Finding Building Blocks for Software Clustering

Kiarash Mahdavi, Mark Harman, and Robert Hierons

Department of Information Systems and Computing
Brunel University
Uxbridge, Middlesex, UB8 3PH
kiarash.mahdavi@brunel.ac.uk

1 Introduction

It is generally believed that good modularization of software leads to systems which are easier to design, develop, test, maintain and evolve [1].

Software clustering using search–based techniques has been well studied using a hill climbing approach [2,4,5,6]. Hill climbing suffers from the problem of local optima, so some improvement may be expected by by considering more sophisticated search-based techniques. However, hitherto, the use of other techniques to overcome this problem such as Genetic Algorithms (GA) [3] and Simulated Annealing [7] have been disappointing.

This poster paper looks at the possibility of using results from multiple hill climbs to form a basis for subsequent search. The findings will be presented in the poster created for the poster session in GECCO 2003.

2 Multiple Hill Climbing to Identify Building Blocks

The goal of module clustering is to arrive at a graph partition in which each cluster maximizes the number of internal edges and minimizes the number of external edges [1]. To capture this in our approach, we use the "Basic MQ" fitness function [4].

The multiple hill climbing process consists of two stages. In the initial stage a set of hill climbs are carried out to produce a set of clusterings. These clusterings are then used to create building blocks for the final stage of the process.

The hill climbers use a nearest neighbor approach used in Bunch [4,6]. In this approach, the nearest neighbors of a clustering are constructed by moving a node (or module) to a new cluster or into an existing cluster. In our approach however, the nodes correspond to building blocks (sets of modules) not individual modules.

The clusterings from the initial stage of the process are compared to identify groups of nodes that are placed in the same cluster across clusterings. To reduce the disruptive effect of highly unfit solutions on the creation of the initial building blocks, cut off points based on the quality of initial hill climbs are used. These are made from the best 10% to the best 100% of the hill climbed, in increments of 10%, resulting in 10 sets of building blocks. These building blocks are then used in the second stage to improve the results of subsequent hill climbing and GAs.

E. Cantú-Paz et al. (Eds.): GECCO 2003, LNCS 2724, pp. 2513–2514, 2003.

3 Using Building Blocks to Reduce the Search Space for Repeated Hill Climbing

Simply reducing the number of nodes by grouping nodes together as building blocks reduces the search space. Furthermore, created building blocks provide a way to identify and preserve useful structures within the solution landscape and remove the destructive effects of mutation on these structures.

4 Using Building Blocks to Seed Genetic Algorithms

GAs can find clustering problems difficult [2,3,6]. This could be attributed to the difficulty they face in preserving building blocks from the destructive effects of the genetic operators. To combat this, our approach is to use the identified building blocks as atomic units by the GA. This reduces the size of the search space and potentially assists the GA in forming solutions at the meta level of combining building blocks.

References

1. Larry L. Constantine and Edward Yourdon. *Structured Design*. Prentice Hall, 1979.
2. D. Doval, S. Mancoridis, and B. S. Mitchell. Automatic clustering of software systems using a genetic algorithm. In *International Conference on Software Tools and Engineering Practice (STEP'99)*, Pittsburgh, PA, 30 August – 2 September 1999.
3. Mark Harman, Robert Hierons, and Mark Proctor. A new representation and crossover operator for search-based optimization of software modularization. In *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1351–1358, New York, 9–13 July 2002. Morgan Kaufmann Publishers.
4. Spiros Mancoridis, Brian S. Mitchell, Yih-Farn Chen, and Emden R. Gansner. Bunch: A clustering tool for the recovery and maintenance of software system structures. In *Proceedings; IEEE International Conference on Software Maintenance*, pages 50–59. IEEE Computer Society Press, 1999.
5. Spiros Mancoridis, Brian S. Mitchell, C. Rorres, Yih-Farn Chen, and Emden R. Gansner. Using automatic clustering to produce high-level system organizations of source code. In *International Workshop on Program Comprehension (IWPC'98)*, pages 45–53, Ischia, Italy, 1998. IEEE Computer Society Press, Los Alamitos, California, USA.
6. Brain S. Mitchell. *A Heuristic Search Approach to Solving the Software Clustering Problem*. PhD Thesis, Drexel University, Philadelphia, PA, January 2002.
7. Brian S. Mitchell and Spiros Mancoridis. Using heuristic search techniques to extract design abstractions from source code. In *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1375–1382, New York, 9–13 July 2002. Morgan Kaufmann Publishers.