View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Repositorio Institucional de la Universidad de Almería (Spain)

Ant Colony Optimization for Requirement selection in Incremental Software development

José del Sagrado and Isabel María del Águila
Department of Languages and Computation
Universidad de Almería
04120 Almería (SPAIN)
jsagrado@ual.es

Search-based software engineering allows the reformulation of the software engineering problems as search-based problems opening the possibility of applying metaheuristic algorithms [9]. Optimization techniques have been applied to a number of software engineering activities, a review of how to use them and the keys required for their application appears in [10].

Classical software engineering methodologies break down the work in several stages: requirements, design, coding and test. This approach originates some problems, especially during the initial stages; once of the major problems we face when developing large and complex software systems is that related with requirements [11]. Requirements problems have a large space of possible solutions, becaming natural candidates for application of search based tecniques [5].

As reaction against these "heavyweight" methodologies, agile software development has been evolved as new approach [3, 14]. Agile methods promote incremental development, teamwork, collaboration, and process adaptability, breaking tasks into small increments or sprints. Iterations are short time and are worked on by a team through a full software development cycle, finishing when a working product is presented to stakeholders or customer. The set of features to be included in the next iteration or increment comes from the negotiation between customers and development team. These features are selected from a prioritized set of high level requirements that cover all customer need.

The problem of which requirements should included in the next iteration as a search problem, also known as next release problem (NRP) [1], has been addressed applying some metaheuristic optimization techniques. Other approaches propose variations and enhancement of this problem [7, 15, 16, 2, 8].

It is assumed that for a software development project, there is a set of customers, $C = \{c_1, \ldots, c_m\}$ and a set of possible software requirements, $R = \{r_1,\ldots, r_n\}$. It is assumed that all customers are not equally important for a given project. In order to satisfy each requirement, some resources need to be allocated and each one has a associated cost: $Cost = \{cost_1, \ldots, cost_n\}$. The increments have a cost boundary (B) that cannot be overrun.

A decision vector $\{x_1,\ldots, x_n\} \in \{0, 1\}$ denotes a problem solution and determines the requirements that have to be included in the iteration. In this vector, x_i is 1 if requirement i is selected and 0 otherwise.

A wide range of different optimization and search techniques can and have been used in software engineering [10]. Specifically related with NRP, greedy algorithms, hill climbing, simulated annealing have been applied [1]. Suitability of weighted and Pareto optimal genetic algorithms, together with the NSGA-II algorithm have been used too [16, 15 , 8]

Other metaheuristic technique is ant colony optimization (ACO), which is inspired by shortest path search behavior of various ant species [5, 6]. These algorithms are a wellknown set of techniques for finding near optimal solutions and have been applied in search based software engineering in testing [13, 12] or model checking [4].

ACO algorithms are essentially construction algorithms: in each step, every ant constructs a solution to the problem by traveling, on a line that connects a food source to their nest. The main mean to maintain the line is a pheromone trail. Ants leave a some pheromone while traveling and each ant probabilistically select to follow a way rich in pheromone. Once an obstacle has appeared, ants cannot continue to follow the pheromone trail and they have to choose turning right or left. Those ants which choose, by chance, the shortest path will more rapidly

reconnect the pheromone trail compared to those who choose the longer path. Thus, the shorter path will receive a greater amount of pheromone and therefore a larger number of ants will choose the shorter path.

This approach can be applied to requirement negotiation problems: the more pheromones assemble in a requirement in R, the more probability the requirement will be selected.

Every ant system cycle, or episode, the pheromone of each r_i is adjust according to the level of attenuation of the pheromone in the search of the solution. After a certain of number episodes the i-th position of the decision vector obtained by ACO techniques, will contain a value 1 if the associated r_i is included in the path with the highest pheromone level and 0 otherwise.

This proposal shows that ACO systems can be applied to problems of requirements selection in software incremental development, with the idea of obtaining better results of those produced by expert judgment alone. The evaluation of the ACO systems should be done through a compared analysis with greedy and simulated annealing algorithms, performing experiments with some problems instances.

References

[1] Bagnall,A. Rayward-Smith, V, and Whittley, I.. The next release problem. Information and Software Technology, 43(14):883–890, Dec. 2001.

[2] Baker, P , Harman, M, Steinhöfel, K, and Skaliotis, A. Search Based Approaches to Component Selection and Prioritization for the Next Release Problem. In 22nd International Conference on Software Maintenance (ICSM 06), pages 176–185, Philadelphia, Pennsylvania, USA, September 24- 27 2006.

[3] Beck, K, E*xtreme Programming explained: Embrace change* Addison-Wesley, 2000.

[4] Chicano F, Alba E. Ant Colony Optimization with Partial Order Reduction for Discovering Safety Property Violations in Concurrent Models. Information Processing Letters. 2007.

[5] Doringo, M., Maniezzo, V. and Coloni, A: Ant System: optimization by colony of cooperating agents, IEEE Trnas. on Sys., Man and Cybernetics, Vol. 26, No. 1, february 1996.

[6] Doringo, M, Birattari, M and Stültzle, T, Ant Colony Optimization: artificial ants as a computational intelligence technique, IEEE Computational Intelligence Magazine, November, 2006

[7] Finkelstein A, Harman M, Mansouri SA, Ren J, Zhang Y. "Fairness Analysis" in Requirements Assignments. En: Proceedings of the 16th IEEE International Requirements Engineering Conference (RE '08). IEEE Computer Society; 2008:115-124.

[8] Greer, D and Ruhe, G. Software release planning: an evolutionary and iterative approach. Information & Software Technology, 46(4):243–253, 2004.

[9] Harman, M, Jones BF. Search-based software engineering. Information & Software Technology. 2001;43(14):833-839.

[10] Harman M. The Current State and Future of Search Based Software Engineering. In: FOSE.; 2007:342-357.

[11] Kotonya G. and Sommerville, I, *Requirements Engineering: Processes and Techniques*, Wiley, 1998.

[12] Lam, C.P., Xiao, J., and Li, H. Ant Colony Optimisation for Generation of Conformance Testing Sequences using a Characterising Set. *Proceeding* (559) Advances in Computer Science and Technology – 2007.

[13] Mahanti PK, Banerjee S. Automated Testing in Software Engineering: using Ant Colony and Self-Regulated Swarms. In: Proc. of the 17th IASTED Int. Conf. on Modelling and simulation (MS '06). ACTA Press; 2006:443-448.

[14] Schwaber K, Beedle M. *Agile Software Development with Scrum*. Prentice Hall; 2001.

[15] Zhang Y, Harman M, Mansouri SA. The Multi-Objective Next Release Problem. En: Proceedings of the 9th annual Conference on Genetic and Evolutionary Computation (GECCO '07). ACM; 2007:1129-1137.

[16] Zhang Y, Finkelstein A, Harman M. Search Based Requirements Optimisation: Existing Work & Challenges. En: Proceedings of the 14th International Working Conference, Requirements Engineering: Foundation for Software Quality (REFSQ '08).Vol 5025. Springer; 2008:88-94.