Defence Science Journal, Vol. 62, No. 3, May 2012, pp. 180-186, DOI: 10.14429/dsj.62.964

Platform-level Distributed Warfare Model-based on Multi-Agent System Framework

Xiong Li and Zhiming Dong

Academy of Armored Force Engineering, Beijing - 100 072, China
E-mail: lixiong2609@126.com

ABSTRACT

The multi-agent paradigm has become a useful tool in solving military problems. However, one of key challenges in multi-agent model for distributed warfare could be how to describe the microcosmic tactical warfare platforms actions. In this paper, a platform-level distributed warfare model based on multi-agent system framework is designed to tackle this challenge. The basic ideas include: Establishing multi-agent model by mapping from tactical warfare system's members, i.e., warfare platforms, to respective agents; performing task decomposition and task allocation by using task-tree decomposition method and improved contract net protocol model technique; and implementing simulation by presenting battlefield terrain environment analysis algorithm based on grid approach. The simulation demonstration results show that our model provides a feasible and effective approach to supporting the abstraction and representation of microcosmic tactical actions for complex warfare system.

Keywords: Computer simulation, distributed warfare model, platform-level modeling and simulation, complex warfare system, agent, multi-agent system

1. INTRODUCTION

Military simulation models of complex, dynamic warfare process, which can be called distributed warfare simulation systems[1-6], bring forward requirements on some properties, such as openness, locality in interaction, heterogeneity, and autonomy of the component. Some conventional methods, however, cannot cater for the requirements usually. For example, linearization, which 'linearises' military problems to derive an analytical solution, comes at the price of realism[1]. Object-oriented methodologies and concepts are unable to effectively model the tactics of entities involved in complex battlefield systems, which require better problem decomposition, more powerful abstraction mechanism and better representation of organizational hierarchy[2].

Multi-agent system emerged, as a scientific area, from the previous research efforts in distributed artificial intelligence started in 1980s[2-8]. For sometime now multi-agent-based modeling and simulation for distributed warfare process, has attracted the interest of researchers far beyond traditional computer science.

But most models have a shortage in platform-level modeling and simulation. For example, the hierarchical interactive theater model constructed and exercised by USA Air Force Studies and Analyses Agency is efficacious[3], but it can only perform unit-level modeling and simulation. Without the platform-level simulation demonstration[9], there would be empty of expression for microcosmic warfare resources application activities and dynamic tactical platforms actions. A platform-level distributed warfare model is needed to describe them.

In this paper, authors established the mapping from tactical warfare system's members, i.e., warfare platforms, to respective agents, and thus design a platform-level distributed warfare simulation model-based on multi-agent system framework to lay a foundation for the abstraction and representation of microcosmic tactical engagement behaviors on future battlefields. In the resolution level view, proposed model puts emphasis on characterising tactical platforms in individual warfare areas, not army units in mission areas. In the role view, it aims at the mapping from real platform-level entities to platform-level agents in simulation. In the technique view, it concentrates on forming tactical tasks list, and imitating the effects of entities running on battlefield terrain grids.

2. MULTI-AGENT MODEL

The internal members of distributed warfare system run with autonomy and interaction. These information interactions include sending, accepting, and informing all kinds of command and control instruction, navigation information, imagery intelligence, signals intelligence, measurement intelligence, surveillance information, and missile launch warning information. Tactical warfare system on distributed battlefield is so alike a distributed multi-agent system in behaviors that we can set up mappings from its internal members, i.e., warfare platforms, to respective agents, e.g., tank $\rightarrow$ tank agent, combat command vehicle $\rightarrow$ combat command vehicle agent.

2.1 Organisation Architecture of System

To develop effective platform-level distributed warfare model system which can be called a whole federation, in the

Received 12 May 2011, revised 31 March 2012, online published 3 May 2012

course of the mapping we should sort not only the function agents (warfare platform entity agents), but also the administration agents and service agents including federation manager agent, declaration manager agent, federation manager agent, time manager agent, data distribution manager agent, which play the roles of demonstration control, simulation evaluation, data base, situation displaying, command practice and battlefield environment. The function agents in red or blue force can be aggregated into the red or blue agents federation, and the administration agents and service agents can be aggregated into the white federation. In this way, we can design the basic multi-agent organization of platform-level distributed warfare simulation system as shown in Fig. 1.

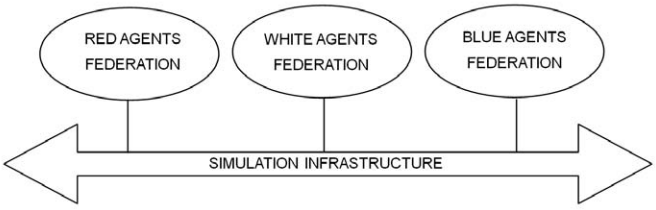


Figure 1. Multi-agent organization of platform-level distributed warfare simulation system.

2.2 Agent Design

In this paper, the condition-event driven rule based system is used in the multi-agent model as the basis for representing knowledge. Because the internal state base and knowledge base are the most important parts of an agent, which are key to realize its functions, we only probe into the internal state base and knowledge base of an entity agent.

In a general way, the information in an agent's internal state base should involve *State_Name*, *State_Type*, *State_Value* and *State_Time*, where:

State_Name is the name of the state and illuminates a state of this agent;

State_Type is the type of the state and differentiates the state' aspects;

State_Value is the value of the state and reflects the current state level;

State_Time is the time symbol of the state and reflects the time sequence of state changes.

The information in an agent's knowledge base should involve military domain knowledge, reasoning knowledge and control knowledge.

Domain knowledge is used to describe task framework. For example, the task of an attack battle scheme can be described as an aggregation with subtasks attack target, attack zone and attack occasion:

task ('*an attack scheme*', '*an attack scheme*')

subtask ('*attack target*', '*attack zone*', '*attack occasion*')

Reasoning knowledge consists of a series of reasoning rules, which is used to reason out a warfare entity agent's communications, collaboration, decision-making and task administration. Its general form can be expressed as:

IF subtask ('*task*', '*subtask*') *AND state* ('*subtask*', *started*)
THEN state ('*task*', *started*)

Control knowledge connects military domain knowledge and reasoning knowledge, by which a warfare entity agent can transform information obtained to domain knowledge after logistic reasoning. Its general form can be expressed as:

IF scheme (*conditions*, *conclusions*) *AND all_true* (*conditions*)
THEN add (*conclusions*)

The pseudo-codes of a platform-level entity agent algorithm can be illustrated as follows, where the agent warfare actions, such as 'MakeDecision' and 'SendMessage' are expressed by task decomposition and task allocation mechanism; the agent interaction with battlefield terrain environment is performed by battlefield terrain environment analysis algorithm.

```
FederatePlatformLevelAgent(){
      Apperceive();/*Apperceiving the
      external affairs(especially the
      affairs inthe battlefield environment)
      */
         While(true){
         Accept();/*Accepting the external affairs */
         Interpret();/*Interpreting the external affairs */
         MakeDecision();/*Estimating, analyzing and making
decision*/
         ModifyState();/*Modifying its state to be adapted */
         SendMessage();/*Sending messages to other platform-
level agents*/
          /*or TakeAction()  Taking relevant action to
accomplish its task*/
          }
      Exit();
      }
      Make_Decision(){
      if (bRuleBase==true) {/* Choosing reasoning system*/
         PreProcess();/*Preprocessing for initialization and
terminating
         conditions*/
         Reason();/*Reasoning for judgment */
         return result;
      else if (BTEData==true) { /*Using battlefield terrain
environment analysis
      algorithm*/
         PreProcess();/*Preprocessing to transform the original
input data into the
         digital data*/
         Run_BETanalysis();/*Inputting the processed data to
battlefield terrain
         environment analysis algorithm and carrying out*/
         return result;
      }
}
```

3. TASK DECOMPOSITION AND TASK ALLOCATION

Whenever agents have to work in a group setting, interactions take place to find a suitable organization (who does what) as well as to enable communication of results (when and to whom). Task decomposition and task allocation in the context of a multi-agent system could be considered in the scope of coordination of the agent's activities in a dynamic environment where resources may be scarce.

3.1 Task Decomposition

Multi-agent system modeling and simulation reflects real tasks relationship. Thus, in the platform-level distributed

warfare model, we can describe the functionality of a tree abstractly by defining a tree structure and introduce a function $De\_of(c_1,c_2)$ to analyze conveniently tactical tasks and their hierarchical structure. An example of the tree is shown in Fig. 2.

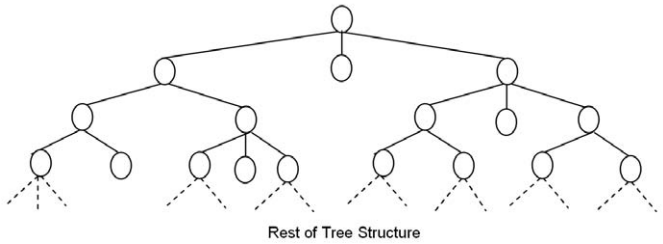


Figure 2. Abstract representation of a tree.

We assume that $De\_of(c_1,c_2)$ denote that c_2 is a sub-concept of c_1 and $De\_of(c_1,c_2)$ meet:

$$(\forall c_1,c_2,c_3)De\_of(c_1,c_2)\wedge De\_of(c_2,c_3)$$
$$\rightarrow De\_of(c_1,c_3)$$
$$(\forall c)\neg De\_of(c,c)$$
$$(\forall c_1,c_2)De\_of(c_1,c_2)\rightarrow \neg De\_of(c_2,c_1)$$
$$(\forall c_1,c_2,c_3)De\_of(c_1,c_2)\wedge De\_of(c_1,c_3)\rightarrow c_2$$
$$=c_3\vee De\_of(c_2,c_3)\vee De\_of(c_3,c_2)$$

Based on the analysis of the multi-agent relationships, we can decompose task for multi-agent-based distributed warfare system. The decomposed task manifests a hierarchy like a tree. The task tree's rootstalk is in fact the total objective of battlefield action and the coequal or same - hierarchy crunodes have an And/Or relationship. We assume that there are n borders from crunode T leading to the crunodes ST_1, ST_2, …, ST_n. If n borders have an And relationship in logic, then $T=ST_1\wedge ST_2\wedge\ldots\wedge ST_n$, which means the fulfillment of task T depends on the final fulfillment of all subtasks. If n borders have an Or relationship in logic, then $T=ST_1\vee ST_2\vee\ldots\vee ST_n$, which means the fulfillment of task T only depends on the fulfillment of a discretional subtask.

According to the task-tree decomposition approach, at first we decompose the tactical task into three subtasks: intelligence reconnaissance, data fusion and processing, and combined striking. Then, we decompose the subtasks into lower hierarchy in term of the correlation and logistic relationship. Thus by means of this task-tree decomposition approach, we can get a task decomposition sketch map of platform-level distributed warfare based on multi-agent system framework as shown in Fig. 3.

Thus, by the above task decomposition, we can obtain the corresponding task tree and-or-tree (T) of platform-level distributed warfare agents (See Fig. 4).

We can also express task tree and-or-tree (T) in event-condition-action (ECA) rule. ECA is an approach to describing task tree by analyzing the relationship of entity's task events, operation conditions and actions, which can be defined as:

WHEN Events

IF Conditions THEN

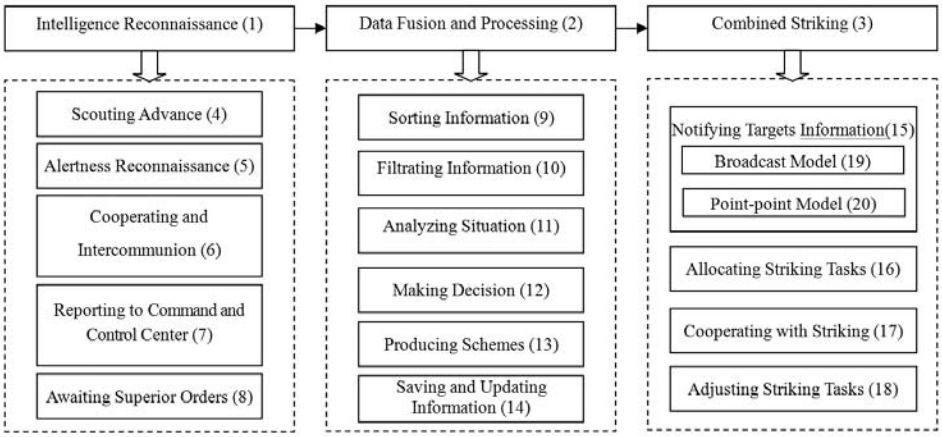


Figure 3. Task decomposition of platform-level distributed warfare.

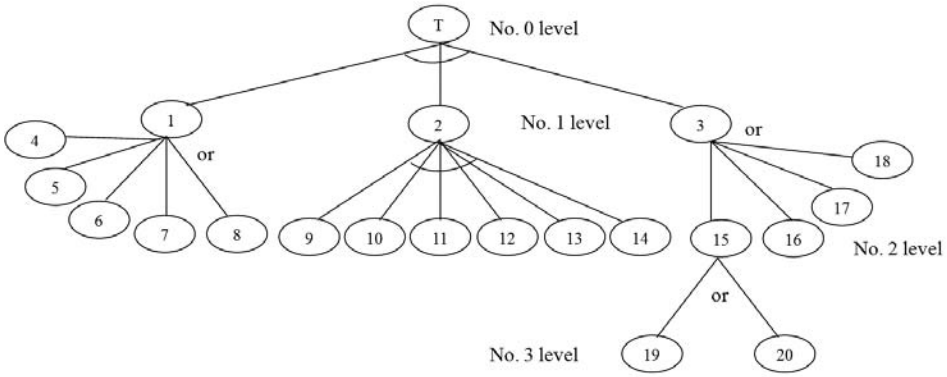


Figure 4. Task tree and-or-tree (T) of platform-level distributed warfare agents.

Action
ENDIF
ENDWHEN

Thus, the Task Tree *and-or-tree*(*T*) of platform-level distributed warfare agents can be expressed as:

On done(1) ∧ *done*(2) ∧ *done*(3) *if union*(1,2,3) = *TRUE then done*(*T*)
On done(4) ∨ *done*(5) ∨ *done*(6) ∨ *done*(7) ∨ *done*(8) *if choice*(4,5,6,7,8) = *TRUE then done*(1)
On done(9) ∧ *done*(10) ∧ *done*(11) ∧ *done*(12) ∧ *done*(13) ∧ *done*(14) *if union*(9,10,11,12,13,14) = *TRUE then done*(2)
On done(15) ∨ *done*(16) ∨ *done*(17) ∨ *done*(18) *if choice*(15,16,17,18) = *TRUE then done*(3)
On done(19) ∨ *done*(20) *if choice*(19,20) = *TRUE then done*(15)

By this task decomposition mechanism and task tree expression, we can obtain a tactical tasks list for platform-level distributed warfare agents. Here, tactical tasks mean microcosmic tactical platforms actions and are corresponding to platform-level tactical engagement behaviors on distributed battlefield.

3.2 Task Allocation

Task allocation has been considered as one of the main goals for coordination among agents in a multi-agent system. In this paper, contract net protocol (CNP)[8] is used to task allocation of platform-level distributed warfare simulation based on multi-agent system framework. Let us take a case to demonstrate the task allocation by CNP in which our platform-level distributed warfare simulation system consists of a Red armored force unit (one combat command vehicle and nine tanks) and a blue army troop (one information processing vehicle, one tank, one missile launch vehicle, one trench mortar, and some other fire platforms).

In proposed model, the manager wishes a task to be performed by one or a group of agents according to some arbitrary function which characterizes the task. The manager issues the call for proposals, and other interested agents can send proposals. In contrast to the original CNP, there is no need to do anything if an agent playing a role of a potential contractor is not interested in submitting proposals. That means that our CNP model from the very beginning relies on the notion of timeout, i.e., some actions need to be performed in the event of a lack of enough proposals or even in the case of a complete lack of proposals.

Thus, we obtain an improved CNP. The proposals are collected by the manager, and then they are rejected or accepted. The accepted proposals can be cancelled, either, by the manager via a cancel action, or by the contractor via a failure action. In case of cancellation other submitted proposals can be reconsidered, or a completely new call for proposals can be issued.

The schematic representation models of agent interaction protocol by the original CNP and our model are respectively shown in Figs 5(a) and 5(b)[6]. In Fig. 5, uncolored parts mean being performed by the manager while colored parts by the contractor, and actions representation is in boldface to distinguish from their contents. In contrast to the original CNP, the improved CNP model has functions of conditions constraints and confirmations for some key tactical tasks. It is obvious that the latter has an advantage of representing microcosmic tactical actions of a platform-level warfare entity.

For example, the interaction is started by the combat command vehicle agent who acts as a manager issuing a call for proposals, e.g. destroying the number 1 target in 59 highland. These tank agents who act as potential contractors respond with proposals, which the combat command vehicle agent either rejected or accepted. Accepted proposals can be

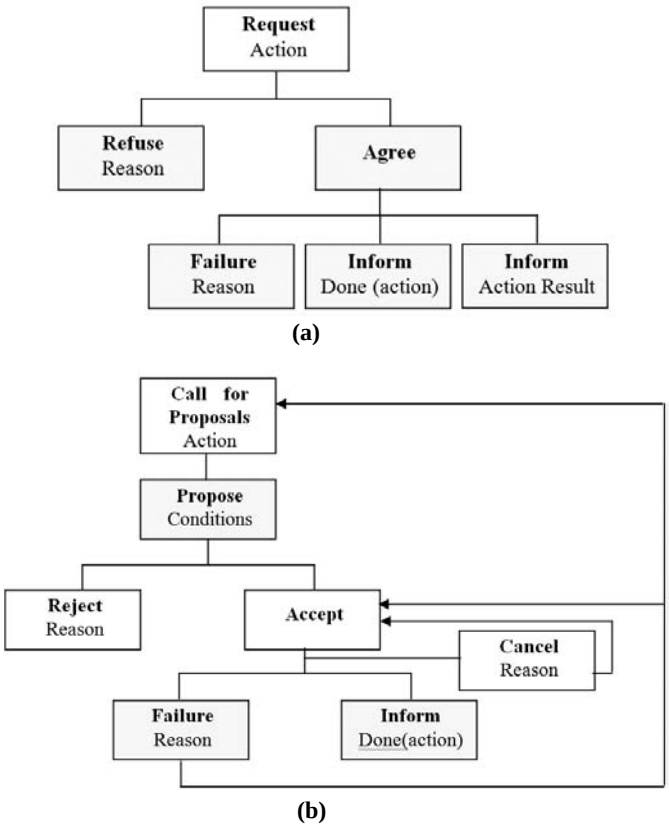


Figure 5. (a) Original CNP and (b) improved CNP.

either cancelled by the combat command vehicle agent, or executed by a certain tank agent, who later informs the combat command vehicle agent of success or failure of the execution.

4. SIMULATION DEMONSTRATION

In implementing simulation demonstration, terrain factor must be taken into account, which is analysed earlier. Actually, terrain environment can be viewed as a 'white agent' since it acts as a part of simulation service, when real platform-level warfare agents operate on a certain terrain.

To establish this interaction model for platform-level warfare agents and terrain environment, grid approach is used. Thus, we can set up lots of warfare zone grids with different granularities, according to simulation demonstration requirement, i.e., implementing platform-level distributed warfare simulation, as shown as Fig. 6.

Warfare zone grids with different granularities not only denote modeling and simulation resolution level, but also

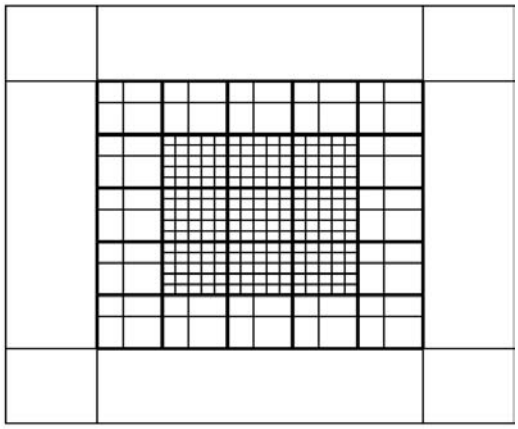


Figure 6. Battlefield terrain grids.

reflect tactical tasks hierarchy. For example, 20 m, 10 m, and 5 m granularities in a certain warfare area are shown as Figs 7(a), 7(b), and 7(c) respectively. It is obvious that only 5 m granularity can meet the needs of platform-level agents simulation because of the sizes of tactical warfare platforms.

In battlefield terrain grids, each cell, i.e., meta-grid, on which platform-level warfare agents run, can be furthermore marked off as Fig. 8. Thus, by the grid approach, we can obtain quantitative analysis.

The gradients in x coordinate axis and y coordinate axis can be computed respectively as:

$$Gradient_x = \frac{(P_7 + 2P_3 + P_6) - (P_8 + 2P_1 + P_5)}{8 \times GridCell} \quad (1)$$

$$Gradient_y = \frac{(P_6 + 2P_2 + P_5) - (P_7 + 2P_4 + P_8)}{8 \times GridCell} \quad (2)$$

The gradient and slope aspect of Point P can be computed respectively as:

$$Gradient = \sqrt{Gradient_x^2 + Gradient_y^2} \quad (3)$$

$$SlopeAspect = Gradient_y / Gradient_x \quad (4)$$

Thus, we can obtain all battlespace data, which can be used to estimate platform-level warfare agents behaviors, such as when a tank agent is climbing 59 highland, if this tank's maximum climbing angle h_1 meets $Gradient \le h_1$, then the tank agent can get it across; otherwise, it can't.

Here we take Visual C++ as the programming language for implementing the simulation demonstration system. In this simulation software, we can run the designed platform-level warfare entity agents in the red and blue agents federation. The intelligent and dynamic interaction actions of these entity agents are administrated by the 'white agents'.

The segmental program of the simulation demonstration system can be expressed as follows:

```
class platform-level distributed warfare simulation_multi-
agent
{
private:
   int iWarfare_Result;/*the results of warfare, which can be
obtained from
   military simulation*/
   Warfareplatform_Agent*Target;/*the enemy entity agent,
i.e., the target of
   this warfare platform*/
   …
   int iState;/*the state of this warfare platform */
   public:
```

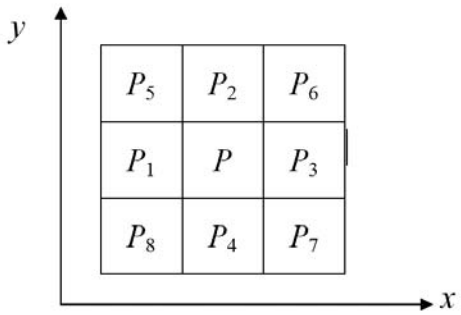


Figure 8. Battlefield terrain grid cell.

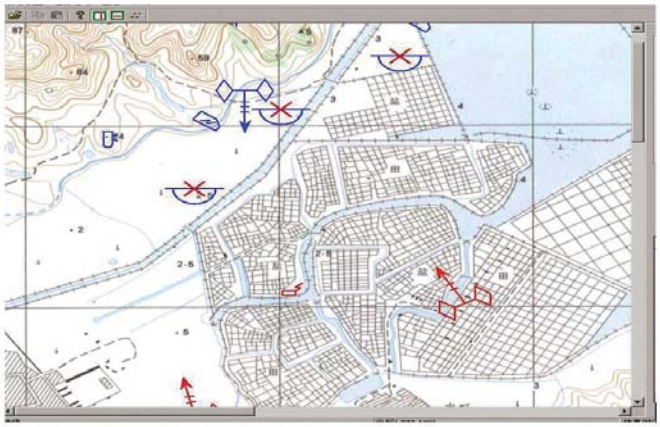
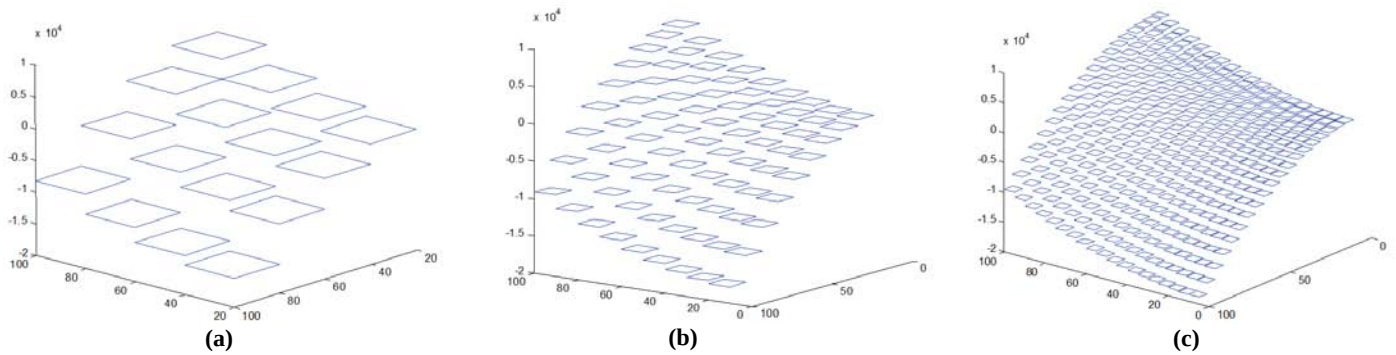


Figure 9. Partial two-dimension battlefield situation.



(a) (b) (c)

Figure 7. Different-granularity battlefield terrain grids simulation models.

```
    VgObject warfareplatformmodel;/*The two-dimension
geometrical model of this
    warfare platform*/
    VgFx*fx;/*environment vision*/
    AwSound*soundfx;/*sound made*/
    Awtactics*tacticsfx;/*tactics designed*/
    …
public:
    Warfareplatform_Agent();/*the constructive functions for
this warfare
    platform entity*/
    …
    Void Message();/*the functions of messages bulletin*/
    Void Superior();/*the functions of superior agent*/
    Void Action();/*the functions of action simulation of this
warfare platform
    agent*/
    int State_Transition();/*the functions of state transition of
this warfare
    platform agent, which presents a dynamic model*/
    private:
    …
}
```

The simulation demonstration system is illustrated in Fig. 9. Figure 9 presents the dynamic and real-time situation information during platform-level distributed warfare entities simulation where the deployment of Red force sensor platforms is approximately transverse. By this system, one can find out easily a certain agent's real-time state information.

According to the military experiences on tactical warfare process on distributed battlefield, we can set appropriate data to the parameters for our system. When we run the simulation demonstration system, we can obtain some results, which are shown in Fig. 10. Three scenarios are used in our simulation. Figures 10(a), 10 (b), and 10(c) give respective parameter result. T represents total time for fulfilling the attack battle task (minute), E represents attack efficiency (min/target) and R represents rate of destroyed force. In Scenario A, the red armored force unit takes a transverse deployment. Column and triangular deployment are taken respectively in Scenario B and Scenario C. Thus by these simulation results one can find that Scenario C is the most effective attack battle plan for the Red armored force unit while Scenario B is the worst one. Moreover, force loss can be obtained by simulation demonstration. Fig. 11 shows the real-time blue force during different distributed warfare periods. The sum of individual warfare platforms in the Blue force is the maximum at period 1, while it decreases stage by stage and gets the minimum at period 6.

We carry through verification, validation, and accreditation (VV&A) for our platform-level distributed warfare simulation model to analyze these results. As far as the concept model, we check whether attributes description, engagement and interactions, e.g., the entities and their tasks are consistent with real force situation. As far as the program model, emphases are put in data to verify their correctness, dependability and performance. By the evaluation, these results that we obtained from battlefield entities simulation are accordant to real distributed warfare situation. The fact proves that our model is feasible and effectual.

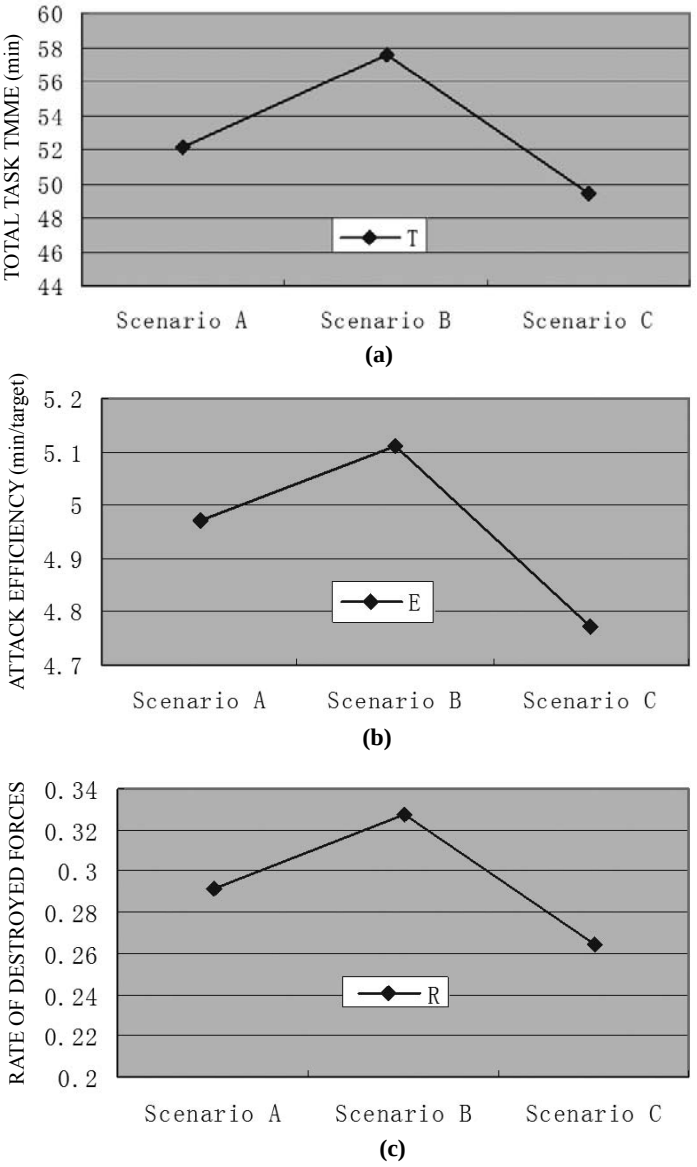


Figure 10. Contrastive results of three scenarios.

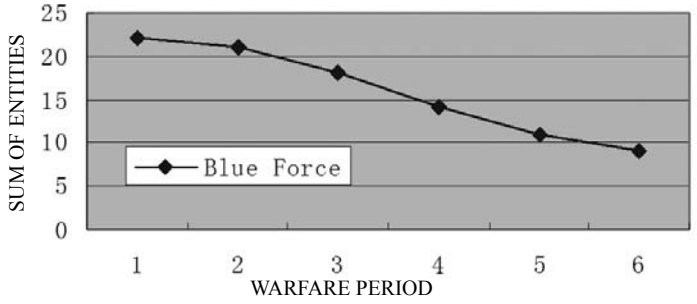


Figure 11. Real-time blue force.

5. CONCLUSIONS

In this paper, a platform-level distributed warfare model based on multi-agent system framework is studied. To solve the problems about microcosmic tactical simulation of intelligent engagement actions, dynamic information processing, and changing battlefield environment, we proposed a feasible and effective approach by putting forward the multi-agent organization of platform-level distributed warfare simulation

system and the architecture of platform-level agents, and furthermore setting up the multi-agent interactions model of task decomposition and task allocation in this system. Although there are only a few platform-level entity agents in the established distributed warfare simulation system model and it needs to be studied furthermore to be more practical, the simulation demonstration system shows that our model can be used to understand the external, complicated and intelligent warfare resources application activities and can realise the dynamic tactical actions simulation.

ACKNOWLEDGEMENT

This work is supported by the National Natural Science Foundation of China (NSFC) Grant #70901075, Military Science Projects for Graduate Supervisors Grant #2010JY0636-366, #2011JY001-015 and Innovation Foundation of AAFE Grant #2011CJ057.

REFERENCES

1. Julian, P.; Christine M. P. & Corrina A. R. Warfare analysis and complexity. Military Operations Research Society, User Report No. A313463. May 1999.
2. Sanjay Bisht; Aparna M. & Taneja, S. B. Modelling and simulation of tactical team behaviour. *Def. Sci. J.*, 2007, **57**(6), 853-64.
3. Richard, K. B.; Gregory, A. M. & Raymond. R. H. Using agent-based modeling to capture airpower strategic effects. *In* Proceedings of 2000 Winter Simulation Conference, Orlando, FL, USA. December 2000. pp. 1739-746.
4. Ibrahim, C. & Murat, M. A multi-agent architecture for modelling and simulation of small military unit combat in asymmetric warfare. *Expert Syst. Applications*, 2010, **37**(2), 1331-343.
5. Hill, R. R.; Champagne, L. E. & Price, J. C. Using agent-based simulation and game theory to examine the WWII Bay of biscay U-boat campaign. *J. Def. Model. Simul.*, 2004, **1**(2), 99-109.
6. Xiong, L.; Gaotian, P.; Zhiming, D.; Dianbo, C. & Hongwei, A. Designing of multi-agent-based of complex warfare system simulation model. *Dynamics of Continuous, Discrete Impulsive Syst., Series A: Math. Analysis*, 2006, **13**(S3), 953-59.
7. Jurgen, S. & Bruce, H. From complex conflicts to stable cooperation. *Complexity*, 2007, **13**(2), 78-91.
8. Junichi, K.; Tomoki, H.; Hiroshi, S.; Takayuki, T.; Toshihisa, F. & Hironori, H. Multi-agent-based autonomous power distribution network restoration using contract net protocol. *Elect. Engg. Japan*, 2009, **166**(4), 45-58.
9. Xiong, L.; Zhiming, D. & Wencheng, P. Platform-level ABM approach and its application to simulation demonstration of multiple sensors. *J. Syst. Simul.*, 2008, **20**(8), 2142-145.

Contributor

Prof Xiong Li received his PhD (Weapon System Engg) in 2009. Presently he is working as a Professor in the Department of Command and Administration, Academy of Armored Force Engineering, China. His current research interests include: Military system engineering, multi-agent systems, complex warfare systems modeling and simulation.