Adaptive Adjustment Method for Command and Control Structure Based on Artificial Bee Colony Algorithm (Postprint)

Authors: Sun Peng, Wu Junsheng, Guanyu Chen, Zhang Jieyong

Date: 2018-05-18T00:00:00+00:00

Abstract

To address the complexity and uncertainty of battlefield environments, adaptive adjustment of command and control structures has emerged as a key research focus. This study describes the basic entities of force organization and command and control structures, proposes a method for decision entity load measurement, establishes optimization models for adaptive adjustment of command and control structures under two battlefield scenarios, designs an artificial bee colony algorithm for solving these problem models, and presents the specific steps and flow of the algorithm. Finally, case simulations are conducted, where the adjustment method based on the artificial bee colony algorithm achieves favorable command and control structure adjustment effects, thereby demonstrating the feasibility of the artificial bee colony algorithm in adaptive adjustment of command and control structures.

Full Text

Preamble

Adaptive Adjustment of Command and Control Structure Based on Artificial Bee Colony Algorithm

Sun Peng[1] ,[2], Wu Junsheng[1] , Chen Guanyu[2], Zhang Jieyong[2]

(1. a. School of Computer Science; b. School of Software and Microelectronics, Northwestern Polytechnical University, Xi'an 710072, China; 2. College of Information & Navigation, Air Force Engineering University, Xi'an 710077, China)

Abstract: To address the complexity and uncertainty of the battlefield environment, research on adaptive adjustment of command and control structure has become a focal point. This paper describes the basic entities and command

and control structure of force organizations, presents a method for measuring the load of decision-making entities, and establishes optimization models for adaptive adjustment of command and control structure under two battlefield scenarios. An artificial bee colony algorithm is designed to solve these models, with detailed steps and procedures provided. Finally, case simulations demonstrate that the proposed method achieves effective command and control structure adjustment, proving the feasibility of the artificial bee colony algorithm for adaptive adjustment of command and control structures.

Key Words: command and control organization; organizational structure; adaptive adjustment; artificial bee colony algorithm

---

0 Introduction

A force organization is an operational whole, with various combat units within the organization closely connected around specific operational objectives. On future complex and dynamic informationized battlefields, organizations face increasingly intense and complicated environmental changes. During mission execution, action plans may change, and unexpected situations such as decision-making entity failures and platform damage may occur. Force organizations must possess adaptive adjustment capabilities to cope with evolving battlefield situations and gain battlefield control over adversaries to achieve established operational objectives.

The core of a force organization is its command and control structure. To enable this structure to respond to dynamic battlefield environments, it must be capable of adaptive adjustment. Current research has achieved certain results in this area, but two limitations require further investigation: (a) when evaluating the performance of the current organizational structure, the impact of past organizational states on the current structure is often not considered; and (b) research on the replacement of damaged decision-making entities is limited. To address these issues, this paper presents a performance measurement method for command and control structures, establishes adaptive adjustment models under two battlefield scenarios, and uses an artificial bee colony algorithm to find optimal solutions.

1.1 Entities in Force Organization

The entities within an organization primarily include task entities (T), platform entities (P), and decision-making entities (DM).

a) Task Entities: Also called combat tasks or simply tasks, these are necessary military actions taken by force organizations to achieve established operational objectives. In a typical task-platform relationship, completing a single task requires deploying one or more platforms. The set of task entities in an organization is denoted as $\{T_1, T_2, ..., T_{N_T}\}$, where N_T is the number of task entities.

b) Platform Entities: Also called platform resources or simply platforms, these carry combat resources within the organization and are the basic units that directly execute and complete combat tasks, such as fighter aircraft formations or infantry companies. The set of platform entities is denoted as $\{P_1, P_2, ..., P_{N_P}\}$, where N_P is the number of platform entities.

c) Decision-Making Entities: Also called decision units, these are responsible for implementing command and control activities within the organization. Based on their responsibilities, decision-making entities can be divided into operational decision-makers (ODM) and tactical decision-makers (TDM). The operational decision-maker exercises centralized control at the global level of the entire force organization, while tactical decision-makers control platform entities to execute specific combat tasks. There is typically only one operational decision-maker in an organization, but multiple tactical decision-makers exist. The set of tactical decision-makers is denoted as $\{TDM_1, TDM_2, ..., TDM_{N_D}\}$, where N_D is the total number of tactical decision-makers.

1.2 Force Organization Structure

The structure of a force organization can be divided into three layers: the decision layer, task layer, and platform layer. These layers are closely interconnected, collectively forming the organizational structure, as shown in [Figure 1: see original paper].

In this structure, relationships between entities include command relationships, collaboration relationships, allocation relationships, execution relationships, and temporal relationships. Command relationships refer to the command and control relationships between the operational decision-maker and tactical decision-makers, as well as between tactical decision-makers and platform entities. Collaboration relationships refer to communication and cooperation among tactical decision-makers. Allocation relationships refer to the demand relationships between task entities and required platform entities. Execution relationships refer to the relationships where tactical decision-makers execute task entities. Temporal relationships refer to the sequential order of task execution.

1.3 Performance Measurement of Force Organization

During combat operations, different command and control structures executing the same action plan result in varying workloads for tactical decision-makers. This occurs because mismatched structures create significant disparities in task assignments among tactical decision-makers, increasing organizational communication and collaboration and consequently raising their workload. Therefore, the performance of a command and control structure can be measured based on the load levels of tactical decision-makers within it.

The load of a tactical decision-maker is typically divided into local load and global load. Local load, also called task load, is the workload incurred by a tactical decision-maker when commanding and controlling platform entities during task execution while collaborating locally with other tactical decision-makers.

For tactical decision-maker TDM_i executing a task, let S_{ij}^{P-TDM} be the set of platform entities it commands and controls, and $S_{ij}^{TDM-TDM}$ be the set of other tactical decision-makers it collaborates with. The load of TDM_i on task T_j is defined as:

$$W_{ij}^{TDM} = \omega_C \cdot |S_{ij}^{P-TDM}| + \omega_{TC} \cdot |S_{ij}^{TDM-TDM}|$$

where ω_C is the command load coefficient and ω_{TC} is the collaboration load coefficient. The total load on TDM_i throughout the entire operation period is:

$$W_i^{TDM} = \sum_{T_j \in TS_i} (W_{ij}^{TDM} + W_i^{TDM}(t))$$

where $W_i^{TDM}(t)$ represents the load accumulated before time t.

The root mean square (RMS) of all tactical decision-maker loads can be calculated as:

$$RMS = \sqrt{\frac{1}{N_D} \sum_{i=1}^{N_D} (W_i^{TDM} - \bar{W})^2}$$

where $\bar{W}$ is the mean load across all tactical decision-makers. A smaller RMS indicates that the mean and variance of task loads remain at lower levels, representing a more rational organizational design. Therefore, RMS is selected as the performance metric for command and control structures.

---

2 Problem Description and Modeling

In battlefield environments filled with uncertainty, two primary situations significantly impact command and control structure performance: changes in action plans and damage to tactical decision-making entities. This paper investigates adaptive adjustment of command and control structures based on these two scenarios.

2.1.1 Problem Description: Action Plan Changes

During combat operations, force organizations face uncertainties such as task additions and platform damage, causing established action plans to change. When action plans change, the original relationships between tactical decision-makers and platform/task entities can no longer address the new battlefield situation. Adaptive adjustment of the pre-change command and control structure is required to ensure optimal organizational performance under the new action plan.

2.1.2 Model Establishment: Action Plan Changes

After an action plan changes, adjusting the command and control structure can improve organizational performance. However, structural changes adversely affect organizational stability, representing an adjustment cost that must not exceed the organization's maximum tolerable limit.

The adjustment cost is evaluated based on platform control transfer—when a platform entity originally commanded by TDM_i is transferred to TDM_j after adjustment. Let $M^{(1)}_{TDM-P}$ represent the pre-adjustment command relationship matrix and $M^{(2)}_{TDM-P}$ represent the post-adjustment matrix. The number of platforms whose control rights are transferred is:

$$Trans = \sum_{i=1}^{N_D}\sum_{j=1}^{N_P} |m^{(1)}_{ij} - m^{(2)}_{ij}|$$

The adaptive adjustment model for command and control structure under action plan changes is:

$$\begin{aligned} \min \quad & RMS \\ \text{s.t.} \quad & Trans \le \sigma \\ & \sum_{i=1}^{N_D} m_{ij} = 1, \quad j = 1, 2, ..., N_P \end{aligned}$$

where σ represents the maximum number of platform control transfers the organization can tolerate. The first constraint limits adjustment cost, while the second ensures each platform is controlled by exactly one tactical decision-maker.

2.2.1 Problem Description: Decision Entity Failure

Tactical decision-making entities are primary targets in combat. When a tactical decision-maker is destroyed and becomes ineffective, the platforms and tasks it commanded should be taken over by other tactical decision-makers. While entity failure degrades command and control performance, adaptive adjustment through appropriate replacement selection can improve the damaged structure's performance.

2.2.2 Model Establishment: Decision Entity Failure

Assume there are N_D normal tactical decision-makers before adjustment, with command relationships represented by matrix $M^{(1)}_{TDM-P}$. After TDM_k fails, the number of tactical decision-makers becomes $N_D - 1$, and the post-failure relationships are represented by $M^{(2)}_{TDM-P}$. Let S_B be the set of platforms originally controlled by the failed entity, with $|S_B| = N_B$, and let S_E be the set of platforms controlled by effective decision-makers.

The adaptive adjustment model after tactical decision-maker failure is:

$$\begin{aligned} \min \quad & RMS \\ \text{s.t.} \quad & \sum_{i=1}^{N_D-1} m_{ij} = 1, \quad j \in S_E \\ & \sum_{i=1}^{N_D-1} m_{ij} = 1, \quad j \in S_B \\ & Trans \leq \varepsilon \end{aligned}$$

where ε is the maximum tolerable number of platform control transfers. The first two constraints ensure each platform is controlled by exactly one tactical decision-maker, while the third limits adjustment cost.

---

3 Artificial Bee Colony Algorithm

The mathematical models for both scenarios are essentially combinatorial optimization problems that can be solved using intelligent algorithms. In 2005, Professor Karaboga from Erciyes University in Turkey proposed the Artificial Bee Colony (ABC) algorithm based on honey bee foraging behavior, which has demonstrated good performance in combinatorial optimization and path planning problems. Therefore, this paper employs the ABC algorithm to solve the proposed models.

3.1.1 Initialization

The optimization objective is to find the tactical decision-maker-platform relationship matrix M_{TDM-P} that minimizes the RMS of tactical decision-maker task loads. The matrix has dimensions $D \times P$, where D and P represent the available numbers of tactical decision-makers and platform entities after encountering unexpected situations. An element $(i, j) = 1$ indicates that platform P_j is commanded by tactical decision-maker TDM_i.

Considering the constraint that each platform can be controlled by only one tactical decision-maker, and each tactical decision-maker must control at least one platform, we generate all feasible relationship matrices $M = \{M_1, M_2, ..., M_N\}$ where each matrix differs from the original by exactly one platform control transfer. These N matrices serve as the initial solutions for the algorithm.

3.1.3 Search Process

The ABC algorithm incorporates both global and local search in each iteration. Global search identifies promising solutions for local refinement, where local search corresponds to neighborhood search.

In each iteration, the N solutions are sorted by RMS value in ascending order. The top $N/2$ solutions are selected for neighborhood search. In this algorithm, the neighborhood of a solution consists of all matrices where exactly one platform's control right is transferred compared to the current matrix, while still satisfying constraints. Using a greedy criterion, the RMS values of all neighboring solutions are calculated, and the one with minimum RMS is selected as the new solution. If this new solution outperforms the original, it replaces the original solution. The number of transferred platform controls is then checked against the organization's capacity limit; if exceeded, the original solution is retained.

After updating solution set M, the fitness of each solution is calculated using Equation (10), and the selection probability for observer bees is computed using Equation (11). Based on these probabilities, $N/4$ solutions are randomly selected via roulette wheel selection for neighborhood search, after which solution set M is updated again. This completes one iteration. The process continues until reaching the maximum iteration count $maxCycle$, at which point the matrix with minimum RMS in M is output as the optimal command and control structure adjustment.

3.2 Basic Steps of the Artificial Bee Colony Algorithm

The fundamental steps of the ABC algorithm are as follows:

a) **Initialize the bee population**: Set population parameters including total bees N, iteration counter $iter = 1$, maximum iterations $maxCycle$, and maximum search limit $Limit$. Generate N feasible $TDM - P$ binary matrices as initial solutions.

b) **Calculate fitness**: Compute the RMS value for each individual. Based on fitness values, classify the top $N/2$ individuals as employed bees and the rest as observer bees. Initialize the trial counter $trial(i) = 1$ for each solution.

c) **Employed bee phase**: Perform neighborhood search on the better $N/2$ solutions. If a new solution from neighborhood search is superior, replace the current solution; otherwise, update $trial(i) = trial(i) + 1$.

d) **Observer bee phase**: Calculate selection probability vector P and use roulette wheel selection to choose $N/4$ observer bees for neighborhood search.

e) **Scout bee phase**: Check if any $trial(i) > Limit$. If so, abandon that solution, randomly generate a new one in the solution space, and reset $trial(i) = 1$.

f) **Update iteration**: Increment $iter = iter + 1$.

g) **Termination check**: If $iter > maxCycle$, proceed to output; otherwise, return to step b).

h) **Output**: The global optimal solution is the required $TDM - P$ relationship matrix.

The algorithm flowchart is shown in [Figure 2: see original paper].

---

4 Adaptive Adjustment Process of Command and Control Structure

The previous sections addressed model construction and algorithmic solution for adaptive adjustment problems. The overall adjustment process is illustrated in [Figure 3: see original paper]. During task execution, when unexpected events occur, the event type is first identified. Corresponding adjustment models are established based on the event type, and the ABC algorithm is applied to obtain the adjusted task plan.

5.1 Simulation Case

A joint operations scenario from literature [17] is used to validate the ABC algorithm's performance in force organizational structure adjustment. Simulations were conducted on Windows 7 using MATLAB 2014a. The initial task-platform allocation relationships and tactical decision-maker-platform allocation relationships are shown in and .

Key parameters were set as follows: command load coefficient $\omega_C = 1$, collaboration load coefficient $\omega_{TC} = 2$, and maximum allowable platform control transfers = 10.

5.2 Simulation Experiments

1) Action Plan Changes

Factors causing action plan changes include task completion, task cancellation, task addition, platform addition, and platform damage—these occur randomly. After the action plan changes, the ABC algorithm solves the first optimization problem.

In 20 experimental runs, one instance randomly involved platform entities P_2 and P_3 being damaged. The pre-adjustment $TDM-P$ relationship matrix is shown in . After ABC algorithm adjustment, the post-adjustment relationship is shown in , where platform P_7's control was transferred to TDM_3. Performance metrics are presented in .

The greedy search algorithm (GSA) can also solve this problem effectively [18]. Comparative results between GSA and ABC algorithms are shown in [Figure 4: see original paper] and [Figure 5: see original paper]. The performance comparison demonstrates that ABC algorithm adjustment reduces RMS values and improves organizational structure performance after action plan changes, with significantly higher efficiency—ABC's runtime is approximately one-third of GSA's.

2) Decision Entity Failure

Scenario 2 is more complex, incorporating tactical decision-maker failure based on Scenario 1. The failed decision-maker is selected randomly. In one simulation where TDM_1 failed after platform addition, the pre-failure relationship is shown in , the immediate post-failure relationship in , and the adjusted relationship after TDM_1's platforms were redistributed to TDM_2 and TDM_3 in .

Twenty simulation runs for Scenario 2 were conducted, with comparative results against GSA shown in . As illustrated in [Figure 6: see original paper] and [Figure 7: see original paper], the ABC algorithm effectively reduces RMS values even in this more complex scenario, significantly improving damaged organizational performance.

5.3 Parameter Analysis

In the above experiments, the collaboration load coefficient ω_{TC} and command load coefficient ω_C were set to 2 and 1 respectively (ratio = 2). The impact of varying this ratio from 0.25 to 3 on algorithm performance was analyzed for both scenarios.

As shown in [Figure 8: see original paper] and [Figure 9: see original paper], RMS values increase significantly as the collaboration load coefficient grows, with larger differences between pre- and post-adjustment RMS. This occurs because the higher collaboration weight makes RMS reduction more sensitive to decreases in inter-decision-maker collaboration achieved through adjustment.

5.4 Simulation Analysis

Simulation results demonstrate that both ABC and GSA algorithms can reduce tactical decision-maker load RMS values. In Scenario 1, GSA performs slightly better in terms of RMS reduction, but ABC is far more computationally efficient. In the more complex Scenario 2, both algorithms show similar runtime, with ABC generally faster and more effective at reducing RMS and improving command and control structure performance.

---

6 Conclusion

This paper investigated adaptive adjustment of force organization command and control structures. An adaptive adjustment method based on the artificial bee colony algorithm was proposed and validated through simulations under two scenarios, with comparisons to the greedy algorithm demonstrating its effectiveness. The results confirm the feasibility and efficiency of applying the artificial bee colony algorithm to adaptive adjustment of command and control structures.

References

[1] Sun Yu, Yao Peiyang, Zhang Jieyong. C2 organization information structure effectiveness measurement and comprehensive evaluation [J]. Systems Engineering and Electronics, 2015, 37(6): 1313-1318.

[2] Zhang Jieyong, Yao Peiyang. C2 organization decision-making entity configuration problem modeling and solution method [J]. Systems Engineering and Electronics, 2012, 34(4): 737-742.

[3] Xiu Baoxin, Zhang Weiming, Liu Zhong, et al. Adaptive design method for C2 organizational structure [J]. Systems Engineering and Electronics, 2007, 29(7): 1102-1108.

[4] Perdu D M, Levis A H. Adaptation as a Morphing process: a methodology for the design and evaluation of adaptive command and control teams [J]. Computational & Mathematical Organization Theory, 1998, 4(1): 5-41.

[5] Levchuk G M, Levchuk Y N, Meirina C, et al. Normative design of organizations part : modeling congruent, robust, and adaptive organizations [J]. IEEE Trans on Systems, Man, and Cybernetics, 2004, 34(3): 337-50.

[6] Mou Liang, Zhang Weiming, Xiu Baoxin, et al. Dynamic adaptive optimization of C2 organization decision layer structure based on rolling horizon [J]. Journal of National University of Defense Technology, 2011, 33(1): 125-131.

[7] Mou Liang. Research on dynamic adaptive optimization method for C2 organizational structure under uncertain mission environment [D]. Changsha: National University of Defense Technology, 2011.

[8] Karaboga D. An idea based on honey bee swarm for numerical optimization [R]. [S. l.]: Engineering Faculty, Erciyes University, 2005.

[9] Karaboga D, Akay B. Artificial bee colony algorithm on training artificial neural networks [C]//Proc of the 15th Signal Processing and Communications Applications. 2007: 1-4.

[10] Karaboga D, Akay B. A comparative study of Artificial Bee Colony algorithm [J]. Applied Mathematics and Computation, 2009, 214(1): 108-132.

[11] Ozturk C, Karaboga D, Gorkemli B. Artificial bee colony algorithm for dynamic deployment of wireless sensor networks [J]. Turkish Journal of Electrical Engineering and Computer Sciences, 2012, 20(2): 255-262.

[12] Karaboga D, Basturk B. On the performance of artificial bee colony (ABC) algorithm [J]. Applied Soft Computing, 2008, 8(1): 687-697.

[13] Rao R S, Narasimham S, Ramalingaraju M. Optimization of distribution network configuration for loss reduction using artificial bee colony algorithm [J]. International Journal of Electrical Power and Energy Systems Engineering, 2008, 1(2): 709-715.

[14] Zhuo Tao, Zhan Ying. Improved artificial bee colony algorithm for cloud computing resource scheduling model [J]. Microelectronics & Computer, 2014, 31(7): 147-150.

[15] Wei Hongkai. Research on artificial bee colony algorithm and its applications [D]. Beijing: Beijing University of Technology, 2014.

[16] Huo Fengcai, Du Ying, Liu Yang. Artificial bee colony algorithm and its applications [J]. Journal of Jilin University: Information Science Edition, 2016(4): 468-476.

[17] Yu F, Tu F, Pattipati K R. Novel congruent organizational design methodology using group technology and a nested genetic algorithm [J]. IEEE Trans on Systems, Man, and Cybernetics, 2006, 36(1): 5-18.

[18] Sun Yu, Yao Peiyang, Li Minghui, et al. Adaptive adjustment method for force organization command and control structure [J]. Systems Engineering and Electronics, 2016, 38(9): 2086-2092.

Note: Figure translations are in progress. See original paper for figures.