Hindawi Publishing Corporation
Discrete Dynamics in Nature and Society
Volume 2012, Article ID 713740, 22 pages
doi:10.1155/2012/713740

Research Article

Simplifying Multiproject Scheduling Problem Based on Design Structure Matrix and Its Solution by an Improved aiNet Algorithm

Chunhua Ju[1,2] and Tinggui Chen[1]

[1] *College of Computer Science & Information Engineering, Zhejiang Gongshang University, Hangzhou 310018, China*
[2] *Contemporary Business and Trade Research Center, Zhejiang Gongshang University, Hangzhou 310018, China*

Correspondence should be addressed to Tinggui Chen, ctgsimon@gmail.com

Received 11 October 2011; Revised 23 February 2012; Accepted 14 March 2012

Academic Editor: Seenith Sivasundaram

Managing multiple project is a complex task involving the unrelenting pressures of time and cost. Many studies have proposed various tools and techniques for single-project scheduling; however, the literature further considering multimode or multiproject issues occurring in the real world is rather scarce. In this paper, design structure matrix (DSM) and an improved artificial immune network algorithm (aiNet) are developed to solve a multi-mode resource-constrained scheduling problem. Firstly, the DSM is used to simplify the mathematic model of multi-project scheduling problem. Subsequently, aiNet algorithm comprised of clonal selection, negative selection, and network suppression is adopted to realize the local searching and global searching, which will assure that it has a powerful searching ability and also avoids the possible combinatorial explosion. Finally, the approach is tested on a set of randomly cases generated from ProGen. The computational results validate the effectiveness of the proposed algorithm comparing with other famous metaheuristic algorithms such as genetic algorithm (GA), simulated annealing algorithm (SA), and ant colony optimization (ACO).

1. Introduction

With widespread availability of the Internet, large-scale distributed projects in manufacturing, production, and others are becoming popular. Project scheduling plays an important role in project management. Scheduling involves the allocation of the given resource to projects to determine the start and completion times of the detailed activities [1]. There may be multiple contending for limited resources, which makes the solution process more complex.

The allocation of scarce resources then becomes a major objective of the problem and several compromises have to be made to solve the problem to the desired level of near-optimality. Traditional tools, such as Gantts, critical path method (CPM), and the program evaluation and review technique (PERT), have serious limitations for project activity scheduling in practice. Furthermore, they are applied to only one project at a time. In many practical environments where project scheduling is an important activity, resources are constrained in number and more than one project is active at any one time. Besides, the activities have multiple execution scenarios (reflecting different ways of performing them), each scenario possibly having a different impact on the activity duration, the costs associated with it, and its resource requirements [2]. Multiple activity modes give rise to several types of trade-offs between (a) the activity duration and its use of resource (time/resource trade-off), (b) the activity duration and its cost (time/cost trade-off), and (c) the quantity and combination of resources employed by the activity (resource/resource trade-off). Consequently, we have a more realistic model, which is the resource-constrained project scheduling problem with multiple execution modes.

In this paper, we take up a challenge to introduce design structure matrix (DSM) to simplify precedence constraints existing in multi-project scheduling. Then, an improved artificial immune network algorithm (aiNet) approach is presented to solve the multi-mode resource-constrained multiproject scheduling problem (MRCMPSP). The remainder of the paper is organized as follows. Section 2 is a literature review. Section 3 describes the MRCMPSP problem and its conceptual model is proposed. Section 4 defines our approach to simplify MRCMPSP based on DSM. Section 5 introduces an improved aiNet algorithm to solve the MRCMPSP. Section 6 details the problem instance generator and reports the computational experiments. Concluding remarks are made in Section 7, along with a discussion about further research.

2. Related Works

The RCMPSP is a generalization of the resource-constrained project scheduling problem (RCPSP). It has been shown by Błażewicz et al. [3] that the RCPSP, as a generalization of the classical job shop scheduling, belongs to the class of NP-hard optimization problem. The RCMPSP and its extensive form MRCMPSP, as a generalization of the RCPSP, are therefore also NP-hard.

RCPSP has aroused a strong interest of academic scholars firstly, and there are many studies involving the scheduling of a single project. For example, Stinson et al. [4], Christofides et al. [5], Demeulemeester and Herroelen [6], and others presented a branch and bound approach to solve the problem and the differences among them lie in branch schemes as well as elimination rules and other details. Montoya-Torres et al. [7] proposed a novel genetic algorithm for the RCPSP and an alternative representation of the chromosomes using a multi-array object-oriented model was developed in order to take advantage of programming features in most common languages for the design of decision support systems. The approach was tested on sets of standard problems and its performance is superior to that of other heuristic algorithms. Xu et al. [8] illustrated how to combine the idea of rollout with priority rule heuristics and justification for the RCPSP, and examined the resulting solution quality and computational cost. They presented empirical evidence that these procedures are competitive with the best solution procedures described in the literature. In addition, Mobini et al. [9] and Ziarati et al. [10] designed the artificial immune algorithm (AIA) and bee algorithm (BA) in order to solve RCPSP, respectively, where AIA is inspired

by vertebrate immune system and BA by intelligent behaviors of honey bees. The results that have been obtained using a standard set of instances, after extensive experiments, proved to be very competitive in terms of number of problems solved to optimality.

Most of the heuristics methods used for solving RCMPSP belong to the class of priority rule-based methods. Several approaches in this class have been proposed in the literatures. Priority-based heuristics developed a schedule by adding one activity at a time to that schedule. A priority rule specifies, for a set of activities that are eligible to be scheduled at a particular point in the algorithm, the one to be placed on the schedule next. The priority values for each activity can be based on a number of factors, including activity duration, the difference between early and late start times, and the number of successor activities. For example, Wiley et al. [11] developed a method utilizing the work breakdown structure (WBS) and the Dantzig-Wolfe decomposition to generate feasible aggregate level multi-project program plans and schedules. The Dantzig-Wolfe procedure provided a means of generating interim solutions and their appropriate funding profiles. The decision maker may then choose any one of these solutions besides the optimal solution based upon his/her own experience and risk tolerance. Lova and Tormos [12] analyzed the effect of the two components of a heuristics based on priority rules—schedule generation scheme and priority rule—over two measures of performance—mean project delay and multi-project duration increase. They then considered two approaches: single-project and multi-project. The study carried out was allowed to conclude the superior performance of the parallel schedule generation scheme in the context of multi-project scheduling. Unlike researches mentioned above, Browning and Yassine [13] conducted a comprehensive analysis of 20 priority rules on 12,320 test problems generated to the specifications project, activity-, and resource-related characteristics. They found several situations in which widely advocated priority rules performed poorly.

However, heuristics methods converge slowly and are easy to be immersed in a local optimum; therefore, other researchers make use of computation for biological engineering to solve RCMPSP. For example, Kim et al. [14] studied a hybrid genetic algorithm with fuzzy logic controller to solve the RCMPSP. The proposed new approach was based on the design of genetic operators with fuzzy logic controller through initializing the revised serial method which outperforms the nonpreemptive scheduling with precedence and resource constraints. Kumanan et al. [15] proposed the use of a heuristic and a genetic algorithm for scheduling a multi-project environment with an objective to minimize the makespan of the project. The proposed method was validated with numerical examples and was found competent.

Furthermore, Joglekar and Ford [16] integrated a traditional control-theory-based derivation of optimal resource allocation and a system dynamics model. They used the control theory model to derive an optimal allocation policy, which they described with a resource allocation policy matrix. The matrix was useful in explaining differences in project performance and developing an intuitive understanding of the characteristics and impacts of different allocation policies. The results showed that and how foresighted policies can improve schedule performance without increasing the total amount of resource. Lambrechts et al. [17] built a robust schedule that met the project deadline and minimized the schedule instability cost. They described how stochastic resource breakdown can be modeled, which reaction was recommended, when resource infeasibility occurred due to a breakdown, and how one can protect the initial schedule from the adverse effects of potential breakdowns. The computational results showed that protection of the baseline schedule, coupled with intelligent schedule recovery, yielded significant performance gains over the use of deterministic scheduling approaches in a stochastic setting. Additionally, Adhau et al. [18] proposed a novel distributed multiagent system using auctions-based negotiation

approach for resolving the resource conflicts and allocating multiple different types of shared resource amongst multiple competing projects. The proposed approach can solve complex large-sized multi-project instances without any limiting assumptions regarding the number of activities, shared resource or the number of projects. In addition, this approach further allowed random project release-time of projects which arrived dynamically over the planning horizon.

In contrast to these researches mentioned above, activities may be performed in more than one mode for MRCMPSP. If the activities select different modes, their execution durations and resource requirements will be changed at the same time. It means the optimal scheduling plan will also be changed. General heuristic methods or intelligent algorithms usually need enough time to seek an optimal plan and are easily immersed in a local optimum. Due to these reasons, we introduce an improved aiNet algorithm, which is more suitable for solving the MRCMPSP. In addition, how to deal with precedence constraints and resource ones is important and difficult in project scheduling. In this paper, we adopt design structure matrix (DSM) to identify precedence relationships among activities and then determine the activities in an eligible set (a set of activities eligible to be scheduled at the current time) during the scheduling of multi-projects in order to simplify constraints and the more detailed process will be discussed in Section 4.

3. Analysis of Uncertainties in Product Development Process

3.1. Problem Description

The scheduling of multiple projects that share a common pool of resources can be carried out with two approaches: multi-project or single-project [19]. Using activity-on-node (AON) network representation, in the former, every project is considered with its corresponding "start" and "end" dummy activities. In the latter, projects are artificially merged together into a single project by the addition of two dummy activities: "start" and "end" of the single project. To obtain a feasible schedule of multi-project we can choose one of the two approaches as shown in Figure 1. Although the start and end dummy activities are not required to solve the scheduling problem, they have been added to formally describe the problem network.

When the multi-project approach is used, the time objective to be minimized is the mean project delay that is calculated with the expression

$$\min\left\{\sum_{i=1}^{I}\alpha_i(F_i - CP_i)\right\}, \tag{3.1}$$

where α_i denotes the weight of the ith project and I represents the number of projects. F_i is completion time of project i. CP_i is the resource unconstrained critical path length of project i. Obviously, minimizing this criterion is equivalent to minimizing the mean resource-constrained completion time of the projects.

When the single-project approach is used, the time objective to be minimized is the multi-project duration increase that can be calculated as follows:

$$\min\left\{\sum_{i=1}^{I}\alpha_i(F_i - CP_i)\right\}, \tag{3.2}$$

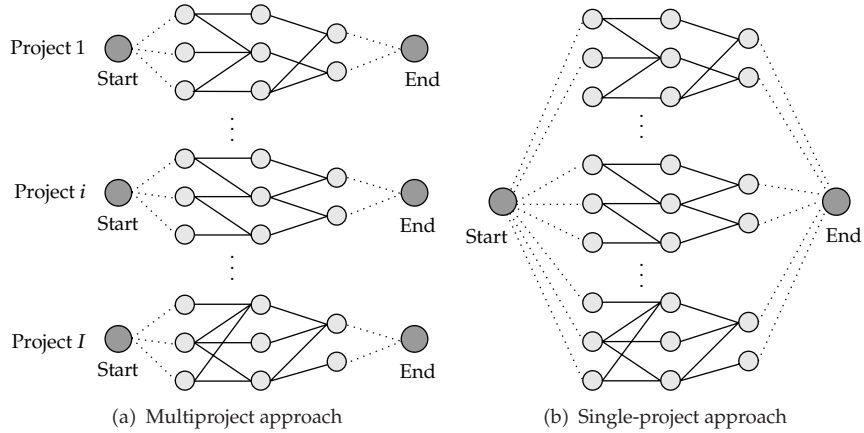


Figure 1: Multiproject scheduling approaches.

where the parameters used in (3.2) are the same as (3.1), and minimizing this criterion is equivalent to minimizing the makespan.

3.2. Conceptual Model of the Problem

The problem consists of the number of projects I, and the following assumptions are taken into consideration.

(1) Activity i may be performed in any one mode j. Each job will have a specific mode and must be finished without changing mode.

(2) Activity i cannot start unless all of its predecessors have been completed.

(3) There are only renewable resources and nonrenewable ones are not considered.

(4) Activity preemption is not allowable.

(5) It is hypothesized that all projects are executed concurrently and precedence constraints among different projects are not considered.

(6) Because time-to-market of products determines whether development process is successful, we can assume that the objective is to minimize the completion time of all projects but not a certain project. Due to this reason, we can adopt single-project approach to solve the problem.

Formally, the problem and the conceptual model will be described as follows. The considered problem consists of I parallel projects, each project $i = 1, \ldots, I$ being composed of J_i activities ij, $j = 1, \ldots, J_i$. Activity ij in project i may be performed in one of the modes $m = 1, \ldots, M_{ij}$. Each activity, once performed in a specific mode, must be finished without changing the mode. The activities are interrelated by two kinds of constrains. One is precedence constraints, and the other is resource constraints. While being processed, activity ij in project i performed in mode m requires q_{ijmr} units of renewable resource type $r = 1, \ldots, R$ during each period of its nonpreemptive duration d_{ijm}. Each resource type r has a fix and limit available amount Q_r. In addition, the optimal objective of the problem is to make the makespan shortest through finding out activity mode distribution scheme

and feasible starting time of activity. Therefore, the model of MRCMPSP can be described as follows:

$$\min\{\max(F_{1J_1}, F_{2J_2}, \ldots, F_{iJ_i}, \ldots, F_{IJ_I})\} \quad (i = 1, 2, \ldots, I), \tag{3.3}$$

subject to

$$F_{il} \leq F_{ih} - d_{ihm} \quad (i = 1, 2, \ldots I,\ h = 1, 2, \ldots, J_i)\ (l \in \wp_i), \tag{3.4}$$

$$\sum_{ij \in BJ(t)} q_{ijmr} \leq Q_r, \tag{3.5}$$

$$F_{iJ_i} \geq 0, \quad i = 1, 2, \ldots, I. \tag{3.6}$$

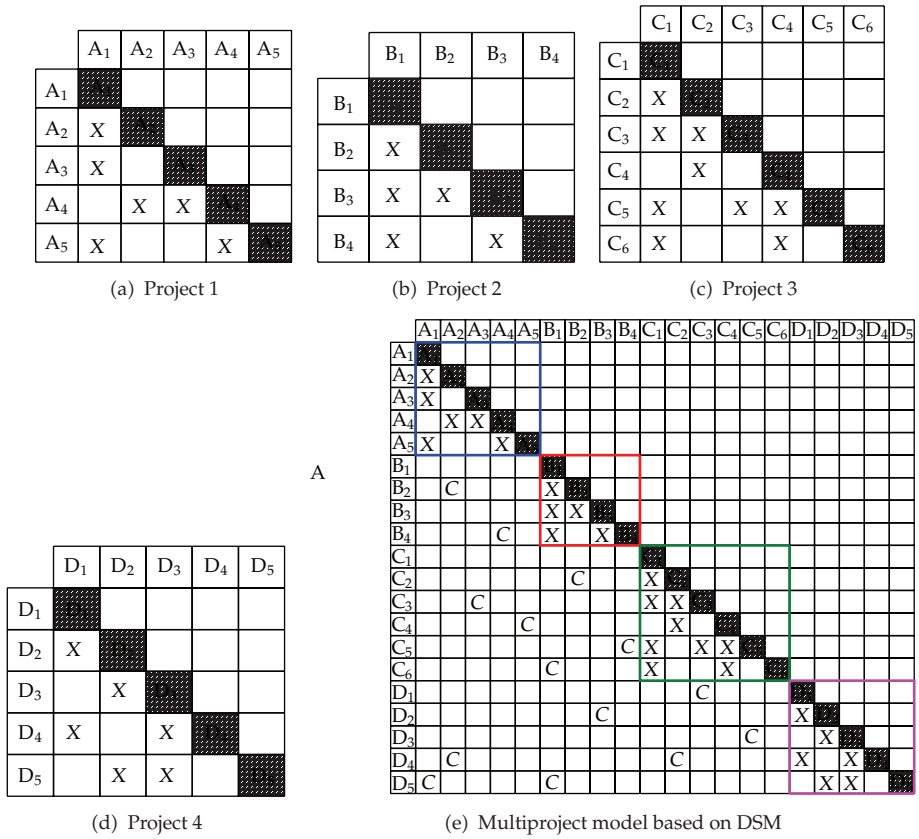
The objective function (3.3) seeks to minimize the performance measure. A constraint (3.4) imposes the precedence relations between activities, and a constraint (3.5) limits the resource demand imposed by the activities being processed at time t to the available capacity. It means the number of available resources will change according to the completion and starting time of activities. Finally, a constraint (3.6) forces the finish times to be nonnegative.

4. Design Structure Matrix Modeling and Simplification of MRCMPSP Model

4.1. Design Structure Matrix Modeling for Multiprojects

As a popular representation and analysis for system modeling, the design structure matrix (DSM) [20] provides a simple, compact, and visual representation of a complex system that supports an innovative solution to the decomposition and integration problems [21]. It had been widely applied as the basis of product development [22] and design iteration [23]. Currently, there are many researches using DSM to analyze RCPSP but few for MRCMPSP. In this paper, we use DSM to find out the activities in an eligible set so as to simplify constraints (3.4). If a project contains a large number of activities, information flows among activities described by matrix form are not only easy to be realized by computer but also compact and visual.

In general, the DSM approach is used only for single-project modeling. In this paper, we propose multi-project modeling approach based on DSM through introducing partitioning operation of DSM. In a multi-project environment, if each project is taken as an independent block, the whole process consisting of I parallel projects may be indicated by I blocks, where relationships among blocks are resource conflicts existing in activities between different projects. For example, Figures 2(a), 2(b), 2(c), and 2(d) represent four independent projects, respectively, and none of the information constraints exist in different projects. Let capital A, B, C, and D denote these four projects, where each project consists of 5, 4, 6, and 4 activities, respectively. Figure 2(e) represents DSM model of multi-project, where character symbol "C" indicates that there exist resource conflicts among projects. For example, the element in seventh row, second column denotes there exists resource competition between activity A_2 in project 1 and activity B_2 in project 2.



(a) Project 1 (b) Project 2 (c) Project 3

(d) Project 4 (e) Multiproject model based on DSM

Figure 2: Design structure matrix modeling for multi-project.

4.2. Simplification of MRCMPSP Model

During the scheduling of multi-projects, some activities will not be performed concurrently due to resource constraints and precedence ones. In this circumstance, precedence constraints among activities should be satisfied firstly so as to determine an eligible activity set. And then, resource conflicts that possibly occurred in this eligible set should be identified in order to decide the activity priority values, issued from the select priority rule. Therefore, the following sections will give a simplifying approach of precedence constraints and the activity priority values, respectively.

4.2.1. Simplification of Precedence Constraints

There exist both precedence constraints and resource ones for multi-project scheduling problem. As a result, activities should compete for limited resources at the premise of satisfying precedence constraints. Furthermore, owing to multi-project environment, resource conflicts among activities contain two situations. One is conflicts between different projects; the other is inside a project. For the former, because we assume that there exist no precedence constraints among different projects, resource conflicts will not occurr unless activities belonging to different projects share the same resource and its usage amount

exceeds the available one at the current time. For the latter, there exist both precedence constraints and resource ones among activities inside a project, therefore, precedence constraints should be satisfied firstly and then resource ones. The concrete process for identifying resource conflicts between activity ij and jk can be described as follows: (1) set up DSM model of multi-project as shown in Figure 2 and use partitioning and banding operations to each block. (2) Construct an eligible activity set, EJ, an activity set being executed, BJ, and an activity set completed, FJ, where EJ can be generated through DSM. That is to say, if activity satisfies (DSM$(ij,:) = K) \in FJ$ from a row, we can obtain $ij \in EJ$ or $ij \notin EJ$. Similarly, we can determine whether activity jk belongs to EJ. In doing so, EJ can be determined. (3) Identify resource conflicts between activities ij and jk. If it exists, perform activities according to priority value; if not, they can be performed concurrently and add the activities that are to be scheduled to BJ. (4) If activities ij, and jk have been fulfilled, update EJ, BJ, and FJ. Determine the next activity set which will possibly cause resource conflicts and repeat the process till all of activities have been fulfilled. Due to these steps, identification approach of resource conflicts can be shown in Pseudocode 1.

4.2.2. Determination of Priority Value of Resource Competition

The same resource can be used by more activities, which will cause resource competition among different activities. It is necessary to use the activity priority value, issued from the select priority rule to obtain the selection probabilities. It is known that many activity priority rules exist. The priority of each activity is subject to many factors, for example, the activity schedule, resources needed, the required earliest or latest beginning time, the number of immediate follow-up activities, and so forth. Different priority setting rules based on these factors will bring different computation performance. A total of 18 priority rules reported from literatures [24, 25] are listed in Table 1.

In this paper, we only select four representative rules such as maximum duration (MaxDur), maximum resource requirement (MaxRR), Early Start Time (EST), and maximum number of immediate successors (MaxSuc). This is because clonal selection of aiNet algorithm has a great ability of local searching. However, the objective of the problem is to minimize the whole multi-projects duration. The effects of scarce resources on project duration should be minimized; therefore, the weights of each of project should be considered.

5. An Improved Artificial Immune Network Algorithm (aiNet) for Solving MRCMPSP

5.1. Improved Strategy of aiNet Algorithm

In artificial immune system (AIS), a newly developed biological computing technology that draws inspiration from vertebrate immune system has become a powerful information processing and problem-solving paradigm in both the scientific and engineering fields with great developmental potential. Researches indicate that AIS is also a kind of stochastic and parallel search method like GA and is an efficient approach to combinatorial optimization problem. So far, AIS has been applied to the traveling salesman problem (TSP), multiobjective optimization, indirect path synthesis, the scheduling problem, capacitor placement, and assembly line balancing with encouraging results. Artificial immune network (aiNet for short) algorithms and models are originally proposed to perform information compression

PROCEDURE OF THE MODEL SIMPLIFICATION AND IDENTIFICATION OF RESOURCE CONFLICT
(1) confirm activities between projects or in one project
(2) ***if*** activities in one project
    if (DSM ([ij, jk],:) = K) $\in FJ$ **then**
      ij, jk belong to EJ
      if (q_{ijr} + q_{ikr}) > Q_r **then**
        resource conflict exists between ij and ik
      else
        there are no resource conflicts between ij and ik
(3) ***if*** activities between projects
    if (q_{ijr} + q_{ikr}) > Q_r **then**
      resource conflict exists between ij and ik
    else
      there are no resource conflicts between ij and ik
(4) update EJ, BJ and FJ
  if there are no activities needed scheduling
    process finished, otherwise go on.
(5) end

Pseudocode 1: Pseudocodes of identification process of resource conflicts.

and data clustering based on Artificial immune system (AIS) theory. Opt-aiNet, a modified version of artificial immune network model specially designed for multimodal function optimization presented by Castro and Timmis [26], has been demonstrated to have powerful multimodal searching ability as well as good stabilization. In this paper, an improved aiNet searching method for MRCMPSP is adopted based on opt-aiNet, and the new meanings of some terms redefined are shown in Table 2. It is notable that the fitness means the makespan of the multi-projects and the affinity denotes the difference between two potential scheduling schemes which will be described in Section 5.2.2. In addition, the stopping criterion has also been revised in order to avoid the early convergence in original algorithm. The detailed flow of the revised algorithm for MRCMPSP cannot be expatiated here for length limitation.

The scheme of the improved aiNet searching method for MRCMPSP, including the nine steps above, is shown in Figure 3.

5.2. Operational Definitions with Specific Details

As discussed earlier, the objective of the MRCMPSP is to schedule the activity such that precedence and resource constraints are satisfied and the makespan of the multi-projects is to be minimized. In order to illustrate the solution procedure of aiNet for MRCMPSP, network cell representation, population initialization, individual evaluation, clonal mutation operator, parameter tuning and so on will be described in the following sections.

5.2.1. Network Cell Representation

The key issue in the implementation of aiNet approach is encoding the antibody of a solution and its representation. In this paper, after extensive experimentation a direct problem

Table 1: Priority rules used for activity selection.

No.	Abbreviation	Name	Formula
(1)	MaxDur	Maximum duration	Max d_{ijm}
(2)	MinDur	Minimum duration	Min d_{ijm}
(3)	MaxRR	Maximum resource requirement	Max $d_{ijm} \sum_{r=1}^{R} q_{ijmr}$
(4)	MinRR	Minimum resource requirement	Min $d_{ijm} \sum_{r=1}^{R} q_{ijmr}$
(5)	MaxSuc	Maximum number of direct successors	Max$\lvert S_{ij}\rvert$
(6)	MinSuc	Minimum number of direct successors	Min$\lvert S_{ij}\rvert$
(7)	EST	Earliest starting time	MinES_{ij}
(8)	EFT	Earliest finishing time	MinEF_{ij}
(9)	LST	Latest starting time	MinLS_{ij}
(10)	LFT	Latest finishing time	MinLF_{ij}
(11)	MaxSLK	Maximum activity-free slack	Max $\mathrm{LF}_{ij} - \sum_{m=1}^{M_{ij}} (d_{ijm}/M_{ij}) - t_{\mathrm{now}}$
(12)	MinSLK	Minimum activity-free slack	Min $\mathrm{LF}_{ij} - \sum_{m=1}^{M_{ij}} (d_{ijm}/M_{ij}) - t_{\mathrm{now}}$
(13)	MaxRPW	Maximum rank positional weight	Max $d_{ijm} + \sum_{ik \in S_{ij}} d_{ikm}$
(14)	MinPRW	Minimum rank positional weight	Min $d_{ijm} + \sum_{ik \in S_{ij}} d_{ikm}$
(15)	MaxCRR	Maximum cumulated resource requirement	Max $\sum_{ij \in SJ} d_{ijm} \sum_{r=1}^{R} q_{ijmr}$
(16)	MinCRR	Minimum cumulated resource requirement	Min $\sum_{ij \in SJ} d_{ijm} \sum_{r=1}^{R} q_{ijmr}$
(17)	MaxCSuc	Maximum cumulated number of successors	Max $\lvert S_{ij} + \sum_{ik \in S_{ij}} S_{ik}\rvert$
(18)	MinCSuc	Minimum cumulated number of successors	Min$\lvert S_{ij} + \sum_{ik \in S_{ij}} S_{ik}\rvert$

Table 2: New meaning of the terms in general aiNet-searching method.

General aiNet model	The opt-aiNet optimization for MRCMPSP
Network cell	Candidate scheduling scheme
Fitness	The makespan of multi-projects
Affinity	Euclidean distance between the parametric vectors of two candidate scheduling schemes
Clone	Candidate scheduling scheme with the same parameters

representation for the MRCMPSP itself is used as a chromosome as shown in Table 3. Complete information of a schedule for the MRCMPSP consists of an activity priority rule, activities, and their corresponding modes in each individual project. The activity priority rule is randomly chosen from the rules such as maximum duration, maximum resource requirement, early start time, and maximum number of immediate successors. For example, priority $(i)/P_{11}/M_{\alpha}$ represents that activity 1 in project 1 is executed in mode α and its activity priority rule is i when existing resource conflicts.

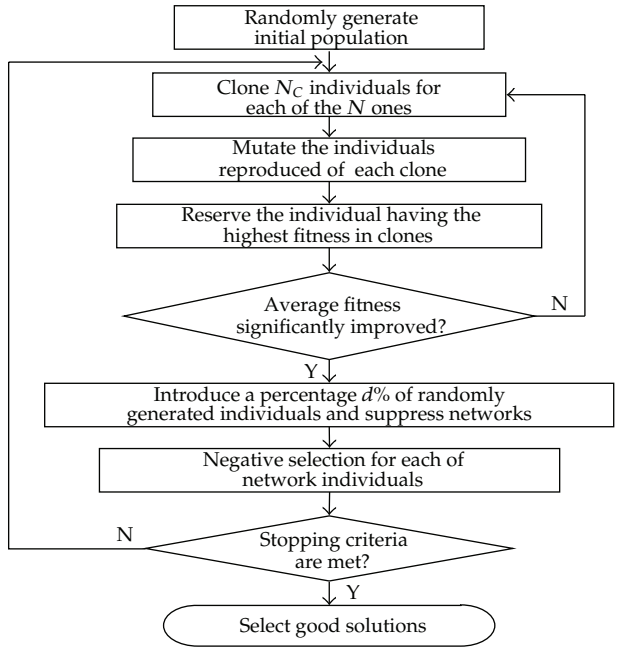


Figure 3: Scheme of the improved aiNet searching method for MRCMPSP.

Table 3: Direct problem representation of a schedule.

$P_1(p - r(x))$			$P_2(p - r(y))$			$P_3(p - r(z))$		
priority (i)	priority (j)	... priority (k)	priority (i)	...	priority (k)	priority (i)	...	priority (k)
$P_{11}M_\alpha$	$P_{12}M_\beta$	$P_{1J1}M_\gamma$	$P_{21}M_\alpha$		$P_{2J2}M_\gamma$	$P_{31}M_\alpha$		$P_{3J3}M_\gamma$

5.2.2. Population Initialization

The proposed aiNet approach deals with an antibody of individual strings. Currently, random techniques or heuristic procedure to generate initial solution has been used in general. However, it has been found that the performance of the aiNet approach with randomly start solutions is superior to that from pre-selected starting solutions. In addition, random initial solutions are helpful to effectively diverse the search space. Therefore, randomly generating an initial solution is adopted in this paper.

5.2.3. Individual Evaluation

Individual evaluation adopts affinity as well as fitness, where fitness function is evaluated according to a problem-specific objective function and affinity is used to get rid of similar individuals.

(1) Fitness Computation

Fitness is developed as follows. The steps involved are as follows: initially set the day as one and select all the activities in the projects to be done in that particular day. Allocate the

available resource as per the rank in the chromosome. If the resources are not available to perform some activities then postpone those activities. Suppose the same resource is required to perform multiple activities in a project then decide which activity should be processed first according to one of the activity priority rules described in Section 4.2.2. After the end of the first day, increase the day to the next day and perform the same until completion of all activities of all projects. The concrete steps are shown in Figure 4.

(2) Affinity Comparison

Different individual corresponds to different schedule. Activity states in a project are decided by activity priority rule (priority (i)) and execution mode (M_α); therefore, the affinity of two individuals X_1 and X_2 can be expressed as shown in formula (5.1), where λ is a scale factor and it represents the difference between activity priority rule and execution mode:

$$d(X_1, X_2) = \sqrt{\sum_{i=1}^{I} \sum_{j=1}^{J_i} \left(\text{priority}(i) - \text{priority}(i')\right)^2 + \lambda \sum_{i=1}^{I} \sum_{j=1}^{J_i} \left(M_{ij} - M_{(ij)'}\right)^2}. \tag{5.1}$$

5.2.4. Parameter Tuning

The parameters affecting the performance of aiNet approach are selected after initial experiments and past experience. Traditionally, these parameters are not determined independently since it is an issue of complex combination optimization. In the aiNet approach, there are three main user-defined parameters, namely, N (number of individuals to be selected for cloning), N_c (number of clones generated) and d (the amount of low-affinity individual to be replaced). These three parameters mainly affect the convergence speed and the computational complexity as well as its ability to perform multi-modal searches. In this paper, we chose 50% of the total individuals for cloning. This is because parameter N strongly influences the size of the population. The larger the value of N, the higher the computational cost to run the algorithm. Therefore, 50% looks to be more appropriate and economical. Further, the parameter N_c is used to find local optimums, which should be not too larger or too smaller. The higher the value of N_c, the faster the occurrence of the convergence in terms of generations. However, the computational time per generation increases linearly with N_c. At the same time, the lower value of N_c may reduce computational time per generation but difficult to converge to the global optimum. As a result, we chose smaller N_c in the early iterations but larger one in the later iterations. In addition, the parameter d is introduced to maintain the diversity in the population. In general, the value of d increases, and the algorithm reaches nearer to the optimal. However, when d approaches 1, the algorithm is the same as random searching. Thus, we set d at 20% for this paper.

5.2.5. Clonal Mutation Operator

Clonal selection plays an important role in adaptive evolution of the population. In essence, the objective of clonal selection is to generate a population of solutions near the solution candidate and then search in the neighborhood space. In other words, clonal selection enhances the local search by enlarging the search scope. In this paper, the clonal mutation procedure is as follows. Firstly, generate N_c clonal individuals. Then, execute mutation operator shown in Table 4. Finally, select the best individual from clone the ones.

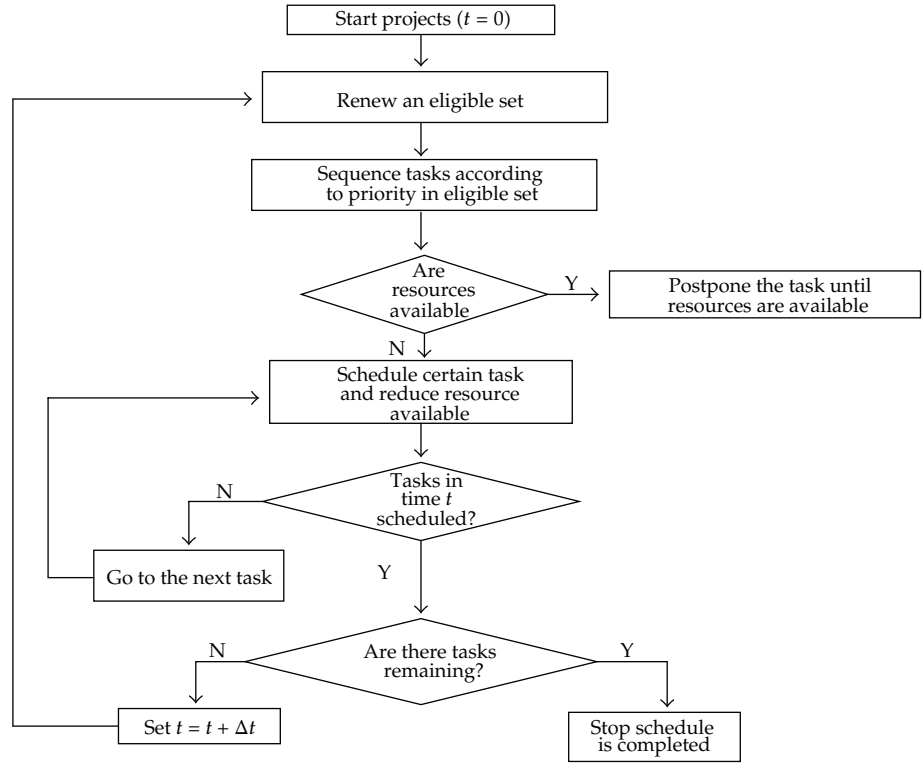


Figure 4: The evaluation process of an individual.

Table 4: Clonal mutation operators.

A schedule of parent chromosome								
$P_1(p - r(x))$			$P_2(p - r(y))$			$P_3(p - r(z))$		
Priority (1)	P riority (2)	Priority (3)	Priority (2)	Priority (1)	P riority (3)	Priority (3)	Priority (1)	Priority (2)
$P_{11}M_1{}^*$	$P_{12}M_2$	$P_{13}M_3{}^*$	$P_{21}M_2$	$P_{22}M_3{}^*$	$P_{23}M_1$	$P_{31}M_2{}^*$	$P_{32}M_1{}^*$	$P_{33}M_2$
⇓Mutation operator								
A schedule of offspring chromosome								
$P_1(p - r(y))$			$P_2(p - r(z))$			$P_3(p - r(x))$		
Priority (3)	Priority (2)	Priority (1)	Priority (2)	Priority (1)	Priority (3)	Priority (1)	Priority (3)	Priority (2)
$P_{11}M_3$	$P_{12}M_2$	$P_{13}M_2$	$P_{21}M_2$	$P_{22}M_1$	$P_{23}M_1$	$P_{31}M_3$	$P_{32}M_2$	$P_{33}M_1$

5.2.6. New Solution Generation

With an appropriate proportion, the operators described above can procedure a new generation:

(A) produce G_1 schedules after clonal selection.

(B) wipe off G_2 similar schedules after negative selection operator using affinity comparison.

(C) generate G_3 new schedules according to proportion at random:

$G_1 - G_2 + G_3 = G$, with G indicating the number of schedules in one generation.

6. Computational Experiments

6.1. Numerical Example

To illustrate the effectiveness of the algorithms described in this paper to solve MRCMPSP, we used a set of standard test problems systematically constructed by the project generator ProGen, which has been developed by Kolisch et al. [27]. They are available in the Project Scheduling Problem Library (PSPLIB) from the University of Kiel. The concrete steps are not described in detail due to the length limitation of the paper, and related parameters include a problem size I, the number of activities J_1 of project i, and the complexity of project i, the total number of renewable resource types R.

In this research, $I = 4$, $R = 3$, and complexity = 1.5. In addition, we also assume that the number of activities in each project is 18, 21, 16, and 17, respectively. Each availability of renewable resource is 13, 10, and 14, respectively, and detailed information about projects is not listed due to length limitation of the paper. DSM is introduced to model multi-project process, and the result is shown in Figure 5 after partitioning and banding algorithms.

The parameters of aiNet used to solve MRCMPSP are shown in Table 5.

Through calculation, the result is shown in Figure 6. After 60 iterations, minimum fitness reduces to 67 (units) and average fitness generates more fluctuation due to introducing new individuals. It means that immune network still search new solutions. The schedule including activity priority values and execution modes by the proposed method is given in Table 7. Furthermore, Figure 7 gives the curve of network individuals with the number of iterations. We can see from Figure 7 that the number of network individuals keeps its stability between 55 and 65. That is to say, the algorithm no longer finds out the better solution. This variation process agrees with variation curve corresponding to minimum fitness in Figure 6, which further verifies the effectiveness and robustness of aiNet algorithm. The comparison of the proposed algorithm with other heuristics approaches is given in Figure 8. The proposed aiNet approach is better than the heuristics approach with four different priority rules.

6.2. More Computational Simulation Experiments

In order to compare the performance of the aiNet algorithm with other heuristic algorithms, extensive experiments to test the algorithm have been illustrated in this section. The multi-project test problems consist of 2, 4, and 10 single projects generated by Kolisch et al. [27]. The number of activities in a project is 30, 60, 90, and 120, respectively. For each problem type, we generated 20 instances. Each activity can use up to four resources and have three modes. To generate the multi-project instances the single project problems were randomly selected network complexity and resource factor. Table 6 shows the combinations of the number of single projects used for the problem.

Setting parameters is a key issue to influence the performance of the algorithm. In order to get the most out of the aiNet algorithm, parameter tuning mentioned in Section 5.2.4 was implemented on a set of randomly selected multi-project problems. Table 8 shows results obtained by the various algorithms on the problem subsets. The proposed algorithm is compared with other approaches, including genetic algorithm (GA), simulated annealing

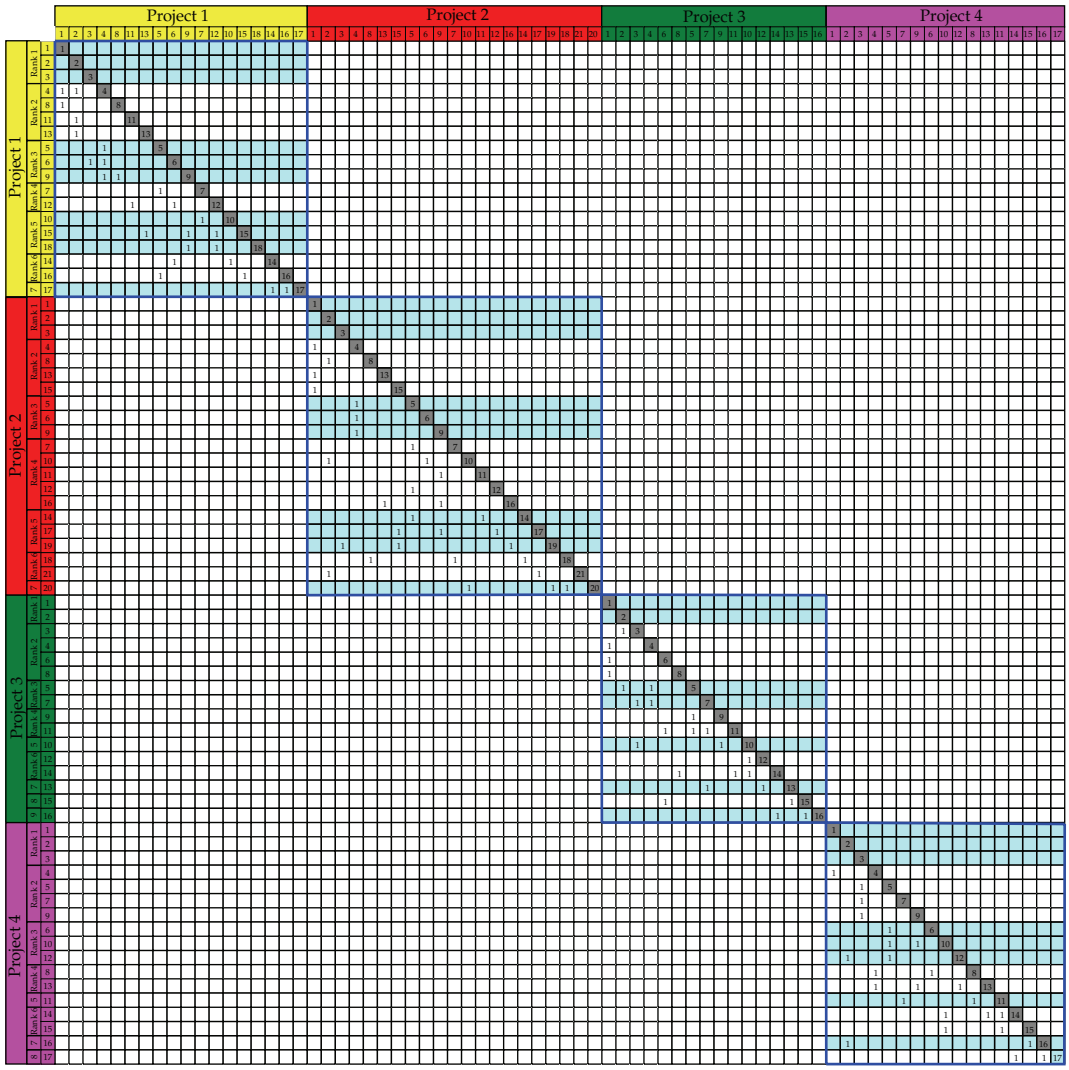


Figure 5: The DSM model of multi-projects.

Table 5: Set parameters of aiNet.

Threshold of affinity	Initial population size	The number of clonal individuals	Maximum number of iterations	Proportion of replacing individuals	Scale factor
$D_s = 0.8$	$N = 20$	$N_c = 10$	$N_{\text{gen}} = 100$	$d\% = 20\%$	$\lambda = 0.8$

(SA), ant colony optimization (ACO), and artificial immune system (AIS), in view of the average project delay (APD) and lower total makespan (LTM) of multi-project. In this table, the first column indicates the problem subset. The second column shows the number of schedules, which is used as the stopping criterion. The third to the fifth columns represent the averages of APD and LTM from various algorithms. From Table 8, it is seen that out of 12 subsets, the aiNet algorithm is superior to others when the problem size is larger. The average

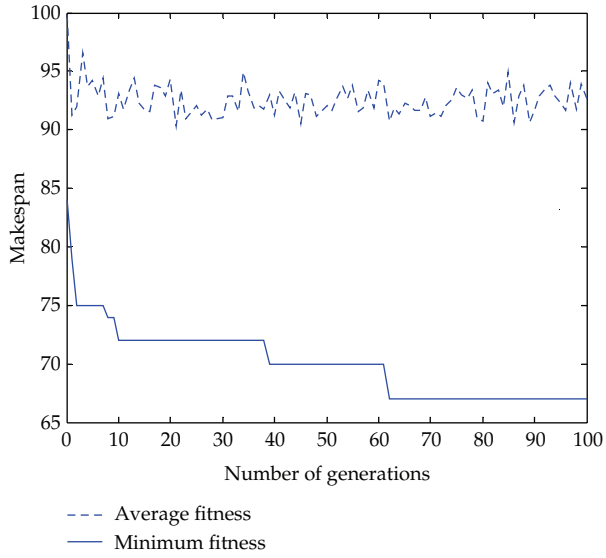


Figure 6: Convergence of makespan with the number of generations.

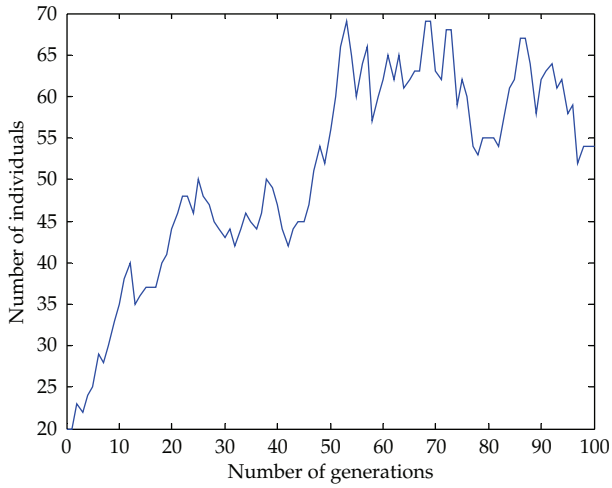


Figure 7: Relationship between individuals and generations.

project delays and lower total makespan obtained by the four approaches for all the problems are also compared, which show that our approach is obviously better than AIS and SA for all the problems but a little worse than ACO when the problem size is small. In addition, when the number of schedules increases, our approach still searches for the optimum but others are hardly influenced by it. This is because our approach introduces operations such as network compression, parameter tuning, and negative selection which are helpful to maintain the diversity of individuals. However, our algorithm may spend more computation time. There are two reasons causing this result: one is that clone selection may occupy more time in order to find out the local optimum; the other is that the new individuals are introduced to replace the more similar ones so as to find out the global optimum.

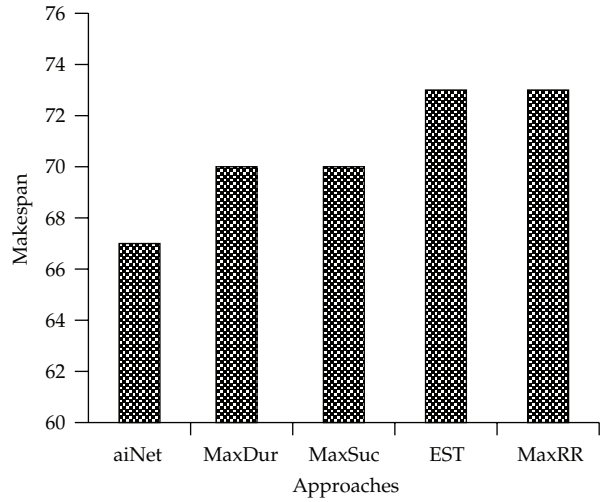


Figure 8: Comparison of schedules generated by the proposed aiNet and other existing approaches.

6.3. Discussions

From the examples mentioned above, the effectiveness of the aiNet method is verified and the following advantages of the aiNet-searching method for solving the multi-mode resource-constrained multi-project scheduling problem can be summarized.

6.3.1. Powerful Global Search Capability

The aiNet-searching method, similar to that of the evolution strategies and artificial neural networks, is based on local search intertwined with global search. In each iteration, a population of individuals is optimized locally through affinity proportional mutation; the global search is implemented by the population evolution operation; the network compression operation and the constant introduction of new randomly generated individuals help to keep the diversity in population. The satisfactory results obtained in a large number of experiments have demonstrated the powerful global search capability as well as the robustness of the aiNet-searching method in solving MRCMPSP despite of the uncertainty in algorithm.

6.3.2. Simplicity and High Speed

In the aiNet-searching method, there is no encoding of individuals required, a direct problem representation is directly used in iterations; there is also no need to use gradient method and mutation operation is directly performed according to individual fitness. In addition, despite of the powerful search capability, the basic operations adopted in aiNet-searching method are only mutation, evaluation, and network compression, which are very easy to be implemented in computer. By simplifying the parameters relevant to MRCMPSP, the number of parameters used for project scheduling problem decreases and, hence, reduces the dimensions of searching space, simplifying the problem and computational complexity.

Table 6: Problem instances for the MRCMPSP problem subset.

Problem subset	NOI	I	J_i	M	R
MP30_2	20	2	30	3	4
MP60_2	20	2	60	3	4
MP90_2	20	2	90	3	4
MP120_2	20	2	120	3	4
MP30_4	20	2	30	3	4
MP60_4	20	2	60	3	4
MP90_4	20	2	90	3	4
MP120_4	20	2	120	3	4
MP30_10	20	2	30	3	4
MP60_10	20	2	60	3	4
MP90_10	20	2	90	3	4
MP120_10	20	2	120	3	4

NOI: no. of instances; I: no. of projects; J_i : no. of activities of project i; M: no. of the modes the activity has; R: resources each activity can use.

Table 7: Optimal activity schedules generated by the proposed aiNet approach.

Proj.	Activity	Mode	Priority	Proj.	Activity	Mode	Priority	Proj.	Activity	Mode	Priority
	1	1	10		2	3	6		3	2	19
	4	1	26		5	1	29		6	1	24
1	7	2	20	1	8	3	31	1	9	2	37
	10	3	39		11	1	46		12	2	49
	13	1	64		14	2	56		15	2	58
	16	1	66		17	2	69		18	1	71
	1	2	1		2	2	5		3	2	4
	4	2	21		5	1	18		6	3	22
	7	1	25		8	2	34		9	2	36
2	10	2	33	2	11	2	43	2	12	2	50
	13	2	44		14	1	54		15	3	42
	16	2	48		17	1	57		18	1	51
	19	1	59		20	2	65		21	2	63
	1	2	8		2	1	3		3	2	11
	4	2	16		5	2	17		6	3	23
3	7	1	35	3	8	1	32	3	9	2	45
	10	2	41		11	2	52		12	1	61
	13	1	60		14	2	68		15	1	70
	16	2	72								
	1	1	2		2	2	7		3	1	9
	4	2	13		5	2	14		6	3	12
4	7	1	15	4	8	1	30	4	9	1	28
	10	1	27		11	2	40		12	2	38
	13	1	47		14	2	55		15	2	53
	16	1	62		17	2	67				

Table 8: Comparison of performance obtained by different algorithms.

Problem subset	Schedules	ai- Net			SA			ACO			AIS		
		APD	LTM	CPU_t	APD	LTM	CPU_t	APD	LTM	CPU_t	APD	LTM	CPU_t
	1000	15.3	72.2	0.08	**13.3**	78.6	**0.04**	17.5	**69.3**	**0.04**	16.6	74.1	0.06
MP30_2	5000	11.0	71.9	0.2	**10.9**	75.9	0.15	17.0	**69.1**	**0.18**	15.9	73.3	0.32
	10000	9.9	70.8	0.66	**9.8**	73.1	**0.50**	16.4	**68.5**	0.85	15.1	73.3	0.92
	1000	18.9	**99.6**	0.52	25.6	116.4	**0.3**	18.8	107.8	0.6	**18.1**	113.2	0.41
MP60_2	5000	**16.3**	**96.6**	2.1	21.5	111.7	**1.2**	18.5	106.5	2.5	17.5	112.1	1.9
	10000	**14.2**	**93.9**	7.8	19.3	109.8	**3**	18.5	106.1	9.8	17.1	110.5	7.8
	1000	28.3	181.2	1.51	**27**	**176.2**	**0.54**	35.6	207.9	0.85	27.1	192.2	0.71
MP90_2	5000	**24.1**	**174**	6.1	26.9	175.9	2.3	34.9	204.6	2.51	26.5	189.1	**1.9**
	10000	**21.9**	**169.3**	13.4	26.9	175.9	6.2	34.5	201.1	6.8	26.1	187.9	**5.8**
	1000	67.2	207	2.4	66.7	264.4	0.75	**65.9**	**202.4**	1.1	87.6	284.6	**0.74**
MP120_2	5000	66.5	203.5	11.2	66.3	242.4	**2.84**	**65.9**	202.4	**3.2**	84.1	262.2	3.5
	10000	**65.8**	**199.1**	25	66.3	232.2	**8.7**	65.9	201.9	9.7	83.8	260	10.1
	1000	21.2	99.6	0.09	**20.1**	**92.5**	**0.06**	22.4	93.5	0.07	25.6	101.4	0.07
MP30_4	5000	19.4	95.3	0.24	19.3	91.3	0.19	**19.2**	**89.2**	**0.18**	24.3	97.5	0.21
	10000	**18.6**	92.2	0.7	18.6	91.3	**0.55**	19	**89**	0.61	22.1	95.1	0.61
	1000	42.3	131	0.72	**41.9**	147.4	**0.41**	42.1	130.4	0.98	**41.9**	**129.8**	0.62
MP60_4	5000	**39.4**	125.4	4.3	41.5	140.3	**1.79**	42.0	**124.7**	3.7	41.9	127.6	3.9
	10000	**35.1**	**121.1**	10.2	41.5	138.6	**4.5**	41.8	124.5	8.8	41.8	127.5	9.5
	1000	69.3	254.3	1.9	87.4	**253.3**	0.9	**67.3**	287.9	1.2	67	296	**0.7**
MP90_4	5000	65.5	**251.4**	9.5	81.3	252.4	**2.76**	**65.1**	277.6	5.1	66.9	275	4.8
	10000	**63.1**	**239.1**	25	75.6	252	**6.1**	64.8	274.1	10.3	66.9	251.4	11.3
	1000	87.5	348.6	3.1	121.6	345	**1.2**	150.4	387.9	1.5	**86.9**	**340.5**	1.34
MP120_4	5000	**85.5**	**325.1**	13.1	119.3	340.1	**5.3**	144.3	366.4	6.1	86.5	338.1	5.77
	10000	**83.1**	**319.6**	34	115.6	335.7	**10.5**	140.2	354.2	12.9	86	335.2	14.8
	1000	31.1	**110.6**	0.12	30.1	152.5	0.09	42.4	193.5	**0.08**	**30**	141.4	0.11
MP30_10	5000	29.4	**105.3**	0.49	29.9	141.3	**0.41**	39.2	189.2	0.4	**29.3**	127.5	0.52
	10000	**28.5**	**102.2**	1.2	28.8	132.9	**0.95**	39	189	1.1	29	115.1	1.3
	1000	62.3	**181**	0.99	88.9	246.4	0.85	82.1	270.1	1.3	**62**	229.1	**0.82**
MP60_10	5000	**59.4**	**175.4**	4.8	85.3	241.4	**3.9**	82.0	254.3	6.3	60.8	227.2	4.1
	10000	**55.1**	**172.1**	12.1	84.1	238.5	**8.1**	81.8	244.5	14.1	59.9	225	11.1
	1000	130.9	**319.5**	2.4	**128.7**	353.1	**1.1**	129.3	387.9	1.7	137	326	1.2
MP90_10	5000	122.5	**311.2**	12.3	126.6	351.4	3.2	**120.1**	361.6	5.3	126.9	324.3	**3.1**
	10000	**113.1**	**306.7**	41	125.6	351	**7.5**	119.8	344.2	13.1	121.1	321.4	12.1
	1000	291.4	497.2	5.2	352.4	652.3	2.7	288.3	487.9	3.1	287.9	540.5	3.34
MP120_10	5000	275.1	483.1	18.3	344.9	640.9	7.3	280.1	486.4	8.2	286.5	538.1	9.77
	10000	261.1	473.1	54	335.1	637.7	15.5	275.6	484.2	17.9	286	535.2	27.6

6.3.3. More Candidate Solutions with Large Diversity

The two most important characters of the aiNet-searching method are that the dynamically adjusted optimum population size through metadynamics (diversity introduction) and network compression. Namely, some of the similar individuals are eliminated to avoid redundancy when the population reaches a stable state; then a number of new randomly generated ones are added to current population, this strategy leads to the large diversity between candidate solutions found.

Besides, it should be noted that when the improved aiNet algorithm is used to solve MRCMPSP, the ideal condition for convergence is that, after several times of iteration, the number of network individuals should not change, indicating that the immune network reach to stabilization and the algorithm cannot go further to find better local solutions. But in real tests, the activity mode selection, which has great impact on its execution cycles and priority, makes the complexity of problem solving exponentially rise with the increase of the number of activity execution modes. For example, for a multi-project scheduling problem containing activities no more than 100 and average number of activity modes no more than 3, the number of configuration schemes for all activities in different modes may be as high as 3^{100} and it is unlikely to go over all the schemes using any algorithms. Due to this reason, if the number of network individuals fluctuates in a certain range during several times of iteration, it will mean that the algorithm meets the convergence conditions.

7. Conclusions

In this paper, an improved artificial immune algorithm used to solve MRCMPSP is put forward. Firstly, the mathematic model of MRCMPSP is set up. Secondly, the DSM method is adopted to simplify the mathematic model of MRCMPSP so as to improve quality and quantity of candidate solutions. And then, operations including clonal selection, negative selection, and network compression are used to realize the local searching, and global searching which will assure that the algorithm has a powerful searching ability and also avoids the possible combinatorial explosion. Subsequently, a set of case studies are given to test the searching ability of the algorithm. The results show that it is efficient and effective comparing to others.

Future works may include the following two aspects: (1) how to use aiNet algorithm to solve multiobject scheduling problem such as time, cost, and resource utilization rate; (2) the vast majority of the research efforts in project scheduling assume complete information about the scheduling problem to be solved and a static deterministic environment. However, in the real world, project activities are subject to considerable uncertainty. As a result, how to introduce risk management strategies to solve project scheduling problem in a dynamic environment is another area that needs to be investigated in future.

Acknowledgments

This research is supported by the National Natural Science Foundation of China (Grant no. 71071141, 71071140, 71001088, and 60905026), Research Fund for the Doctoral Program of Higher Education of China (Grant no. 20103326110001, 20103326120001, and 20093326120004), Humanity and Sociology Foundation of Ministry of Education of China (Grant no. 11YJC630019 and 11YJA630161), Zhejiang Provincial Natural Science Foundation

of China (no. Z1091224, Y7100673, Y6090413, Y1091164, and Y1111039) and the Scientific Research Fund of Zhejiang Province, China (Grand no. 2011C23008).

References

[1] J. F. Gonçalves, J. J. M. Mendes, and M. G. C. Resende, "A genetic algorithm for the resource constrained multi-project scheduling problem," *European Journal of Operational Research*, vol. 189, no. 3, pp. 1171–1190, 2008.

[2] B. De Reyck and W. Herroelen, "Multi-mode resource-constrained project scheduling problem with generalized precedence relations," *European Journal of Operational Research*, vol. 119, no. 2, pp. 538–556, 1999.

[3] J. Błażewicz, J. K. Lenstra, and A. H. G. Rinnooy Kan, "Scheduling subject to resource constraints: classification and complexity," *Discrete Applied Mathematics*, vol. 5, no. 1, pp. 11–24, 1983.

[4] J. P. Stinson, E. W. Davis, and B. M. Khumawala, "Multiple resource-constrained scheduling using branch and bound," *AIIE Trans*, vol. 10, no. 3, pp. 252–259, 1978.

[5] N. Christofides, R. Álvarez-Valdés, and J. M. Tamarit, "Project scheduling with resource constraints: a branch and bound approach," *European Journal of Operational Research*, vol. 29, no. 3, pp. 262–273, 1987.

[6] E. L. Demeulemeester and W. Herroelen, "A branch and bound procedure for the multiple resource-constrained project scheduling problem," *Management Science*, vol. 38, no. 12, pp. 1803–1818, 1992.

[7] J. R. Montoya-Torres, E. Gutierrez-Franco, and C. Pirachicán-Mayorga, "Project scheduling with limited resources using a genetic algorithm," *International Journal of Project Management*, vol. 28, no. 6, pp. 619–628, 2010.

[8] N. Xu, S. A. McKee, L. K. Nozick, and R. Ufomata, "Augmenting priority rule heuristics with justification and rollout to solve the resource-constrained project scheduling problem," *Computers and Operations Research*, vol. 35, no. 10, pp. 3284–3297, 2008.

[9] M. Mobini, Z. Mobini, and M. Rabbani, "An Artificial Immune Algorithm for the project scheduling problem under resource constraints," *Applied Soft Computing Journal*, vol. 11, no. 2, pp. 1975–1982, 2011.

[10] K. Ziarati, R. Akbari, and V. Zeighami, "On the performance of bee algorithms for resource-constrained project scheduling problem," *Applied Soft Computing Journal*, vol. 11, no. 4, pp. 3720–3733, 2011.

[11] V. D. Wiley, R. F. Deckro, and J. A. Jackson, "Optimization analysis for design and planning of multi-project programs," *European Journal of Operational Research*, vol. 107, no. 2, pp. 492–506, 1998.

[12] A. Lova and P. Tormos, "Analysis of scheduling schemes and heuristic rules performance in resource-constrained multiproject scheduling," *Annals of Operations Research*, vol. 102, pp. 263–286, 2001.

[13] T. R. Browning and A. A. Yassine, "Resource-constrained multi-project scheduling: priority rule performance revisited," *International Journal of Production Economics*, vol. 126, no. 2, pp. 212–228, 2010.

[14] K. Kim, Y. Yun, J. Yoon, M. Gen, and G. Yamazaki, "Hybrid genetic algorithm with adaptive abilities for resource-constrained multiple project scheduling," *Computers in Industry*, vol. 56, no. 2, pp. 143–160, 2005.

[15] S. Kumanan, G. Jegan Jose, and K. Raja, "Multi-project scheduling using an heuristic and a genetic algorithm," *International Journal of Advanced Manufacturing Technology*, vol. 31, no. 3-4, pp. 360–366, 2006.

[16] N. R. Joglekar and D. N. Ford, "Product development resource allocation with foresight," *European Journal of Operational Research*, vol. 160, no. 1, pp. 72–87, 2005.

[17] O. Lambrechts, E. Demeulemeester, and W. Herroelen, "Proactive and reactive strategies for resource-constrained project scheduling with uncertain resource availabilities," *Journal of Scheduling*, vol. 11, no. 2, pp. 121–136, 2008.

[18] S. Adhau, M. L. Mittal, and A. Mittal, "A multi-agent system for distributed multi-project scheduling: an auction-based negotiation approach," *Engineering Applications of Artificial Intelligence*. In press.

[19] I. S. Kurtulus and S. C. Narula, "Multi-project scheduling: analysis of project performance," *IIE Transactions*, vol. 17, no. 1, pp. 58–66, 1985.

[20] D. V. Steward, "The design structure system: a method for managing the design of complex systems," *IEEE Transactions on Engineering Management*, vol. 28, no. 3, pp. 71–74, 1981.

[21] H. F. Hung, H. P. Kao, and Y. S. Juang, "An integrated information system for product design planning," *Expert Systems with Applications*, vol. 35, no. 1-2, pp. 338–349, 2008.

[22] R. Xiao and T. Chen, "Research on design structure matrix and its applications in product development and innovation: an overview," *International Journal of Computer Applications in Technology*, vol. 37, no. 3-4, pp. 218–229, 2010.

[23] R. Xiao, T. Chen, and C. Ju, "Research on product development iterations based on feedback control theory in a dynamic environment," *International Journal of Innovative Computing, Information and Control*, vol. 7, no. 5, pp. 2669–2688, 2011.

[24] S. Hartmann and R. Kolisch, "Experimental evaluation of state-of-the-art heuristics for the resource-constrained project scheduling problem," *European Journal of Operational Research*, vol. 127, no. 2, pp. 394–407, 2000.

[25] H. Chtourou and M. Haouari, "A two-stage-priority-rule-based algorithm for robust resource-constrained project scheduling," *Computers and Industrial Engineering*, vol. 55, no. 1, pp. 183–194, 2008.

[26] N. J. Castro and J. Timmis, "An artificial immune network for multimodal function optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '02)*, pp. 699–674, Hawaii, Hawaii, USA, May 2002.

[27] R. Kolisch, A. Sprecher, and A. Drexl, "Characterization and generation of a general class of resource constrained project scheduling problems," *Management Science*, vol. 41, no. 10, pp. 1693–1703, 1995.