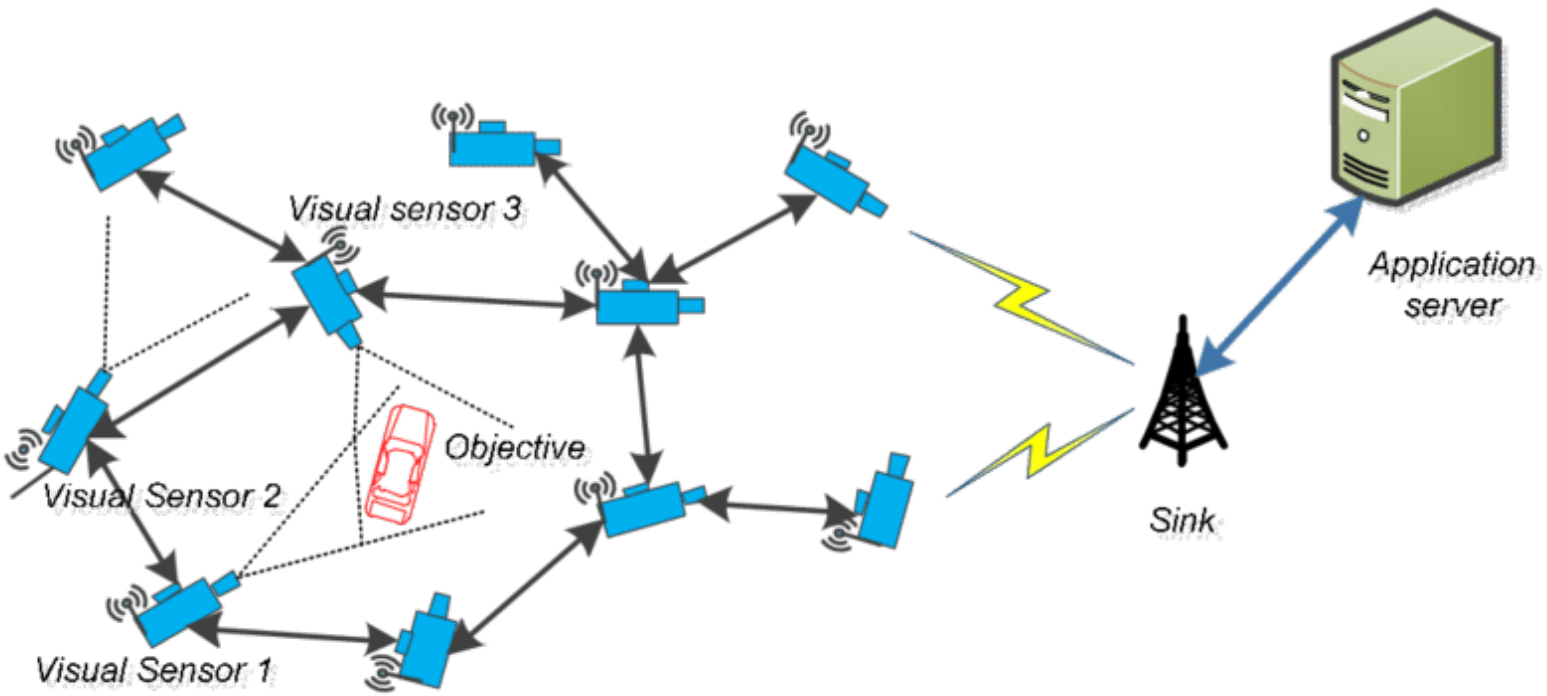
Tối ưu hóa Q-bao phủ trong mạng cảm biến thị giác có hướng quay rời rạc

Nguyễn Trung Hải - 20204545
Hồ Sỹ Thế - 20200614

Giới thiệu

Giới thiệu

- Một mạng cảm biến thị giác bao gồm một số lượng lớn các cảm biến thị giác (camera).
- Các cảm biến thị giác này có thể có hướng cố định hoặc đa hướng.



Một mạng cảm biến có thể được triển khai theo hai cách:

(i) **Sắp đặt trước**
Các cảm biến có thể được bố trí vào các vụ trí được tính toán từ trước để đạt điều kiện bao phủ, chỉ phù hợp với các mạng cỡ nhỏ và vừa.

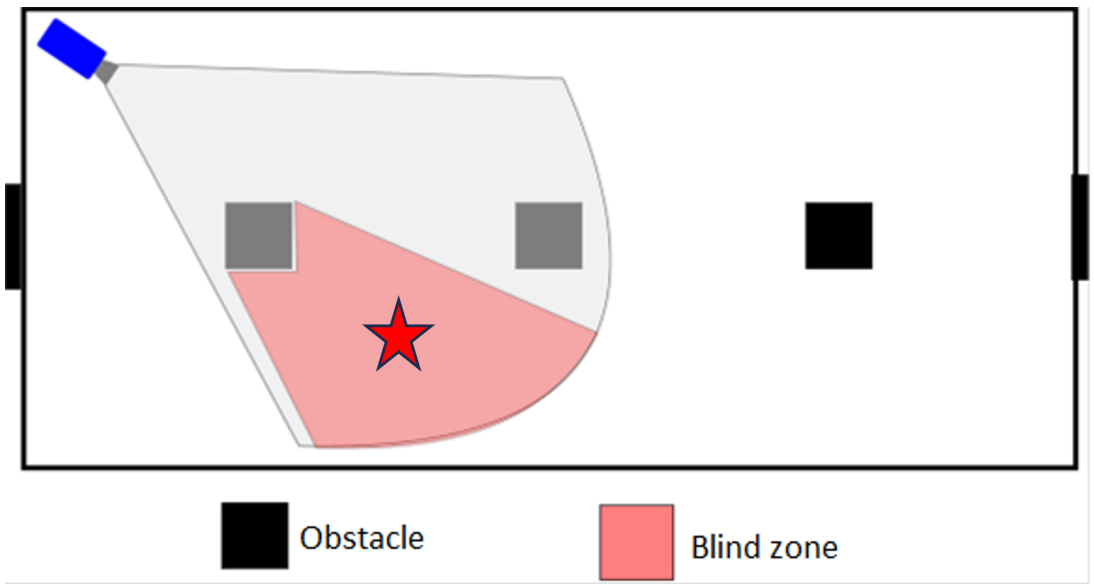
(ii) **Phân bố ngẫu nhiên**
Cách tiện lợi đối với các mạng cỡ lớn.

Vấn đề đặt ra: Điều chỉnh hướng quay của các cảm biến.

Giới thiệu

Bài toán Q bao phủ

- Trong môi trường thực tế, mạng cảm biến có thể bị mất bao phủ với mục tiêu.



- Mỗi mục tiêu yêu cầu chất lượng giám sát khác nhau.
- Trong bài toán Q bao phủ (Q coverage), mỗi mục tiêu có yêu cầu về mức độ bao phủ (số cảm biến theo dõi nó) khác nhau.

Giới thiệu

Mạng cảm biến trong miền thiếu giám sát

- Trong thực tế, chúng ta thường đối mặt với vấn đề thiếu hụt cảm biến.
- Tập trung vào các mạng cảm biến trong miền thiếu giám sát, khi mà yêu cầu bao phủ của các mục tiêu là không thể đạt được với số lượng cảm biến đã cho.
  - Ưu tiên bao phủ các mục tiêu có yêu cầu bao phủ lớn.
  - Cân bằng độ bao phủ đạt được giữa các mục tiêu trong cùng nhóm bao phủ.

Các tài liệu liên quan

Các tài liệu liên quan

[1] Al Zhishan et al, "Maximizing heterogeneous coverage in over and under provisioned visual sensor networks", Journal of Network and Computer Applications, Volume 124, 2018, Pages 44-62, ISSN 1084-8045, https://doi.org/10.1016/j.jnca.2018.09.009.

[2] A.Kaveh (2017) "Advances in Metaheuristic Algorithms for Optimal Design of Structures", Springer, 2nd ed.

[3] R. Tanabe and A. S. Fukunaga, "Improving the search performance of SHADE using linear population size reduction", 2014 IEEE Congress on Evolutionary Computation (CEC), Beijing, China, 2014, pp. 1658-1665, doi: 10.1109/CEC.2014.6900380.

Thông tin sơ bộ

Thông tin sơ bộ

Mô hình mạng cảm biến

- Một mạng cảm biến thị giác bao gồm các cảm biến thường là các camera đa hướng quay. Góc nhìn (FoV) của các camera này là span của phần diện tích chúng có thể quan sát được với một hướng quay nào đó của camera.
- Giả thiết các camera chỉ quay theo phương nằm ngang. Biểu diễn vùng cảm nhận hay một góc quay của camera bằng một hình nón quạt trong mặt phẳng hai chiều.

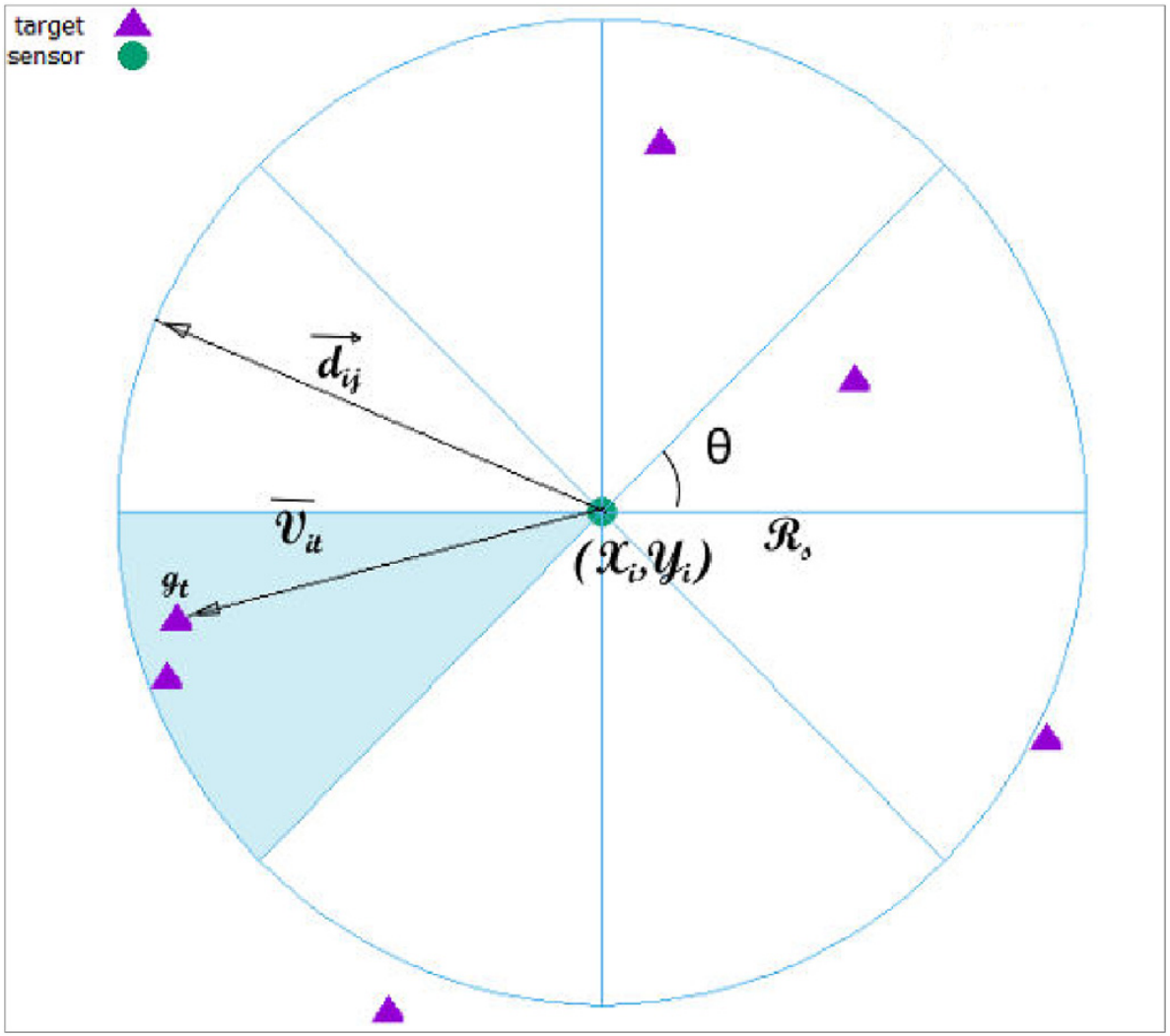
Các tham số:

- (x_i, y_i): Vị trí của camera s_i.
- θ: Góc quan sát (AoV) / góc bao phủ lớn nhất của một camera ở hướng bất kỳ.
- R_s: Bán kính bao phủ của một camera.
- $\overrightarrow{d_{ij}}$: một vector đơn vị nằm trên tia phân giác của một góc quay, thể hiện camera s_i quay tới góc p_j.
- $\overrightarrow{v_{it}}$: vector đi từ camera s_i tới mục tiêu g_t.

Giả thiết rằng các camera là đồng nhất, mỗi camera có một lượng giới hạn các góc quay và những góc quay này là rời rạc.

Thông tin sơ bộ

Mô hình mạng cảm biến



Thông tin sơ bộ

Kiểm tra mục tiêu nằm trong góc quay (TIWP)

- Kiểm tra mục tiêu nằm trong FoV của camera s_i bằng điều kiện:

$$\overrightarrow{v_{it}} \cdot \overrightarrow{d_{ij}} \geq \left|\overrightarrow{v_{it}}\right| \cos\left(\frac{\theta}{2}\right)$$

- Kiểm tra mục tiêu g_t nằm trong phạm vi bao phủ của camera bằng điều kiện: $\left|\overrightarrow{v_{it}}\right| \leq R_s$.

Nếu các điều kiện được thỏa mãn $\Rightarrow$ camera s_i bao phủ mục tiêu g_t với góc quay p_j.
$\mathcal{T}_{ij}$: tập các mục tiêu mà s_i bao phủ được khi quay hướng p_j.

Thông tin sơ bộ

Phát biểu bài toán

Cho:

- Một tập các mục tiêu, $\mathcal{G} = \{g_1, g_2, g_3, \ldots, g_m\}$.
- Một bộ các số nguyên dương, $\mathcal{K} = (k_1, k_2, k_3, \ldots, k_m)$, k_t là độ bao phủ yêu cầu của g_t; mục tiêu g_i và g_j nằm trong cùng nhóm bao phủ nếu $k_i = k_j$.
- Một tập các camera đồng nhất, $\mathcal{S} = \{s_1, s_2, s_3, \ldots, s_n\}$, mỗi camera có thể quay theo q hướng quay rời rạc.
- Một tập các hướng quay rời rạc (không chồng lấn), $\mathcal{P} = \{p_1, p_2, p_3, \ldots, p_q\}$.
- Một tập tất cả các cặp (cảm biến, hướng quay) có thể, $\mathcal{F} = \{(s_i, p_j) | \text{cảm biến } s_i \text{ được quay về hướng } p_j\}$.

Bài toán HCT: *Tìm tập con $\mathcal{C}$ của $\mathcal{F}$ sao cho, mỗi cảm biến có tối đa một góc quay tương ứng trong $\mathcal{C}$ và độ bao phủ đạt được của g_t được tối đa với điều kiện g_t được bao phủ bởi ít nhất k_t cảm biến và tối thiểu hóa số cảm biến được kích hoạt.*

Mô hình bài toán

Mô hình bài toán

Quy hoạch nguyên tuyến tính

Thông số cho cho ILP:

- n: số cảm biến.
- m: số mục tiêu.
- q: số hướng quay của các cảm biến.
- α_t: *độ bao phủ thực tế* của mục tiêu g_t.
- ψ_t: *độ bao phủ đạt được* của mục tiêu g_t, $\psi_t = \min(\alpha_t, k_t)$.
- $\chi_{(i,j)}$: một biến nhị phân nhận giá trị 1 nếu cảm biến s_i quay về hướng p_j và 0 nếu ngược lại.
- *Iversion bracket*: P là một mệnh đề đúng hoặc sai; khi đó *Iversion bracket* được định nghĩa như sau: $[P] = \begin{cases} 1, & \text{nếu } P \text{ là đúng} \\ 0, & \text{ngược lại} \end{cases}$

α_t có thể được tính như sau: $\alpha_t = \sum_{i=1}^{n} \sum_{j=1}^{q} [g_t \in \mathcal{T}_{ij}] \chi_{(i,j)}$

Mô hình ILP cho bài toán:

$$\textbf{maximize: } \sum_{t=1}^{m} \psi_t - \rho \sum_{i=1}^{n} \sum_{j=1}^{q} \chi_{(i,j)}$$

thỏa mãn các điều kiện:

- $\frac{\alpha_t}{n} \leq \psi_t \leq \alpha_t, \quad \forall t = 1, 2, \ldots, m$
- $\psi_t = \in \{0, 1, \ldots k_t\}, \quad \forall t = 1, 2, \ldots, m$
- $\sum_{j=1}^{q} \chi_{(i,j)} \leq 1, \quad \forall i = 1, 2, \ldots, n$
- $\chi_{(i,j)} = 0$ or $1, \quad \forall i = 1, 2, \ldots, n, \forall j = 1, 2, \ldots, q$

Mô hình bài toán

Hệ số phạt

Hệ số phạt ρ là một số thực dương, dùng để phạt số lượng cảm biến được kích hoạt. Nếu có hai lời giải, trong đó một lời giải có tổng mức bao phủ lớn hơn lời giải còn lại, thì hệ số phạt phải đảm bảo thuật toán sẽ chọn lời giải này mà không cần xét đến số cảm biến được kích hoạt.

Bổ đề

Đặt n là số cảm biến. Khi đó, trong mô hình ILP, hệ số phạt ρ nên nhỏ hơn $\frac{1}{n}$ để mô hình lựa chọn lời giải có độ bao phủ lớn hơn mà không xét đến số cảm biến được kích hoạt.

Nếu $\rho > 1$, việc giảm số lượng cảm biến kích hoạt sẽ được ưu tiên hơn tối đa hóa tổng mức bao phủ.

Mô hình bài toán

Quy hoạch bậc hai

- Ý tưởng sử dụng mô hình quy hoạch bậc hai để giải bài toán này là tối thiểu hóa khoảng cách giữa hai vector độ bao phủ đạt được và độ bao phủ yêu cầu với lượng cảm biến được kích hoạt là nhỏ nhất.
- Tối thiểu hóa khoảng cách $d = \| \overrightarrow{k} - \overrightarrow{\psi} \|_2$ và số cảm biến được kích hoạt.
- Hàm mục tiêu:

$$\textbf{minimize: } \sum_{t=1}^{m} (k_t - \psi_t)^2 + \rho \sum_{i=1}^{n} \sum_{j=1}^{q} \chi_{(i,j)}$$

Tất cả các ràng buộc khác tương tự với mô hình ILP.

Ràng buộc cho mạng cảm biến trong miền thiếu giám sát

Ràng buộc cho mạng cảm biến trong miền thiếu giám sát

Ưu tiên bao phủ - *pIQP*

- Đặt ưu tiên cho các mục tiêu với độ bao phủ yêu cầu lớn hơn, độ bao phủ yêu cầu càng lớn, mục tiêu càng được ưu tiên.
- Điều chỉnh hàm mục tiêu của IQP để thể hiện sự ưu tiên:

$$\textbf{minimize: } \sum_{t=1}^{m} k_t(k_t - \psi_t)^2 + \rho \sum_{i=1}^{n} \sum_{j=1}^{q} \chi_{(i,j)}$$

Tất cả các ràng buộc khác tương tự với mô hình ILP.

Ràng buộc cho mạng cảm biến trong miền thiếu giám sát

Cân bằng trong nhóm bao phủ - **rvIQP**

- Giảm thiểu phương sai về độ bao phủ đạt được đối với các mục tiêu trong cùng nhóm bao phủ. Thể hiện mong muốn các mục tiêu trong cùng nhóm bao phủ sẽ được đạt được độ bao phủ tương tự nhau.
- Điều chỉnh hàm mục tiêu của IQP để thêm yêu cầu về tính cân bằng:

minimize: $$\sum_{t=1}^{m}(k_t - \psi_t)^2 + \sum_{t=1}^{m}\frac{(\psi_t - \mu_{k_t})^2}{m_{k_t}} + \rho\sum_{i=1}^{n}\sum_{j=1}^{q}\chi_{(i,j)}$$

Trong đó:

- m_{k_t} là số mục tiêu yêu cầu k_t-bao phủ.
- μ_{k_t} là độ bao phủ trung bình của nhóm k_t-bao phủ, μ_{k_t} được tính như sau:

$$\mu_{k_t} = \frac{\sum_{i=1}^{m} \psi_i \left[k_i = k_t\right]}{m_{k_t}}$$

Tất cả các ràng buộc khác tương tự với mô hình ILP.

Mô hình IQP này giảm thiểu phương sai trong các nhóm bao phủ, đồng thời tối thiểu hóa khoảng cách giữa hai vector đã trình bày.

Thuật toán

Thuật toán
SOGA

- Việc giải chính xác các mô hình trên rất tốn thời gian và không khả thi đối với các mạng quy mô lớn.
- Al Zishan et al đề xuất thuật toán tham lam với độ phức tạp đa thức để giải bài toán HCT.
- SOGA là một thuật toán lặp. Mỗi vòng lặp, thuật toán sẽ kích hoạt một cảm biến đạt được giá trị hàm lợi ích lớn nhất.
- Hàm lợi ích này sẽ được tùy chọn theo bản chất thuật toán.

Thuật toán: SOGA

Algorithm 1 SOGA (Sensor Oriented Greedy Algorithm) to solve HCT problem.

1: $\mathcal{F}$ {a set of all possible (sensor, orientation) pairs}
2: $\mathcal{C} \leftarrow \emptyset$ { a subset of $\mathcal{F}$, final output of this algorithm}
3: $\mathcal{N} \leftarrow \mathcal{G}$ {a set of unachieved targets}
4: $a_t \leftarrow 0$ {achieved coverage for each target g_t upto previous iteration}
5: **While** $\mathcal{F} \neq \emptyset$ **and** $\mathcal{N} \neq \emptyset$ **do**
6: $(s_{i^*}, p_{j^*}) \leftarrow \arg \max_{(s_i,p_j)\in\mathcal{F}} benefit(s_i, p_j)$
7: $\mathcal{C} \leftarrow \mathcal{C} \cup \{(s_{i^*}, p_{j^*})\}$
8: **for all** $g_t \in \mathcal{T}_{i^*j^*}$ **and** $g_t \in \mathcal{N}$ **do**
9: $a_t \leftarrow a_t + 1$
10: **if** $a_t = k_t$ **then**
11: $\mathcal{N} \leftarrow \mathcal{N} - \{g_t\}$
12: **remove** all pairs for sensor s_{i^*} from $\mathcal{F}$
13: **return** $\mathcal{C}$

SOGA

Thuật toán

SOGA, IQP

Algorithm 2 *benefit*(s_i, p_j).

1: $value \leftarrow 0$
2: **for all** $g_t \in \mathcal{T}_{ij}$ and $a_t < k_t$ **do**
3: **if** *linear* **then**
4: $value \leftarrow value + 1$ (i.e. modified from Fusco and Gupta (Fusco and Gupta, 2009))
5: **if** *quadratic* **then**
6: $value \leftarrow value + [(k_t - a_t)^2 - (k_t - a_t - 1)^2]$
7: **return** *value*

Hàm lợi ích cho ILP và IQP

Algorithm 3 *benefit*(s_i, p_j) (for prioritized-IQP).

1: $value \leftarrow 0$
2: **for all** $g_t \in \mathcal{T}_{ij}$ and $a_t < k_t$ **do**
3: $\quad value \leftarrow value + k_t[(k_t - a_t)^2 - (k_t - a_t - 1)^2]$
4: **return** *value*

Hàm lợi ích cho *pIQP*

Algorithm 4 *benefit*(s_i, p_j) (for reduced-variance IQP).

1: $value \leftarrow 0$

2: **for all** $g_t \in \mathcal{T}_{ij}$ and $a_t < k_t$ **do**

3: $\quad \mu \leftarrow \mu_{k_t}$ {average of k_t-coverage group}

4: $\quad g \leftarrow m_{k_t}$ {number of targets in k_t-coverage group}

5: $\quad old \leftarrow (k_t - a_t)^2 + \frac{(a_t - \mu)^2}{g}$

6: $\quad new \leftarrow (k_t - a_t - 1)^2 + \frac{(a_t - \mu + 1 - \frac{1}{g})^2}{g}$

7: $\quad value \leftarrow value + (old - new)$

8: **return** *value*

Hàm lợi ích cho *rvIQP*

Các cá thể trong quần thể đại diện cho một cách chọn hướng quay cho tất cả các cảm biến trong mạng và được mã hóa thành một mảng, mỗi phần tử của mảng thể hiện hướng quay của một cảm biến:

sensor	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	s_{10}
pan	3	4	1	0	7	8	4	8	6	1

Trong đó:

$$\text{pan} = \begin{cases} 0, 1, \ldots, q-1 & \text{cảm biến quay về hướng pan} \\ q & \text{cảm biến không được kích hoạt} \end{cases}.$$

Chọn theo tỉ lệ r_c các cá thể tốt nhất trong quần thể để tiến hành lai ghép.

- Lai ghép nhị phân.
  Hai cá thể cha mẹ X_A và X_B lai ghép cho một cá thể con:

$$X[i] = \begin{cases} X_A[i] & rand < 0.5 \\ X_B[i] & rand \geq 0.5 \end{cases}$$

- Lai ghép k điểm cắt.
  Hai cá thể cha mẹ X_A và X_B lai ghép cho hai cá thể con có các gen được lấy lần lượt từ cá thể cha và cá thể mẹ theo các điểm cắt.

Chọn ngẫu nhiên theo tỉ lệ r_m các cá thể trong quần thể để tiến hành đột biến.

- Chọn ngẫu nhiên một gen của cá thể được đột biến X.
- Thay đổi ngẫu nhiên gen này với một giá trị thuộc $\{0, 1, \ldots, q\}$

Thuật toán

PSO

- Particle Swarm Optimization (PSO) là một giải thuật di truyền dựa trên quần thể. Trong thuật toán PSO, các cá thể trong quần thể sẽ lưu lại trạng thái tốt nhất của nó (khám phá/exploration) và lấy trạng thái của cá thể tốt nhất trong quần thể (khai thác/exploitation) để điều chỉnh trạng thái của chính nó.

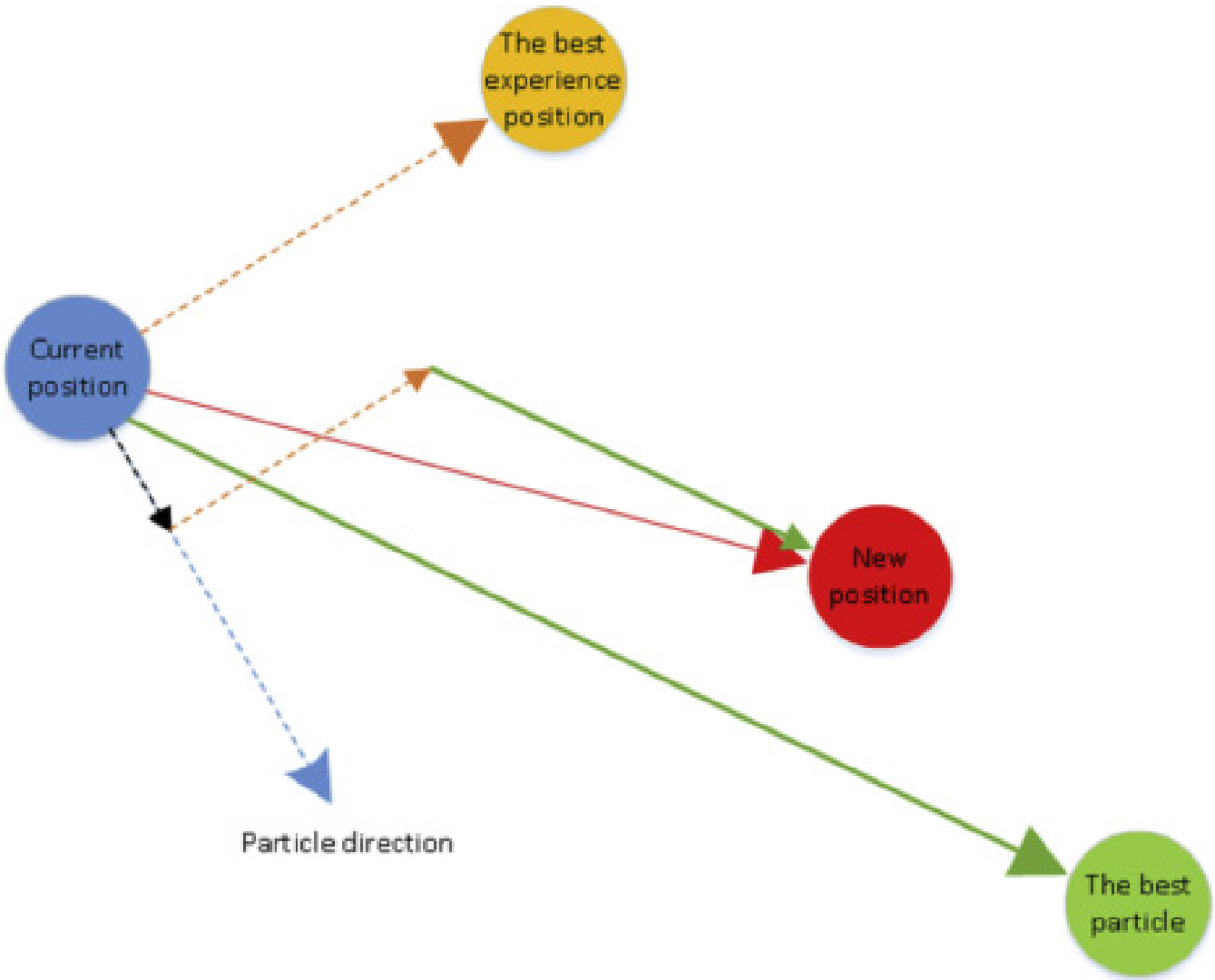
$$v_{i,j}^{k+1} = v_{i,j}^{k} + c_1 r_1 \left(xbest_{i,j}^{k} - x_{i,j}^{k} \right) + c_2 r_2 \left(xgbest_{j}^{k} - x_{i,j}^{k} \right)$$

$$x_{i,j}^{k+1} = x_{i,j}^{k} + v_{i,j}^{k+1}$$

- Thuật toán PSO nguyên thủy thường vướng phải vấn đề hội tụ sớm về một cực trị địa phương do các cá thể trong quần thể nhanh chóng co cụm mà không có khả năng khám phá những trạng thái mới.

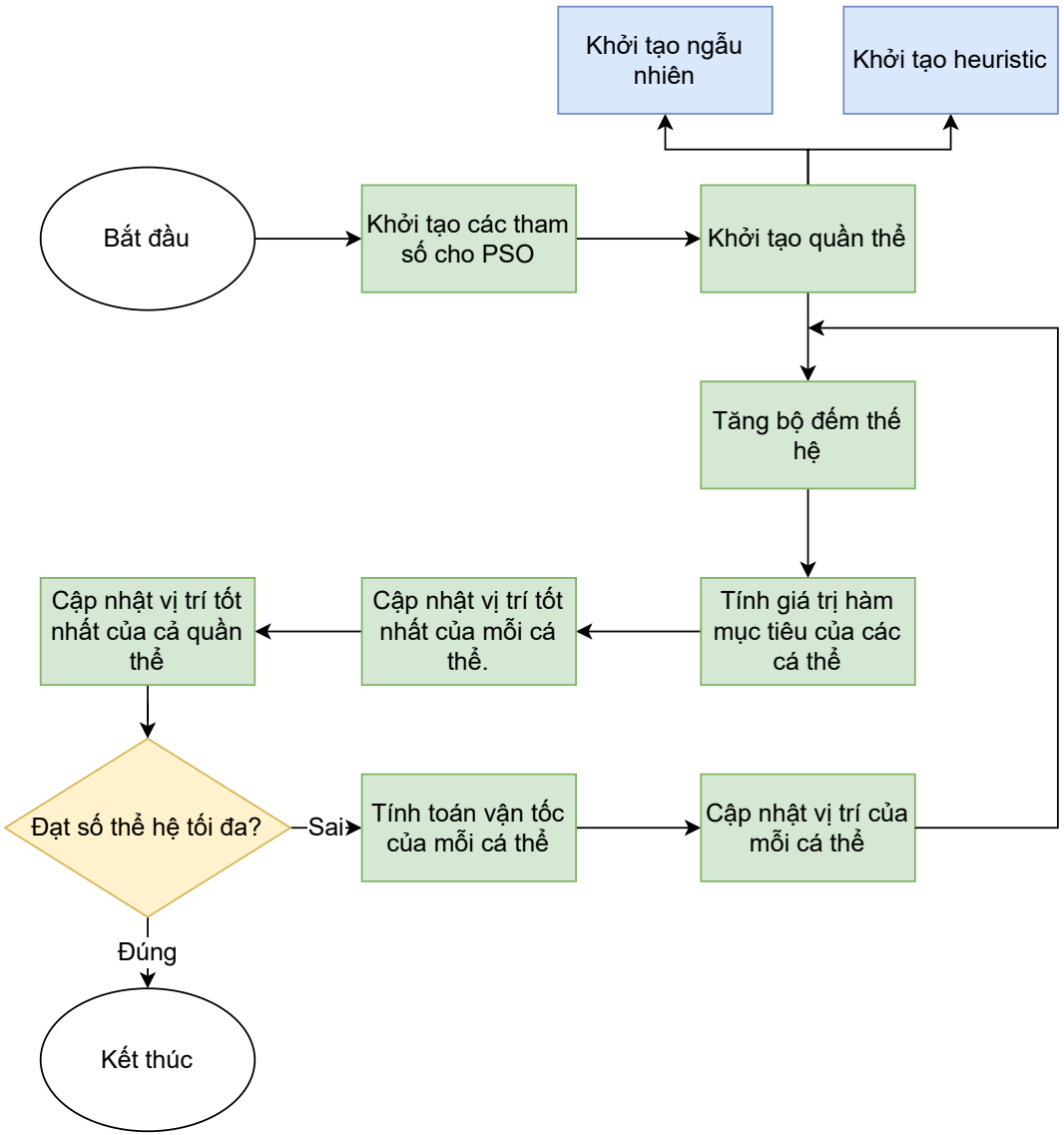
Thuật toán

PSO



Thuật toán

PSO



Thuật toán

DPSO

Democratic Particle Swarm Optimization (DPSO) là một biến thể của thuật toán PSO. Với thuật toán DPSO, các thể trong quần thể lấy thông tin từ tất cả các cá thể khác thay vì chỉ cá thể tốt nhất, quần thể trở nên dân chủ hơn.

Thuật toán

DPSO

Ý tưởng của thuật toán DPSO thể hiện qua công thức vận tốc sau:

$$v_{i,j}^{k+1} = \chi \left[\omega v_{i,j}^{k} + c_1 r_1 \left(xbest_{i,j}^{k} - x_{i,j}^{k} \right) + c_2 r_2 \left(xgbest_{j}^{k} - x_{i,j}^{k} \right) + c_3 r_3 d_{i,j}^{k} \right]$$

Trong đó $d_{i,j}^{k}$ là thành phần thứ j của vector D_i. Vector D_i thể hiện ảnh hưởng của trạng thái của các cá thể khác lên cá thể i.

Thuật toán

DPSO

Các công thức liên quan:

- $D_i = \sum_{k=1}^{n} Q_{ik}(X_k - X_i)$, Q_{ik} là mức ảnh hưởng của cá thể k tới cá thể i.
- $Q_{ik} = \dfrac{E_{ik}\frac{obj_{best}}{obj(k)}}{\sum_{j=1}^{n} E_{ij}\frac{obj_{best}}{obj(j)}}$
- $E_{ik} = \begin{cases} 1 & \dfrac{obj(k) - obj(i)}{obj_{worst} - obj_{best}} > rand \vee obj(k) < obj(i) \\ 0 & else \end{cases}$

Thuật toán

DPSO, Mã hóa biểu diễn

Các cá thể trong quần thể đại diện cho một cách chọn hướng quay cho tất cả các cảm biến trong mạng và được mã hóa thành một mảng, mỗi phần tử của mảng đại diện cho một cảm biến có hai trường thông tin:

sensor	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	s_{10}
state	-0.5	0.7	-0.2	1	1	-1	-0.3	0.3	-0.9	0.9
pan	3	4	3	6	1	2	0	0	6	7

- state: Số thực trong đoạn $[0, 1]$. Cảm biến tắt nếu state < 0 và bật nếu state ≥ 0.
- pan: Hướng quay của cảm biến, số thực trong tập $\{-1, 1, \ldots, q-1\}$.

Thuật toán

L-SHADE

- Success-history based parameter adaptation for Differential Evolution (SHADE) là giải thuật tiến hóa sai phân áp dụng kỹ thuật thích ứng tham số sử dụng bộ nhớ lịch sử của các tham số điều khiển, để điều chỉnh một cách thích ứng các tham số điều khiển của nó dựa trên sự thành công hay thất bại của các giải pháp trước đó.
- L-SHADE (SHADE with Linear Population Size Reduction) là một biến thể của SHADE với cải tiến việc giảm kích thước quần thể theo một hàm tuyến tính

Thuật toán

L-SHADE, Mã giả

Algorithm 2: L-SHADE algorithm

```
// Initialization phase
G = 1, N_G = N^init, Archive A = ∅;
Initialize population P_G = (x_{1,G}, ..., x_{N,G}) randomly;
Set all values in M_CR, M_F to 0.5;
// Main loop
while The termination criteria are not met do
    S_CR = ∅, S_F = ∅;
    for i = 1 to N do
        r_i = Select from [1, H] randomly;
        If M_{CR,r_i} = ⊥, CR_{i,G} = 0. Otherwise
        CR_{i,G} = randn_i(M_{CR,r_i}, 0.1);
        F_{i,G} = randc_i(M_{F,r_i}, 0.1);
        Generate trial vector u_{i,G} according to
        current-to-pbest/1/bin;
    for i = 1 to N do
        if f(u_{i,G}) ≤ f(x_{i,G}) then
            x_{i,G+1} = u_{i,G};
        else
            x_{i,G+1} = x_{i,G};
        if f(u_{i,G}) < f(x_{i,G}) then
            x_{i,G} → A;
            CR_{i,G} → S_CR, F_{i,G} → S_F;
    If necessary, delete randomly selected individuals
    from the archive such that the archive size is |A|.
    Update memories M_CR and M_F (Algorithm 1);
    // Optional LPSR strategy
    Calculate N_{G+1} according to Eq. (10);
    if N_G < N_{G+1} then
        Sort individuals in P based on their fitness
        values and delete lowest N_G − N_{G+1} members;
        Resize archive size |A| according to new |P|;
    G++;
```

Thuật toán

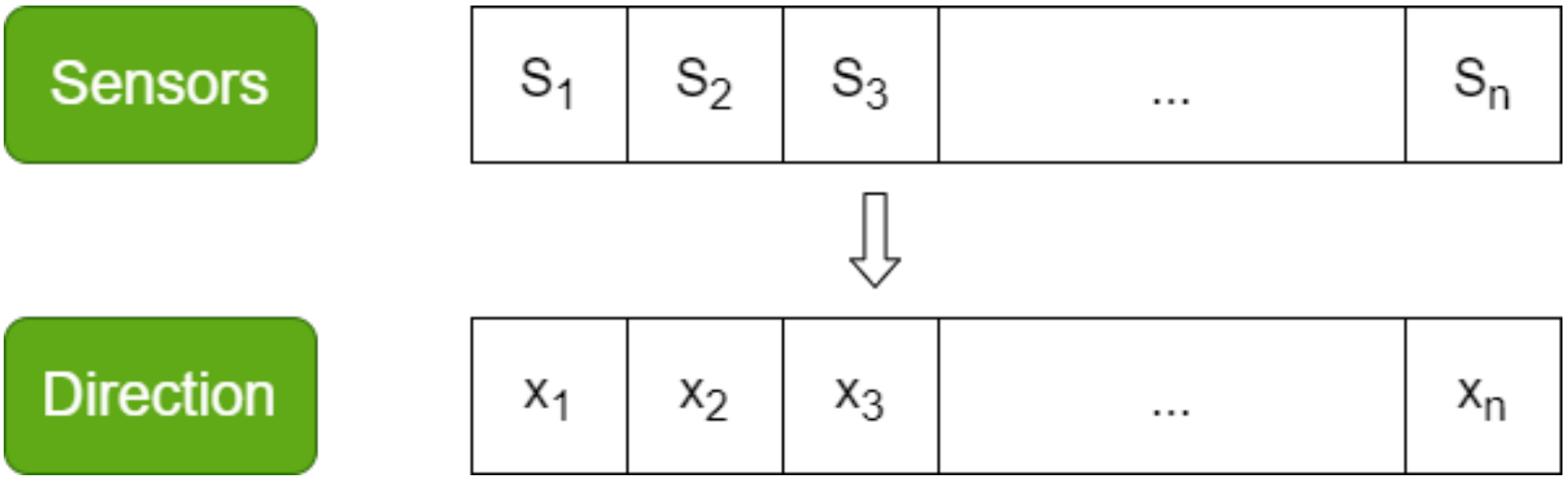
L-SHADE, Ý tưởng

L-SHADE sử dụng 5 tập lưu trữ

- Tập lưu trữ **A** lưu những cá thể thất bại
- Tập **MF**, **MCR** ($|\mathbf{MF}| = |\mathbf{MCR}| = H$) lưu các giá trị scale factor và crossover hoạt động tốt ở các thế hệ trước
- Tập **SF**, **SCR** lưu trữ các giá trị CR_i và F_i tạo ra trial vector tốt hơn so với cá thể gốc
- **MF**, **MCR** khởi tạo bằng 0.5 và được cập nhật bằng weighted Lehmer mean
- **SF**, **SCR** được khởi tạo rỗng

Thuật toán

L-SHADE, Mô hình hóa



$$\text{Chromosome} : \{x_1, x_2, \ldots, x_N\}$$

$$\mathbf{x_i} \in (-0.5, 8.5) \quad \forall i \in 1..N$$

$$dir_i \in [0, 8] \rightarrow \text{genome} : (dir_i - 0.5, dir_i + 0.5) \quad \forall i \in 1..N$$

Note:

- $dir_i \in [0, 7]$: quay các hướng từ 0 đến 7
- $dir_i = 8$ nghĩa là sensor S_i được tắt

Thuật toán

L-SHADE, Phép đột biến

Trial vector: $\mathbf{v}_i = \mathbf{x}_i + F_i\left(\mathbf{x}_{pbest} - \mathbf{x}_i\right) + F_i\left(\mathbf{x}_{r_1} + \mathbf{x}_{r_2}\right)$

- $\mathbf{x}_i$: cá thể đột biến
- $\mathbf{x}_{pbest}$: chọn ngẫu nhiên từ p% cá thể tốt nhất trong quần thể
- $\mathbf{x}_{r_1}$: chọn ngẫu nhiên trong quần thể
- $\mathbf{x}_{r_2}$: chọn ngẫu nhiên trong quần thể và tập **A**

Từ trial vector ở thế hệ g: $\mathbf{v}_i^{(g)}$:
Tính cá thể mới theo công thức:

$$\mathbf{o}_i^{(g)} = \begin{cases} \mathbf{v}_{i,j}^{(g)} & \text{if rand}[0,1) \leq CR_i \text{ or } j = j_{rand} \\ \mathbf{x}_{i,j}^{(g)} & \text{otherwise} \end{cases} \quad , \forall j \in 1..N$$

Thuật toán

L-SHADE, Hàm đánh giá

Từ cá thể $\mathbf{x}_i$ tính độ bao phủ của mục tiêu:$\mathbf{a}$ và số sensor hoạt động: c
Fitness f:

$$f(\mathbf{x}_i) = \sum_{j=1}^{m} (\mathbf{k}_j * |\mathbf{k}_j - \mathbf{a}_j|) + \lambda * c$$

- $\mathbf{k}_j$: Độ bao phủ yêu cầu của target j
- λ: Hằng số phạt đối với số sensor hoạt động

Chọn lọc:

$$\mathbf{x}_i^{(g+1)} = \begin{cases} \mathbf{o}_i^{(g)} & \text{if } f(\mathbf{o}_i^{(g)}) \leq f(\mathbf{x}_i^{(g)}) \\ \mathbf{x}_i^{(g)} & \text{ngược lại} \end{cases}$$

Giảm kích thước quần thể:

$$NP_{g+1} = round\left[\frac{NP_{min} - NP_{init}}{MAX_{NFE}} * NFE + NP_{init}\right]$$

Cập nhật các tập lưu trữ **A**, **MF**, **MCR**, **SF**, **SCR**

Đánh giá

Đánh giá

- Triển khai mạng trên mặt phẳng hai chiều, diện tích vùng triển khai kích thước 200×200 đơn vị.
- Các mục tiêu và các cảm biến được coi là các điểm trên mặt phẳng và có vị trí ngẫu nhiên.
- Triển khai các cảm biến theo phân phối đều.
- Hàm mục tiêu của mô hình IQP.

- **Distance Index** $(\mathcal{DI})$

  $$\mathcal{DI} = \frac{\sum_{t=1}^{m} k_t^2 - \sum_{t=1}^{m} (k_t - \psi_t)^2}{\sum_{t=1}^{m} k_t^2}$$

  Chỉ số $\mathcal{DI}$ càng cao thể hiện các mục tiêu được bao phủ càng tốt so với yêu cầu.
- **Activated Sensors** Nếu một thuật toán cho độ bao phủ tương đương với một thuật toán khác nhưng sử dụng ít cảm biến hơn thì thuật toán đó là tốt hơn.

- **Variance** Để đánh giá độ cân bằng trong mỗi nhóm bao phủ, ta cần tích phương sai của mỗi nhóm và tính tổng các phương sai này:

$$\sum_{t=1}^{m} \frac{(\psi_t - \mu_{k_t})^2}{m_{k_t}}$$

Chỉ số này cho giá trị nhỏ hơn nghĩa là độ cân bằng bao phủ tốt hơn.

Đánh giá

Các chỉ số

- **Coverage Quality** là tổng chất lượng bao phủ đạt được đối với mỗi mục tiêu và được tính bởi công thức:

$$\mathcal{CQ} = \sum_{i,j,t} U(i,j,t)\,[s_i \textit{ quay về hướng } p_j]$$

Trong đó:

$$U(i,j,t) = \begin{cases} 1 - \left(\dfrac{\left|\overrightarrow{v_{it}}\right|^2}{R_s} \right), & \text{nếu } g_t \in \mathcal{T}_{ij} \text{ và } \overrightarrow{v_{it}} < R_s \\ 0, & \text{ngược lại.} \end{cases}$$

$\mathcal{CQ}$ càng cao thể hiện các mục tiêu được bao phủ với chất lượng càng tốt.

GA

- Lai ghép hai điểm cắt.
- $r_c = 0.4$
- $r_m = 0.05$
- Khởi tạo heuristuc 40%, các cảm biến được bật khi có mục tiêu trong bán kính và được quay theo những hướng có nhiều mục tiêu hơn.
- Kích cỡ quần thể là 200, số thế hệ tối đa là 1000.

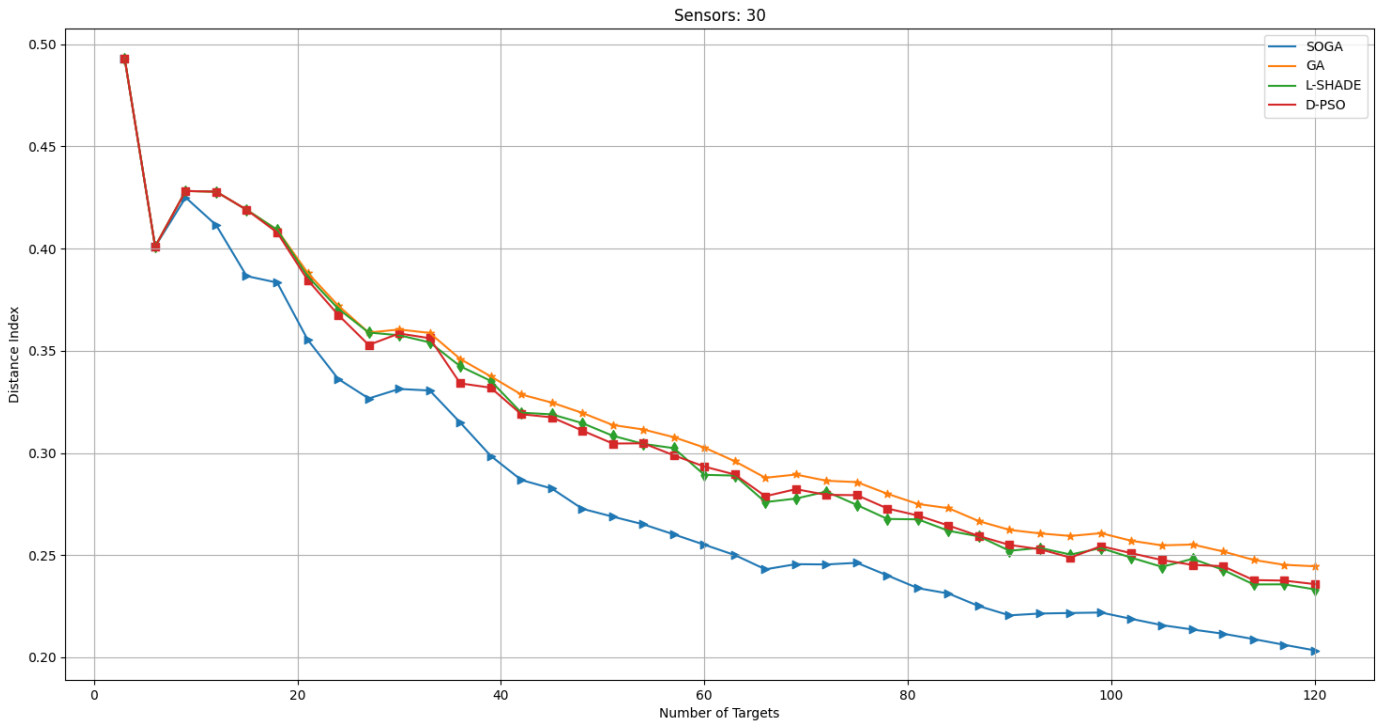
DPSO

- $\chi = 1.25$
- $\omega = 1.3$
- $c_1 = 0.45$
- $c_2 = 0.25$
- $c_3 = 0.15$
- Khởi tạo heuristuc 40%, các cảm biến được bật khi có mục tiêu trong bán kính và được quay theo những hướng có nhiều mục tiêu hơn.
- Kích cỡ quần thể là 200, số thế hệ tối đa là 1000.

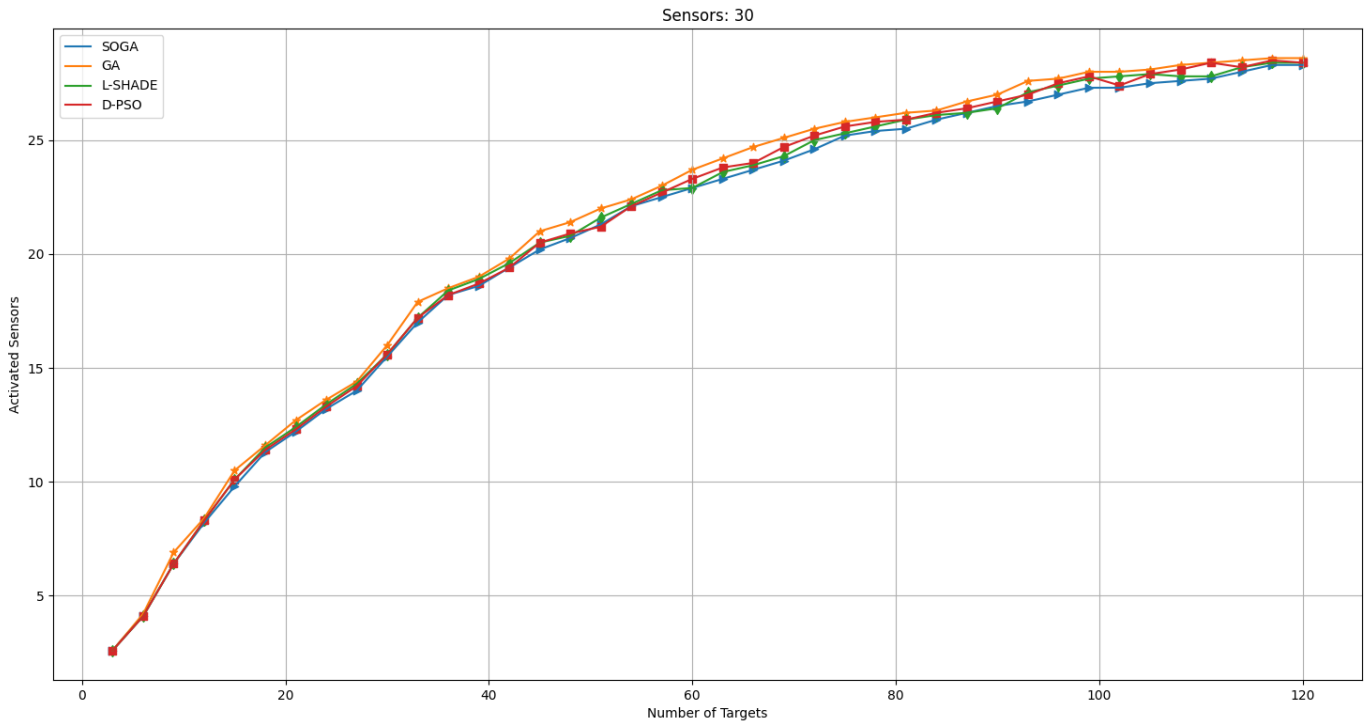
L-SHADE

- Kích thước quần thể khởi tạo: $NP_{init} = 18.D$
- Kích thước quần thể tối thiểu: $NP_{min} = 4$
- $H = 15$
- Kích thước tập lưu trữ $\mathbf{A}$: $|\mathbf{A}|_g = 2.6NP_g$
- $MAX_{NPE} = 5000 \cdot D$
- $\lambda = \frac{1}{D+1}$
- Khởi tạo đều các cảm biến được bật.

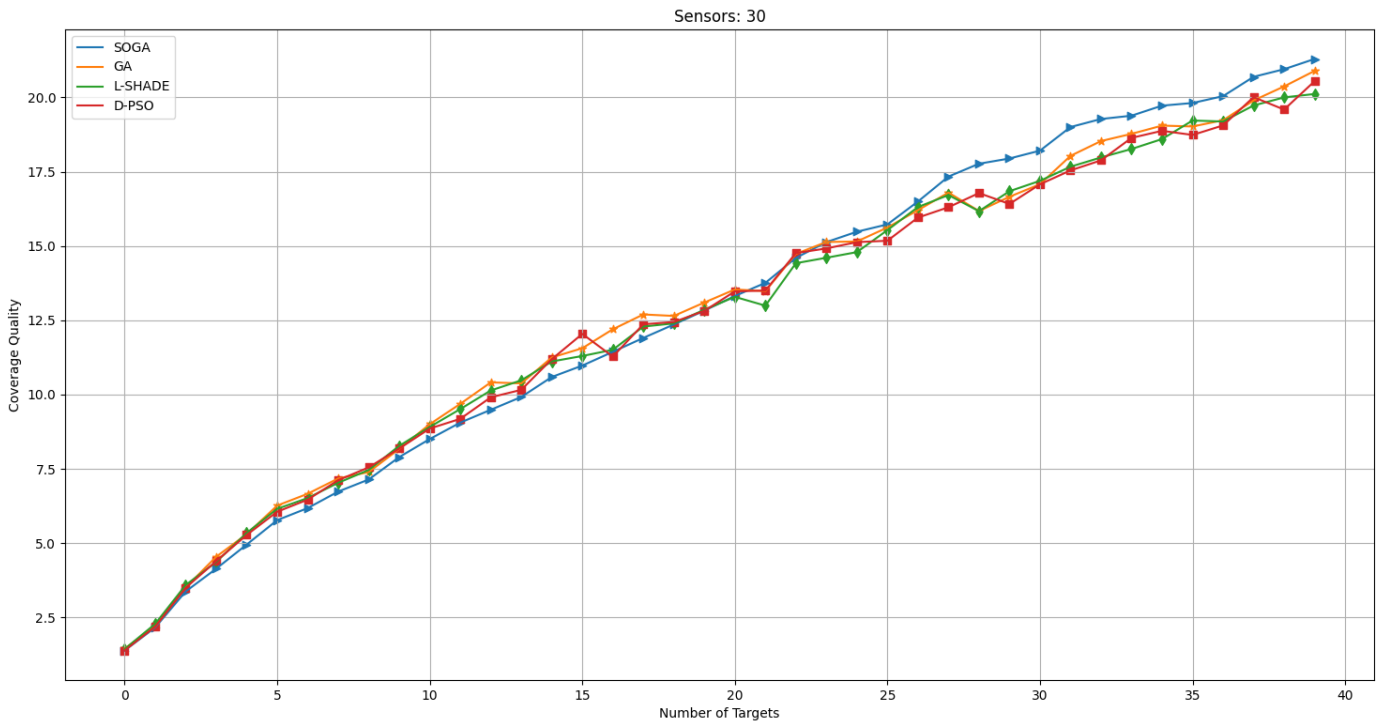
Distance Index



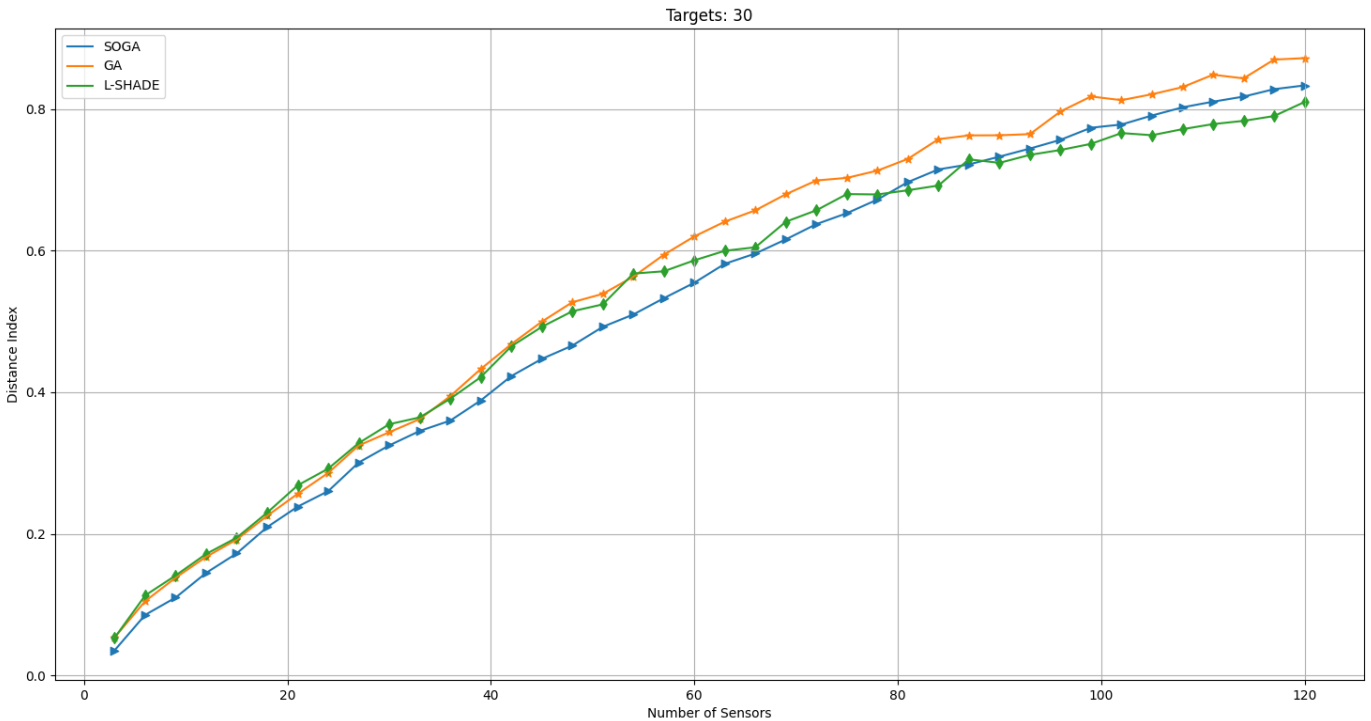
Activated Sensors



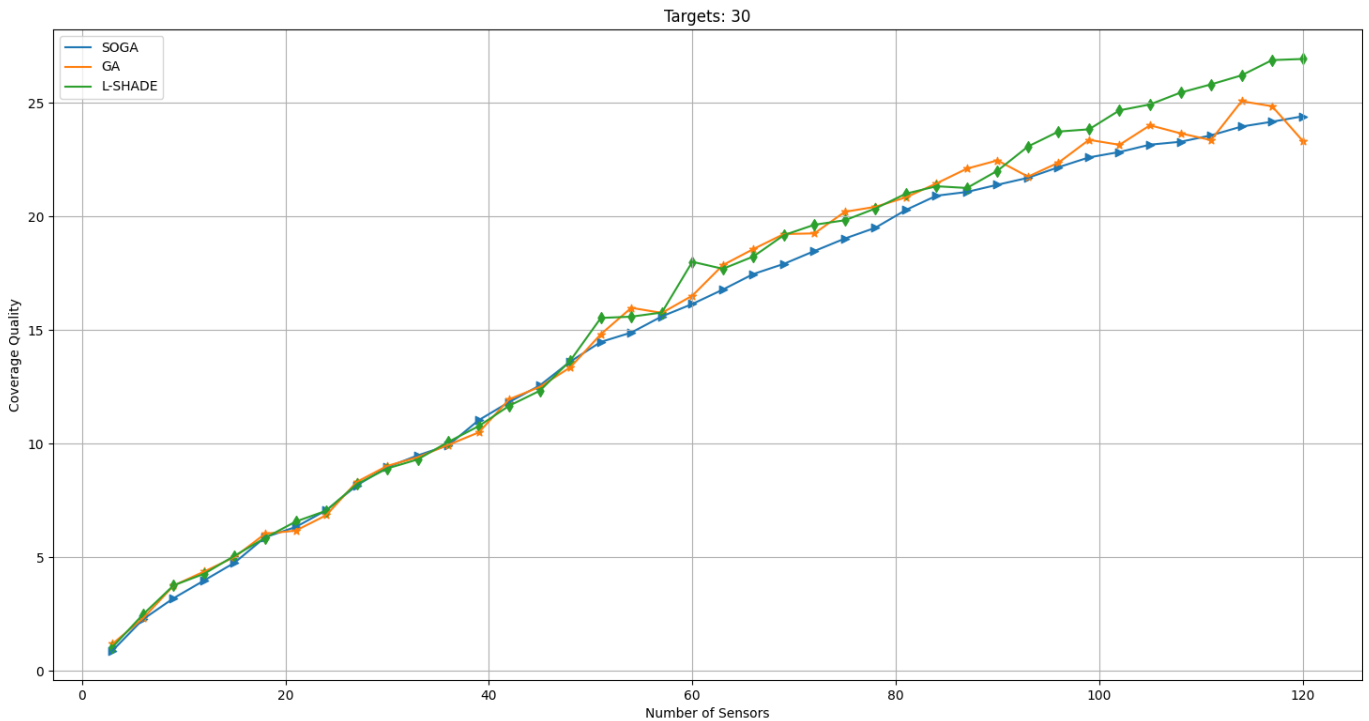
Coverage Quality



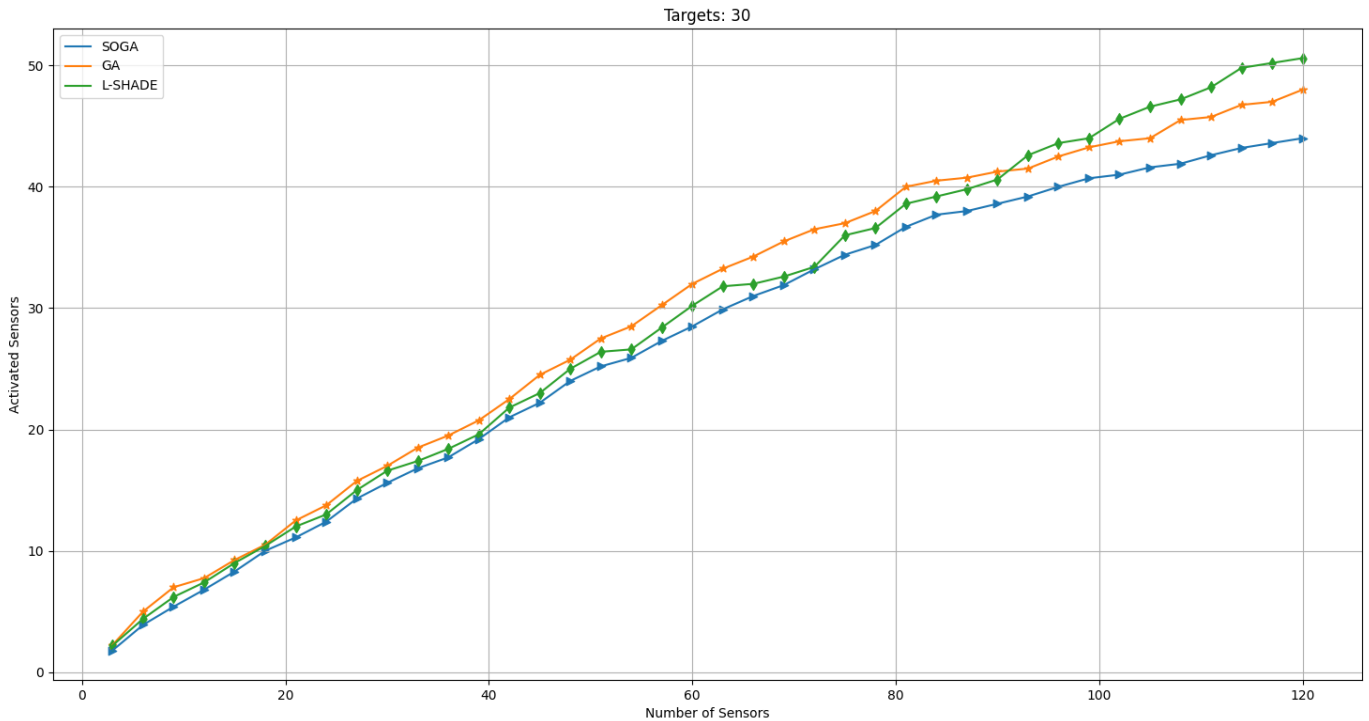
Distance Index



Coverage Quality



Activated Sensors



Kết luận

Kết luận

Công việc thực hiện

- Trong quá trình thực hiện đề tài, nhóm đã nghiên cứu bài toán Q-bao phủ trong mạng cảm biến thị giác.
- Đã cái đặt thử nghiệm giải chính xác bài toán theo hai mô hình ILP và IQP,

Kết luận

Kết quả

- Thuật toán tham lam SOGA có ý tưởng đơn giản nhưng cho kết quả khá và có thời gian chạy nhanh.
- Các thuật toán GA, DPSO và LSHADE có khả năng đưa ra lời giải gần tối ưu trong thời gian chấp nhận được.
- Cả thời gian tính toán và chất lượng lời giải đều chịu ảnh hưởng bởi cả thuật toán và cách mã hóa biểu diễn.
- Đối với các thuật toán DPSO và LSHADE, việc chọn các tham số hợp lý là vô cùng quan trọng.

Kết luận

Hướng phát triển

- Nghiên cứu các phương pháp mã hóa biểu diễn phù hợp hơn.
- Điều chỉnh tham số của các thuật toán.
- Thử nghiệm với các tập dữ liệu lớn hơn.