

# Paper Four Summary

Matthew Neal, Joseph Sankar, and Alexander Sobran

September 26, 2015

## Reference

Vandecruys et al [1] listed below.

## Important Keywords

**Ant Colony Optimization (ACO)** A metaheuristic inspired by the ant behavior of determining shortest path by concentration of pheromone present on the path. Ants returning faster to the colony leave a greater concentration of pheromone by doubling over on their path which is used as an indicator of the shortest path for other ants. ACO's artificial ants iteratively and stochastically construct solutions and add pheromones to these solutions. The pheromone is defined as the number of ants recently choosing a solution. When a new artificial ant encounters a decision it is more likely to choose path with a greater concentration of pheromone. The concentration is adjusted based on the evaluation of the solution. An evaporation constant causes the pheromones to evaporate and lowers concentration for each path.

**Software mining** The use of data mining to support and improve the development of software. The paper found three areas of interest covered by recent papers on this field: Fault prediction, the use of change histories to detect incomplete changes, and effort prediction.

**COCOMO II** A well-known example of a structural model designed to optimize the allocation of resources for planning and developing software projects. The authors note that the constants in COCOMO II are determined from trial and error and the scaling factors and effort multipliers are discrete, limiting the feasibility and accuracy of the model.

**Classification** A type of data mining where a data point is assigned to a group according to some set of its characteristics. The authors used classification to assign software modules to either an "erroneous" or "correct" class depending on characteristics such as LOC (lines of code) metrics, cyclomatic complexity, and the number of unique operands.

## Feature Extraction

**Motivational statements** The motivation of this paper is to apply AntMiner+, a rule-based Ant Colony Optimization classification technique to public repositories and assess its prediction and inference capabilities against other previously successful classification techniques in the domain of software quality.

**Patterns** Different types of preprocessing were applied on the data. First, the authors had to discretize all the input variables since AntMiner+ only accepts discrete variables. Second, they had to narrow the selection of which input variables to consider to the ones with the highest predictive power. Last, they oversampled the training observations that contained erroneous software modules in order to eliminate a bias toward correct software modules.

**Sampling procedures** The authors note that in about half of the papers they summarized, datasets that are not publicly accessible were used. They recommend using public datasets as it allows other researchers to replicate the results and to better compare different data mining techniques. These are the reasons why, they state, that they have chosen the most commonly used state-of-the-art classification techniques and apply them to publicly available datasets, namely three from NASA: PC1, PC4, and KC1.

**Scripts** The pseudocode of the AntMiner+ algorithm is provided. The algorithm builds a graph representing the state space (see Fig. 3 in the paper), then simulates running ants between the Start and Stop vertices, constructing a rule along the way. Better rules will contain more pheromone along them as more ants traverse the same path. Due to some evaporation that occurs over time, there will eventually be a very high amount of pheromone on the best path and very low amounts on the other paths. Once this occurs, the rule corresponding to the path with the high amount of pheromone is added to the rule set, and the process continues until rules are discovered to account for all the data in the training set.

## Possible Improvements

## Connection to Other Papers

## References

- [1] Vandecruys, Olivier and Martens, David and Baesens, Bart and Mues, Christophe and De Backer, Manu and Haesen, Raf. Mining Software Repositories for Comprehensible Software Fault Prediction Models. *J. Syst. Softw.*, 81(5):823–839, May 2008.