Open access Journal International Journal of Emerging Trends in Science and Technology

Improving Evolutionary Algorithm Design for Complex Real Time Problems

Authors
Volga Benjamin F[1], Jose Hormese[2]
[1]M.Tech in CSE, Marian Engineering College, Trivandrum
[2]Associate Professor in CSE, Marian Engineering College, Trivandrum

Abstract
Several types of evolutionary algorithms (EAs) have been applied to solve the project scheduling problem (PSP). The performance of these EAs highly depends on design choices for the EA. Based on the dedications of particular tasks the employee can work on multiple jobs simultaneously. This consist of normalizing employees' dedication for different tasks to ensure they are not working overtime; a fitness function that requires fewer pre-defined parameters and provides a clear gradient towards feasible solutions; and an improved representation and mutation operator. Both the theoretical and empirical findings show that the design is very effective. A repair mechanism is that which facilitates the search for feasible schedules without overwork. Their repair mechanism considers the maximum total dedication of any employee at any point of time during the generated schedule. The problem of overwork can be alleviated and hence can remove a crucial obstacle in the search process of EAs by using the following an approach: normalisation. Combining the use of normalization to a population gave the best results in the experiments, and normalization was a principle insight for the practical effectiveness of the existing system. Existing system concludes that normalisation is not always effective. The proposed work is based on comparison of an earlier technique used in this area called 'repair mechanism'. Proposing the collaboration of both techniques to arrive at the best optimal solutions for the PSP and at the end testing the feasibility of the proposed idea.

Index terms:- *Evolutionary Algorithm(EA), Genetic Algorithm(GA), Project Scheduling Problem(PSP), Normalisation, Repair*

1. INTRODUCTION

Scheduling is the way we actually manage a project. Without scheduling, nothing or nobody is managing the project and hence amounts to failure of a project. Scheduling describes guidance and pathway for a project to run. It defines certain milestones and deliverables which need to be achieved on a timely basis for successful completion of a project. Monitoring the schedule provides an idea of the impact the current problems are having on the project, and provides opportunities to enhance or reduce the scope of a milestone/phase in the project. It also provides a medium for continuous feedback on how the project is progressing and if there are issues that need to be dealt with or if the client needs to be told about a delay in delivery.

Project scheduling problem is to determine the schedule of allocating resources so as to balance the total cost and the completion time. This paper considers a type of project scheduling problem with activity duration times.

Effort: The estimated tasks and activities required to manage the project and produce deliverables. • *Schedule*: The estimated tasks and events needed to complete the project, organized into a structured sequence to meet a specified project end date. • *Resources*: The estimated staff resources needed to complete the project, according to number, type, work hours, and skills. • *Budget*: The estimated cost of the project, allocated to tasks, resources and phases as needed to complete the project. • *Vendors and Procurement*: The anticipated performance of contractors, vendors and suppliers to deliver goods and services according to contracts and project requirements. • *Management Process*: Management standards can serve as a constraint on project performance, adding quality control overhead.

Evolutionary computation, offers practical advantages to the researcher facing difficult optimization problems. The evolutionary algorithm can be applied to problems where heuristic solutions are not available or generally lead to unsatisfactory results. As a result, evolutionary algorithms have recently received increased interest, particularly with regard to the manner in which they may be applied for practical problem solving. Compared to other global optimization techniques, evolutionary algorithms (EA) are easy to implement and very often they provide adequate solutions

1.1 EXISTING SYSTEM

This theoretical approach has been applied to many problems from combinatorial optimisation, and it has led to many interesting results about how EAs perform. Inspired by theoretical insight, a more practical contribution is made: a new mechanism for normalizing dedication values. Alba and Chicano showed that GAs spend most of their effort avoiding overwork. This is a major problem even on simple instances, as GAs have a very low hit rate in finding feasible schedules. Our approach normalizes dedication values automatically. Xin Yao et al. shows that Normalisation [16] is embedded into the genotype-phenotype mapping, extending the mapping. Whenever an employee has a total dedication greater than the maximum dedication, all dedication values to active tasks are scaled accordingly. This completely removes overwork from the problem. Instead of struggling to find a solution without overwork, our approach allows EAs to focus on the solution quality. In addition, we introduce a tailored mutation operator and a new way of dealing with infeasible solutions, guiding EAs to reach feasibility. Both theoretical and empirical results showed that the developments are very effective. A (1+1) EA[9] in this existing design performs better than the existing GA . It reaches feasible solutions in 100% of all tested cases, whereas the GA struggles to reach feasibility. Also in terms of solution quality this approach proved better.

1.2 PROBLEM STATEMENT

The existing system has proved that hit rates are very high and time complexity has reduced by introducing normalization as a tool. In the previous works repair mechanisms were used to optimize the project scheduling problem using EAs. Repair mechanism considers the maximum total dedication of any employee at any point of time during the generated schedule, This implies that overwork is eliminated, but this is done at the expense of increasing the execution time of all tasks. The theoretical and empirical results have confirmed that normalization is very effective in avoiding overwork, and that EAs with normalization quickly and consistently evolve schedules of good quality.

But it could not be proved that normalization is always better than repair, with regard to the completion time only. However, what can be said is that the optimal completion time with normalization is never more than the optimal completion time with repair. This is because the all-ones dedication matrix has every employee always working full time, leading to a minimal completion time.

2. LITERATURE SURVEY

2.1 EVOLUTIONARY ALGORITHM

Evolutionary algorithms often perform well approximating solutions to all types of problems because they ideally do not make any assumption about the underlying fitness landscape; this generality is shown by successes in fields as diverse as engineering, art, biology, economics, marketing, genetics, operations research, robotics, social sciences, physics, politics and chemistry

Techniques from evolutionary algorithms applied to the modeling of biological evolution are generally limited to explorations of micro evolutionary processes and planning models based upon cellular processes. The computer simulations Tierra and

Avida attempt to model macro evolutionary dynamics.

During the past 35 years the Evolutionary Computation (EC) research community has been studying properties of Evolutionary Algorithms (EA). Many claims have been made – these varied from a promise of developing an automatic programming methodology to solving virtually any optimisation problem (as some Evolutionary Algorithms are problem independent). However, the most important claim was related to applicability of Evolutionary Algorithms[5][8] to solving very complex business problems, i.e. problems, where other techniques failed.

2.2 IMPLEMENTATION OF BIOLOGICAL PROCESS

1. Generate the initial population of individuals randomly - first Generation
2. Evaluate the fitness of each individual in that population
3. Repeat on this generation until termination (time limit, sufficient fitness achieved, etc.): Select the best-fit individuals for reproduction – parents
4. Breed new individuals through crossover and mutation operations to give birth to offspring
5. Evaluate the individual fitness of new individuals
6. Replace least-fit population with new individuals

3. GENETIC ALGORITHM

In the computer science field of artificial intelligence, a genetic algorithm (GA) [11] is a search heuristic that mimics the process of natural selection. This heuristic (also sometimes called a meta-heuristic) is routinely used to generate useful solutions to optimization and search problems. Genetic algorithms belong to the larger class of evolutionary algorithms (EA), which generate solutions to optimization problems using techniques inspired by natural evolution, such as inheritance, mutation, selection, and crossover.

Genetic algorithms find application in bioinformatics, phylogenetics, computational science, engineering, economics, chemistry, manufacturing, mathematics, physics, pharmacometrics and other fields.

4 APPLICATIONS OF EVOLUTIONARY ALGORITHMS

During the past 35 years the Evolutionary Computation (EC) research community has been studying properties of Evolutionary Algorithms (EA). Many claims have been made – these varied from a promise of developing an automatic programming methodology to solving virtually any optimisation problem (as some Evolutionary Algorithms are problem independent). However, the most important claim was related to applicability of Evolutionary Algorithms[5][8] to solving very complex business problems, i.e. problems, where other techniques failed.

4.1 SOFTWARE PROJECT MANAGEMENT

Software project management [12] is, like many other activities in the software process, a problem-solving issue. It involves what is to be done, a decision regarding how to do it, the control of how it is being done, and an evaluation (or measurement) of what was done. The issue on “what” typically takes the form of a plan. Tausworthe introduced WBS into software project planning in the early 80's. WBS provides a hierarchical view for the whole project, but the precedence relationships among the work packages are not clearly identified in the WBS. Currently, most of the project planning techniques used today are based on network-based techniques, such as PERT and CPM [13], which was originated in the early 50's. However, the classical project management models are inadequate for large-scale distributed software project management, since they are poor in modeling and analysis of the concurrent and evolutionary characteristics embedded in a software project Liu et al. The issue on “how” is the allocation of resources (e.g. a schedule or budget). Unfortunately, according to the survey in resource allocation still heavily relies on software managers. That is, software managers need to manually assign the resources to different tasks based on their experience in order to optimize resource usage, shorten the cycle time, and control the evolutionary nature of project development. This is extremely difficult for large-scale software projects, which involve hundreds of tasks, programmers, and a wide variety of hardware and software resources[2]. Since the search space for such a problem class is typically huge, even with computer tools we will not be able to find optimal solutions reasonably. In order to solve this problem, we propose a new approach to generate near-optimal resource allocation and scheduling based on genetic algorithms [2].

5. PROJECT SCHEDULING PROBLEM

A Project Scheduling Problem [13] for Software Development is a variant of Project Scheduling Problem where the software development model can be presented as a set of software activities, a set of developer skills and a set of resources specified on money and the total time divided on time per activity. This paper presents an instance set of Project Scheduling Problem for Software Development for projects of software development.

The Project Scheduling Problem (PSP) is a generic name given to a whole class of problems in which it is necessary to schedule in an optimal way, the time, cost and resources of projects . The application areas are usually are defined in terms of: technical elements (development of software, pharmaceutical drugs or civil engineering, planning of production systems), elements of the administration (project scheduling problems, manu-facturing management, technology management, contracts with the government or development of new products), and groups of industry (industrial engineering, automobiles, chemicals or financial services).

Project planning is difficult because it inevitably involves uncertainty. Uncertainty in real-world projects arises from the following characteristics:

□□□*uniqueness* (no similar experience)

□□□*variability* (trade-off between performance measures like time, cost and quality)

□□□*ambiguity* (lack of clarity, lack of data, lack of structure and bias in estimates)

Many different techniques and tools have been developed to support better project scheduling, and these tools are used seriously by a large majority of project managers Yet, quantifying uncertainty is rarely prominent in these approaches.

6. NORMALISING DEDICATIONS

Instead of penalizing overwork and letting an EA search for feasible solutions, we normalize [16] the dedication values. If at some point of time the total dedication of an employee e_i across all active tasks is $d_i > 1$ then her/his dedication for all tasks is divided by d_i. This reflects a very natural way of an employee dividing her/his attention to several tasks.

For instance, assume there are two tasks t_1, t_2 suitable for employee e_1. Assume also that the employee works on both tasks at overlapping time intervals. If $x_{1,1} + x_{1,2} > 1$, the employee works overtime whenever she/he works on both t_1 and t_2 at the same time. If $x_{1,1} + x_{1,2} < 1$, on the other hand, resources are wasted when the employee works on a single task. So, no matter the values for $x_{1,1}$ and $x_{1,2}$, there will always be overwork or resources wasted—unless both tasks start and finish at exactly the same time. Note that, depending on $x_{1,1}$ and $x_{1,2}$, there could be even both resources wasted when the employee is working on a single task and overwork when working on both tasks at the same time (whenever $x_{1,1} < 1 \wedge x_{1,2} < 1 \wedge x_{1,1} + x_{1,2} > 1$). Note that we do not normalise "underwork", i. e., total dedications less than 1. This would otherwise remove the possibility of balancing cost vs. completion time. Normalisation allows for much more fine-grained schedules as employees can automatically re-scale their dedications as soon as tasks are finished or new tasks are started.

7. EA ALGORITHM

The following explains how the existing algorithm works :

1. Input is given as two sections : the admin and employee sections
2. Employee details such as skills , experience and their proficiencies are required for the scheduling purpose
3. Admin manages the project part.

Admin can be either the team leader, or the project manager itself or anybody with such powers

When a new project is rewarded the admin first enters the project into the system. Admin identifies various modules and separates the project into various modules. Each module requires specific skills to perform it successfully. The required skills are also identified and noted separately.

4. Scheduling

The main objective of this algorithm is to find out the best of the schedules which are generated by assigning free modules to appropriate employees. Few steps are involved in finding out the optimal schedules.

a. Fetch free Modules

Search if there is any module free or which is not yet associated with any employee. If any such module is found then start the search. Compare skill_mod (skills required to complete a module) with emp_skill (employee skill set). We many get many number of matches. Meanwhile a counter variable A is maintained in order to count the number of matching skills found with each employee with the existing module.

b. These matched employees are again iterated to find out best match with respect to experience , proficiency, etc. The obtained results are again given as input to the GA to obtain optimal results ie, optimized schedules.

5. Fetch Assigned Modules

Assigned Modules will be flagged A. so we can easily identify the assigned modules from the others while viewing the schedules. Following are the steps involved in this section:

a. Initiate GA pool(Input)
b. Check for free modules
   i. If no free modules found then go back to home
   ii. If found then

Get best match and save it by iterating in GA

c. Flag assigned modules
d. Exit

Normalisation

a. Get employees' assigned tasks
b. Get dedications associated with each task
c. Scale dedications for each task
d. Exit when all tasks ends

Repair

a. Get employees assigned tasks.
b. Get dedication associated with each task. Dedication is divided equally among all assigned tasks.
c. Exit when job finishes.

8. CONCLUSION

Evolutionary algorithms led to the development of Genetic algorithms[7] which use the functionalities similar to the human DNA to solve complex problems.(1+1) EA is such an algorithm which uses *mutation* and *fitness function* to produce the desired outcome. The existing system uses this algorithm to solve Project Scheduling Problem by introducing normalization. Normalization proved better the repair mechanisms here. But a drawback is that normalization is not always feasible.

Even though genetic algorithms (GAs) have been used for solving the project scheduling problem (PSP), it is not well understood which problem characteristics make it difficult/easy for GAs. This theory has inspired a new evolutionary design, including normalisation of employees' dedication for different tasks to eliminate the problem of exceeding their maximum dedication. Theoretical and empirical results show that this design is very effective in terms of hit rate and solution quality.

Though Normalisation has proved effective in many projects, in certain schedules it has not worked properly. So we have introduced a Repair mechanism collaborated with the Normalisation techniques.

REFERENCES

1. L.L. Minku, D. Sudholt, and X. Yao, “Evolutionary Algorithms for the Project Scheduling Problem: Runtime Analysis and Improved Design,” Proc. 14th Int’ l Conf. Geneticand Evolutionary Computation Conf. (GECCO 12), pp. 1221-1228, 2012.
2. M. Di Penta, M. Harman, and G. Antoniol, “The Use of Search-Based Optimization Techniques to Schedule and Staff Software Projects: An Approach and an Empirical Study,” Software: Practice and Experience, vol. 41, no. 5, pp. 495-519, 2011.
3. E. Alba and J.F. Chicano, “Software Project Management with GAs,” Information Sciences, vol. 177, pp. 2380-2401, 2007.
4. F. Luna, D. Gonz_alez- _ Alvarez, F. Chicano, and M.A. Vega Rodrıguez, “On the Scalability of Multi-Objective Metaheuristics for the Software Scheduling Problem,” Proc. 11th Int’l Conf. Intelligent System Design and Applications (ISDA ’11), pp. 1110- 1115, 2011.
5. J. Ren, M. Harman, and M. Penta, “Cooperative Co-Evolutionary Optimization of Software Project Staff Assignments and Job Scheduling,” Proc. Third Int’l Conf. Search Based Software Eng., vol. 6956, pp. 127-141, 2011.
6. D. Sudholt, “A New Method for Lower Bounds on the Running Time of Evolutionary Algorithms,” IEEE Trans. Evolutionary Computation, vol. 17, no. 3, pp. 418-435, June 2013.
7. C. Grosan and A. Abraham, “Hybrid Evolutionary Algorithms:Methodologies, Architectures, and Reviews”
8. Zbigniew Michalewicz “Evolutionary Algorithms for Real-World Applications ”, School of Computer Science University of Adelaide Adelaide, SA 5005, Australia
9. Benjamin Doerr, Sebastian Pohl, “Run-Time Analysis of the (1+1) Evolutionary Algorithm Optimizing Linear Functions Over a Finite Alphabet”, Max Planck Institute for Computer Science, Campus E1 4, 66123 Saarbrücken, Germany
10. Carl K. Chang, Chikuang Chao, Thinh T. Nguyen, Mark Christensen, “Software Project Management Net: A New Methodology on Software Management ”
11. Gareth Jones,” Genetic and Evolutionary Algorithms”, University of Sheffield, UK
12. Leandro L. Minku and Xin Yao, “Software Effort Estimation as a Multi-objective Learning Problem ” The University of Birmingham
13. Ruiz-Vanoye Jorge A. Universidad Popular Autónoma del Estado de Puebla, Mexico, Pathiyamattom-Joseph Sebastian , Centro de Investigación en Energía, Mexico. Fernández-Medina Miguel A., Fuentes-Penna Alejandro Universidad Popular Autónoma del Estado de Puebla, Mexico, Díaz-Parra Ocotlán Universidad Autónoma del Estado de Morelos, Mexico. “Project Scheduling Problem for Software Development Library - PSPSWDLIB ”
14. Stefan Droste ∗, Thomas Jansen, Ingo Wegener,” On the analysis of the (1 + 1) evolutionary algorithm ” ,FB Informatik, LS 2, Universitat Dortmund, 44221 Dortmund, Germany, February 2001
15. Pietro S. Oliveto Jun He Xin Yao ,” Time Complexity of Evolutionary Algorithms for Combinatorial Optimization: A Decade of Results ”, The Center of Excellence for Research in Computational Intelligence and Applications (CERCIA), School of Computer Science, University of Birmingham, Edgbaston, Birmingham B15 2TT, UK, July 2007
16. Leandro L. Minku, Dirk Sudholt, and Xin Yao, Fellow, IEEE, “Improved Evolutionary Algorithm Design for the Project Scheduling Problem Based on Runtime Analysis ”, IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 40, NO. 1, JANUARY 2014