Search–based techniques for optimizing software project resource allocation

G. Antoniol* and **M. Di Penta*** and **M. Harman****
antoniol@ieee.org dipenta@unisannio.it mark.harman@brunel.ac.uk

* RCOST - Research Centre on Software Technology
University of Sannio, Department of Engineering
Palazzo ex Poste, Via Traiano 82100 Benevento, Italy

**Department of Information Systems and Computing,
Brunel University, Uxbridge, Middlesex, UB8 3PH, UK

March 19, 2004

Abstract

In software development, testing and maintenance, as in other large scale engineering activities, effective project planning is essential. Failure to plan and/or poor planning can cause delays and costs that, given timing and budget constraints, are often unacceptable, leading to business–critical failures. Traditional tools such as the Project Evaluation and Review Technique (PERT), the Critical Path Method (CPM), Gantt diagrams and Earned Value Analysis help to plan and track project milestones. While these tools and techniques are important, they cannot assist with the identification of optimal scheduling assignment in the presence of configurable resource allocation.

However, most large scale software projects involve several teams of programmers and many individual project work packages. As such, the optimal allocation of teams of programmers (the primary resource cost drivers) to Work Packages (WPs) is an important problem which cannot be overlooked.

In this work we study this problem from the perspective of a massive software maintenance project. Typical examples are the Y2K remediation, Euro conversion or phone numbering change, involving a large number of applications simultaneously. Such maintenance activities present particularly acute problems for managers, since they have fixed hard deadlines and cut right across an entire software portfolio, touching almost every software asset possessed by the organisation.

When a massive maintenance request arrives, it is split in WPs according to the project Work-Breakdown Structure (WBS). An analogy estimate [5] can be used to determine the effort required to maintain each WP. Having obtained estimates for effort, the next task is to determine

the order in which WPs flow into the queuing system to be dealt with by the next available team of programmers [1].

The order of presentation of WPs is a way of describing the allocation of programmer teams to WPs. Such a resource allocation problem is an example of a bin packing problem, the solution of which is NP-hard and, for which, evolutionary algorithms are known to be effective [2, 4, 3].

We performed an empirical study, using historical data from a real–world massive Y2K maintenance intervention, aiming at addressing the applicability of evolutionary algorithms to software resource allocation problems. While optimization techniques have been widely used in other fields, they have never been used for that specific field.

Three optimization techniques, namely hill climbing, simulated annealing and genetic algorithms, have been evaluated. Each was applied to two very different encoding strategies. Each encoding represents the way in which the work packages of the overall project are to be allocated to teams of programmers.

The first encoding used is a scanning genome encoding, which combines the evolutionary algorithms with a queuing simulation model (the genome encoding represent the order in which the WPs flow to the maintenance process queuing model). The second genome is a vector-based genome, encoding the allocation of WPs to different maintenance teams.

Overall, the scanning genome encoding was found to outperform a vector-based genome encoding. In particular, when comparing different optimization techniques, we found that for the less optimal encoding (vector–based) the GA performed significantly better than the other approaches. For the optimal encoding (scanning), though GA starts better simulated annealing and hill climbing approaches soon catch up, so that the overall difference between the three approaches appears to be small, compared to the problem of establishing an effective encoding.

Work-in progress is to perform in–depth analyses as well as to model more complex maintenance process, in which the ordering is also constrained by the precedence between maintenance tasks.

Keywords: Software Project Management, Genetic Algorithms, Queuing Networks

References

[1] G. Antoniol, A. Cimitile, G. A. Di Lucca, and M. Di Penta. Assessing staffing needs for a software maintenance project through queuing simulation. *IEEE Transactions on Software Engineering*, 30(1):43–58, Jan 2004.

[2] L. Davis. Job-shop scheduling with genetic algorithms. In *International Conference on GAs*, pages 136–140. Lawrence Erlbaum, 1985.

[3] E. Falkenauer. *Genetic Algorithms and Grouping Problems*. Wiley-Inter Science, Wiley - NY, 1998.

[4] E. Hart, D. Corne, and P. Ross. The state of the art in evolutionary scheduling. *Genetic Programming and Evolvable Machines*, 2004 (to appear).

[5] M. Shepperd and C. Schofield. Estimating software project effort using analogies. *IEEE Transactions on Software Engineering*, 23(11):736–743, 1997.