Chinese Journal of Aeronautics 24 (2011) 493-505

Contents lists available at ScienceDirect

Chinese Journal of Aeronautics

journal homepage: www.elsevier.com/locate/cja

A Distributed Cooperative Dynamic Task Planning Algorithm for Multiple Satellites Based on Multi-agent Hybrid Learning

WANG Chong, LI Jun*, JING Ning, WANG Jun, CHEN Hao

College of Electronic Science and Engineering, National University of Defense Technology, Changsha 410073, China

Received 13 December 2010; revised 24 March 2011; accepted 27 May 2011

Abstract

Traditionally, heuristic re-planning algorithms are used to tackle the problem of dynamic task planning for multiple satellites. However, the traditional heuristic strategies depend on the concrete tasks, which often affect the result's optimality. Noticing that the historical information of cooperative task planning will impact the latter planning results, we propose a hybrid learning algorithm for dynamic multi-satellite task planning, which is based on the multi-agent reinforcement learning of policy iteration and the transfer learning. The reinforcement learning strategy of each satellite is described with neural networks. The policy neural network individuals with the best topological structure and weights are found by applying co-evolutionary search iteratively. To avoid the failure of the historical learning caused by the randomly occurring observation requests, a novel approach is proposed to balance the quality and efficiency of the task planning, which converts the historical learning strategy to the current initial learning strategy by applying the transfer learning algorithm. The simulations and analysis show the feasibility and adaptability of the proposed approach especially for the situation with randomly occurring observation requests.

Keywords: multiple satellites dynamic task planning problem; multi-agent systems; reinforcement learning; neuroevolution of augmenting topologies; transfer learning

1. Introduction

Earth observing satellites (EOSs) receive the remote sensing information from the surface of Earth sent by satellite borne sensor in space, which have advantages of no constraint on any country, long observation, wide coverage, etc. They are widely applied in environment monitoring, military reconnaissance and so forth. In the actual satellite observation process, a mass of observation requests occur randomly. How to support multiple satellites to fulfill complex and randomly occurring tasks cooperatively in the dynamic environment through the effective planning strategy is the current problem of multi-satellite task planning research to be solved urgently [1].

To guarantee the calculation speed, the traditional satellite task planning algorithms [2-6] for the dynamic environment, mostly adjust the existent plan based on heuristic rules. However, the heuristic strategy has greater dependence on the concrete tasks, so that the optimality of the results could not be guaranteed as it is restricted by the task. Moreover, the distributed cooperative task planning for multiple satellites needs to respond to the dynamic environment quickly, but it does not pursuit one-sidedly the solving speed for the algorithm. Balancing the calculation speed, optimization and adaption to the dynamic environment is the key to solve multiple satellites cooperative task planning.

In the process of planning, historical planning information will have an important influence on the follow-up planning results, but the current satellite task planning research lacks utilization of historical information. To tackle the problem of multiple satellite dy-

*Corresponding author. Tel.: +86-731-84574439.
E-mail address: junli@nudt.edu.cn
Foundation item: National High-tech Research and Development Program of China (2007AA120203)

doi: 10.1016/S1000-9361(11)60057-5

namic cooperative planning effectively caused by randomly incoming task, a hybrid learning algorithm based on multi-agent reinforcement learning (MARL) [7-9] and transfer learning (TL) [10-12] is proposed. The reinforcement learning policy is designed based on neural networks which is capable of describing the large-scale state and action space generated by observation request. The accumulated historical cooperative planning information is converted to the current available planning information by applying TL algorithm, which both accelerates the calculation speed and guarantees the result's optimality.

2. Distribution of Multi-satellite Cooperative Dynamic Task

2.1. Problem description

In the multi-satellite cooperative task planning system composed of multiple EOSs with autonomous planning capability, each satellite is considered as an agent for the randomly occurring observation request. Realizing the dynamic distribution for the random tasks by coordination, each agent expects that tasks are assigned quickly and reasonably to every satellite based on autonomous computing, so that global observation obtains maximum benefit under the condition of meeting the constraints of each satellite.

Multi-satellite dynamic task planning (MSDTP) has characteristics as follows:

(1) Satellite agents have differences in capability. The number of resources consumed by different satellite may not be equal when they perform the same task.

(2) Cooperative environment information is partially visible to the agent. Each agent only has limited resources and capability, and the agent could only make the decision in accordance with its own state and the partial environment information.

(3) Planning environment changes due to the randomly occurring observation request, which increases the complexity of solving the problems. In the process of cooperative planning, not only coordination among multiple satellites is considered, but also new tasks are required to be integrated into historical planning results.

In the multi-satellite cooperative dynamic task planning, variables are involved as follows:

(1) Given that planning period is $w_{\text{schedule}}=[t_S,\ t_E]$, where t_S is the start time of the planning and t_E the end time.

(2) N_S agents in total with heterogeneous capability are involved in the cooperative task planning, which are denoted as $\text{SAT}=\{\text{sat}_1, \text{sat}_2, \cdots, \text{sat}_{Ns}\}$. $\forall \text{sat}_k \in \text{SAT}$, $\text{sat}_k=<R_{k,\text{Vst}}, R_{k,\text{Mem}}, R_{k,\text{Eng}}>$, where $R_{k,\text{Vst}}$ means available object visiting window resource of sat_k, $R_{k,\text{Mem}}$ is the available storage resource of sat_k, $R_{k,\text{Eng}}$ means the current available energy.

(3) N_A randomly occurring observation requests in the $[t_S,\ t_E]$ are assumed, denoted by $\text{TSK}=\{\text{TSK}_{t_1}, \text{TSK}_{t_2}, \cdots, \text{TSK}_{t_{N_A}}\}$, $t_S \le t_1 < t_2 < \cdots < t_{N_A} \le t_E$, the observation requests emerging at the moment of i are collected, indicated by $\text{TSK}_{t_i}=\{\text{tsk}_1, \text{tsk}_2, \cdots, \text{tsk}_{N_T}\}$, and $|\text{TSK}_{t_i}|=N_T$. $\forall \text{tsk}_j \in T$ is expressed as $\text{tsk}_j=\{u_j, A_j(k)\}$, $\text{sat}_k \in \text{SAT}$, where u_j is the evaluation gained by finishing tsk_j; $A_j(k)=<A_{j,\text{Vst}}(k),\ A_{j,\text{Mem}}(k),\ A_{j,\text{Eng}}(k)>$ means resource vector demanded by tsk_j from sat_k. Due to the heterogeneous capability, resource demanding vector of different satellites is not equal for the same object tsk_j. $A_{j,\text{Vst}}(k)$, $A_{j,\text{Mem}}(k)$, and $A_{j,\text{Eng}}(k)$ respectively represent the time window resource, memory capacity and energy consuming which are taken by satellite sat_k who observes tsk_j. The necessary condition of the object tsk_j which could be observed is $A_{j,\text{Vst}}(k) \le R_{k,\text{Vst}} \wedge A_{j,\text{Mem}}(k) \le R_{k,\text{Mem}} \wedge A_{j,\text{Eng}}(k) \le R_{k,\text{Eng}}$.

2.2. Model construction

MSDTP could be expressed as a tuple:$<\{S_k\}_{k=1}^{N_S}, \{A_k\}_{k=1}^{N_S}, \{\Omega_k\}_{k=1}^{N_S}, P, O, \{C_k\}_{k=1}^{N_S}>$. In which,

(1) S_k is the state space of each satellite agent. $s_{k,t} \in S_k$ is the state of sat_k at time t which is expressed as a tuple $<\text{CR}_{k,t}, \text{ST}_{k,t}>$, in which $\text{CR}_{k,t}=<\text{cr}_{t,\text{Vst}}^k, \text{cr}_{t,\text{Mem}}^k, \text{cr}_{t,\text{Eng}}^k>$ is the available capability vector of sat_k at time t, i.e. the remaining capability vector. And $\text{ST}_{k,t} \subseteq \bigcup_{i=1}^{i \le t} \text{TSK}_{t_i}$ is the current selected task set.

(2) $A_k=\{a_{k,t} | a_{k,t} \in P(\bigcup_{i=1}^{i \le t} \text{TSK}_{t_i})\}$ is the action set of sat_k, where $P(\bigcup_{i=1}^{i \le t} \text{TSK}_{t_i})$ means the power set of the observation request set $\bigcup_{i=1}^{i \le t} \text{TSK}_{t_i}$ at time t.

(3) $\Omega_k=\{\text{ST}_{i,t} | I \in \text{SAT} \wedge i \ne k\}$ is task planning information of sat_k interacting with other satellites at t.

(4) $P=\Pr(s_1', \cdots, s' | s_1, a_1, \cdots, s_k, a_k)$ means the joint state transfer probability with consideration of the current state and selected action, determined by action of each satellite.

(5) O is the object of MSDTP, while satisfying the constraints for each satellite, it determines $\text{ST}=\bigcup_{\text{sat}_k \in \text{SAT}} \text{ST}_k$ (the set of tasks to be observed), which makes the most benefit by SAT through completing TSK (the set of dynamic task). That is,

$$O = \max \sum_{k \in \text{SAT}} \sum_{j \in \text{ST}_k} u_j \cdot x_j(k) \tag{1}$$

where the decision variable is

$$x_j(k) = \begin{cases} 1 & \text{sat}_k \text{ observes tsk}_j \\ 0 & \text{Otherwise} \end{cases} \tag{2}$$

(6) C_k is the constraints of each satellite, consisting of Eqs. (3)-(6):

$$\sum_{k \in \text{SAT}} \sum_{j \in \text{ST}_k} A_{j,\text{Eng}}(k) \le R_{\text{Eng}}^k \tag{3}$$

$$\sum_{k\in \mathrm{SAT}} \sum_{j\in \mathrm{ST}_k} A_{j,\mathrm{Mem}}(k) \le R^k_{\mathrm{Mem}} \tag{4}$$

$$\sum_{k\in \mathrm{SAT}} \sum_{j\in \mathrm{ST}_k} A_{j,\mathrm{Vst}}(k) \le R^k_{\mathrm{Vst}} \tag{5}$$

$\forall \mathrm{tsk}_{j_1}, \mathrm{tsk}_{j_2} \in T, j_1 \neq j_2$:

$$r_{j_1,\mathrm{Vst}}(k) \wedge r_{j_2,\mathrm{Vst}}(k) = \varnothing \tag{6}$$

where $r_{j,\mathrm{Vst}} \in R_{k,\mathrm{Vst}}$ means the time window resource taken by tsk_j, and $s_{j,k}$ and $e_{j,k}$ are start time and end time of the task.

Eq. (3) means the observation process could not violate the energy constraint of satellite; Eq. (4) prevents the observation from exceeding the memory capacity; Eq. (5) means the time window of the total selected tasks could not exceed the available time window; Eq. (6) is to ensure that the time window resources do not collide caused by different tasks of the same satellite.

3. Framework of Hybrid Learning Algorithm

Through analyzing the model, it is known that the process of solving MSDTP is a problem of decentralized Markov decision process (DEC-MDP). Bernstein pointed out that the optimal strategy of solving DEC-MDP has the complexity of NEXP-complete [13].

To tackle the DEC-MDP problem, the method of reinforcement learning is mostly adopted. If the reinforcement learning algorithm [14-15] is adopted based on value function iteration (VFI) to tackle MSDTP, the amount of time and storage space is needed although optimal strategy could be found, and there are two factors which restrict the VFI, with the task scale expanding caused by increasing observation requests. One is the convergence speed: in order to find optimal policy, VFI has to experience the whole state-action space, in the process of which consumes a lot of time, therefore the reinforcement learning algorithm based on VFI converges slowly. The other is the reuse of historical planning information: the result of reinforcement learning always depends on the concrete indication of state-action, while in MSDTP the state and action space change with observation request increasing, which causes the historical cooperative planning useless, and the complete relearning will pay a very high price.

Based on the analysis above, a multiple satellites dynamic hybrid learning algorithm (MSDHLA) combined with reinforcement learning based on policy search [16-21] (specifically, the multi-agent cooperative NEAT (MACoNEAT) algorithm described in Section 4) and transfer learning [10-12] is proposed for randomly occurring observation requests. Fig. 1 shows the

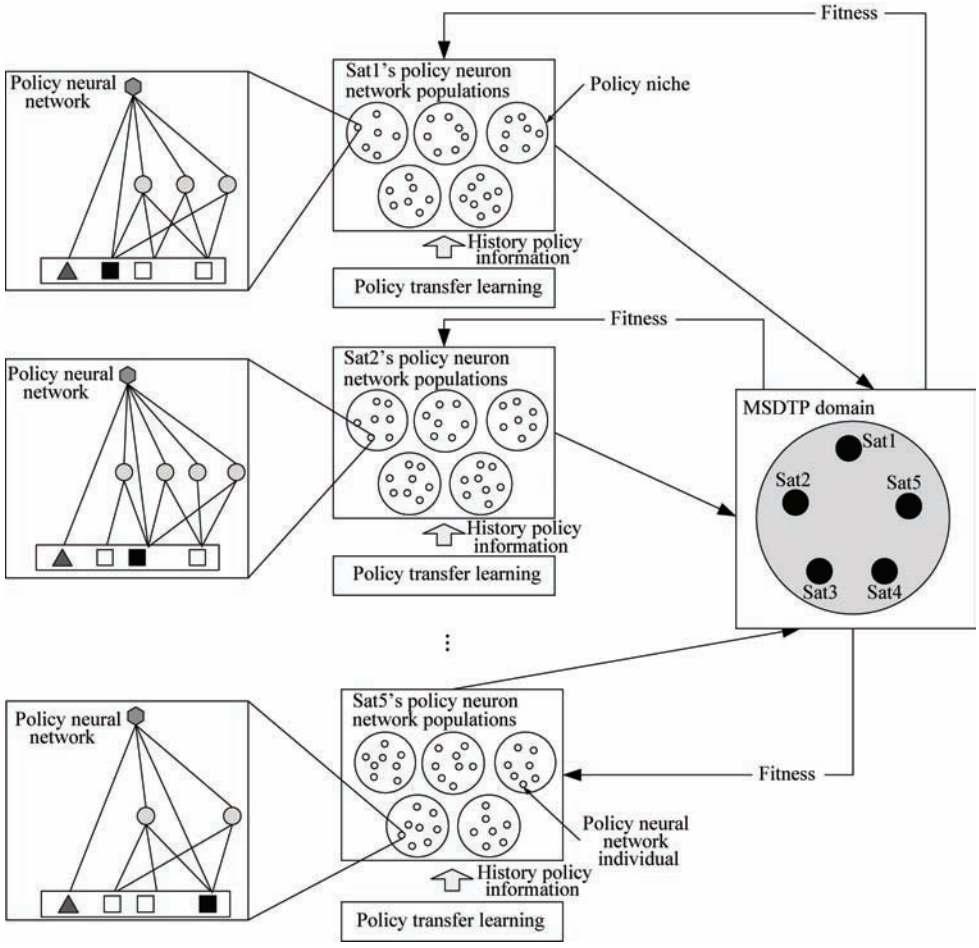

Fig. 1 Framework of multiple-satellite dynamic hybrid learning algorithm.

framework of the algorithm. A search policy is described by a neural network. Some approximate policy neural network individuals form a policy inche. Therefore, the populations in every satellite agent consist of such inches. All the policy populations of the agent constantly search optimal planning strategy iteratively through coevolution. The output of each neural network denotes a task sequence of such satellite. Thus in order to calculate the fitness of every individual in the policy population, the individual should be decoded in MSDTP domain, and then is calculated with representative individuals from policy population from other satellites, which is detailed in Section 4.4. When the change of state space is caused by randomly occurring observation requests, the learning rate for newly added tasks is improved through learning experience, and results of historical assigned tasks for multiple satellites are reused by transfer learning.

4. MACoNEAT Algorithm for Multiple Satellites

In the MSDTP, an exponential relationship between state and action space and current cumulative task scale will lead to "curse of dimensionality" if the method of temporal difference (TD) and so forth are used for iterative estimation based on VFI. Different from the algorithm of TD solved by VFI iteration, policy search [16-22] in the reinforcement learning searches policy space by direct utilization of optimization algorithm to find the optimal policy. And neuro evolution (NE) algorithm [17-18,21-22] iteratively optimizes neural network populations by using search capability of NE, which could tackle effectively reinforcement learning problem in high dimensional state space.

However, since traditional NE approaches [18,22] have determined the structure of neural network in advance, including number of hidden nodes and the connections, the weight of neural network is just optimized through NE algorithm. And what is more, the topology of neural network has great influence on the optimality of the result.

For that, according to the idea of NeuroEvolution of augmenting topologies (NEAT), the algorithm of MACoNEAT is proposed, combining the multi-satellite cooperative task planning problem. In MACoNEAT, the topological structure and the weight of policy individual of neural network evolve at the same time in the populations of each satellite agent. Avoiding the task observation redundancy of multiple satellites, the joint fitness is calculated among agents, which improves the global planning effectiveness.

4.1. Genetic encoding

Before genetic encoding, the first thing is to describe the task planning problem of each satellite in MSDTP in the way of neural network. For the agent sat_k in the MSDTP, each neural network represents the task selection policy of sat_k under the current task distribution. Each input node in the neural network represents the segment of learning policy at time t; each output represents the task selection action of sat_k. In order to make the subsequent operation of crossover and mutation, a neural network is firstly transferred into a chromosome. Considering the characteristics of the policy neural network which describes the problem of MSDTP, chromosome is generated by adopting the length-variable genetic encoding for policy neural network of agent. As illustrated in Fig. 2, every single chromosome contains some connection genes, where each connection gene contains interconnected input and output node identity, connection weight, current state and global historical identity (the details are in Section 4.3). The length of each chromosome is determined by input nodes, output nodes, hidden nodes

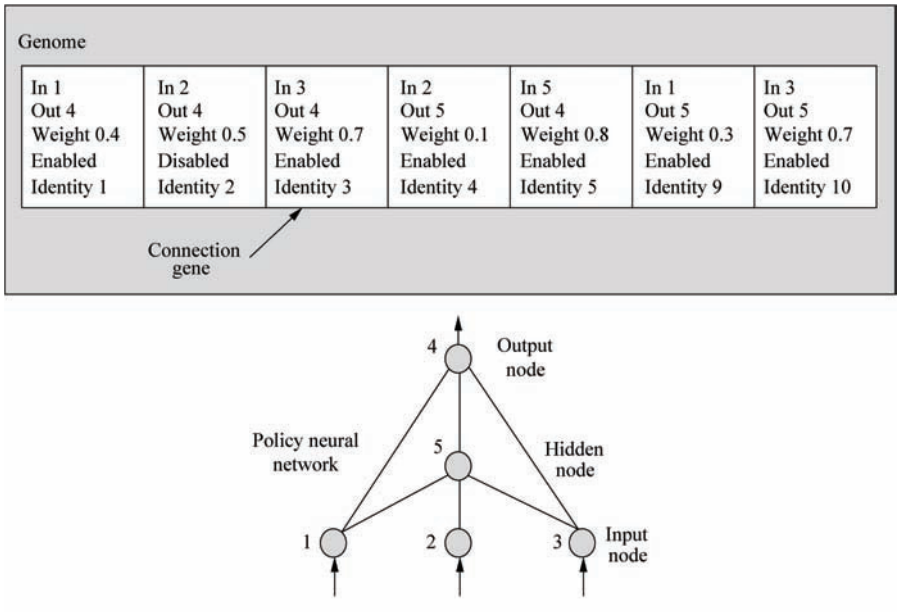

Fig. 2 Encoding of reinforcement learning policy neuron network.

and the connection way between nodes.

It should be noted that all the neural network individuals of initial policy population in each satellite agent only consist of connection gene constructed by input and output nodes.

4.2. Mutation operators

In MACoNEAT, mutation consists of two mutation operators, which are node connection weight mutation operator and network structure mutation operator. The mutation operator of connection weight is the same as other NE algorithms, which is to adjust the weight of each connection gene with the probability of m_c. Structural mutation operators achieve the changes of structure of chromosome group by two ways of connection mutation and node mutation. The connection mutation operators establish connection by stochastic weight between unconnected nodes in the existent neural network (see Fig. 3(a)); node mutation is operated by adding the node n_{new} to the current connection, cutting off existent connection and generating two new connections, and setting n_{new} as the connection weight of input node to 1 while n_{new} as the output node consistent with original connection (see Fig. 3(b)).

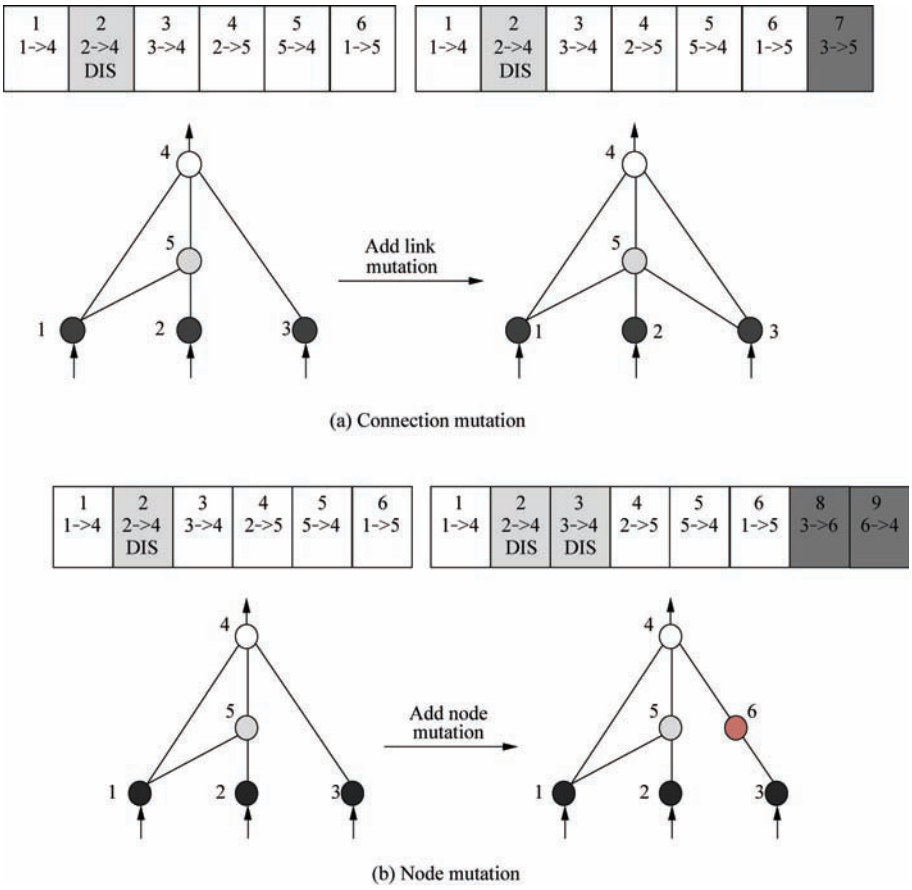

Fig. 3 Illustration of structural mutation operators.

4.3. Crossover operators

With the evolution of MACoNEAT, the neural network individual containing in each agent policy population changes the topological structure of neural network through a variety of operations. The first thing is to match the connection gene of parent chromosome if crossover operation is to be made. For this reason, a global historical identity (GHI) is employed to record the additional connection together for all individuals in populations, and the same connection gene has the same GHI.

Before performing the crossover operation, both of connection genes in parent chromosome are firstly matched according to their GHIs: if their GHIs are equal, the genes are matched; otherwise, the matched connection genes do not exist in the two parent chromosomes. There are two cases classified as “disjoint” and “excess” based on the position of unmatched connected gene, as shown in Fig. 4. When the crossover operation is performed, firstly, two parent individuals are sorted according to the GHI order. The matched genes in parent individuals are stochastically selected as offspring gene. When “disjoint” and “excess” happen, the connection genes are inherited from the more suitable parent. In this case, equal fitness is assumed, so the disjoint and excess genes are also inherited randomly.

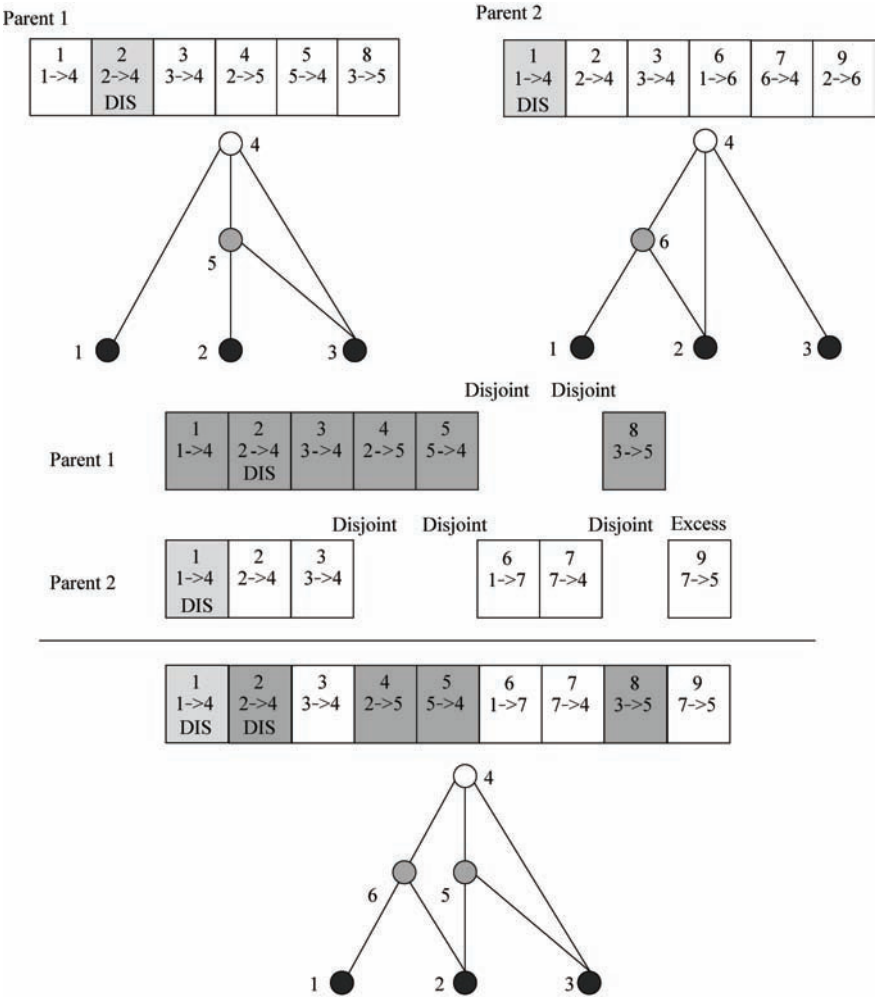

Fig. 4 Illustration of crossover operators.

In this way, GHIs allow NEAT to perform crossover using linear genomes without the need for expensive topological analysis.

4.4. Fitness calculation

There are two factors to be considered for calculating fitness of policy neural network individuals in the policy population of every agent: first, infeasible policy individuals violating constraints of satellite are necessarily generated after performing the operations of crossover and mutation. Mitsuc, et al. [23] pointed out that optimal solution would appear on the boundary between feasible solution domain and infeasible solution domain, for that the punishment function method is employed and the optimal solution could be approached by two ways of feasible solution domain and infeasible solution domain; second, while each satellite agent iteratively searches for the optimal policy, the redundant observation among satellites due to repeated observation is necessarily to be considered. Based on Tan's [24] innovation of joint fitness calculation based on cooperative coevolution, the fitness of individual in the population is determined, as illustrated in Fig. 5. For that, the fitness of policy individual is defined by

$$\text{Fitness}_k = f^k_{\text{utility}} - f^k_{\text{punish}} - f^k_{\text{stimulate}} \tag{7}$$

where $f^k_{\text{utility}} = \sum_{j\in \text{sat}_k} u_j \cdot x_j(k)$ is the sum of utilities of current selected tasks, f^k_{punish} the punishment function of constraints defined by

$$f^k_{\text{punish}} = c_{\text{c}} \left(\frac{R^{\text{ex}}_{k,\text{Vst}}}{R_{k,\text{Vst}}} \cdot \frac{\sum_{\text{tsk}_j \in \text{TSK}^{\text{Vst}}_{k,\text{CONS}}} u_j}{|\,\text{TSK}^{\text{Vst}}_{k,\text{CONS}}\,|} + \frac{R^{\text{ex}}_{k,\text{Mem}}}{R_{k,\text{Mem}}} \cdot \frac{\sum_{\text{tsk}_j \in \text{TSK}^{\text{Mem}}_{k,\text{CONS}}} u_j}{|\,\text{TSK}^{\text{Mem}}_{k,\text{CONS}}\,|} + \frac{R^{\text{ex}}_{k,\text{Eng}}}{R_{k,\text{Eng}}} \cdot \frac{\sum_{\text{tsk}_j \in \text{TSK}^{\text{Eng}}_{k,\text{CONS}}} u_j}{|\,\text{TSK}^{\text{Eng}}_{k,\text{CONS}}\,|} \right) \tag{8}$$

where $R^{\text{ex}}_{k,\text{Vst}}$ is the excess time window of current selected tasks, compared with $R_{k,\text{Vst}}$. $R^{\text{ex}}_{k,\text{Mem}}$ is the excess memories. $R^{\text{ex}}_{k,\text{Eng}}$ is the excess energy consumption, compared with $R_{k,\text{Eng}}$. $\text{TSK}^{\text{Vst}}_{k,\text{CONS}}$, $\text{TSK}^{\text{Vst}}_{k,\text{CONS}}$

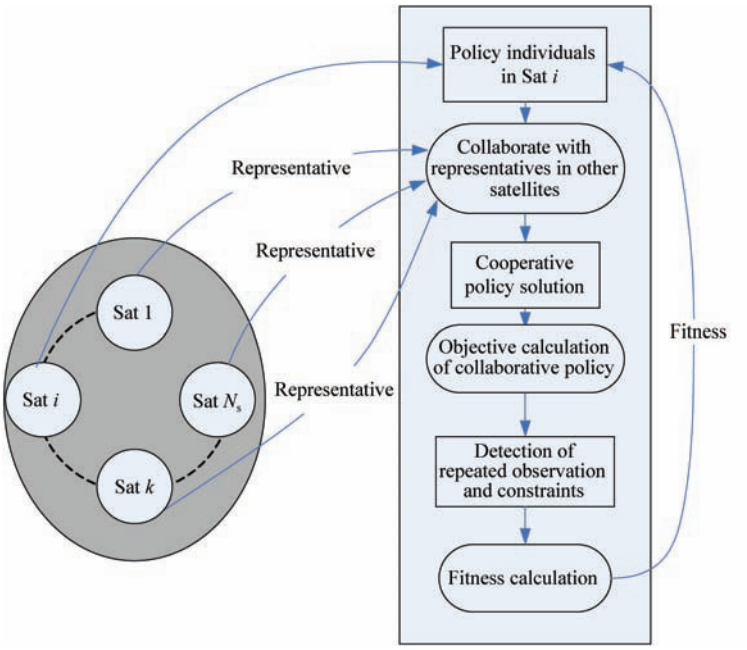

Fig. 5 Illustration of multi-satellite cooperative fitness assignment.

and $\mathrm{TSK}_{k,\mathrm{CONS}}^{\mathrm{Vst}}$ mean the task set of violating visiting window, memory storage, energy consumption respectively. $c_c \in (0, 1)$ is the coefficient of constraint punishment. $f_{\mathrm{stimulate}}$ is the multi-satellite stimulated function, determined by

$$f_{\mathrm{stimulate}}^{k} = c_s \left\{ \sum_{\mathrm{tsk}_j \in \mathrm{TSK}_{\mathrm{RDN}}} \left\{ \frac{r_j}{r_{\max}} \cdot \frac{2u_{\max}}{1+\exp[(u_j - u_{\min})/(u_{\max} - u_j)]} \right\} \right\} \tag{9}$$

where $\mathrm{TSK}_{\mathrm{RDN}}$ means the task set of repeated visits generated by the current satellite and other satellite, r_j is the number of repeated observation of tsk_j, $r_{\max}$ is the maximal number of repeated observation, $u_{\max}$ and $u_{\min}$ respectively mean the maximal utility and minimal utility of tasks, c_s is the coefficient of stimulation.

4.5. Specification

After the crossover and mutation operation, the initial fitness of offspring individual of policy neural network with new topological structure may be low. To ensure that offspring individuals with topological innovation are able to store in the subsequent evolution to maintain the diversity of the policy population, the individual with similar topological structure is composed of sub-population, and is optimized iteratively within the niche.

Based on GHI, the similarity of topological structure of individuals is calculated and the speciation is achieved, which is to avoid complex topological calculation between different individuals. Thus, species is established. The compatibility distance between individuals is defined by

$$\varphi = \frac{c_1 E}{N} + \frac{c_2 D}{N} + c_3 \overline{W} \tag{10}$$

where E and D are respectively the number of connection gene "excess" and "disjoint" by two chromosomes, N means the number of the connection genes in one of the two chromosomes larger in scale, $\overline{W}$ is the average weight difference of match genes, c_1, c_2, and c_3 are respectively the weight coefficients of E, D and $\overline{W}$.

Based on Eq. (10), φ for a new chromosome g is calculated with chromosome representatives of each species. If the compatibility is less than the threshold value φ_t, g will be added to the species. Otherwise, a new species is created with g as its representative.

In the process of speciation, explicit fitness sharing [19] is applied in the population of agent to calculate the fitness of species $F'(i)$ based on Eq. (11), in order to avoid the situation that any good individual in some species takes over the entire population excessively and then planning results get trapped into the suboptimal.

$F'(i)$ is determined by the compatibility distance of i and other individuals in the population:

$$F'(i) = \frac{F(i)}{\sum_{j=1}^{n} \mathrm{sh}(\varphi(i,j))} \tag{11}$$

The sharing function sh(·) is set to 0 when the distance $\varphi(i,j)$ is above the threshold φ_t, otherwise, sh($\varphi(i,j)$) is set to 1. Thus explicit fitness sharing lowers the individual fitness with larger scale in sub-population. Every species is assigned a potentially different number of offspring in proportion to the sum of adjusted finesses $F'(i)$ of its member avoiding that any

species controls the entire population. Thus, the diversity of population is guaranteed and the premature is avoided.

5. Incremental Planning Strategy Transfer Learning

For TSK_{t_t}, the set of randomly occurring observation requests at t_t, agent needs to integrate it with the historical observation set $\bigcup_{i=1}^{t-1}\mathrm{TSK}_{t_i}$ to ensure the global optimum of multi-satellite cooperative planning results, and to re-learn for new observation set $\bigcup_{i=1}^{t-1}\mathrm{TSK}_{t_i}$, which is very time-consuming. The expansion of the new observation set in MSDTP expands is based on historical observation, and the learning speed of the new task set shall be improved by using multi-satellite historical planning policy information.

Based on the idea of TL [10-12], we design an incremental planning strategy transfer learning algorithm (IPSTL). Satellite agent learning policy mapping function is given by

$$\pi_{k,\mathrm{target}} = \rho_{k,H}(\pi_{k,\mathrm{source}}) \tag{12}$$

where $\pi_{k,\mathrm{source}}$ means a individual in policy neural network population while sat_k turns towards the historical task set, $\pi_{k,\mathrm{target}}$ the individual in the target initial policy neural network population of sat_k for the new observation set $\bigcup_{i=1}^{t}\mathrm{TSK}_{t_i}$, and $\rho_{k,H}$ the transfer learning function of sat_k.

Satellite agent learning policy is described by neural network in MACoNEAT, thus $\rho_{k,H}$ is used to establish the mapping function between individuals of historical policy neural network and policy target neural network.

The state and action space of history and target reinforcement learning have changed due to TSK_{t_t}, which leads to the corresponding changes of the input and output in policy neural network. Therefore, the mapping functions $\chi_{H,X}$ and $\chi_{H,A}$ which denote the input and output in history-target policy neural network are designed. $x_{j,\mathrm{source}}=\chi_{H,X}(x_{i,\mathrm{target}})$ means each input node in the mapping target policy neural network is mapped to the most relevant input node in history policy neural network; in a similar way, $x_{j,\mathrm{source}}=\chi_{H,A}(x_{i,\mathrm{target}})$ means each output node in the mapping target policy neural network is mapped to the most relevant output node in history policy neural network.

Under the given conditions of $\chi_{H,X}$, $\chi_{H,A}$ and the history policy individual $\pi_{k,\mathrm{source}}$, hidden nodes with the same number as history policy neural network are added to initial target policy neural network. The defined function $h_{\mathrm{source}}=\delta(h_{\mathrm{target}})$ represents the corresponding relationship of the hidden nodes between history and target policy neural network. Therefore, each node n in $\pi_{k,\mathrm{target}}$ could be mapped to $\pi_{k,\mathrm{source}}$ by function ψ.

$$\psi(n)=\begin{cases}\chi_{H,X}(n) & \text{if } n \text{ is an input node}\\ \chi_{H,A}(n) & \text{if } n \text{ is an output node}\\ \delta(n) & \text{if } n \text{ is an hidden node}\end{cases} \tag{13}$$

According to ψ, the corresponding node in $\pi_{k,\mathrm{source}}$ is copied and $\pi_{k,\mathrm{target}}$ connection is established. For the nodes n_i and n_j in $\pi_{k,\mathrm{target}}$, if there exists a connection between the corresponding nodes $\psi(n_i)$ and $\psi(n_j)$ in $\pi_{k,\mathrm{source}}$, a connection with equal weighted value between n_i and n_j will be established.

Through $\rho_{k,H}$, the history learning policy is transferred to the initial target learning policy. All target learning policies inherit the topological structure and weigh information to accelerate the learning speed of new observation for multiple satellites. The specific process for TL algorithm of sat_k policy is given in Fig. 6.

Algorithm: Incremental history policy transfer learning (S_k A_k, p_k, $P_{k,\mathrm{HIS}}$)

// S_k: *set of all states in* sat_k, A_k: *set of all actions in* sat_k, p_k: *population size of* sat_k

// $P_{k,\mathrm{HIS}}$: *history learned policy population*

1: for $\alpha \leftarrow 1$ to p_k do

2: $P_k[\alpha] \leftarrow$ CONSTRUCT-POLICY-NEURAL-NETWORK(S_k, A_k)

// construct a target neural network where nodes are determined by the current tasks

3: ADD-HIDDEN-NODES ($P_k[\alpha]$, $P_{k,\mathrm{HIS}}[\alpha]$)

//add the same number of hidden nodes to $P_k[\alpha]$ as $P_{k,\mathrm{HIS}}[\alpha]$

4: for each pair of nodes n_i and n_j in $P_k[\alpha]$ do

5: if *link* ($\psi(n_i, n_j)$) in then

6: ADD-LINK(n_i, n_j) *//Add link* (n_i, n_j) *with weight identical to link* ($\psi(n_i, n_j)$)

Fig. 6 Pseudocode of policy transferring algorithm.

Therefore, the detailed process of multiple satellites dynamic hybrid learning algorithm is shown in Fig. 7.

6. Convergence Analysis

Based on the analysis above, it is obvious that the IPSTL algorithm does not affect the convergence of MSDHLA, which is decided by MACoNEAT.

In order to analyze the convergence of MACoNEAT, the definition of Nash optimal is given:

Definition 1 (Nash Optimal) For the MSDTP and a set of policies $P^*=\{P_1^*, P_2^* \cdots, P_{N_S}^*\}$, if $\forall P_i$ in sat_i satisfies Eq. (14), then it is considered that the process of finding joint policies of multiple satellites achieves the Nash equilibrium. And then, P^* is the Nash optimal.

$$F_i(\mathrm{ind}_i, \{\mathrm{ind}_{j\neq i}^*\}) \le F_i(\mathrm{ind}_i^*, \{\mathrm{ind}_{j\neq i}^*\}) \quad \forall j=1, 2, \cdots, N_S \tag{14}$$

From the concept of Nash optimal, it is seen that each decision maker in a distributed system can only decide its own decision variables. A rational decision maker should choose the policy without affecting other decision makers. Because the decision results would affect each other, each decision maker could not change its decisions to get better performance of the system, then the system achieves equilibrium.

Algorithm: MSDHLA (S_k, A_k, p_k, $m_{k,n}$, $m_{k,l}$, $m_{k,c}$, e, g, $P_{k,HIS}$)

1: *//S_k: set of all states in* sat$_k$, *A_k: set of all actions in* sat$_k$, *p_k: population size of* sat$_k$,
2: *//$m_{k,n}$: node mutation rate, $m_{k,l}$: connection mutation rate, $m_{k,c}$: crossover rate,*
3: *//g: number of generations, $P_{k,HIS}$: history learned policy population, e: elite population size of* sat$_k$,
4:
5: P[]←Incremental history policy transfer learning (S_k, A_k, p_k, $P_{k,HIS}$)
// create new population P based on history learned policy neural networks
6: for i←1 to g do
7: for j←1 to p_k do
8: $a_{k,i}$←argmax activation ($P[j]$), $s_{k,i}$, $a_{k,i}$ *// select action with the highest activation*
9: $P[j]$.fitness=*TakeCollabrationAction* ($a_{k,i}$) *// calculate the fitness with other satellites' actions*
10: P'[]←new array of size p *// new array will store next generation*
11: P'[]←CHOOSE-ELITE-INDI (P[]) *// choose elite individuals from P*
12: for j←e to p do
13: $P'[j]$← ROULETTE-SELECTION ((P'[]) *//select individual to mate in roulette way*
14: $P'[j+1]$←ROULETTE-SELECTION (P'[])
15: with probability $m_{k,n}$: ADD-NODE-MUTATION ($P'[j]$) *//add node to new network*
16: with probability $m_{k,l}$: ADD-CONNECTION-MUTATION ($P'[j+1]$) *//add connection to new network*
17: with probability $m_{k,c}$: CROSSOVER ($P'[j]$, $P'[j+1]$) *//crossover with individuals*
18: P[]← P'[]

Fig. 7 Pseudocode of MSDHLA.

For the MSDTP, there is an implication in definition 1 that: each satellite optimizes his local policy under the assumption of the optimal of other satellites in the process of local optimization have been given. Only in this way could the system get the Nash optimal. It is known that the fitness of policy neural network individual in each satellite is calculated with the best individuals of other satellite subpopulations. Therefore, as long as each satellite converges to its local optimal policy, the distributed system could converge to the Nash optimal.

Next, we just need to prove that the individuals in the policy neural network subpopulations could converge to the optimal.

Lemma 1 [25] Corresponding to general genetic algorithm with the finite homogeneous Markov chains, if the population keeps the best solution before selection operation in each generation, the genetic algorithm converges to the optimal with probability 1.

Theorem 1 In MACoNEAT, the policy neural network subpopulations of each satellite converge to each optimal of its own subpopulation with probability 1.

Proof The framework of MACoNEAT shows that once the length of chromosome and number of subpopulations is determined and finite, both the individual space formed by all individuals and subpopulation space formed by subpopulations in each generation are finite. Because the parameters used in the algorithm do not change over time, and the state transition is not dependent on generation but on the selection, crossover, and mutation operators, MACoNEAT is the one of the genetic algorithms with the finite homogeneous Markov chains. So it is known that MACoNEAT converges to optimal with the probability 1. The original proposition is proved.

Therefore, the MSDHLA could converge to the Nash optimal.

7. Experiments

In order to test the performance of MSDHLA, we design some experiments as follows. The observation requests are randomly generated in different scales under uniform distribution and two-dimensional Gaussian distribution in the area of latitude 0°-60° and longitude 0°-180°. The utilities of tasks are random integer in $[u_{min}, u_{max}]$, u_{min}=1, u_{max}=3.

The algorithm simulates with 5 satellites. The time window of satellites for object observed is calculated by STK [26], and the details for the data set are given in Table 1, in which, SN means the serial number of test data, OR the number of objects, TR the number of time window generated according to each satellite, and DS the distribution scheme of the objects. For comparison, all satellites in the planning are presumed to have the same spatial resolution.

The program of the algorithm is written by C#, whose compiler environment is Microsoft Visual Studio 2005, running on the PC with configuration of Pentium E5300 2.6 GHz, 2 GB RAM. Algorithm parameters used are as follows: maximum number of iterations MaxIter=100 000, the population sizes are PopSize=100. m_n=0.6, m_l=0.9, m_c=0.8, c_1=1.0, c_2=1.0, c_3=2.0, φ_t=3.0, λ=2.5, and γ=3.5.

Table 1 Test data set

SN	OR	TR	DS
PH120	30	55	NORMAL
PH121	30	49	GAUSS
PH122	30	61	RANDOM
PH123	50	89	NORMAL
PH124	50	114	GAUSS
PH125	50	96	RANDOM
PH126	200	495	NORMAL
PH127	200	542	GAUSS
PH128	200	955	RANDOM
PH129	300	1 244	NORMAL
PH130	300	1 613	GAUSS
PH131	300	1 573	RANDOM
PH131	300	1 573	RANDOM

7.1. Comparison of experimental results

The planning results of the three algorithms without considering the conditions of history information of TL are compared under the condition of different scales and distributions of tasks. Where, nonstationary converging policies (NSCP) is a cooperative algorithm based on temporal difference reinforcement learning. ECNP is an extended algorithm of task distribution base on contract net protocol [27]. As shown in Fig. 8, whatever the condition of scale and distribution of tasks is, the results of MACoNEAT are all superior to the other two algorithms. Under the conditions of the large-scale task, example PH126-PH131, the advantages of MACoNEAT are more obvious. Due to a higher degree of multi-satellite potential collision for objects visited under the large-scale task, MACoNEAT adopts the method of solving the fitness for joint individuals of multi-population to reduce the collisions re-visited by multi-satellite tasks, taking full advantage of relatively limited observation resources to gain the better planning results.

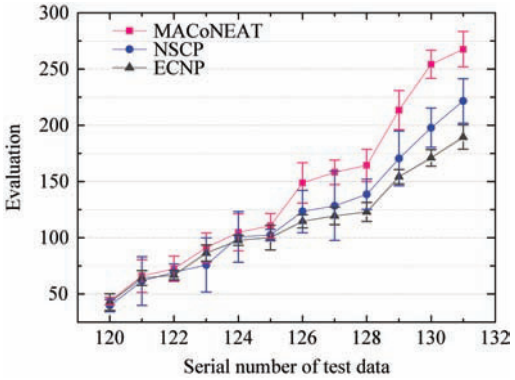

Fig. 8 Comparison of evaluation with different algorithms.

7.2. Capability of algorithm improved by TL

To test the improvement of algorithm by TL, two examples with different scales and distributions are divided into a group. The hybrid learning of IPSTL and MACoNEAT is adopted and the results are recorded, as shown in Table 2.

Table 2 Comparison between MSDHLA and MACoNEAT in different combinations of test data

No.	FIRST	SECOND	MSDHLA				MACoNEAT				IMP/%	
			AVG	MAX	CPU/s	TASK	AVG	MAX	CPU/s	TASK	TIME	EVA
1	PH120	PH123	73.52	84.44	589.37	48.2	68.76	75.69	908.17	42.7	35.10	6.92
2		PH124	83.26	89.4	615.9	55.4	83.73	87.2	962.45	55.2	36.01	–0.56
3		PH125	72.8	130.57	599.85	51.6	68.1	73.68	884.57	50.9	32.18	6.9
4		PH129	216.96	221.38	1 196.45	117.5	211.32	212.4	1 378.9	116.7	13.23	2.67
5		PH130	199.6	207.87	1 241.7	121.5	201.74	209.63	1 441.28	122.7	13.84	–1.06
6		PH131	227.85	231.92	1 139.93	155.4	229.17	235.77	1 374.94	159.1	17.09	–0.58
7	PH121	PH123	77.47	79.52	591.62	65.7	70.87	75.46	871.04	68.5	32.07	9.31
8		PH124	82.89	89.45	612.98	64.9	77.52	80.11	992.44	74.4	38.23	6.93
9		PH125	90.74	101.46	598.83	72.5	85.95	90.48	1 004.29	70.8	40.37	5.57
10		PH129	187.46	190.2	1 089.45	108.4	188.52	194.1	1 314.32	110.8	17.10	–0.56
11		PH130	201.39	205.85	1 297.13	115.1	198.28	200.46	1 479.91	113.4	12.35	1.57
12		PH131	219.47	223.67	1 124.67	122.3	219.56	225.77	1 403.4	125.1	19.86	–0.04
13	PH122	PH123	88.62	90.11	637.38	56.1	86.33	91.21	991.52	54.3	35.72	2.65
14		PH124	76.79	87.14	621.62	48.8	75.46	79.98	948.06	46.8	34.43	1.76
15		PH125	94.71	96.76	633.08	65.2	97.49	99.36	981.87	69.1	35.52	–2.85
16		PH129	197.76	200.47	1 235.75	106.8	195.14	199.57	1 524.73	104.9	18.95	1.34
17		PH130	225.17	227.38	1 267.54	152.6	220.94	224.39	1 489.74	148.9	14.92	1.91
18		PH131	213.79	215.6	1 194.51	116.5	213.1	218.5	1 338.16	116.2	10.73	0.32

Continued

No.	FIRST	SECOND	MSDHLA				MACoNEAT				IMP/%	
			AVG	MAX	CPU/s	TASK	AVG	MAX	CPU/s	TASK	TIME	EVA
19	PH123	PH123	134.91	138.72	766.49	85.4	136.88	141.25	1 378.64	88.1	44.4	−1.44
20		PH124	148.16	152.31	787.22	89.2	149.16	155.6	1 422.98	91.2	44.68	−0.67
21		PH125	137.22	138.96	743.16	82.7	134.58	138.69	1 298.34	80.9	42.76	1.96
22		PH129	341.84	362.17	1 625.38	223.8	337.1	344.48	3 137.58	217.5	48.2	1.411
23		PH130	369.41	377.25	1 739.92	256.9	362.24	375.17	3 468.84	251.7	49.84	1.98
24		PH131	352.14	359.64	1 688.43	239.4	345.63	354.01	3 257.12	230.4	48.16	1.88
25	PH124	PH123	137.61	141.18	728.31	87.1	135.98	137.84	1 352.17	85.2	46.14	1.2
26		PH124	142.89	148.35	747.14	90.5	144.91	148.58	1 398.44	91.4	46.57	−1.39
27		PH125	140.22	141.2	723.39	88.5	135.73	140.13	1 329.66	84.3	45.6	3.31
28		PH129	343.51	350.6	1 583.42	235.9	347.26	356.21	3 284.94	240.6	51.8	−1.08
29		PH130	389.42	396.14	1 672.35	269.4	387.04	393.7	3 887.51	265.2	56.98	0.61
30		PH131	367.58	372.63	1 627.45	248.9	380.16	385.14	3 741.23	254.6	56.5	−3.31
31	PH125	PH123	152.34	155.68	675.89	93.6	155.45	157.49	1 429.87	95.5	52.73	−2.0
32		PH124	160.77	168.21	715.46	106.7	157.36	164.23	1 567.62	103.3	54.36	2.17
33		PH125	158.34	163.12	681.49	98.1	152.17	159.81	1 532.72	94.6	55.54	4.05
34		PH129	371.85	379.46	1 425.94	252.6	367.45	374.49	3 621.59	248.3	60.63	1.2
35		PH130	400.21	405.27	1 518.22	273.9	398.35	405.89	4 025.16	269.2	62.28	0.48
36		PH131	386.42	392.13	1 584.26	264.8	391.24	402.54	3 871.64	277.5	59.08	−1.23

Notes: FIRST is the first example number, SECOND the second example number; AVG and MAX are separately the average value and the maximum value of results; CPU is the time of calculation (unit: s); TASK the quantity of average completion, IMP the increased ratio for comparison between MSDHLA and MACoNEAT, TIME the increased ratio for the time to reach the same threshold value of capability, and EVA the increased ratio of results.

As shown in Table 2, the planning results are not significantly improved by comparing hybrid learning and MACoNEAT, and some examples are slightly lower, while the planning time of hybrid learning is obviously less than MACoNEAT among all examples of test. By analyzing any two examples of groups of all examples, it could be found that the calculation speed is accelerated obviously. The main reason is that the previous large-scale examples accumulate more cooperative planning information, compared with the following example. Thus, MSDHLA converges faster.

It is seen that, compared with MACoNEAT, hybrid learning accelerates the convergence speed, especially, which is reflected more obviously when the scale of previous is larger than the following example.

7.3. Evolution process of learning policy

In order to further analyze the improvement of algorithm performance enhanced by the transfer learning strategy of the algorithm, the evolutionary curve for the group of case PH127 and PH125 based on two algorithms is given in Section 6, as shown in Fig. 9. From the comparison between the evolutionary curves of two algorithms, the hybrid learning converts the learning policy of case PH127 to the follow-up groups, which not only improves the initial solution, but also takes the advantage that history learning information accelerates the learning rate of algorithm. Thus, the convergence speeds of hybrid learning are all faster than MACoNEAT in Table 2.

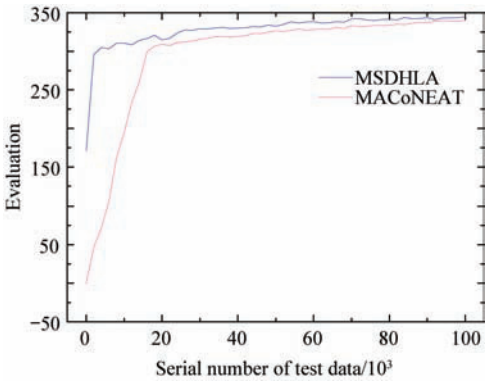

Fig. 9 Evolutionary curves of two algorithms.

7.4. Comparison of TL time in different policies

To test the influence of different policies of TL, evolutionary generations, using three TL policies to perform all examples accumulatively and converge to the same threshold, are compared with those of

MACoNEAT. ρ_{half}, ρ_{quarters} and ρ_{final} mean 50 000 generations, 750 000 generations and 100 000 generations are respectively taken as history learning information to transfer.

It could be seen from Fig. 10, ρ_{quarters} could get faster convergence speed than that of the other two transfer learning policies by taking ρ_{quarters} as the transfer learning policy. It suggests that learning history planning by using transfer policy is faced with the problems of insufficient learning and excessive learning. Thus, the selection for suitable transfer policy should be considered with specific problems. ρ_{quarters} is selected as transfer learning policy for MSDTP in this paper.

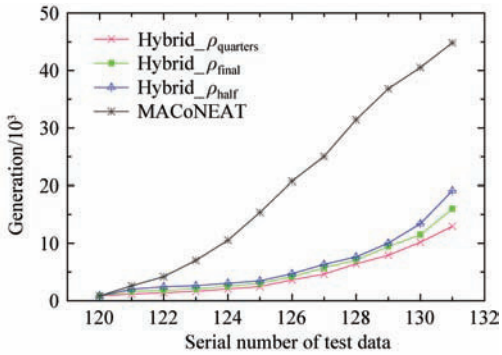

Fig. 10 Comparison of running time with different transfer strategies.

8. Conclusions and Future Works

Considering the characteristic that task planning requests occur randomly in multi-satellite dynamic task planning problem, a transfer learning strategy is introduced to multi-agent reinforcement learning, and a multi-agent hybrid learning algorithm is proposed. The algorithm describes the learning policy by using neural network which is encoded in a chromosome. By augmenting network topologies, the learning policy is optimized iteratively. And the fitness of the individual in the population is calculated in a fitness sharing way, which avoids the observation redundancy and improves the efficiency of resource utilization of satellites.

For the stochastic incoming observation requests, TL takes full advantages of historical cooperative planning information, which not only guarantees the cooperative planning quality but also accelerates the learning speed. Examples verify that the proposed algorithm in this paper is more suitable for the dynamic randomly occurring observation requests.

The future work is to achieve the knowledge produced during the former multi-satellite cooperative task planning and to transfer them to the following task planning. Currently, our TL method just focuses on the action transference to accelerate the learning speed for emerging task using historical task planning results. However, the transfer learning is not only to simulate the action, but also to discover the knowledge and transfer them for the incoming cooperative planning.

References

[1] Morris R A, Dungan J L, Bresina J L. An information infrastructure for coordinating earth science observations. Proceedings of the 2nd IEEE International Conference on Space Mission Challenges for Information Technology. Washington D. C., USA: IEEE, 2006; 397-404.

[2] Liu Y. Research on imaging reconnaissance satellite dynamic replanning model, algorithm and its application, PhD thesis, National University of Defense Technology, 2004. [In Chinese]

[3] Kramer L A, Smith S F. Maximizing flexibility: a retraction heuristic for oversubscribed scheduling problems. Proceedings of the 18th International Joint Conference on Artificial Intelligence. 2003; 1218-1223.

[4] Kramer L A, Smith S F. Task swapping for schedule improvement: a broader analysis. Proceeding of the 14th International Conference on Automated Planning and Scheduling. 2004; 235-243.

[5] Wang J M. Research on robust scheduling approach and its application of imaging satellites. PhD thesis, National University of Defense Technology, 2008. [In Chinese]

[6] Chen H. Planning and scheduling method for the earth's surface electromagnetic environment detection satellite resources. PhD thesis, National University of Defense Technology, 2009. [In Chinese]

[7] Peter S. Learning and multiagent reasoning for autonomous agents. Proceeding of International Joint Conference on Artificial Intelligence. 2007; 13-30.

[8] Liviu P, Sean L. Cooperative multi-agent learning: the state of the art. Autonomous agents and multi-agent systems, Springer, 2005; 11(1): 387-434.

[9] Lucian B, Robert B, Bart D S. A Comprehensive survey of multiagent reinforcement learning. IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews 2008; 38(2):156-172.

[10] Matthew E T, Peter S. Transfer learning for reinforcement learning domains: a survey. Journal of Machine Learning Research 2009; 10: 1633-1685.

[11] Matthew E T, Shimon W, Peter S. Transfer via inter task mappings in policy search reinforcement learning. Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems, 2007.

[12] Matthew E T, Gregory K, Peter S. Autonomous transfer for reinforcement learning. Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems. 2008.

[13] Daniel S B, Robert G, Neil I, et al. The complexity of decentralized control of markov decision processes. Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence. 2000; 32-37.

[14] Christopher J C H W. Learning from delayed rewards. PhD thesis, King's College, 1989.

[15] Michael W, Jeffrey S R, Best-response multiagent learning in non-stationary environments. Proceedings of 3rd International Joint Conference on Autonomous

Agents Multiagent System. 2004; 506-513.
[16] Richard S S, Andrew G B. Reinforcement learning: an introduction. Cambridge, MA. MIT Press, 1998.
[17] Xin Y. Evolving artificial neural networks. Proceedings of the IEEE 1999; 87(9): 1423-1447.
[18] Kenneth O S, Risto M. Evolving neural networks through augmenting topologies. Evolutionary Computation 2002; 10(2): 99-127.
[19] Richard S S, David M, Satinder S, et al. Policy gradient methods for reinforcement learning with function approximation. Advances in Neural Information Processing Systems. 2000; 1057-1063.
[20] Kohl N, Stone P. Policy gradient reinforcement learning for fast quadrupedal locomotion. Proceedings of the IEEE International Conference on Robotics and Automation. 2004; 2619-2624.
[21] David E M, Alan C S, John J G. Evolutionary algorithms for reinforcement learning. Journal of Artificial Intelligence Research 1999; 11: 199-229.
[22] Shimon W. Adaptive Representations for reinforcement learning. PhD thesis, University of Texas at Austin, 2007.
[23] Mitsuo G, Cheng R W, Genetic algorithms and engineering optimization. New York: Wiley, 2000.
[24] Tan K C, Yang Y J, Goh C K, A distributed cooperative coevolutionary algorithm for multiobjective optimization. IEEE Transactions on Evolutionary Computation 2006; 10(5): 527-549.
[25] Wang L. Intelligent optimization algorithm and its applications. Beijing: Tsinghua University Press, 2001. [in Chinese]
[26] Analytical Graphics Inc. Satellite tool kit 8.1: users guide [Online], available: http://www.agi.com, May 2, 2009.
[27] Samir A, Suzanne P, Melvin F S. An extended multi-agent negotiation protocol. Autonomous Agents and Multi-Agent Systems 2004; 8(1): 5-45.

Biographies:

WANG Chong Born in 1982, he received B.S. and M.S. degrees from Aviation University of Air Force in 2004 and 2007 respectively. And now he is a Ph.D. candidate in National University of Defense Technology. His main research interests are satellite intelligent planning and scheduling, and multi-agent systems.
E-mail: cwang.nudt@gmail.com

LI Jun Born in 1973, he received Ph.D degree from National University of Defense Technology in 2000. And now he is a professor and a master instructor in National University of Defense Technology. His main research interests are spatial information systerns and application technology, remote Sensing and Geospoltial Information Integration.
E-mail: junli@nudt.edu.cn