An Improved Ant Colony Optimization Algorithm for Software Project Planning and Scheduling

Avinash Mahadik

Department Of Computer Engineering
Alard College Of Engineering And Management, Marunje, Pune

Email-avinash.mahadik5@gmail.com

Abstract—Research into developing effective computer aided techniques for planning software projects is important and challenging for software engineering. Different from projects in other fields, software projects are people-intensive activities and their related resources are mainly human resources. Thus, an adequate model for software project planning has to deal with not only the problem of project task scheduling but also the problem of human resource allocation. But as both of these two problems are difficult, existing models either suffer from a very large search space or have to restrict the flexibility of human resource allocation to simplify the model. To develop a flexible and effective model for software project planning, this paper proposes a novel approach with an ant colony optimization (ACO) algorithm. The proposed approach represents a plan by a task list and a planned employee allocation matrix. In this way, both the issues of task scheduling and employee allocation can be taken into account.

Keywords - Software project planning, project scheduling, resource allocation, workload assignment, ant colony optimization (ACO).

I INTRODUCTION

There is a rapid development of software industry so software companies are now facing a highly competitive market. To succeed, companies have to make efficient project plans to reduce the cost of software construction. However, in medium to large-scale projects, the problem of project planning is very complex and challenging. Due to the importance and difficulty of software project planning, there is a growing need for developing effective computer aided tools for software project planning in recent years. To plan a software project, the project manager needs to estimate the project workload and cost and decide the project schedule and resource allocation. To plan a software project, the project manager needs to estimate the project workload and cost and decide the project schedule and resource allocation. To build more suitable models and tools, traditional project management techniques need to be further extended.

In this paper, we propose a practical and effective approach for the task scheduling and human resource allocation problem in software project planning with an ant colony optimization (ACO) algorithm. The representation scheme is composed of a task list and a planned employee allocation matrix. The task list defines the priorities of tasks to consume resources, and the planned employee allocation matrix specifies the originally planned workload assignments. In this way, the representation takes both the issues of task scheduling and resource allocation into account. ACO was proposed by Dorigo and Gambardella in the early 1990s and by now has been successfully applied to various combinatorial optimization problems. As ACO builds solutions in a step-by-step manner and enables the use of problem-based heuristics to guide the search direction of ants, it is possible to design useful heuristics to direct the ants to schedule the critical tasks as early as possible and to assign the project tasks to suitable employees with required skills. Therefore, ACO promises to converge fast and perform well on the considered problem. Various ACO variants have been developed. Two of the best performing ACO variants include ant colony system (ACS) and max-min ant system (MMAS). In this paper, we follow the ACS
variant to develop the ACO approach to the considered software project planning problem. The main characteristics of the ACS are in two aspects. First, in the solution construction procedure, the ACS applies a pseudorandom proportional selection rule which aggressively biases selecting the components with the maximum pheromone and heuristic values. In this way, the ACS strongly exploits the past search experience of ants and has a fast convergence speed. Second, in the pheromone management procedure, ACS has two pheromone updating rules, namely, the global updating and the local updating.

II LITERATURE REVIEW
In the literature review, several works have been done on developing search-based approaches for software project planning. Duggan et al. [2] and Barreto et al. [3] built models for the staffing problem of software projects and proposed genetic algorithm (GA) approaches. But, their models only focused on staffing and the problem of task scheduling was not considered. Chang et al. proposed the software project management net (SPMnet) model [4] and the project management net (PM-net) model [5] successively, and then further improved the models to a richer version with a GA [6]. Other GA-based approaches were also proposed in [6] and [10], [11]. In these approaches, a plan is described by a 2D matrix which specifies the workload of each employee on each task. But, as this representation is inadequate for modeling resource conflict, these models all implicitly uses the “Mongolian Horde” strategy [7] that assumes an unlimited number of employees can be assigned to a task and an employee can join an unlimited number of tasks simultaneously, which is usually not the case in practice. Bellenguez and Ne’ron [8], [9] proposed a multiskill scheduling model by extending the traditional RCPSP model. The model considers both the problems of human resource allocation and task scheduling, and takes the skill proficiency of employees and resource conflict into account. Tabu search (TS) [10], branch and bound [10], and GA [11] have been developed for the model. In all of the above-mentioned models, there is an assumption that pre-emption is not allowed.

As discussed, this assumption reduces the flexibility of human resource allocation for software projects. Task pre-emption in software projects is only considered in a few studies. In Chang’s recent work [3], he improved his previous scheduling model by introducing a 3D matrix representation, specifying the workload assignment of each employee for each task on each time period. Although this representation is much more flexible, it makes the search space very large and suffers from the problem of desultory assignment of workloads.

III OBJECTIVES
The aim of the project is to develop an efficient and improved Ant Colony Optimization (ACO) algorithm for multiple Quality of Service unicast routing mechanism. The project uses unicast routing technique with the ant colony optimization algorithm. The Ant colony algorithm simulates the behavior of ants to find the optimal path from the source to destination. Similar to ant’s behavior of using pheromone to mark the path travelled by them and select the optimal path among all paths travelled, the algorithm uses the pheromone concept to find the optimal paths to deliver the packets from source to destination across the network. The main reason is that, differently from other projects, a software project is a people-intensive activity and its related resources are mainly human resources. Different software project tasks require employees with different skills, and skill proficiency of employees significantly influences the efficiency of project execution. A plan for a project must specify when the tasks of the project are processed and how the workloads of employees are assigned to the tasks.

IV PROPOSED METHODOLOGY
An ACO algorithm works by dispatching a group of artificial ants to build solutions to the problem iteratively. In general, an ACO algorithm can be viewed as the interplay and the repeated execution of the following three main procedures:

1. Solution construction—During each iteration of the algorithm, a group of
ants set out to build solutions to the problem.

2. Pheromone management—Along with the solution construction procedure, pheromone values are updated according to the performance of the solutions built by ants.

3. Daemon actions—Daemon actions mean the centralized operations that cannot be done by single ants.

Figure 1. Proposed Model Of Ant Colony Optimization Algorithm

The different modules of the proposed Ant Colony Optimization approach are described below.

**Input module**: The following data pertaining to the problem are given as input: Number of Tasks (n), number of machines in the shop (m), number of operations Ji of each task i.

**Initialization module**: The number of ants is defined, and the pheromone trails used by them for constructing solutions are initialized. This problem uses two pheromone trails: pheromone trail intensity for route selection gives information about the desirability of choosing route r for operation Oij at iteration tn and pheromone trail intensity, which indicates the desirability of choosing operation Oij directly after the operation Oi’j’ is loaded on machine k, is used for task conflict resolution while generating feasible schedule.

**Solution construction and Evaluation module**: Each ant constructs a solution in two stages. In the 1st stage, an ant, at each construction step, allocates an operation of a particular task to one of its available resources. The ants use a probabilistic choice rule which is a function of the pheromone trail and a heuristic information based on processing time. In the 2nd stage, on allocation of all operations to the machines, each ant generates a schedule based on algorithm.

**Sorting module**: The best solution of the current iteration and the global best are sorted and stored separately.

**Termination Check module**: A specified number of iterations (no_iter) is estimated to terminate the algorithm depending on the size of the problem. Termination directs to the output module; otherwise, continue to the pheromone updating module.

**Pheromone updating module**: At the end of iteration, the pheromone trails corresponding to only one single ant is updated. This ant may be the one which found the best solution in the current iteration or the one which found the best solution from the beginning of the trial. A dynamically mixed strategy of pheromone updating is used where best solution is chosen as default for updating the pheromones and using global solution only every fixed number of iterations. The frequency of using global solution for the pheromone update is increased during the search.

**Output Module**: This module prints the best solution of the optimal route choices of all operations and schedule for minimum make span time criterion.

**V ANT COLONY OPTIMIZATION APPROACH**

**Overview of the ACO Algorithm**

To solve the software project planning problem, this paper proposes an ACO approach. The underlying idea of ACO is to simulate the foraging behavior of ants. When ants search for food, they usually deposit a special chemical on the path they travel through. This kind of chemical, which is called pheromone, serves as a medium for ants to communicate with each other. By sensing the concentration of pheromone, other ants can follow the path to find the food.
1. Solution construction—During each iteration of the algorithm, a group of ants set out to build solutions to the problem. Each ant builds a solution in a constructive manner by selecting components step by step to form a complete solution. The selections are made according to pheromone and heuristic information.

2. Pheromone management—Along with the solution construction procedure, pheromone values are updated according to the performance of the solutions built by ants. Ants tend to deposit more pheromone to the components of better-performed solutions.

3. Daemon actions—Daemon actions mean the centralized operations that cannot be done by single ants. In the design of ACO algorithms, daemon actions are optional.

By now, various ACO variants have been developed. Two of the best performing ACO variants include ant colony system (ACS) and max-min ant system (MMAS). In this paper, we follow the ACS variant to develop the ACO approach to the considered software project planning problem. The main characteristics of the ACS are in two aspects. First, in the solution construction procedure, the ACS applies a pseudorandom proportional selection rule which aggressively biases selecting the components with the maximum pheromone and heuristic values. In this way, the ACS strongly exploits the past search experience of ants and has a fast convergence speed. Second, in the pheromone management procedure, ACS has two pheromone updating rules, namely, the global updating and the local updating. The global updating rule makes the components corresponding to the best-so-far solution become more attractive. The local updating rule reduces the pheromone on the components just selected by ants to increase the search diversity of the algorithm.

Solution Construction
Construction of Task List A task list is an order of tasks t1,t2,......tn that satisfies the precedence constraints defined by the TPG. Here we first define the pheromone and the heuristic for task list construction. Then the procedure for an ant to build a task list will be.

particular, although there are effective heuristics for guiding the construction of task list and the selection of employees, it is difficult to find an effective heuristic that can tell us how many working hours an employee works for a specific task is the most suitable. The procedure is performed by mutating the best-so-far solution found by ACO using either of the following mutation operators.

Operator 1—Task list mutation: First, a task tj is randomly selected from the task list. Then a destination position is randomly generated.

Operator 2—Employee allocation matrix mutation: First, a task tj is randomly selected. Then an employee i who has been assigned to work for this task is randomly selected and his working hours for this task are reset to pw. Finally, a new employee u who has not been selected to work for the task is randomly selected and his working is described in detail.
VI LIMITATION

The solution construction method used in the proposed ACO makes it capable to manage through the entire solution space and provide all possible instances that an enumerative search can and is therefore capable of finding the optimal or near-optimal solutions. Since the proposed ACO uses schedule generation procedure for generating active feasible schedules, therefore, the proposed ACO can be easily adapted to generate schedules for any scheduling objective. In future research, it will be interesting to consider employee experience and the training model [6] to make the considered problem more comprehensive. Including uncertainty treatment in the software project planning model is also a promising research topic.

VII CONCLUSION

ACO is a recently proposed heuristic approach for solving hard combinatorial optimization problems. Artificial ants implement a randomized construction heuristic which makes probabilistic decisions. The accumulated search experience is taken into account by the adaptation of the pheromone trail. ACO shows great performance with the structured problems like network routing. In ACO Local search is extremely important to obtain good results. The proposed algorithm manages to yield better plans with lower costs and more stable workload assignments compared with other existing approaches. In addition, since the model proposed in this paper provides a flexible and effective way for managing human resources, it is promising to apply the proposed approach to other complex human-centric projects like consulting projects.

REFERENCES


