Multi-agent decision support system for missile defense based on improved PSO algorithm

Article *in* Journal of Systems Engineering and Electronics · June 2017

DOI: 10.21629/JSEE.2017.03.11

3 authors, including:

Cheng Zilong
China Astronaut Research and Training Center
5 PUBLICATIONS **14** CITATIONS

SEE PROFILE

Journal of Systems Engineering and Electronics
Vol. 28, No. 3, June 2017, pp.514 – 525

Multi-agent decision support system for missile defense based on improved PSO algorithm

Zilong Cheng[1,*], Li Fan[2], and Yulin Zhang[1]

1. College of Aerospace Science and Engineering, National University of Defense Technology, Changsha 410073, China;
2. Academy of Equipment, Beijing 101416, China

Abstract: Ballistic missile defense system (BMDS) is important for its special role in ensuring national security and maintaining strategic balance. Research on modeling and simulation of the BMDS beforehand is essential as developing a real one requires lots of manpower and resources. BMDS is a typical complex system for its nonlinear, adaptive and uncertainty characteristics. The agent-based modeling method is well suited for the complex system whose overall behaviors are determined by interactions among individual elements. A multi-agent decision support system (DSS), which includes missile agent, radar agent and command center agent, is established based on the studies of structure and function of BMDS. Considering the constraints brought by radar, intercept missile, offensive missile and commander, the objective function of DSS is established. In order to dynamically generate the optimal interception plan, the variable neighborhood negative selection particle swarm optimization (VNNSPSO) algorithm is proposed to support the decision making of DSS. The proposed algorithm is compared with the standard PSO, constriction factor PSO (CFPSO), inertia weight linear decrease PSO (LDPSO), variable neighborhood PSO (VNPSO) algorithm from the aspects of convergence rate, iteration number, average fitness value and standard deviation. The simulation results verify the efficiency of the proposed algorithm. The multi-agent DSS is developed through the Repast simulation platform and the constructed DSS can generate intercept plans automatically and support three-dimensional dynamic display of missile defense process.

Keywords: agent-based modeling, missile defense system, decision support system (DSS), variable neighborhood, negative selection, particle swarm optimization (PSO).

DOI: 10.21629/JSEE.2017.03.11

1. Introduction

Ballistic missile defense system (BMDS) is important for its special role in ensuring national security and maintaining strategic balance [1]. Due to the huge cost of development, testing and employment of BMDS [2], establishing a BMDS simulation model beforehand is a practical method before developing a real one which requires lots of manpower and resources. BMDS is a typical complex system for its nonlinear, adaptive and uncertainty characteristics. The interactions between the elements of BMDS must be considered to evaluate the effectiveness of the whole BMDS.

Constructing a multi-layer missile defense system is an inevitable trend for the future development of BMDS [3]. Lots of efforts have been done to evaluate the effectiveness of BMDS [4]. Some scholars studied on the multi-layer missile defense system which considers the combat effectiveness and operational costs by using traditional deterministic modeling methods [5]. Others explored the use of system engineering thinking [6,7] or the computational experiments approach [8] to solve the complex BMDS modeling and simulation problem.

Traditional modeling and simulation methods cannot reflect the inherent complexity of the complex adaptive system [9]. The agent-based modeling method is a bottom-up modeling method [10] which is suitable for complex system research and various types of agent-based modeling platforms have already been developed [11]. More and more scholars begin to employ agent-based modeling methods to complex system modeling [12] like transportation system [13] and economic system [14]. Ibrahim [15] et al. proposed a two layer hybrid agent architecture for modeling and simulation of small military unit combat in asymmetric warfare. Holland [16] et al. established a BMDS kill chain model by using the agent-based modeling method, quantitatively analyzed the interactions among the BMDS and evaluated the uncertainty of the kill chain. Christopher [17] et al. created an agent-based BMDS model through system development framework which integrates the human-in-the-loop decision maker into the design. Connors [18] et al. designed an experiment to simulate and analyze a new air-to-air missile which yields useful insights about the complex interactions of different actors on the battlefield.

Manuscript received March 22, 2016.
*Corresponding author.

All of these excellent works are trying to solve the problem of command and control of missile defense, but these preliminary efforts did not realize the autonomous decision making of BMDS. The decision making time for assigning weapons to intercept offensive missiles is always restricted to a few tens of minutes, which is a great challenge for commanders. Constructing a missile defense decision support system (DDS) will greatly improve the effectiveness of BMDS [19]. Tanerguclu [20] et al. developed a DSS to determine the optimal positions for air defense weapons and radars. Some scholars brought the optimization algorithm into the DSS to provide more fast and reliable solutions for missile defense missions. Li [21] et al. proposed a modified particle swarm optimization (PSO) algorithm to solve the problem of complex BMDS interceptor resource planning.

This paper is organized as follows. The second section discusses the agent-based BMDS DSS modeling and the third section describes and analyzes the proposed improved PSO algorithm. The fourth section describes the design and realization of missile defense DSS with the Repast simulation platform. Conclusions are given in the last section.

2. Agent-based BMDS DSS modeling

A typical missile defense mission process consists of a series of steps, including early warning, tracking and recognition, command and control, and missile interception which is achieved through the collaboration of BMDS components. The early warning radar and tracking radar are mainly made up of antenna, signal transmitter and receiver, signal processor and terminal devices. The function of the early warning radar is to obtain rough position information about the offensive missile and provide early warning information for the command center. The tracking radar guides the intercept missile to accomplish the intercept mission under the instruction from the command center. The command center is mainly made up of communication link and information processing centers. Its functions include estimating the offensive missile trajectory according to early warning radar warning information, sending guidance information to the tracking radar, generating interception plan and issuing emission instruction to intercept missile when the interception time is arrived. Intercept missiles are made of booster rockets and warheads. Its mission includes starting the interception mission when receiving emission instruction, correcting trajectory under the guidance information from tracking and recognition radar during the interception process, performing terminal guidance maneuver if reaching the terminal guidance distance. In order to achieve the closed-loop missile defense simulation, the offensive missile is also needed.

2.1 Agent modeling

An agent is a concept first introduced in distributed artificial intelligence research [22] which is expected to solve complex problems through negotiation between individuals. A typical agent is made up of sensor, controller and actuator. The sensor is responsible for obtaining the environment information within the perception range. The actuator is the execution composition of the agent which is responsible for applying the agent feedback to the environment. The controller is the core of the agent which is made of knowledge base, rule base and reasoning engine. The structure of the agent is shown in Fig. 1.

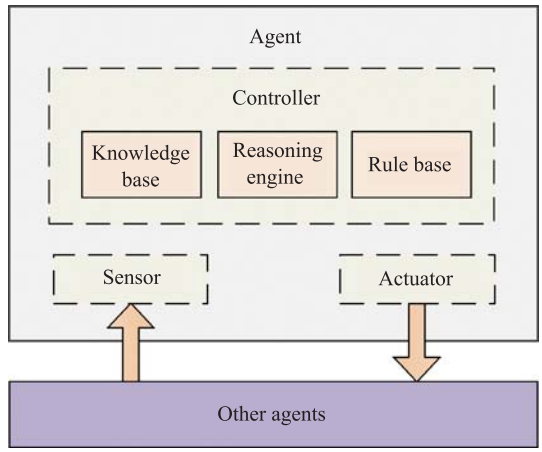

Fig. 1 Structure of agent

The priori knowledge of the agent is stored in the knowledge base. The decision rules and selection preferences are stored in the rule base. The agent can make choices according to a logic statement or the output of a calculation model. Reasoning engine is the most important part of the agent which achieves the reasoning and decision-making mission. It works as follows: constructing the constraint model in accordance with the perception information, selecting the most appropriate behavior based on the priori knowledge and rule base. The reasoning engine of the complex agent can integrate all kinds of complex input information. Apart from the ability of reasoning using priori rules, it can also automatically correct or create rules after training.

Taking the middle-course missile defense interception as an example, the components of BMDS are modeled using the agent-based modeling method which supports the autonomous evolution of BMDS. The multi-agent missile defense DSS will generate the interception plan dynamically which supports the computational experiments on BMDS as well as the system-of-systems effectiveness evaluation and optimization. The agents in BMDS are shown in Fig. 2.

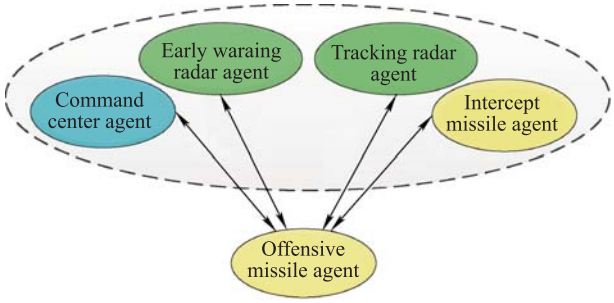

Fig. 2 Agents in BMDS

As shown above, agents in the same color indicate that they have similar features. The primary agents in the BMDS are missile agent, radar agent and command center agent.

2.1.1 Command center agent

The command center agent achieves target weapon assignment and sending missile launch instruction. Its knowledge is shown in Table 1.

Table 1 Knowledge of command center agent

Property	Value
Command center position	Longitude, latitude, altitude
Intercept missile position	Longitude, latitude, altitude
Intercept missile range angle	Calculated by spherical geometry formula
Intercept missile launch time	Calculated by the Lambert equation
Intercept missile fly time	Calculated by the Lambert equation
Intercept probability	Determined by the intercept missile

The rules of the command center agent include the threat evaluation rule and the intercept missile selection rule. The rules are supported by the decision-making model described below.

(i) Weapon target assignment calculation model

This model achieves the intercept weapon assignment which is the key factor to ensure the success of the interception mission. Before generating the intercept plans, all related factors must be taken into consideration. These related factors include position, type and number of intercept missiles, the threat level and the number of offensive missiles, and the time when the threat is found. These considered factors will be integrated into an optimization objective function.

(ii) Expected intercept point calculation model

The expected intercept point is set to a point from the trajectory of the offensive missile which is equal to the maximum intercept height of the intercept missile. The trajectory of the offensive missile is estimated by the cumulative observational information from early warning and tracking radar. The maximum intercept height of the intercept missile is determined by the capacity of the intercept missile.

(iii) Intercept missile launch time calculation model

According to the expected intercept point time, intercept missile launch time T_L can be calculated as follows:

$$T_L = T_e - T_f - T_p \tag{1}$$

where T_e denotes the time of expected intercept point, T_f denotes the flight time of intercept missile which can be calculated by Lambert equation, and T_p denotes the launch preparation time of intercept missile.

(iv) Missile range angle calculation model

The range angle θ_f is calculated by the spherical trigonometry formula according to the launch point and the target point. Set the launch point as A, target point as B, north pole as C, the three points and geocentric o form a spherical triangle cone. The A, B, C represents the angle between two facets while a, b, c represents the geocentric angle between two points. After replacing variables in the formula with $a = 90 - T_{lat}$, $b = 90 - L_{lat}$, $C = T_{lon} - L_{lon}$, we can obtain

$$\cos c = \cos(90 - T_{lat})\cos(90 - L_{lat}) + \sin(90 - T_{lat})\sin(90 - L_{lat})\cos(T_{lon} - L_{lon}) \tag{2}$$

where L_{Lon} and L_{lat} represent the longitude and latitude of the launch point while T_{lon} and T_{lat} represent the longitude and latitude of the target point. Finally we get the range angle $\theta_f = \arccos(\cos c)$.

(v) Missile shutdown parameters calculation model

Setting geocentric as the origin of the coordinate system, and the connection of geocentric and launch points as the X axis, we can obtain a polar coordinate system. According to Lambert equation [23], the missile shutdown velocity V_s satisfies

$$V_s = \sqrt{\frac{\mu}{|\boldsymbol{r}_1|}\frac{|\boldsymbol{r}_2|(1-\cos\theta_f)}{|\boldsymbol{r}_1|\cos\gamma^2 - |\boldsymbol{r}_2|\cos(\theta_f+\gamma)\cos\gamma}} \tag{3}$$

where $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ are the launch point vector and the intercept vector from geocentric, θ_f is the range angle which can be obtained by $\theta_f = \cos^{-1}(\boldsymbol{r}_1 \cdot \boldsymbol{r}_2/|\boldsymbol{r}_1||\boldsymbol{r}_2|)$. γ represents the trajectory inclination angle which can be obtained by $\gamma = (\pi - \theta_f)/4$ under minimum energy trajectory. μ represents the gravity constant. Since the missile flight trajectory is an ellipse, the missile flight time can be calculated as follows:

$$t_f = \frac{|\boldsymbol{r}_1|}{V\cos\gamma}\{\frac{\tan\gamma(1-\cos\theta_f)+(1-\lambda)\sin\theta_f}{(2-\lambda)|\boldsymbol{r}_1|/|\boldsymbol{r}_2|} + \frac{2\cos\gamma}{\lambda[(2/\lambda)-1]^{\frac{3}{2}}}\arctan\frac{[(2/\lambda)-1]^{\frac{1}{2}}}{\cos\gamma\cot(\theta_f/2)-\sin\gamma}\} \tag{4}$$

where variable λ is determined by $\lambda = |\boldsymbol{r}_1|V^2/\mu$. The decision-making process of the command center is shown below.

If Received warning information from radar

Then Execute threat evaluation

If Meets requirements of intercept mission

Then Execute weapon target assignment, cal-

culate intercept missile launch time T_L, generate interception plan
If Meets launch conditions
Then Send launch instruction which includes offensive missile ID, intercept missile ID, intercept missile launch time T_L, longitude, latitude and altitude of the expected intercept point
End If
End If
End If

2.1.2 Missile agent

The missile agent is the base class of the offensive missile and the intercept missile agent. The knowledge base stores properties to support calculation of missile trajectory which includes the position and velocity of the missile. Note that the properties of different missiles have different initial values.

The rules of the intercept missile include the missile launch rule and the trajectory correction rule. The missile launch rule is "if-then" judgment, e.g. the intercept missile will launch when receiving the launch instruction. The trajectory correction rule consists of two sub-rules. One sub-rule is revising its trajectory when the track cumulative error exceeds the threshold value. The other one is revising its trajectory by using the proportional guidance law when the terminal guidance distance is arrived. The decision-making process is shown below.

If Received launch instruction.
Then Calculate intercept missile range angle, estimate shutdown velocity V_s and velocity inclination angle
If Arrived launch time
Then Launch intercept missile and update its velocity and position vector
End If
If The accumulated error between flight trajectory and expected trajectory exceeds the threshold
Then Receive guidance information from tracking radar and execute trajectory revision
End If
If Reached the terminal guidance distance
Then Revise its trajectory by using proportional guidance law
End If
If Reached the effective damage distance or miss the target
Then Intercept missile explosion
End If
End If

2.1.3 Radar agent

The radar agent is the base class of early warning radar and tracking radar. Its function is providing early warning and guidance information for BMDS. The controller of the radar agent is depicted below. The radar knowledge base mainly stores properties that support the calculation of the radar detection range and angle coverage. The rules of radar include the target detection rule and the forecast rule. The target detection rule calculates the radar detection range through radar equation. The forecast rule reports the warning information in case of finding offensive missile three times in a row.

Assuming the missile is in the shape of cylindrical, its radar cross section (RCS) can be calculated using the following equation according to the empirical formula.

$$RCS = \frac{2\pi r h_1^2}{\lambda} + \frac{2\pi r h_2^2}{\lambda} + \frac{2\pi r h_3^2}{\lambda} + \frac{8\pi r}{9\lambda h}(r^2 + h_0^2)^{\frac{3}{2}} \quad (5)$$

where r denotes the missile section radius, h denotes the missile warhead length, and λ denotes the detection signal wavelength. h_1, h_2, h_3 denote the first stage, second stage and third stage booster rockets respectively. h_0 represents the length of warhead. In fact, the RCS of the missile is also related to the angle between the radar antenna and the vertical direction of missile axial. Thus, the real value of RCS σ can be obtained by $\sigma = RCS * \cos\theta$, where θ represents the direction intersection angle between the radar antenna and the missile axial. The maximum detection range of radar can be calculated according to radar equation. The decision-making process of radar agent is shown below.

If Radar initialization, which includes position and detection parameters
Then Judge if the target lies in the radar detection area
If Target lies in the radar detection coverage area
Then Calculate the radar effective detection range $R_{\max}$ by using radar equation
If Distance between the target and the radar less than current radar detection range
Then Continue performing scan task
If Find target three times in a row
Then Report warning information
Else It is a false alarm, continue tracking mission
End If
End If
End If
End If

2.2 BMDS decision-making constraint model

Taking multi-layer missile defense as an example, we need specific symbols to represent objects and variables which support the subsequent modeling and description. Symbols and definitions in the constraints model are listed in Table 2.

Table 2 Symbols and definitions

Symbol	Definition
m	Total number of intercept missile
i	Number of intercept missile, $i \in \{1, 2, 3, \ldots, m\}$
n	Total number of offensive missile
j	Number of offensive missile, $j \in \{1, 2, 3, \ldots, n\}$
V_i	Cost of intercept missile i
T_j	Threat of offensive missile j
R_j	Responding time to intercept offensive missile j
C_j	Commander intervention factor
D_j	Distance from offensive missile to target
S_j	Velocity constraint factor
W_k	Weight factor
KP_{ij}	Intercept probability of intercept missile i to offensive missile j which satisfies $0 \leqslant KP_{ij} \leqslant 1$
x_{ij}	Boolean variable which determines whether the intercept missile i is selected to intercept offensive missile j

Note that the number of intercept is bigger than the number of offensive missile. The main constraints in the decision-making of BMDS are described below.

(i) Intercept and offensive missile constraint

In order to ensure the security of the defensive asset, each offensive missile needs at least one intercept missile to intercept. In order to maximize the interception effectiveness, we should give priority to the offensive missile of higher threat with a lower cost.

$$Fitness = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} KP_{ij}x_{ij} \cdot T_j}{\sum_{i=1}^{m}\sum_{j=1}^{n} V_i x_{ij}}. \tag{6}$$

The threat of offensive missile j is represented by T_j, the intercept probability of offensive missile j is represented by KP_{ij}, the cost of intercept missile i is represented by V_i. The Boolean variable x_{ij} is set as 1 when the intercept missile i is chosen to intercept offensive missile j.

(ii) Response time brought by radar

The effectiveness of the intercept plan depends on the detection and warning ability of the radar. The poor radar detection ability will lead to limited response time. The offensive missile which has short responding time has a higher priority to be intercepted.

(iii) Commander intervention constraint

The intercept plan is also determined by the command intervention. The command intervention factor is obtained by instruction from superior commander or priori knowledge such as experience and intelligent information. The offensive missile will be intercepted first which has a higher commander intervention.

(iv) Offensive missile distance constraint

The distance from the offensive missile to defense assets is also an important constraint factor. The intercept missile will intercept the offensive missile which is near to the defense assets.

(v) Offensive missile velocity constraint

The offensive missile is preferred to be intercepted when its velocity is higher. Thus, the fitness function is updated as follows:

$$Fitness = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} KP_{ij}x_{ij} \cdot (w_1 T_j) \cdot (w_2 S_j) \cdot (w_3 C_j)}{\sum_{i=1}^{m}\sum_{j=1}^{n} (w_4 V_i x_{ij}) \cdot (w_5 R_j) \cdot (w_6 D_j)} \tag{7}$$

where x_{ij} is the Boolean variable which indicates whether the intercept missile is used, R_j represents the response time brought by radar detection capacity, S_j represents the velocity of the jth offensive missile, C_j represents the commander intervention factor to the jth offensive missile, D_j represent the distance from defensive assets to the jth offensive missile.

In order to increase the adaptability and expansibility of the proposed constraint model, let w_k represent the weight factor of each constraint factor and its initial value be equal to one. Different missile defense missions lead to different configurations of the weight factor.

3. Variable neighborhood negative selection particle swarm optimization (VNNSPSO) algorithm

The PSO algorithm is inspired from the biological group behavior which was first proposed by Eberhart and Kennedy [24] in 1995. PSO is initialized to a group of random particles and the position and velocity of particles are updated during iterations. Each position represents a potential solution which has a specific fitness. The moving velocity of particle is changing with its updating equation which decides the moving distance of particle in the search space. The current position of the particles constituted by n dimensional vector is described as $\boldsymbol{X} = (X_1, X_2, \ldots, X_n)$. The ith element of particle is described as $\boldsymbol{X}_i = (x_{i1}, x_{i2}, \ldots, x_{iD})^{\mathrm{T}}$ in the search space S. The current velocity of the particle is described as $\boldsymbol{V}_i = (V_{i1}, V_{i2}, \ldots, V_{iD})^{\mathrm{T}}$. The individual best position is described as $\boldsymbol{P}_i = (P_{i1}, P_{i2}, \ldots, P_{iD})^{\mathrm{T}}$ while the global best

position is represented by $\boldsymbol{P}_g = (P_{g1}, P_{g2}, \ldots, P_{gD})^{\mathrm{T}}$. The individual fitness is determined by the constraint model described in the earlier section. The individual and global best fitness value will be updated until the termination condition is reached. The velocity and position of each particle is updated by using the following equation.

$$V_{id}^{k+1} = w \cdot V_{id}^{k} + c_1 r_1 (P_{id}^{k} - X_{id}^{k}) + c_2 r_2 (P_{gd}^{k} - X_{id}^{k})$$

$$X_{id}^{k+1} = X_{id}^{k} + V_{id}^{k+1} \tag{8}$$

where w represents the inertia weight which plays the role of balancing the individual and global search while superscript k denotes the current iteration number. V_{id} represents the d dimension velocity of particle i. r_1 and r_2 are random numbers within range [0,1] and c_1 and c_2 are non-negative constant numbers called acceleration factors. In order to avoid invalid searches, the position and velocity of particle are restricted within the range $[-X_{\max}, X_{\max}]$ and $[-V_{\max}, V_{\max}]$ respectively.

The fitness of the particle is calculated and updated during the iteration of the proposed algorithm. The new fitness value will be compared with current individuals and global best values. Then, the individual and global best value will be updated with the bigger one. When the iteration number meets the maximum iteration number $T_{\max}$, the optimization process finishes and the current global best value is the solution of the proposed algorithm.

3.1 Algorithm design

(i) Particle encode

The missile defense can be described as choosing j missiles from i intercept missiles to fulfill the intercept mission. In order to ensure the safety of the defensive asset, each offensive missile needs to be intercepted at least by one intercept missile. The coding rule is shown in Fig. 3 where the number of each missile is integer.

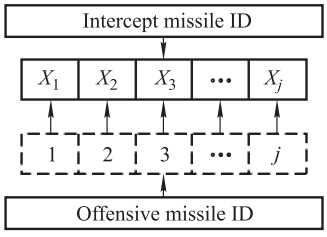

Fig. 3 Coding rule

In Fig. 3, X_j denotes the identification number of the intercept missile.

(ii) Particle velocity and position update

In order to obtain the suitable missile number, we need to take the integer part of the updated position. What's more, the newly generated position must have different values within the range $[1, 2, 3, \ldots, i]$ for the reality that one intercept missile can intercept only one offensive missile. The newly generated particle need to be modified to make it a feasible solution. This can be realized by replacing the duplicate elements in the newly generated position.

During the iteration process, the velocity of the particle might go beyond the range. In that case, the newly generated velocity and position need to be revised using the following equation:

$$V^{k+1} = \begin{cases} V_{\max}, & V^{k+1} > V_{\max} \\ V^{k+1}, & V_{\min} \leqslant V^{k+1} \leqslant V_{\max} \\ V_{\min}, & V^{k+1} < V_{\min} \end{cases} \tag{9}$$

$$X^{k+1} = \begin{cases} X_{\max}, & X^{k+1} > X_{\max} \\ X^{k+1}, & X_{\min} \leqslant X^{k+1} \leqslant X_{\max} \\ X_{\min}, & X^{k+1} < X_{\min} \end{cases} \tag{10}$$

where V^{k+1} represents the updated velocity obtained from the velocity updating equation. The above-mentioned equation shows that the velocity and position will be substituted by the boundary value when it is out of range. Note that if the boundary value is already the element of the particle, a new element is generated which does not duplicate with the existing elements. X^{k+1} represents the updated position obtained from the position updating equation. The maximum velocity and position are set to $V_{\max} = 8$ and $X_{\max} = 16$ in current operational scenario.

(iii) Inertia weight linearly decrease

In order to balance the exploration and exploitation of the algorithm, the inertia weight will be set to decrease linearly with the iteration process as follows:

$$w = w_{\mathrm{end}} + (w_{\mathrm{initial}} - w_{\mathrm{end}})\left(\frac{T_{\max} - k}{T_{\max}}\right) \tag{11}$$

where w_{initial} indicates the initial value of the inertia weight and w_{end} represents the end value during the iteration process. $T_{\max}$ represents the maximum iteration number and k indicates the iteration number of PSO.

(iv) Learning factor

c_1 is the self-learning factor and c_2 is the social learning factor. In order to improve the exploring ability and exploitation ability of the PSO algorithm, the asynchronous varying learning factors are designed as follows:

$$\begin{cases} c_1 = c_{1\,\min} + (c_{1\,\max} - c_{1\,\min})(iter)/(iter_{\max}) \\ c_2 = c_{2\,\min} + (c_{2\,\max} - c_{2\,\min})(iter)/(iter_{\max}) \end{cases} \tag{12}$$

where c_1 and c_2 indicate the learning factor, $c_{\min}$ and $c_{\max}$ represent the minimum and maximum value of the learning factor, $iter$ indicates the iteration number and $iter_{\max}$ indicates the maximum iteration number.

(v) Constriction factor

In order to improve the convergence of the algorithm, some scholars proposed the PSO with the constriction factor. The velocity update equation is updated as follows:

$$V_{id}^{k+1} = K[V_{id}^{k} + c_1 r_1 (P_{id}^{k} - X_{id}^{k}) + c_2 r_2 (P_{gd}^{k} - X_{id}^{k})] \tag{13}$$

where K indicates the constriction factor and it is determined by $K = 2/|2-\varphi-\sqrt{\varphi^2 - 4\varphi}|$ and $\varphi = c_1 + c_2 > 4$.

(vi) Variable neighborhood

In order to balance the exploration and the exploitation ability of PSO, the variable neighborhood algorithm is proposed to avoid getting into the local extreme. Its main idea is to modify the particle's neighborhood radius with the iteration of the algorithm. The diversity of the algorithm can be preserved by the adjustment of the neighborhood radius. The velocity update equation can be revised as follows:

$$V_{id}^{k+1} = wV_{id}^{k} + c_1 r_1 (P_{id}^{k} - X_{id}^{k}) + c_2 r_2 (P_{N_r d}^{k} - X_{id}^{k})] \tag{14}$$

where P_{id}^{k} indicates the individual extreme value and $P_{N_r d}^{k}$ represents the neighborhood extreme value. The subscript of the neighborhood extreme is the neighborhood radius N_r which is variable in the iteration process. Its value ranges with the iteration number. In this paper, we take a variable neighborhood strategy in the early stage of the optimization algorithm.

(vii) Negative selection

Variable neighborhood will improve the ability to jump out of the local extreme value to a certain extent and it is difficult to achieve the desired convergence effect. Here negative selection mechanism is proposed to maintain the diversity of particles and improve the convergence of PSO. This mechanism is derived from the theory of biological immunity. Its main idea is to update partial particles in proportion P_u when the algorithm converges. The convergence of the algorithm is determined by whether the affinity of the particles is bigger than the affinity threshold T_{aff}. The affinity of particle i in the d dimension is determined by the following equation:

$$A_{id} = 1 - |P_{gd} - x_{id}|/(X_{\max} - X_{\min}) \tag{15}$$

where A_{id} is the affinity of particle i in the d dimension. P_{gd} indicates the global extreme value and x_{id} represents the current value. $X_{\min}$ and $X_{\max}$ are the maximum and minimum values of the variables. The affinity of particle i is the average affinity of each dimension which is shown as follows:

$$A_i = \frac{1}{D}\sum_{d=1}^{D} A_{id}. \tag{16}$$

The convergence of the algorithm is determined by whether the affinity of all the particles is bigger than affinity threshold T_{aff}. The negative selection mechanism is shown in Fig. 4.

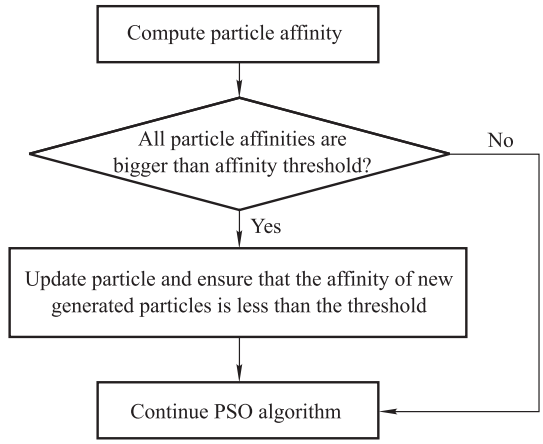

Fig. 4 Negative selection process

The negative selection mechanism will be triggered when all particle affinities are bigger than the affinity threshold. Based on the above analysis, we propose the VNNSPSO algorithm and its implementation process is shown in Fig. 5.

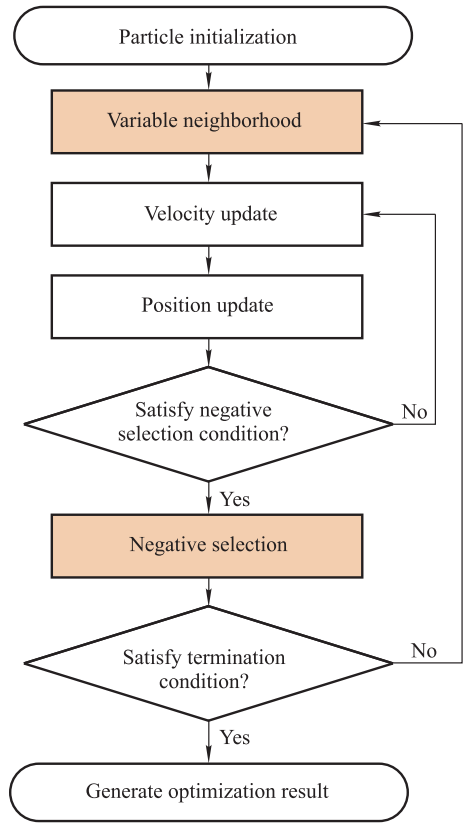

Fig. 5 VNNSPSO algorithm

The proposed VNNSPSO algorithm combines the advantage of variable neighborhood and negative selection mechanism. The variable neighborhood mechanism is adopted in the early stage of the optimization algorithm and negative selection mechanism is adopted in the middle and later period of the algorithm. The proposed modified PSO will enhance the diversity of the particles and avoid falling into local extreme value.

3.2 Analysis and comparison

Set the number of iterations as 1 000, the initial particle number as 200. The inertia weight is equal to 0.729 and learning factors are equal to 1.494 45.

(i) Parameter sensitivity analysis

The main parameters of the proposed algorithm are affinity threshold T_{aff} and update ratio P_u. Let the affinity threshold value vary from 0.91 to 0.99 and the update ratio as 0.3, we can obtain the convergence rate with different affinity thresholds, which is shown in Fig. 6.

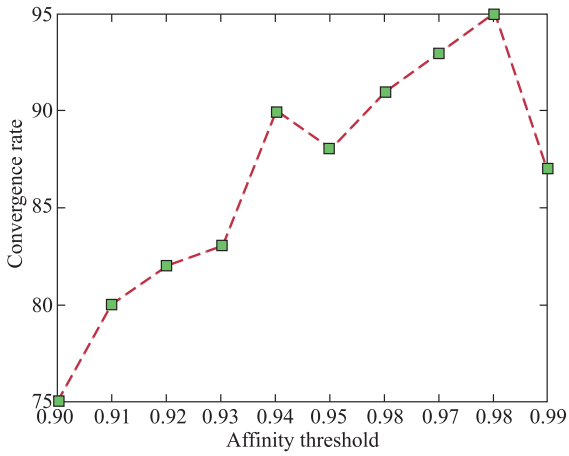

Fig. 6 Convergence rate with different affinity thresholds

According to Fig. 6, the algorithm has a better convergence rate when the affinity threshold is around 0.98 and the update ratio is equal to 0.3. When the affinity threshold is small, the algorithm is easy to get converged. The negative selection mechanism will make the particle swarm update frequently which makes it difficult to converge to the optimal solution. When the affinity threshold is big enough, the algorithm has already fall into the local extreme value.

Let the update ratio vary from 0.1 to 0.5 and the affinity threshold as 0.98, we can obtain Fig. 7.

As we can see, the algorithm has a better convergence rate when the update ratio is around 0.35. When the update ratio is too small, it is hard to jump out of the local extreme value. When the update ratio is too big, the optimization result will fluctuate with the update operation and is hard to converge to the optimal solution. The affinity threshold and update ratio will set as 0.98 and 0.35 in the subsequent study.

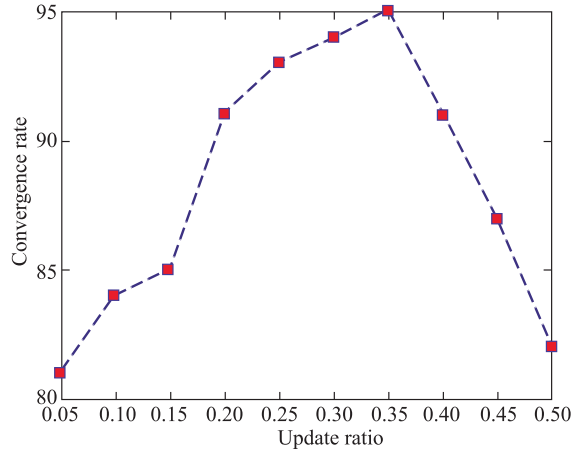

Fig. 7 Convergence rate with different update ratios

(ii) Algorithm convergence analysis

Let the convergence condition be 99.5% of the optimal solution, the affinity threshold as 0.98 and the update proportion as 0.35. After running 1 000 times, the convergence rate of different versions of PSO is compared as shown in Table 3.

Table 3 Convergence rate

Algorithm	Convergence rate/%	Average convergence iteration number
PSO	80.3	975
CFPSO	85.1	937
LDPSO	87.6	850
VNPSO	91.5	715
VNNSPSO	96.8	720

Table 3 shows that the standard PSO has the lowest convergence rate and the proposed VNNSPSO algorithm has the best convergence. The convergence of variable neighborhood PSO (VNPSO) is better than construction factor PSO (CFPSO) and linear decrease PSO (LDPSO) which indicates that the variable neighborhood mechanism will improve the diversity of particle swarm. The iteration number indicates that the proposed VNNSPSO can converge to the optimal solution rapidly. The negative selection mechanism makes the iteration number of VNNSPSO slightly bigger than the VNPSO.

(iii) Algorithm consistency analysis

The average fitness and standard deviation of different versions of PSO are depicted in Table 4.

Table 4 Average and standard deviation

Algorithm	Average	Standard deviation
PSO	6.845 2	0.268 9
CFPSO	6.818 3	0.238 6
LDPSO	6.720 5	0.171 1
VNPSO	6.674 0	0.291 4
VNNSPSO	6.539 4	0.313 1

Note that the average fitness value is obtained by the fitness before the algorithm converged. The data in Table 4 show that the improved PSO has lower average fitness which means that the convergence velocity of the improved PSO algorithm is fast. The LDPSO has the lowest standard deviation since its weight is decreasing during the algorithm iterative process. The standard deviation of VNPSO and VNNPSO is bigger since its variable neighborhood mechanism will enhance the variability of particle swarm. The negative selection mechanism makes the standard deviation of VNNPSO slightly bigger than the VNPSO. The negative selection mechanism will improve the ability to jump out of the local extreme value.

4. DSS design and realization

4.1 Missile defense DSS framework

The DSS is a computer-based information system that helps people make decisions on problems that may be rapidly changing and not easily specified in advance. In the BMDS case, the response time for missile defense intercept is valuable and limited which is typically less than ten minutes. To solve this dilemma, one of the feasible methods is computing and generating interception plan automatically by DSS. The structure of the proposed DSS is shown in Fig. 8.

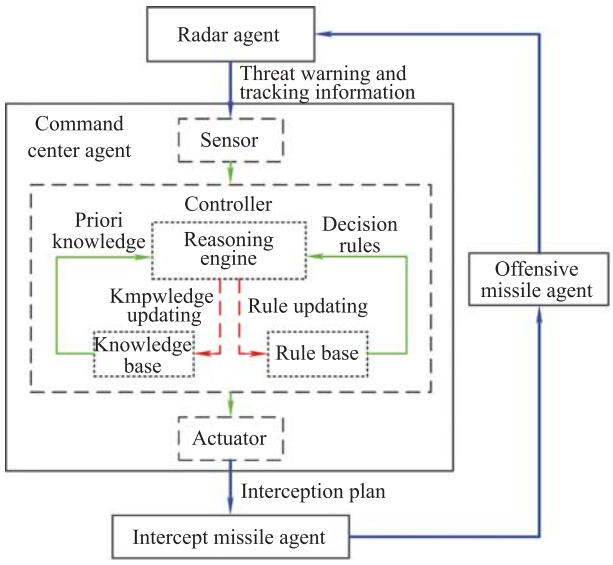

Fig. 8 Framework of missile defense DSS system

Note that the blue arrows indicate the interactions among agents while the green arrows indicate the internal information interaction of the command center agent. The DSS is mainly made of offensive missile agent, radar agent, intercept missile agent and command center agent as shown in Fig. 8. The radar agent percepts the offensive missile and provides warning and tracking information of potential threats to the command center agent. The command center agent is the core of the DSS which composes of a controller, a sensor and an actuator. The decision making reasoning engine gets missile parameters from intercept missile agent and executes decision reasoning under the support of knowledge base and rule base. The modified PSO algorithm is embedded into the reasoning engine of the command center agent which supports the automatic generation of the interception plan.

4.2 Realization of DSS

Lots of agent-based toolkits are developed to assist the development of the agent-based system. The recursive porous agent simulation toolkit (Repast) is a widely used, free and open-source, cross-platform, agent-based modeling and simulation toolkit which is developed by social computing research center at the University of Chicago. Agents will be reserved and managed by the context provided by Repast which can be regarded as an environment agent.

The whole missile defense system-of-systems consists of middle-course interception system, terminal high altitude interception system and terminal low altitude interception system which has sixteen intercept missiles. Each defense layer has a certain number of intercept missiles and different intercept missiles have different intercept capabilities for different offensive missiles.

After embedding the modified PSO algorithm into the reasoning engine of command center agent, we can establish the DSS of missile defense with the Repast platform [25]. After calling and loading the NASA's 3D world wind model into the simulation system, the developed missile defense DSS is shown in Fig. 9.

The running interface of DSS consists of control console interface on the left side and the display interface on the right side. The control console interface is used for parameter configuration which consists of run options, scenario tree, parameters and user panel option box. In the right part of the figure, the output figures are displayed while its title is on the bottom of the figure.

Missile defense operational scenario is described as follows. There are three radars in the missile defense system which has different detection abilities. The red star represents the defensive asset. The command center is deployed around the defensive asset. For the convenience of description, we use the numerals 1st, 2nd, 3rd to indicate different radars. The 1st radar locates near the defensive asset which is represented by a yellow area. The 2nd radar locates in the southeast of the defensive asset which is represented by a green area. The 3rd radar locates in the northeast of the defensive asset which is represented by a red coverage area. Assume that there are eight ballistic missiles aiming at defensive asset using minimum energy trajectory and the eight orbits in color cyan represent the trajectory of offensive missiles.

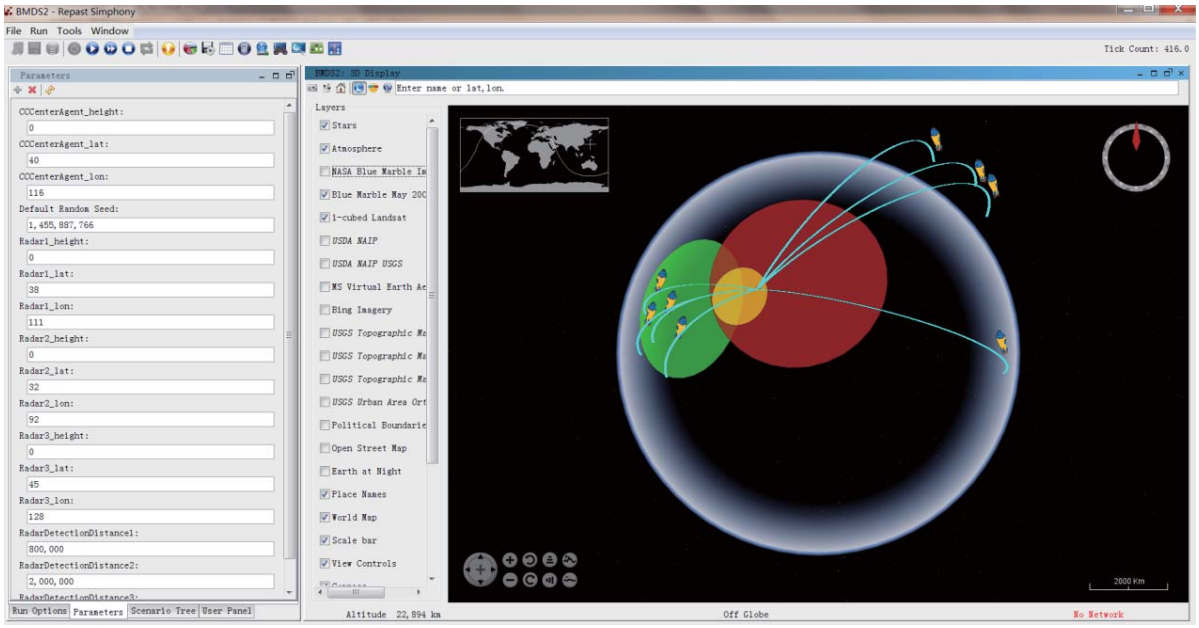

Fig. 9 Missile defense decision support system running interface

4.3 Results and discussion

After initializing running parameters of the DSS simulation system, including missile agent, radar agent, command center agent as well as the parameters of the modified PSO algorithm, we can obtain the following radar detection range curves shown in Fig. 10.

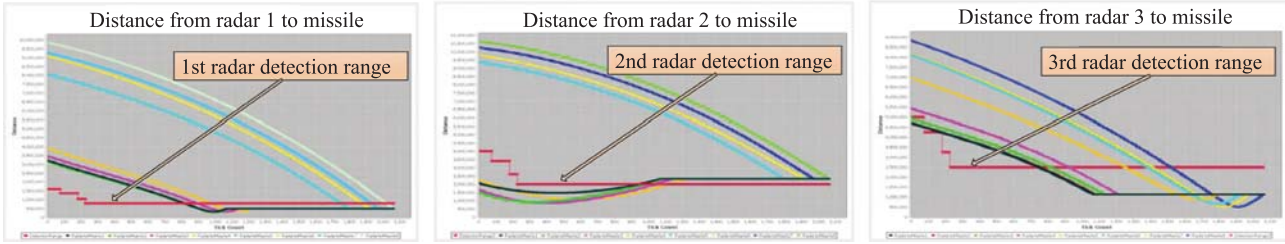

Fig. 10 Radar detection range curves

The horizontal axis represents the simulation time and the vertical axis represents the distance between the offensive missile and radar. The figures from left to right represent the radar detection range curve of radars numbered 1st, 2nd, 3rd. The red curve represents the radar detection range while the other eight curves represent the distance from eight offensive missiles to the radar.

Table 5 shows the generated parameters to form the interception plan.

Table 5 Generated parameters for missile defense

Time/s		First discovery time of 1st radar/s	First discovery time of 2nd radar/s	First discovery time of 3rd radar/s	Intercept missile shutdown velocity/(m/s)	Intercept missile launch time/s	Intercept missile flight time/s
Offensive missile ID	1	814	0	0	520	923	133
	2	815	0	0	851	827	219
	3	888	0	0	1 240	754	316
	4	972	0	920	894	958	223
	5	1 745	–	1 249	1 434	1 402	338
	6	1 874	–	1 378	748	1 752	159
	7	1 931	–	1 409	749	1 880	154
	8	2 018	–	1 521	1 255	1 766	263

The first three columns in Table 5 are first discovery time of offensive missile by radars. The number '0' means that the offensive missile will be detected since it was launched. The symbol "–" means that the radar cannot detect the offensive missile during its whole flight. The command center agent will calculate the offensive missile trajectory and the expected intercept point for intercept missile. After receiving the expected intercept point, the intercept missile agent will calculate the missile launch parameters including shutdown velocity, launch time and flight time. Finally, the DSS will generate the interception plan as shown in Fig. 11.

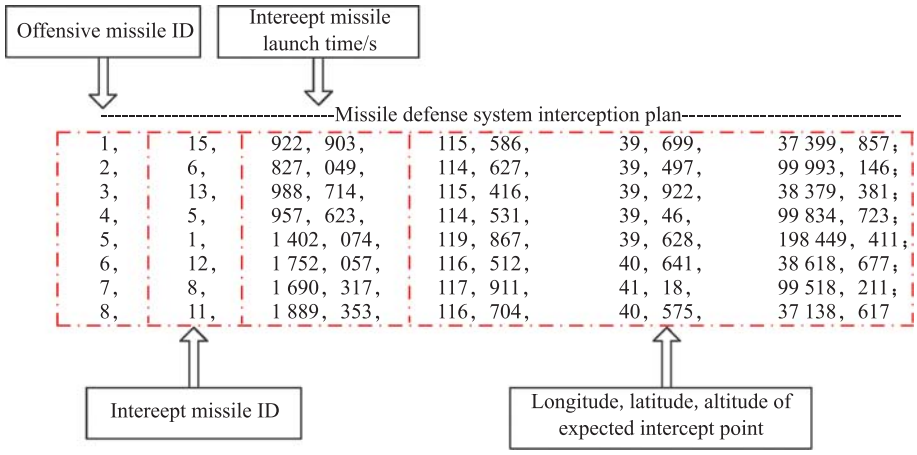

Fig. 11 Missile defense system interception plan

As shown in Fig. 11, the interception plan is constituted by four types of elements which are offensive missile ID, intercept missile ID, intercept missile launch time and expected intercept point. The plan gives the intercept missile launch time and the expected intercept position which will provide important reference information for the decision-making of missile defense.

5. Conclusions

Missile defense DSS plays a decisive role on the effectiveness of missile defense systems. The agent-based modeling and simulation method can support the individual characteristics modeling as well as the complex interactions among agents which provide a new way for the autonomous evolution of the complex system. This paper establishes a multi-agent DSS for missile defense which includes radar agent, missile agent, and command center agent. The constraint model of missile defense is established which takes the constraints brought by radar, intercept missile, offensive missile and command intervention into consideration. In order to realize the optimization of the interception plan, the VNNSPSO is proposed which combines the advantage of variable neighborhood and negative selection mechanism. The proposed algorithm is compared with the standard PSO, CFPSO, inertia weight LDPSO, VNPSO algorithm from the aspects of convergence rate, iteration number, average fitness value and standard deviation. The simulation results verify the efficiency of the proposed algorithm. A missile defense DSS simulation system is designed and realized by using the Repast multi-agent simulation platform. The simulation results show that the DSS can achieve the three-dimensional dynamic display of missile defense operational process and dynamic generation of the interception plan. In a subsequent study, we will explore more intelligent optimization algorithms and efficient organizational structure of the multi-agent system to meet the requirements of large scale, multi-constrained coupling and real-time missile defense problem.

References

[1] R. Handberg. The symbolic politics of ballistic missile defense: seeking the perfect defense in an imperfect world. *Defense & Security Analysis*, 2015, 31(1): 44 – 57.
[2] E. Marshall. A midcourse correction for U.S. missile defense system. *Science*, 2013, 339(6127): 1508 – 1509.
[3] S. Ganguly. India's pursuit of ballistic missile defense. *The Nonproliferation Review*, 2014, 21(3/4): 373 – 382.
[4] U. B. Nguyen. Assessment of a ballistic missile defense system. *Defense & Security Analysis*, 2014, 30(1): 4 – 16.
[5] M. J. Armstrong. Modeling short-range ballistic missile defense and Israel's iron dome system. *Operations Research*, 2014, 62(5): 1028 – 1039.
[6] S. Sommerer, M. D. Guevara, M. A. Landis, et al. Systems-of-systems engineering in air and missile defense. *John Hopkins APL Technical*, 2012, 31(1): 5 – 20.
[7] K. Robert, G. S. Anderson, N. T. Baron, et al. Managing the interstitials, a system of systems framework suited for the ballistic missile defense system. *Systems Engineering*, 2011, 14(1): 87 – 109.
[8] Z. L. Cheng, L. Fan, Y. L. Zhang. A framework for equip-

ment systems-of-systems effectiveness evaluation using parallel experiments approach. *Journal of Systems Engineering and Electronics*, 2015, 26(2): 292 – 300.
[9] J. H. Holland. Studying complex adaptive systems. *Journal of System Science & Complexity*, 2006, 19: 1 – 8.
[10] C. M. Macal, M. J. North. Tutorial on agent-based modeling and simulation. *Journal of Simulation*, 2010, 4(3): 151 – 162.
[11] B. Heath, R. Hill, F. Ciarallo. A survey of agent-based modeling practices. *Journal of Artificial Societies and Social Simulation*, 2009, 12(4): 1 – 35.
[12] M. Barbatia, G. Bruno, A. Genovese. Applications of agent-based models for optimization problems: a literature review. *Expert Systems with Applications*, 2012, 39: 6020 – 6028.
[13] F. P. Goksal, I. Karaoglan, F. Altiparmak. A hybrid discrete particle swarm optimization for vehicle routing problem with simultaneous pickup and delivery. *Computers & Industrial Engineering*, 2013, 65(1): 39 – 53.
[14] R. Kumar, D. Sharma, A. Sadu. A hybrid multi-agent based particle swarm optimization algorithm for economic power dispatch. *Electrical Power and Energy Systems*, 2011, (33): 115 – 123.
[15] I. Cil, M. Mala. A multi-agent architecture for modelling and simulation of small military unit combat in asymmetric warfare. *Expert Systems with Applications*, 2010, 37(2): 1331 – 1343.
[16] O. T. Holland, S. E. Wallace. Using agents to model the kill chain of the ballistic missile defense system. *Naval Engineer Journal*, 2011, 123(3): 141 – 151.
[17] C. J. Lynch, S. Y. Diallo, A. Tolk. Representing the ballistic missile defense system using agent-based modeling. *Proc. of the Military Modeling & Simulation Symposium*, 2013: 1 – 8.
[18] C. D. Connors, J. O. Miller, B. J. Lunday. Using agent-based modeling and a designed experiment to simulate and analyze a new air-to-air missile. *The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology*, 2015, 12(5): 1 – 10.
[18] T. Bui, J. Le. An agent-based framework for building decision support systems. *Decision Support Systems*, 1999, 25(3): 225 – 237.
[20] T. Tanerguclu, H. Maras, C. Gencer. A decision support system for locating weapon and radar positions in stationary point air defense. *Information System Frontiers*, 2012, 14(2): 423 – 444.
[21] L. Y. Li, F. X. Liu, G. Z. Long, et al. Modified particle swarm optimization for BMDS interceptor resource planning. *Applied Intelligence*, 2015, 43(2): 1 – 18.
[22] M. Wooldridge. *An introduction to multi agent systems*. Chichester: John Wiley & Sons, 2002.
[23] R. H. Gooding. A procedure for the solution of Lambert's orbital boundary-value problem. *Celestial Mechanics and Dynamical Astronomy*, 1990(48): 145 – 165.
[24] R. Eberhart, J. Kennedy. A new optimizer using particle swarm theory. *Proc. of the Sixth International Symposium on Micro Machine and Human Science*, 1995: 39 – 43.
[25] M. J. North, N. T. Collier, J. Ozik, et al. Complex adaptive systems modeling with repast symphony. *Complex Adaptive Systems Modeling*, 2013, 1(3): 1 – 26.

Biographies

Zilong Cheng was born in 1988. He received his M.S. degree in control science and engineering from National University of Defense Technology, in 2012. Currently he is a doctoral candidate in School of Aerospace Science and Engineering, National University of Defense Technology. His research interests focus on agent-based modeling and optimization.
E-mail: kdchengzilong@163.com

Li Fan was born in 1977. She received her Ph.D. degree from School of Aerospace Science and Engineering, National University of Defense Technology, in 2006. Currently she is an associate professor in Academy of Equipment. Her research interests include aircraft design, complex system modeling and control.
E-mail: fanlinudt@163.com

Yulin Zhang was born in 1958. He received his Ph.D. degree from Zhejiang University, in 1988. Currently he is a professor in School of Aerospace Science and Engineering, National University of Defense Technology. His research interests include space systems dynamics and control, distributed and responsive space systems, cis-lunar space development and infrastructure construction.
E-mail: y.l.zhang@tsinghua.edu.cn