT LIỆU & XÂY DỰNG
Tạp chí Vật liệu và Xây dựng
ISSN 1859 - 381X
Website: www. jomc.vn

Tối ưu bố trí cơ sở vật chất trên mặt bằng công trình xây dựng sử dụng thuật toán lai ghép kiến sư tử (ALO)

Phạm Vũ Hồng Sơn[1*], Tô Diệu Lâm[2]

[1] Khoa kỹ thuật xây dựng, trường Đại học bách khoa thành phố Hồ Chí Minh

[2] Khoa kỹ thuật xây dựng, trường Đại học bách khoa thành phố Hồ Chí Minh

TỪ KHOÁ

Thuật toán kiến sư tử ALO
Bài toán phân công bậc hai
Lập kế hoạch bố trí mặt bằng xây dựng
Bố trí mặt bằng
Thuật toán tối ưu bầy đàn

TÓM TẮT

Lập kế hoạch bố trí mặt bằng xây dựng (CSLP) là vấn đề quan trọng trong việc quản lý xây dựng. Các cơ sở vật chất bị xung đột mục đích sử dụng trong không gian có sẵn trên mặt bằng xây dựng, làm tăng luồng vận chuyển cơ sở vật chất không hiệu quả và đó là nguyên nhân chính dẫn đến mất năng suất hoạt động, làm tăng chi phí xây dựng dự án. Cho nên việc lập, bố trí cơ sở vật chất được xác định vào các vị trí thích hợp để tìm ra một giải pháp tối ưu trong không gian mặt bằng có sẵn là vấn đề cần giải quyết dựa trên phương pháp Quadratic Assignment Problems (QAP). Trước đây từng có nhiều cách để giải quyết vấn đề QAP bằng phương pháp meta-heuristic như (GA), (MIP), (ABC). Tuy nhiên mỗi phương pháp đều có ưu và nhược điểm riêng. Vì vậy, bài nghiên cứu này đề xuất một thuật toán mới lai ghép giữa phương pháp đột biến và trao đổi chéo, phương pháp lựa chọn cạnh tranh (Tournament selection), phương pháp học dựa trên sự đối diện (Opposition-based learning) với thuật toán cải tiến (IALO) dựa trên thuật toán Ant Lion Optimizer (ALO) để giải quyết vấn đề QAP tối ưu hóa bố trí cơ sở vật chất trên mặt bằng xây dựng tìm ra một kết quả tối ưu nhất trong khoảng thời gian ngắn nhất. Kết quả qua bảng so sánh đánh giá đã cho thấy rằng thuật toán lai ghép cải tiến mới có kết quả tốt hơn so với các thuật toán trước đây như thuật toán di truyền (GA), (MIP), và thuật toán gốc (ALO), kết quả cho thấy thuật toán mới đã vượt trội hơn về tốc độ hội tụ, kết quả tìm kiếm với độ chính xác cao hơn các thuật toán nghiên cứu giải quyết vấn đề QAP trước đây.

KEYWORDS

Ant Lion Optimizer (ALO)
Quadratic Assignment problems (QAP)
Construction site layout planning
Site layout
Particle swarm optimization algorithm

ABSTRACT

Construction site layout planning (CSLP) problem is important in construction management. Facilities have conflicting purposes in the available space on the construction site, increasing the inefficient transportation of facilities. This is the main cause leading to the loss of operational productivity and increasing project construction costs. Therefore, the facilities planning and layout to be established in the appropriate locations to find an optimal solution in the available space is a problem to be solved using quadratic assignment problems (QAP) method. In the past, there were several ways to solve the QAP problem using metaheuristic methods such as genetic algorithm (GA),mixed integer programming (M.I.P), and artificial bee colony. However, each method has both advantages and disadvantages. Therefore, this study proposes a new algorithm that combines mutation, crossover, tournament selection (TS), and opposition-based learning with improved ant lion optimization based on ant lion optimizer (ALO) to solve the QAP problem of optimizing facilities layout on the construction site to find the most optimal results in the shortest time. The comparison results in the research paper belowhave shown that the new, improved hybrid algorithm outperformed previous algorithms such as the GA, MIP, and original ALO algorithm.

1. Giới thiệu

Bố trí công trình xây dựng (Contruction site layuot problems – CLSP)[1], bố trí các cơ sở vật chất kỹ thuật công trình để phục vụ trong quá trình thi công ngoài công trường và là vấn đề không thể thiếu trong công tác quản lý xây dựng. Đặc biệt như các công trình xây dựng công nghiệp, công trình thi công nhà cao tầng có các hoạt động cơ sở vật chất dồi dào, không gian thi công được bố trí thuận tiện thì các công tác sec được tiếp nối nhau suôn sẻ. Ví dụ như hệ thống giao thông giữa các công trình tạm, khu tập kết vật tư, thiết bị cơ giới, phòng ban chỉ huy, phòng thiết kế, lán trai công nhân viên, nhà bảo vệ, cổng ra vào công trình.

Vấn đề cơ sở vật chất trên mặt bằng ở các lĩnh vực, môi trường khác như trong môi trường sản xuất, thiết kế tổng mặt bằng bố trí cơ sở vật chất phù hợp thì hệ thống dịch vụ sẽ phản ứng linh hoạt đối với sự thường xuyên thay đổi không gian trong quy trình sản xuất [2].

* Liên hệ tác giả: pvhson@hcmut.edu.vn
Nhận ngày 21/10/2022, giải trình ngày 15/11/2022, chấp nhận đăng 02/12/2022
Link DOI: https://doi.org/10.54772/jomc.06.2022.468

Trong môi trường kho, bãi chứa, vật chất được bố trí kém sẽ dẫn đến tồn kho, quá tải việc sử lý vật liệu trong quá trình làm việc, và hàng hóa chờ lâu. Việc lập kế hoạch bố trí mặt bằng hiệu quả sẽ tiết kiệm được khoảng 10% đến 30% trong tổng 50% chi phí sản xuất [3]. Cho thấy rằng việc bố trí mặt bằng rất quan trọng trong sản xuất, ảnh hưởng truejc tiếp đến kết quả kinh doanh về chi phí, thời gian kết thúc quy trình và năng suất [4]. Bài toán bố trí cơ sở vật chất là một bài toán lý thuyết tính toán có độ phức tạp cao (nondeterminitic polynimial-hard). NP-khó khi NP đầy đủ L quy về H trong đa thức [5] bài toán không được giải quyết trong thời gian ngắn, độ phức tạp có thể tăng lên theo cấp số nhân với số luojng n cân được giải quyết. Ví dụ như bố trí 20 cơ sở vật chất thí số lượng lựa chon là 20! = 2,432902e + 18 [6]. Bài toán cơ sở được xây dụng đưa về bài toán phân công bậc hai (QAP) (Quadratic assignment problem) [7] là có hai vấn đề không giải quyết trong việc phân bổ nguồn lực . Cả hai vấn đề này được chỉ định sẵn vị trí trên mặt bằng, công thức giải quyết vấn đề này được đề xuất bởi Koopmans và Beckmann năm 1957, ngoài ra cũng có các phương pháp khác như quy hoạch động [8], phương pháp nhánh cận[9], quy hoạch nguyên tuyến tính [10].

Ngày nay phương pháp giải quyết bằng thuật toán (meta-heuristic) được các nhà khoa học lựa chọn để giải quyết vấn đề rợi rạc trong (QAP), các phương pháp này tìm ra lời giải tối ưu bằng các tìm kiếm và khám phá. Một số thuật toán cũ và thuật toán được cải tiếng được sử dụng rộng rãi như: (Genetic algorithm_GA)[11], (particle swarm optimization_PSO)[12], (artificial bee colony_ABC)[13]. Các nghiên cứu trước đây thường sử dụng (GA), để giải quyết vấn đề rời rạc trong (QAP), tuy nhiên các nghiên cưu trước đây thường bị hạn chế lập đi lập lại các thuật toán cũ, thuật toán mới cải tiến chưa khắc phục triệt để các hạn chế của thuật toán cũ, dữ liệu đầu vào xử lý cồng kềnh và hạn chế so sanh kết quả với các thuật toán khác.

Bài nghiên cứu này áp dụng thuật toán kiến sư tư (ALO) được giới thiệu vào năm 2015 bởi nhà toán học Mirjalili [14]. Đến nay, thuật toán (ALO) đã từng giải quyết được nhiều vấn đề trong các lĩnh vực như nguồn điện, khoa học máy tính, ứng dụng y tế, mạng không dây, kỹ thuật điều khiển và các lĩnh vực khác nhưng là hoàn toàn mới trong giải quyết vấn đề bố trí công trình xây dựng (CLSP), được mô phỏng như (QAP). Theo các nghiên cứu trước đây cho thấy thuật toán (ALO) phù hợp giải quyết các vấn đề tối ưu, có nhiều ưu điểm như thông số đầu vào ít, hội tụ cao, có thể mở rộng, cân bằng giữa thăm dò và khai thác. Tuy nhiên (ALO) có khuyết điểm tối ưu muộn hơn các thuật toán khác không có nhiều kết quả tối ưu chính xác. Đến nay đã có hơn 10 tạp chí đăng bài báo thuật toán (ALO) trong đó có nhiều bài nghiên cứu nâng cấp thuật toán (ALO)[15], như Emary và cộng sự [15] đã đề xuất 3 biến thể nhị phân của ALO sử dụng 2 cách tiếp cận (bALO)[16], Kumar và cộng sự [17] cải tiến bánh xe roulette thay thế bằng phương trình cập nhật kiến (IAO), Subhashini và Satapathy[18] giới thiệu một phiên bản nâng cao của ALO (e-ALO) áp dụng một hàm ngẫu nhiên để tạo ra số ngẫu nhiên giữa [0,1] thay vì hàm phân phối đồng nhất. Bài nghiên cứu này đề xuất cải tiến thuật toán (ALO-MOLT) bằng việc lai ghép thuật toán cải tiến (IALO) [19] với phương pháp Mutation and crossover strategy (MC) [20], Opposition-based learning (OBL) [21], và phương pháp cạnh tranh Tournament selection (TS) [22] từ thuật toán gốc (ALO) để khắc phục các hạn chế và tăng tốc hội tụ cho ra kết quả tối ưu trong khoảng thời gian ngắn nhất.

2. Phương pháp nghiên cứu

2.1. Thuật toán kiến sư tử (ALO)

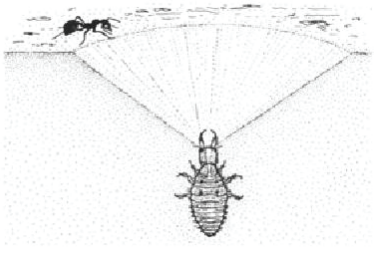
2.1.1. Cảm hứng của thuật toán

Thuật toán tối ưu kiến sư tử - AntLion Optimizer (ALO) được đề xuất bởi nhà toán học Seyedali Mirjalili. Thuật toán ALO bắt chước hành vi thông minh của loài kiến sư tử (antlion) trong việc săn kiến trong tự nhiên. Năm bước chính trong việc săn mồi gồm: *(i) Sự di chuyển ngẫu nhiên của các con kiến (con mồi),(ii) xây bẫy,(iii) nhốt kiến trong bẫy,(iv) bắt mồi,(v) xây dựng lại bẫy.*

Kiến sư tử có hành vi săn mồi độc đáo với con mồi yêu thích của chúng. AntLion sẽ đào một cái hố hình nón trên cát bằng cách di chuyển theo một đường tròn và ném cát ra ngoài bằng chiếc hàm khổng lồ của nó. Hình 1 cho thấy một số hố hình nón với các kích cỡ khác nhau và hình dạng kiến sư tử - AntLion. Sau khi đào bẫy AntLion ẩn bên dưới đáy hình nón (như một loài săn mồi ngồi chờ) và chờ côn trùng (tốt nhất là kiến) bị nhốt trong hố. Hình 2

Hành vi thú vị khác là sự liên quan giữa kích thước của hố bẫy và hai yếu tố: *(i) mức độ đói và (ii) hình dạng của mặt trăng.* Antlions có xu hướng đào ra những cái bẫy lớn hơn khi chúng trở nên đói. Cảm hứng chính của thuật toán ALO đến từ hành vi tìm kiếm thức ăn của kiến sư tử antlion.

Hình 1. Hố bẫy hình nón do kiến sư tử antLion đào [23]



Hình 2. Kiến sư tử antlion chờ con mồi (kiến) trong hố bẫy [23].

2.1.2. Mô hình toán học

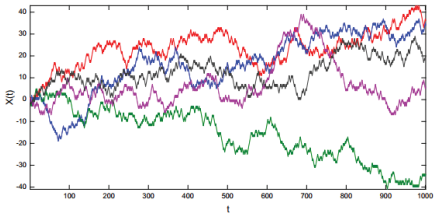
Kiến sư tử đi săn mồi bắt đầu với các bước đi ngẫu nhiên trong không gian tìm kiếm, thể hiện qua phương trình sau:

$$X(t) = [0, cumsum(2r(t_1) - 1), cumsum(2r(t_2) - 1), \dots, cumsum(2r(t_n) - 1)] \quad (1)$$

Trong đó X (t) đại diện cho lần đi bộ ngẫu nhiên, n là số lần lặp tối đa, t đại diện cho lần lặp hiện tại, cumsum đề cập đến tổng tích lũy và r (t) đề cập đến một hàm ngẫu nhiên được định nghĩa như sau:

$$r(t) = \begin{cases} 1 & if \quad rand > 0{,}5 \\ 0 & if \quad rand \le 0{,}5 \end{cases} \tag{2}$$

Trong đó t chỉ ra bước đi ngẫu nhiên (tính lặp trong nghiên cứu này) và *rand* là một số ngẫu nhiên được tạo ra với phân phối đồng nhất trong khoảng [0,1]. Hình 3 biểu diễn minh họa ba lần di chuyển ngẫu nhiên trên 1000 lần lặp. Hình này cho thấy bước đi ngẫu nhiên dao động đáng kể xung quanh vị trí gốc, có xu hướng tăng hoặc có xu hướng giảm dần (đường cong màu xanh).



Hình 3. Kiến sư tử antlion chờ con mồi (kiến) trong hố bẫy [24].

Thuật toán (ALO) dựa trên số lượng kiến và lưu trữ kiến trong ma trận sau:

$$M_{Ant} = \begin{bmatrix} A_{1,1} & A_{1,2} & \dots & \dots & A_{1,d} \\ A_{2,1} & A_{2,2} & \dots & \dots & A_{2,d} \\ : & : & : & : & : \\ : & : & : & : & : \\ A_{n,1} & A_{n,2} & \dots & \dots & A_{n,d} \end{bmatrix} \tag{3}$$

$$M_{Antlion} = \begin{bmatrix} AL_{1,1} & AL_{1,2} & \dots & \dots & AL_{1,d} \\ AL_{2,1} & AL_{2,2} & \dots & \dots & AL_{2,d} \\ : & : & : & : & : \\ : & : & : & : & : \\ AL_{n,1} & AL_{n,2} & \dots & \dots & AL_{n,d} \end{bmatrix} \tag{4}$$

trong đó n là số lượng kiến trong quần thể. M_{Ant} là ma trận lưu vi trí các con kiến mồi, $A_{i,j}$ cho thấy giá trị của chiều thứ *j* của con kiến thứ *i*, *n* là số lượng con kiến mồi $M_{Antlion}$ là ma trận lưu vị trí kiến sư tử, $AL_{i,j}$ biểu thị giá trị chiều thứ *j* của antlion thứ *i*, *n* là số con kiến sư tử, *d* là số lượng chiều.

Để đánh giá sức mạnh của từng con kiến mồi và kiến sư tử, một hàm mục tiêu được sử dụng và lưu trữ trong ma trận sau:

$$M_{OA} = \begin{bmatrix} f([A_{1,1} \ A_{1,2} \ \dots \ \dots \ A_{1,d}]) \\ f([A_{2,1} \ A_{2,2} \ \dots \ \dots \ A_{2,d}]) \\ : \\ : \\ f([A_{n,1} \ A_{n,2} \ \dots \ \dots \ A_{n,d}]) \end{bmatrix} \tag{5}$$

$$M_{OAL} = \begin{bmatrix} f([AL_{1,1} \ AL_{1,2} \ \dots \ \dots \ AL_{1,d}]) \\ f([AL_{2,1} \ AL_{2,2} \ \dots \ \dots \ AL_{2,d}]) \\ : \\ : \\ f([AL_{n,1} \ AL_{n,2} \ \dots \ \dots \ AL_{n,d}]) \end{bmatrix} \tag{6}$$

trong đó M_{OA} là ma trận lưu giữ hàm thích nghi của từng con kiến, $A_{i,j}$ cho thấy giá trị của chiều thứ *j* của con kiến thứ *i*, *n* là số lượng con kiến mồi. M_{OAL} là ma trận để lưu giữ hàm thích nghi của từng kiến sư tử, $AL_{i,j}$ biểu thị giá trị chiều thứ *j* của antlion thứ *i*, *n* là số con kiến sư tử, *d* là số lượng chiều và *f* là hàm mục tiêu.

Chuẩn hóa vi trí của kiến, giữ cho kiến đi lại ngẫu nhiên trong không gian tìm kiếm bằng phương trình sau:

$$X_i^t = \frac{(X_i^t - a_i) \times (d_i^t - c_i^t)}{(b_i - a_i)} + c_i^t \tag{7}$$

trong đó i là chỉ số biến, t là số lặp, a_i là bước ngẫu nhiên nhỏ nhất, b_i là giá trị bước ngẫu nhiên lớn nhất, c^t_i là giá trị biến nhỏ nhất và d^t_i là giá trị biến lớn nhất của vị trí kiến sư tử được cập nhật tại mỗi lần lặp như theo sau.

Khi những con kiến mồi rơi vào bẫy của kiến sư tử thì theo bản năng muốn thoát ra khỏi đó. Con kiến sư tử bắt đầu ném cát vào những con kiến cản trở việc chạy trốn của chúng và đưa chúng về phía đáy hố. Được thể hiện qua các phương trình (8), (9), (10), (11), (12).

$$c_i^t = Antlion_i^t + c^t \tag{8}$$

$$d_i^t = Antlion_i^t + d^t \tag{9}$$

$$c^t = \frac{c^t}{I} \tag{10}$$

$$d^t = \frac{d^t}{I} \tag{11}$$

$$I^t = 10^w \frac{t}{T} \tag{12}$$

trong đó c^t, d^t lần lượt là giá trị nhỏ nhất và lớn nhất của tất cả các biến tại lần lặp thứ t, c^t_i; d^t_i là giá trị nhỏ nhất và tối đa của tất cả các biến đối với kiến thứ i tương ứng, $Antlion^t_j$ cho biết vị trí kiến sư tử thứ j tại lần lặp thứ t, và I là tỷ lệ trượt thay đổi như trong phương trình sau.

$$\mathrm{I} = \begin{cases} 1 + 10^6 \frac{t}{Tmax} & \text{if } 0{,}95\mathrm{Tmax} < \mathrm{t} < \mathrm{Tmax} \\ 1 + 10^5 \frac{t}{Tmax} & \text{if } 0{,}9\mathrm{Tmax} < \mathrm{t} < 0{,}95\mathrm{Tmax} \\ 1 + 10^4 \frac{t}{Tmax} & \text{if } 0{,}75\mathrm{Tmax} < \mathrm{t} < 0{,}9\mathrm{Tmax} \\ 1 + 10^3 \frac{t}{Tmax} & \text{if } 0{,}5\mathrm{Tmax} < \mathrm{t} < 0{,}75\mathrm{Tmax} \\ 1 + 10^2 \frac{t}{Tmax} & \text{if } 0{,}1\mathrm{Tmax} < \mathrm{t} < 0{,}5\mathrm{Tmax} \\ 1 & \text{otherwise} \end{cases} \tag{13}$$

trong đó *Tmax* là lần lặp tối đa. Các con kiến mồi đi bộ và được định vị xung quanh kiến sư tử ưu tú và kiến sư tử được chọn theo phương pháp bánh xe roulette với mô hình toán học sau.

$$Ant_i^t = \frac{R_A^t + R_E^t}{2} \tag{14}$$

trong đó R^t_A là kiến sư tử được chọn bằng phương pháp roulette wheel ở lần lập thứ t và R^t_E là kiến sư tử ưu tú thu được bằng phương trình (7) cho mỗi lần lặp thứ t. Sau khi săn mồi, những con kiến sư tử cập nhật vị trí của chúng với vị trí của những con kiến mồi theo fitness value bằng phương trình sau:

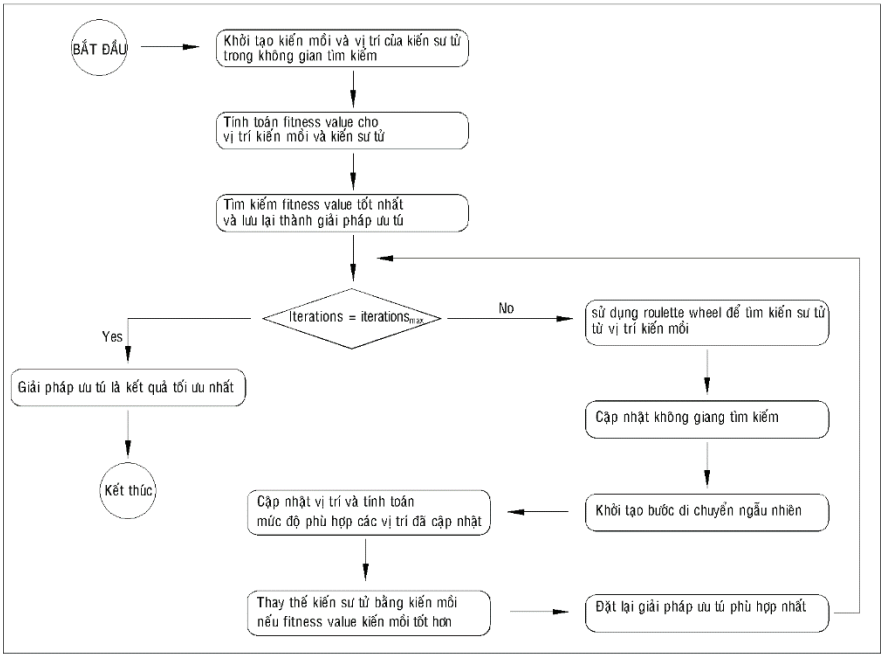
$$Antlion_i^t = Ant_i^t \text{ nếu } f(Ant_i^t) < f(Antlion_i^t) \tag{15}$$

trong đó f là hàm thích nghi (fitness value).

2.2. Cải tiến thuật toán kiến sư tử (IALO) [26]

Giảm kích thước đi bộ ngẫu nhiên trong quần thể kiến mồi đoạn đầu thuật toán (ALO), hiệu xuất thăm dò và khai thác của thuật toán (ALO) dưới 20 % trong lần lập tối đa, thay đổi 20 % số lần lập tối đa trong mô hình dđi bộ ngẫu nhiên của kiến mồi được thể hiện trong đoạn mã code sau đây:

$$X(t) = [0, \cdots, cumsum(2r(tn) - 1)], n = 1{,}2, \cdots, Max\_iter/5 \textbf{ for } every\ ant \tag{16}$$



Hình 4. Lưu đồ thuật toán ALO [25].

Cải tiến cách lựa chọn con kiến sư tử trong giai đoạn roulette wheel bằng cách tăng độ lớn (fitness value) và chọn lun giá trij không tốt (negative fitness value), vào cuối thuật toán cập nhật kiến sư tử ưu tú (elite), kết hợp tìm kiếm và sắp xếp quần thể, so sánh các cặp fitness value của kiến mồi và kiến sư tử, nếu như fitness value của kiến mồi tốt hơn fitness value của kiến sư tử thì vị trí kiến mồi sẽ được cập nhật thành kiến sư tử được thể hiện qua đoạn code sau:

Chọn kiến sư tử bằng roulette wheel để xây dựng bẫy

$$\frac{|f(Antlion_i^{-1})|}{\sum_{j=1}^{n}|f(Antlion_i^{-1})|}, i = 1,2,\dots,n \quad (17)$$

Kiến mồi trượt vào bẫy

$$\left.\begin{matrix} c_i^t = \text{Antlion}_i^t + c^t \\ d_i^t = \text{Antlion}_i^t + c^t \end{matrix}\right\} if\ 0.75 < option < 1$$
$$\left.\begin{matrix} c_i^t = \text{Antlion}_i^t - c^t \\ d_i^t = \text{Antlion}_i^t - c^t \end{matrix}\right\} if\ 0.5 < option < 0{,}75$$
$$\left.\begin{matrix} c_i^t = -\text{Antlion}_i^t + c^t \\ d_i^t = -\text{Antlion}_i^t + c^t \end{matrix}\right\} if\ 0.25 < option < 0{,}5$$
$$\left.\begin{matrix} c_i^t = -\text{Antlion}_i^t - c^t \\ d_i^t = -\text{Antlion}_i^t - c^t \end{matrix}\right\} if\ 0 < option < 0{,}25 \quad (18)$$

Cập nhật vị trí kiến.

$$Ant_i^t = \frac{R_A^{r(tn)} + R_E^{r(tn)}}{2}, r(t_n): rand\ number\ in\ [0 t_n], n = 1,2\dots, \frac{Max_{iter}}{5}$$

trường hợp kiến mồi nằm ngoài không gian tìm kiếm, thuật toán cải tiến sẽ đưa kiến mồi vào lại không gian tìm kiếm được trình bày qua đoạn code sau:

$$Antit = b_{low} + rand \times (b_{up} - b_{low})$$
$$if\ (Ant_i^t > b_{up})\ or\ (Ant_i^t < b_{low}) \quad (18)$$

end for

Tính toán fitness value của kiến mồi

so sánh fitness value kiến mồi với kiến sư tử, nếu kiến mồi tốt hơn fitness value kiến sư tử thì vị trí kiến mồi thành vị trí kiến sư tử. Nếu ngược lại thì kiến sư tử vẫn giữ nguyên vị trí

$$\text{Antlion}_i^t = \text{Ant}_i^t\ if\ f(Ant_i^t) < f(Antlion_i^t) \quad (19)$$

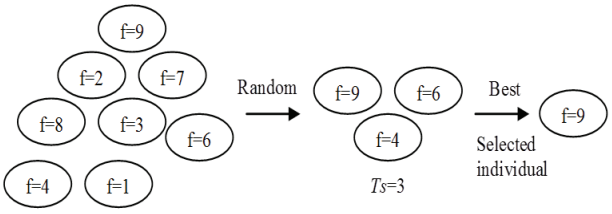
cập nhật vị trí kiến sư tử

end while

Return elite antlion

2.3. Phương pháp lựa chọn cạnh tranh - Tournament selection (TS) [27]

Tournament selection là phương pháp lựa chọn đơn giản, n cá thể được chọn trong quần thể lớn, và cho cạnh tranh với nhau, cá thể có fitness cao nhất sẽ được chọn, mỗi lần chọn thường là hai cá thể (binary tournament) và sắp xếp vào nhóm mới (Hình 5)



Hình 5. Phương pháp Selection với Tourament [27]

Trong thuật toán (ALO) việc lựa chọn kiến sư tử tại bước đi ngẫu nhiên được xác đinh lại với một hệ số sẽ làm chậm qua trình tối ưu. Phương pháp Tournament selection cập nhật lại bước đi ngẫu nhiên vị trí kiến mồi trong bẫy tại roulette wheel, viết lai qua đoạn code:

for *every ant*

Chọn kiến sư tử bằng Tournament selection để xây dựng bẫy

Kiến mồi trượt vào bẫy như (18).

trong đó, Tournament, nó chọn ngẫu nhiên 2 con kiến sư tử và chọn ra một con tốt nhất để tham gia vào nhóm.

2.4. Phương pháp đột biến và trao đổi chéo - Mutation and crossover (MC) [28]

Phương pháp này tạo ra một số biến thể của thuật toán di chuyển trong các giai đoạn của quá trình tối ưu hóa.

2.4.1. Crosover – trao đổi chéo

Thuật toán di chuyển kết hợp hai nhiễm sắc thể (bố mẹ) để tạo ra một nhiễm sắc thể mới (con) với xác xuất trao đổi chéo **Px** tạo ra cá thể mới (con) tốt hơn.

Parent A Parent B offspring

{**1 2 3 4 5** 6 7 8 9}+ {**4 5 3 6 8** 9 7 2 1} = {**1 2 3 4 5** 6 8 9 7} (20)

2.4.2. Mutation – đột biến

Sau khi trao đôi chéo, cá thể con bị đột biến, đột biến giúp ngăn thuật toán bị tối ưu cục bộ, khám phá toàn bộ không gian tìm kiếm, duy trì sự đa dạng trong quần thể .Sửa đổi các gen của nhiễm sắc thể được chọn với xác xuất đột biến **Pm.** Đảo ngược các bit của nhiễm sắc thể được chọn.

{1 **2** 3 4 5 6 **8** 9 7} => {1 **8** 3 4 5 6 **2** 9 7} (21)

2.5. Phương pháp học dựa trên sự đối diện - Opposition-based learning (OBL) [29]

Opposition-based learning (OBL) được giới thiệu bởi Tizoosh và cộng sự. [30] đảm bảo sự tăng tốc trong quá trình hội tụ bằng cách so sánh số lượng đối diện với kết quả đã được chọn và tìm ra kết quả tốt hơn để giữ lại cho lần quần thể tiếp theo để đẩy nhanh sự hội tụ. Chu kì hội tụ phụ thuộc vào khoản cách của điểm kết quả ban đầu đến điểm kết quả tối ưu.

2.5.1 Định nghĩa 1

Opposition-based learning (OBL) được Rahnamayan và cộng sự [31] khái niệm rằng không gian tìm kiếm được chuyển đổi thành không gian đối diện mới, không gian tìm kiếm một chiều có thể được mở rộng nhiều chiều. "Nếu y là điểm tìm kiếm trong không gian, trong đó $p \in [LB, UB]$, thì một vị trí tìm kiếm mới là p^* trong không gian đã biến đổi thể hiện qua công thức sau:

$$p^* = LB + UB - p \quad (22)$$

2.5.2 Định nghĩa 2:

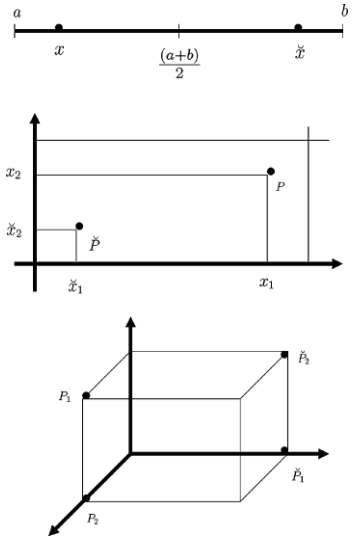
Opposition-based learning (OBL) cho thứ nguyên cao hơn (hight dimension) thì Rahnamayan và cộng sự [31] khái nệm rằng, nếu $P \in (p_1, p_2 \ldots, p_D)$ thì D là một điểm trong không gian tìm kiếm, $p_1, p_2 \ldots, p_D \in R$, trong đó R là không gian Euclidean và $p_i \in [LB_i, UB_i]$, $\forall_i \in (1, 2, \ldots, D)$ thì điểm biến đổi mới được xác định bởi công thức:

$$p_i^* = LB_i + UB_i - p_i \quad (23)$$

quần thể ban đầu và quần thể đối diện mới được đánh giá và chọn ra fitness value tốt nhất thông qua công thức

$$f(p) \leq f(p^*) \quad (23)$$

trong đó $f(p)$ xác định finess value tại điểm đầu và $f(p^*)$ xác định fitness value điểm đối diện, nếu $f(p) \leq f(p^*)$ thì y sẽ được thay thế bằng p^*, nếu không thì tiếp tuïcc tìm kiếm.



Hình 6. Hình minh họa điểm đối diện trong một, hai, và ba chiều [31].

2.6. Construction site layout problems – CSLP [1]

Bố trí mặt bằng công trình xây dựng (Construction site layout problems_CSLP) được xem mô phỏng như vấn đề bài toán phân công bậc hai (Quadratic assignment problem-QAP). Mục tiêu (QAP) tìm ra vị trí tối ưu khoảng cách cho các cơ sở vật chất (nhà bảo vệ, nhà kho, khu tập kết vật tư, etc..). Bài toán chỉ định n cở sở gán và m vị trí có sẵn được mô phỏng qua công thức sau đây [33,34]

$$min\, z = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{m=1}^{n} A_{ijkm} X_{ij} X_{km} \quad (24)$$

$$\sum_{i=1}^{n} X_{ij} = 1, j = 1, \ldots\ldots, n$$

$$X_{ij} \in \{0,1\}, i, j = 1, \ldots\ldots, n$$

trong đó:

x_{ij} = 1, nếu cơ sở i được gán cho địa điểm j, nếu không thì bằng 0

x_{km} = 1, nếu cơ sở k được gán cho địa điểm m, nếu không thì bằng 0

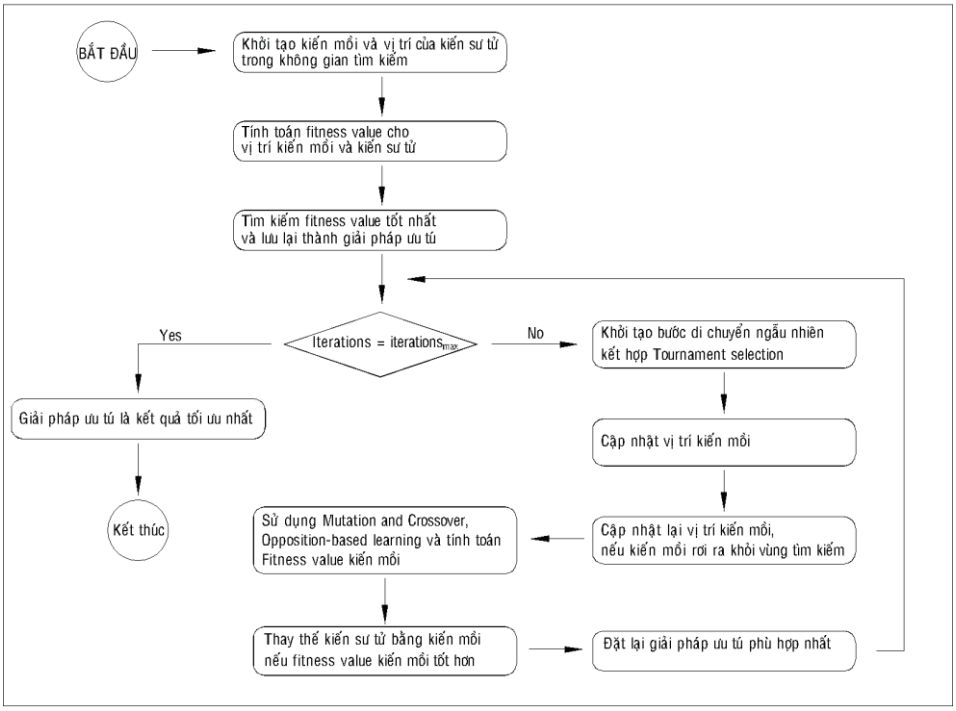
c_{ij} = chi phí cố định (vị trí) liên quan đến chỉ định cơ sở i tới địa điểm j

d_{jm} = khoảng cách giữa vị trí j và m (tỉ lệ với chi phí vận chuyển), d_{jj} = 0

f_{ik} = luồng công việc giữa các cơ sở i và k, f_{ii} = 0

A_{ijkm} = f_{ik}, d_{jm} nếu i ≠k hoặc j ≠m

A_{ijkm} = c_{ij} nếu i = k hoặc j = m



Hình 7. Lưu đồ thuật toán lai ghép ALO-MOLT.

3. Trường hợp tính toán

Bài toán này được sử dụng từ nghiên cứu của C.Huang, C.K.Wong, C.M.Tam [32], lập kế hoạch bố trí cơ sở vật chất trên kho chứa tầng hầm trong một công trình nhiều tâng, xét chuyên động ngang của kho chứa và chuyển động vật liệu theo phương đứng theo vận thăng. Qúa trình phân phối vật liệu có thể hình dung qua mặt cắt công trình xem hình 8 và mặt bằng lưu kho điển hình (Hình 9).

3.1. Hàm mục tiêu

$$Min(\sum_{j=1}^{J}\sum_{l=1}^{L}\sum_{k=1}^{K} Q_{j,m}\left(C_{j,l}^{V}+D_{l,k}C_{j}^{H}\right)x_{j,l,k}+\sum_{m=1}^{M}\sum_{j=1}^{J} Q_{j,m}(C_{j,m}^{V}-C_{j,l}^{V})\delta_{j,l,k,m} \quad (25)$$

trong đó:

j Loại vật liệu

l Tầng trong một tòa nhà để chứa vật liệu làm nguồn cung cấp

k Các ô trên các tầng của tòa nhà để lưu trữ

m Tầng trong một tòa nhà yêu cầu vật liệu

J Tổng số loại vật liệu

L Tổng số tầng lưu trữ trong một tòa nhà

K Tổng số ô trong một tầng của tòa nhà

M Tổng số cấp trong một tòa nhà

$Q_{j,m}$ Nhu cầu loại vật liệu j trên tầng m trong nhà

C^{h}_{j} Chi phí vận chuyển đơn vị theo chiều ngang của loại vật liệu j

$C^{v}_{j,l}$ Chi phí vận chuyển đơn vị dọc của vật liệu loại j lên tầng l của công trình từ mặt đất

$C^{v}_{j,m}$ Chi phí vận chuyển đơn vị dọc của vật liệu loại j đến tầng m của công trình từ mặt đất

$D_{l,k}$ Khoảng cách từ ô k đến palăng vật liệu trên tầng l

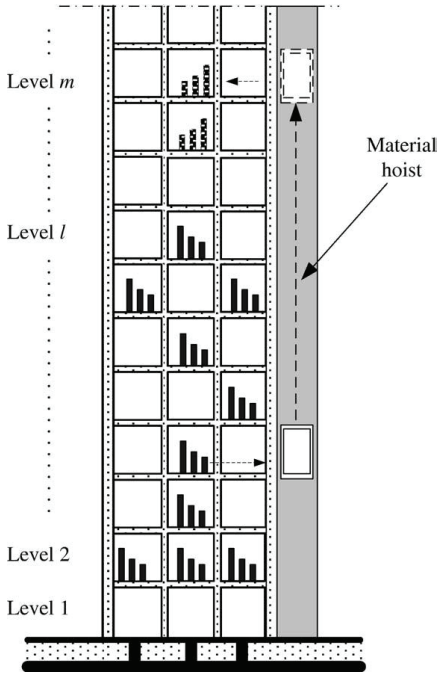
$x_{j,l,k}$ Biến quyết định nhị phân lưu trữ vật liệu j bên trong ô k ở mức l

$\Delta_{j,m}$ Nhu cầu loại vật liệu j trên cấp công trình m

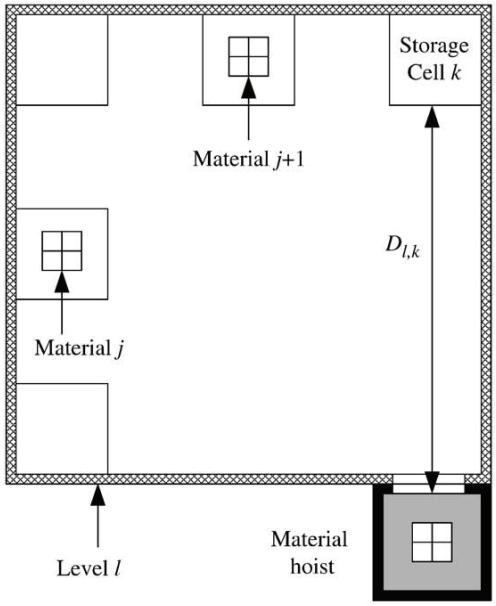
$\delta_{j,l,k,m}$ Biến kiểu nhị phân bổ trợ trong đó '1' có nghĩa là vật liệu j được chuyển từ ô tầng l k sang ô tầng m nhưng ngược lại là '0'

Bảng 1 Nhu cầu về số lượng vật liệu

Loại vật liệu, j	Tên loại vật liệu	Nhu cầu của loại vật liệu $Q_{j,m}$ (in kg)
1	Kính	2000
2	Hoàn thiện sàn	1500
3	Bê tông và xi măng	6000
4	Trần thạch cao	1500
5	Thép và nhôm	4000
6	Gỗ	3000
7	Gạch và vách ngăn	2500
8	Đèn	1000
9	Sơn màu	500
10	Dây cáp điện và dây điện	700



Hình 9. Mặt cắt qua vận thăng công trình.



Hình 10. Mặt bằng tầng l điển hình, với 5 ô chứa vật liệu.

Bảng 2. Khoảng cách theo phương ngang từ ô k trên tầng 1 đến vận thăng.

Ô k	$D_{l,k}$ (in meters)				
Tầng l	1	2	3	4	5
1	10	15	20	15	10
2	10	15	20	15	10
3	10	15	20	15	10
4	10	15	20	15	10
5	10	15	20	15	10
6	10	15	20	15	10
7	10	15	20	15	10
8	10	15	20	15	10

Bảng 3. Chi phí vận tải ngang C_j^h

Loại vật tư, j	C^H_j (in $/m/kg)
1	20,0
2	22,5
3	15,0
4	23,0
5	10,0
6	10,0
7	23,0
8	14,0
9	15,0
10	16,0

Bảng 4 Chi phí vận chuyển dọc (tính bằng USD / kg) từ mặt đất lên tầng l, $C^v_{j,l}$.

Floor,l	Material type, j									
	1	2	3	4	5	6	7	8	9	10
1	15,0	20,0	13,0	20,0	7,0	7,0	26,0	11,0	12,0	13,0
2	16,2	21,2	14,2	21,2	8,2	8,2	27,2	12,2	13,2	14,2
3	17,4	22,4	15,4	22,4	9,4	9,4	28,4	13,4	14,4	15,4
4	18,6	23,6	16,6	23,6	10,6	10,6	29,6	14,6	15,6	16,6
5	19,8	24,8	17,8	24,8	11,8	11,8	30,8	15,8	16,8	17,8
6	21,0	26,0	19,0	26,0	13,0	13,0	32,0	17,0	18,0	19,0
7	22,2	27,2	20,2	27,20	14,0	14,2	33,2	18,2	19,2	20,2
8	23,4	28,4	21,4	28,4	15,4	15,4	34,4	19,4	20,4	21,4

Bảng 5 Chi phí vận chuyển phương đứng (in $/kg) từ mặt đất tới tầng m, tầng l $C^V_{j,m}$.

Tầng m, l	Loại vật tư, j									
	1	2	3	4	5	6	7	8	9	10
1	15,0	20,0	13,0	20,0	7,0	7,0	26,0	11,0	12,0	13,0
2	16,2	21,2	14,2	21,2	8,2	8,2	27,2	12,2	13,2	14,2
3	17,4	22,4	15,4	22,4	9,4	9,4	28,4	13,4	14,4	15,4
4	18,6	23,6	16,6	23,6	10,6	10,6	29,6	14,6	15,6	16,6
5	19,8	24,8	17,8	24,8	11,8	11,8	30,8	15,8	16,8	17,8
6	21,0	26,0	19,0	26,0	13,0	13,0	32,0	17,0	18,0	19,0
7	22,2	27,2	20,2	27,2	14,2	14,2	33,2	18,2	19,2	20,2
8	23,4	28,4	21,4	28,4	15,4	15,4	34,4	19,4	20,4	21,4
9	24,6	29,6	22,6	29,6	16,6	16,6	35,6	20,6	21,6	22,6
10	25,8	30,8	23,8	30,8	17,8	17,8	36,8	21,8	22,8	23,8
11	27,0	32,0	25,0	32,0	19,0	19,0	38,0	23,0	24,0	25,0
12	28,2	33,2	26,2	33,2	20,2	20,2	39,2	24,2	25,2	26,2
13	29,4	34,4	27,4	34,4	21,4	21,4	40,4	25,4	26,4	27,4
14	30,6	35,6	28,6	35,6	22,6	22,6	41,6	26,6	27,6	28,6
15	31,8	36,8	29,8	36,8	23,8	23,8	42,8	27,8	28,8	29,8
16	33,0	38,0	31,0	38,0	25,0	25,0	44,0	29,0	30,0	31,0
17	34,2	39,2	32,2	39,2	26,2	26,2	45,2	30,2	31,2	32,2
18	35,4	40,4	33,4	40,4	27,4	27,4	46,4	31,4	32,4	33,4
19	36,6	41,6	34,6	41,6	28,6	28,6	47,6	32,6	33,6	34,6
20	37,8	42,8	35,8	42,8	29,8	29,8	48,8	33,8	34,8	35,8
21	39,0	44,0	37,0	44,0	31,0	31,0	50,0	35,0	36,0	37,0
22	40,2	45,2	38,2	45,2	32,2	32,2	51,2	36,2	37,2	38,2
23	41,4	46,4	39,4	46,4	33,4	33,4	52,4	37,4	38,4	39,4
24	42,6	47,6	40,6	47,6	34,6	34,6	53,6	38,6	39,6	40,6
25	43,8	48,8	41,8	48,8	35,8	35,8	54,8	39,8	40,8	41,8
26	45,0	50,0	43,0	50,0	37,0	37,0	56,0	41,0	42,0	43,0
27	46,2	51,2	44,2	51,2	38,2	38,2	57,2	42,2	43,2	44,2
28	47,4	52,4	45,4	52,4	39,4	39,4	58,4	43,4	44,4	45,4
29	48,6	53,6	46,6	53,6	40,6	40,6	59,6	44,6	45,6	46,6
30	49,8	54,8	47,8	54,8	41,8	41,8	60,8	45,8	46,8	47,8

Bảng 6. Thông số đầu vào trong các thuật toán.

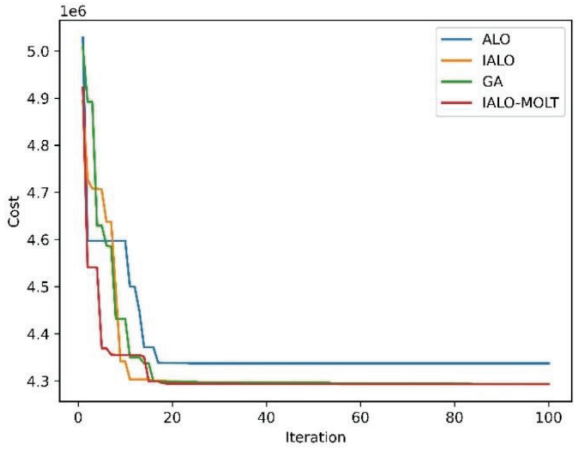
Thuật toán áp dụng	GA	ALO	IALO	IALO-MOLT
Số cá thể trong quần thể	10	10	10	10
Số vòng lặp tối đa	100	100	100	100

Cho chạy bài toán 20 lần vòng lập ba thuật toán (GA), (ALO), (IALO-MOLT) để so sánh kết quả trên phần mềm PyCharm 2022, máy HP ProBook 450 G2, kết quả cuối cùng được lấy từ lần chạy tốt nhất.

Bảng 7. Kết quả tối ưu tìm được.

	GA	ALO	IALO	IALO-MOLT
Tổng chi phí tối ưu ($)	4.295.180	4.301.340	4.293.140	4.293.020
Thời gian hoàn thành (s)	51	56	174	2

Bảng 7 cho thấy các kết quả thu được bởi thuật toán ALO-MOLT với ba thuật toán (GA), (ALO), (IALO) thông qua 30 lần chạy vòng lập thì cho ta thấy kết quả IALO-MOLT ra chi phí thấp tôi ưu nhất trong khoảng thời gian nhanh nhất co thể. Gía trị chi phí thấp nhất dduojc in trong Bảng 7.



Hình 8. So sánh kết quả và tốc độ hội tụ giữa thuật toán GA, ALO, IALO-MOLT.

Qua Hình 8 cho thấy đường cong hội tụ thu được từ thuật toán IALO-MOLT cho mỗi lần chạy có kết quả ổn định nhất cho việc giải quyết vấn đề QAP. Thuật toán IALO-MOLT xác nhận khả năng tăng tốc hội tụ cao, gia tăng sô lần lặp và tìm kiếm không gian tooanf cục tối ưu hiệu quả.

Dựa trên kết quả của Bảng 7, Thuật toán IALO–MOLT có thể đạt được kết quả tối ưu trong thời gian ngắn nhất so với các thuật toán (GA), (ALO) và (IALO) sau 20 lần lặp lại. Các đường cong hội tụ của IALO–MOLT và các thuật toán (GA), (ALO) và (IALO) được hiển thị trong hình 8 minh họa tốc độ mà thuật toán mới IALO–MOLT hội tụ và lưu trữ kết quả so với các thuật toán khác nhanh như thế nào. Thuật toán IALO–MOLT được kết hợp để khắc phục những hạn chế của thuật toán gốc (ALO). Hạn chế chính của (ALO) là vấn đề phát hiện quần thể chỉ phù hợp với quy mô vừa và nhỏ, cuối cùng là có xu hướng hội tụ về một điểm. Do đó, thuật toán IALO–MOLT được cải tiến và kết hợp với phương pháp cạnh tranh (TS), phương pháp đột biến và trao đổi chéo chéo (MC), cải tiến thuật toán (IALO) và phương pháp học dựa trên sự đối diện (OBL) từ thuật toán gốc (ALO) có thể được sử dụng để thúc đẩy tối ưu hóa cục bộ và tăng khả năng khám phá cũng như khai thác. (IALO) đã cải thiện phương trình tìm kiếm của kiến trong roulette wheel, giảm quy mô các bước đi ngẫu nhiên trong quần thể kiến và thay đổi số lần lặp lại tới 20 % trong các bước đi ngẫu nhiên của kiến. (TS) sử dụng lựa chọn cạnh tranh để tìm ra những cá thể phù hợp nhất trong một quần thể rộng lớn. (MC) tạo ra một số biến đổi mới trong quá trình tối ưu hóa và đột biến giúp ngăn chặn tối ưu cục bộ. Bằng cách so sánh kết quả quá phần tử với kết quả đã chọn (OBL) đã được chứng minh là đẩy nhanh quá trình hội tụ.

4. Kết luận và đề xuất

Trong bài nghiên cứu này đã cải tiến một vài nhược điểm trong thuật toán gốc Antlion Optimization(ALO) được lấy cảm hứng cơ chế kiến sư tử trong tự nhiên. IALO-MOLT thuật toán mới được cải tiến cơ chế đi bộ ngẫu nhiên, phương pháp lựa chọn, không gian tìm kiếm, đột biến và trao đổi chéo được thực hiện trong thuật toán (ALO). Các phương pháp đổi mới thực hiện trong quá trình kiến mồi trượt vào bẫy của kiến sư tử và được lấy kết quả so sánh với các thuật toán khác cho thấy kết quả tối ưu chi phí có độ chính xác cao. Thời gian chạy của thuật toán IALO-MOLT cũng được giảm đáng kể, tốt hơn nhiều so với các thuật toán ban đầu do những phương pháp được lai ghép vào thuật toán.

Việc áp dụng thuật toán IALO-MOLT vào giải quyết vấn đề bố trí mặt bằng xây dựng được mô phỏng như (Quadratic Assignment Problems – QAP) thu được hiệu suất tốt, tìm ra chi phí tối ưu nhất khi so sánh với các thuật toán cũ (GA), (ALO), (IALO). Nghiên cưu mở rộng ra nhiều hướng mới trong tương lai như là mở rộng thêm vấn đề để triển khai thuật toán mới IALO-MOLT xử lý, như lập kế hoạch vận chuyên kho bãi tối ưu chi phí, đa mục tiêu chi phí logictic, các vấn đề xử lý hình ảnh, kỹ thuật xây dựng, y tế.

Tài liệu tham khảo

[1]. S. Jajodia, I. Minis, G. Harhalakis, and J.-M. Proth, 1992. CLASS: Computerized Layout Solutions using Simulated annealing. International Journal of Production Research, Volume 30, pp. 95-108.

[2]. J. S. Kochhar, B. T. Foster, and S. S. Heragu, 1998. HOPE: A genetic algorithm for the unequal area facility layout problem. Computers & Operations Research, Volume 25, pp. 583-594

[3]. J. A. Tompkins, J. A. White, Y. A. Bozer, and J. M. A. Tanchoco, 2010. Facilities Planning, 4th Edition. New York: Wiley.

[4]. P. Kouvelis, A. A. Kurawarwala, and G. J. Gutiérrez, 1992. P. Kouvelis, A. A. Kurawarwala, and G. J. Gutiérrez,. European Journal of Operational Research, Volume 63, pp. 287-303.

[5]. S. Sahni and T. Gonzalez, 1976. P-Complete Approximation Problems. J. ACM, Volume 23, pp. 555-565.

[6]. K. L. Mak, Y. S. Wong, and F. T. S. Chan, 1998. A genetic algorithm for facility layout problems. Computer Integrated Manufacturing Systems, Volume 11, pp. 113-127.

[7]. T. Koopmans and M. Beckmann, 1957. Assignment Problems and the Location of Economic Activities. Econometrica, Volume 25, pp. 53-76.

[8]. J.-C. Picard and M. Queyranne, 1981. On the One-Dimensional Space Allocation Problem. Operations Research, Volume 29, pp. 371-391.

[9]. Simmons, D. M., 1969. Single row space allocation: An ordering algorithm.. Operations Research, Volume 17, p. 812–826

[10]. R. F. Love and J. Y. Wong, 1976. On solving a single row space allocation problem with integer programming. INFOR, Volume 14, pp. 139-143.

[11]. M. W. Guo, J. S. Wang, L. F. Zhu, S. S. Guo, and W. Xie, "Improved Ant Lion Optimizer Based on Spiral Complex Path Searching Patterns," *IEEE Access*, vol. 8, pp. 22094–22126, 2020, doi: 10.1109/ACCESS.2020.2968943.

[12]. Z. Zhang, R. Yang, H. Li, Y. Fang, Z. Huang, and Y. Zhang, "Antlion optimizer algorithm based on chaos search and its application," *Journal of*

Systems Engineering and Electronics, vol. 30, no. 2, pp. 352–365, Apr. 2019, doi: 10.21629/JSEE.2019.02.14

[13]. H. Tizhoosh, "Opposition-Based Learning: A New Scheme for Machine Intelligence," *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06)*, vol. 1, pp. 695–701, 2005.

[14]. S. Mirjalili, "The ant lion optimizer," *Advances in Engineering Software*, vol. 83, pp. 80–98, 2015, doi: 10.1016/j.advengsoft.2015.01.010.

[15]. S. Assiri, A. G. Hussien and M. Amin, "Ant Lion Optimization: Variants, Hybrids, and Applications," in *IEEE Access*, vol. 8, pp. 77746-77764, 2020, doi: 10.1109/ACCESS.2020.2990338.

[16]. Emary, Eid, Hossam M. Zawbaa, and Aboul Ella Hassanien. "Binary ant lion approaches for feature selection." *Neurocomputing* 213 (2016): 54-65.

[17]. N. Kumar, I. Hussain, B. Singh, B. K. Panigrahi, "Maximum power extraction from partially shaded pv panel in rainy season by using improved antlions optimization algorithm", in: 2016 IEEE 7th Power India International Conference (PIICON), IEEE, 2016, pp. 1-6.

[18]. K. R. Subhashini, J. K. Satapathy, "Development of an enhanced ant lion optimization algorithm and its application in antenna array synthesis", Applied Soft Computing 59 (2017) 153-173

[19]. K. Zhang, J. Ma, X. Zhao, X. Liu, Y. Zhang, "Parameter identi cation and state of charge estimation of nmc cells based on improved antlion optimizer", Mathematical Problems in Engineering 2019.

[20]. J. Vilma Roseline and D. D. Saravanan, "Crossover and Mutation Strategies applied in Job Shop Scheduling Problems," in *Journal of Physics: Conference Series*, Nov. 2019, vol. 1377, no. 1. doi: 10.1088/1742-6596/1377/1/012031.

[21]. H. Tizhoosh, "Opposition-Based Learning: A New Scheme for Machine Intelligence," *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06)*, vol. 1, pp. 695–701, 2005.

[22]. Y. Fang and J. Li, "A Review of Tournament Selection in Genetic Programming," *Advances in Computation and Intelligence*, vol. 6382, pp. 181–192, 2010, doi: 10.1007/978-3-642-16493-4_19.

[23]. Petrović, M., et al. "The ant lion optimization algorithm for flexible process planning." *Journal of Production Engineering* 18.2 (2015): 65-68.

[24]. Kilic, Haydar, Ugur Yuzgec, and Cihan Karakuzu. "A novel improved antlion optimizer algorithm and its comparative performance." *Neural Computing and Applications* 32.8 (2020): 3803-3824.

[25]. Filali, W., et al. "A Novel Parameter Identification Approach for C–V–T Characteristics of Multi-Quantum Wells Schottky Diode Using Ant Lion Optimizer." *Russian Microelectronics* 48.6 (2019): 428-434.

[26]. Kılıç, Haydar, and Ugur Yuzgec. "Improved antlion optimization algorithm for quadratic assignment problem." *Malaysian Journal of Computer Science* 34.1 (2021): 34-60.

[27]. Razali, Noraini Mohd, and John Geraghty. "Genetic algorithm performance with different selection strategies in solving TSP." *Proceedings of the world congress on engineering*. Vol. 2. No. 1. Hong Kong, China: International Association of Engineers, 2011.

[28]. Abdoun, Otman, Jaafar Abouchabaka, and Chakir Tajani. "Analyzing the performance of mutation operators to solve the travelling salesman problem." *arXiv preprint arXiv:1203.3099* (2012).

[29]. Dinkar, Shail Kumar, and Kusum Deep. "An efficient opposition based Lévy Flight Antlion optimizer for optimization problems." *Journal of computational science* 29 (2018): 119-141.

[30]. Tizhoosh, Hamid R. "Opposition-based learning: a new scheme for machine intelligence." *International conference on computational intelligence for modelling, control and automation and international conference on intelligent agents, web technologies and internet commerce (CIMCA-IAWTIC'06)*. Vol. 1. IEEE, 2005.

[31]. Rahnamayan, Shahryar, Hamid R. Tizhoosh, and Magdy MA Salama. "Opposition versus randomness in soft computing techniques." *Applied Soft Computing* 8.2 (2008): 906-918.

[32]. Huang, Chun, Chi Kwong Wong, and Chi Ming Tam. "Optimization of material hoisting operations and storage locations in multi-storey building construction by mixed-integer programming." *Automation in Construction* 19.5 (2010): 656-663.

[33]. CHIANG, W.-C. & CHIANG, C. J. E. J. O. O. R. 1998. Intelligent local search strategies for solving facility layout problems with the quadratic assignment problem formulation. 106, 457-488.