Building Algorithm Portfolios for Memetic Algorithms

Mustafa Mısır
Singapore Management University
Singapore
mustafamisir@smu.edu.sg

Stephanus Daniel Handoko
Singapore Management University
Singapore
dhandoko@smu.edu.sg

Hoong Chuin Lau
Singapore Management University
Singapore
hclau@smu.edu.sg

ABSTRACT

The present study introduces an automated mechanism to build algorithm portfolios for memetic algorithms. The objective is to determine an algorithm set involving combinations of crossover, mutation and local search operators based on their past performance. The past performance is used to cluster algorithm combinations. Top performing combinations are then considered as the members of the set. The set is expected to have algorithm combinations complementing each other with respect to their strengths in a portfolio setting. In other words, each algorithm combination should be good at solving a certain type of problem instances such that this set can be used to solve different problem instances. The set is used together with an online selection strategy. An empirical analysis is performed on the Quadratic Assignment problem to show the advantages of the proposed approach.

1. INTRODUCTION

Memetic algorithms [7] were introduced as a version of genetic algorithms [3] supported by local search. A memetic algorithm considers successive application of crossover, mutation and local search operators. One of the critical design element is the selection of appropriate operators of these three types for a target problem. Although such selection decisions are usually done based on personal experience, there are methods to address this problem. Algorithm selection [8] is a research area focusing on automated selection of algorithms. The idea is to choose the best algorithm from an algorithm set to solve a given problem instance. Algorithm portfolios [4] treat the algorithm selection problem in a broader perspective. The goal is the selection of a useful algorithm subset or generating them automatically besides solely selection from a given set. The sub-set selection operation is performed in a way that the sub-set can be successful for a wide range of problem types.

This paper proposes an algorithm portfolio procedure for memetic algorithms. Each algorithm here is a combination of three operators from crossover, mutation and local search types. The portfolio approach considers determining a good algorithm sub-set involving such combinations. For delivering such a portfolio, an approach called Clustering-based Algorithm Portfolios for Memetic Algorithms (CPM) is introduced. CPM clusters algorithms with respect to a set of problem-independent and performance related features by applying k-means clustering. In order to represent different types of algorithms in the portfolio, the best performing algorithm from each large-enough cluster is used to build the portfolio. A simple online algorithm selection approach, i.e. uniform random selection, that chooses an algorithm from the portfolio and applies it to a solution at each step of a generation is utilised. The performance of CPM with online algorithm selection on the quadratic assignment problem (QAP) showed its effectiveness.

GECCO'14, July 12–16, 2014, Vancouver, BC, Canada.
ACM 978-1-4503-2881-4/14/07.
http://dx.doi.org/10.1145/2598394.2598455.

2. BUILDING A PORTFOLIO

Unlike the existing algorithm portfolio approaches delivering portfolio of single solvers, this study focuses on building a portfolio of algorithm combinations. Each combination consists of a crossover operator, a mutational heuristic and a local search method. The goal is to specify a small group of algorithm combinations that can successfully solve a large set of instances from a given problem domain. In order to have such a portfolio, it is initially required to generate a performance database revealing the *behaviour* of each combination. Behavior here is denoted as the generic and problem-independent features primarily used in the hyper-heuristic studies [6]. Hyper-heuristics [2] are defined as high-level search and optimisation strategies that can solve any problem requiring search without any domain knowledge. A class of hyper-heuristics, i.e. selection hyper-heuristics, aims at efficiently managing a given set of heuristics by selecting a heuristic(s) at each decision step. Due to the selection element in hyper-heuristics and their generic nature, the following features, including number of new best (N_{best}), improving (N_{imp}), worsening (N_{wrs}), equal quality solutions (N_{eql}) and amount of improvement ($\triangle imp$), worsening ($\triangle wrs$) are used to characterise algorithm combinations for memetic algorithms.

CPM starts by collecting performance data regarding each algorithm combination a_x. For this purpose, each instance i_y is solved by a memetic algorithm successively using a randomly selected algorithm combination a_x. It should be noted that the performance data generation process differs for the cases where offline algorithm selection is applied. In the offline case, each algorithm is separately trained since these algorithms neither interact nor share solutions. Considering that an online selection device is employed and solutions are shared, it is vital to gather the performance data by running all the algorithms while they are selected online and operating on the same solutions.

The corresponding crossover (c_x), mutation (m_x) and local search (l_x) operators of a_x are applied in a relay fashion. The performance data generation process ends after each instance is solved within a given time limit (t_{limit}). The resulting performance data is used to generate features for each algorithm, $F(a_x)$. Each feature vector is composed of the normalised versions of the following 6 features for each instance where N_{moves} refers to the number of times an algorithm combination is chosen. $f_1 = N_{best}/N_{moves}$, $f_2 = N_{imp}/N_{moves}$, $f_3 = N_{wrs}/N_{moves}$, $f_4 = N_{eql}/N_{moves}$, $f_5 = \triangle_{imp}/N_{moves}$, $f_6 = \triangle_{wrs}/N_{moves}$. As a result, each algorithm combination has $\#instances \times 6$ features. k-means clustering is applied afterwards to identify the (dis-)similarity of the algorithm combinations. The clusters that are large enough ($size(cl) \geq n \times 0.1$) are considered the major clusters that can be used to build a portfolio. The best performing algorithm combination from each selected cluster is contributed to the portfolio. An online algorithm selection procedure is then applied to manage the portfolio.

3. COMPUTATIONAL RESULTS

3.1 Quadratic assignment problem

The QAP [5] requires the assignment of n facilities to n locations. $min \sum_i^n \sum_j^n f_{\pi_i \pi_j} d_{ij}$ is the objective function for the QAP. $f_{\pi_i \pi_j}$ is the flow between the facilities π_i and π_j. π refers to a solution where each element is a facility and the locus of each facility shows its location. d_{ij} is the distance between the location i and j. The objective is to minimise the total distance weighted by the flow values.

3.2 Experimental settings

The current memetic algorithm design involves 4 crossovers, 1 mutation operator and 3 local search heuristics. The crossovers are CYCLE, DISTANCE PRESERVING, ORDER and PARTIALLY MAPPED. The local search heuristics involve BEST-2-OPT, FIRST-2-OPT and RANDOM-2-OPT which are the 2-OPT algorithm based methods. Since the mutation operator needs a mutation rate to be set, 6 different values are set as {0.0, 0.2, 0.4, 0.6, 0.8, 1.0}. In total, 72 algorithm combinations are generated. Subset of these combinations proposed by CPM are applied using a random selection within memetic algorithm. The population size fot the memetic algorithm is set to 40. 20 solutions are produced during each generation.

The 137 QAP instances from QAPLIB [1] were used to evaluate the performance of CPM. For generating features concerning algorithm combinations, t_{limit} is set to 300 seconds. The training phase is performed using 31 instances that give the opportunity to gather enough performance data for each algorithm combination within the aforementioned time limit. CPM is tested on the remaining 106 instances with the per-instance execution time limit of 100 seconds for 10 trials using Java on an Intel Core I5 1.7 GHz PC.

3.3 Results

3 algorithm combinations used in the portfolio suggested by CPM are listed as follows.

- $a_1 = c : CYCLE$ crossover + m : mutation rate of 0.2 + $l : FIRST\_2\_OPT$
- $a_2 = c : ORDER$ crossover + m : mutation rate of 0.2 + $l : BEST\_2\_OPT$
- $a_3 = c : CYCLE$ crossover + m : mutation rate of 0.4 + $l : RANDOM\_2\_OPT$

The first (a_1) and third (a_3) algorithm combinations utilise $CYCLE$ crossover while the second combinations (a_2) involve $ORDER$ crossover. For the mutation operation, the mutation rates changes between 0.2 and 0.4. Each combination consists a different local search method.

The empirical results indicate that using the 3 algorithms suggested by CPM provides better performance than using all the 72 combinations. In particular, the % average difference of 10 trials to the best known solution is 0.73 for the full case while it is 0.62 for CPM. % differences for the best trial results are 0.38 and 0.36 respectively.

4. ACKNOWLEDGMENT

This research is supported by Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office, Media Development Authority (MDA).

5. REFERENCES

[1] R. E. Burkard, S. E. Karisch, and F. Rendl. Qaplib–a quadratic assignment problem library. *Journal of Global Optimization*, 10(4):391–403, 1997.

[2] E. Burke, M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Ozcan, and R. Qu. Hyper-heuristics: A survey of the state of the art. *Journal of the Operational Research Society*, 64:1695–1724, 2013.

[3] D. E. Goldberg and J. H. Holland. Genetic algorithms and machine learning. *Machine Learning*, 3(2):95–99, 1988.

[4] C. Gomes and B. Selman. Algorithm portfolio design: Theory vs. practice. In *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence (UAI'97)*, pages 190–197, Providence/Rhode Island, USA, August 1–3 1997.

[5] E. Lawler. The quadratic assignment problem. *Management Science*, 9(4):586–599, 1963.

[6] M. Mısır. *Intelligent Hyper-heuristics: A Tool for Solving Generic Optimisation Problems*. PhD thesis, Department of Computer Science, KU Leuven, 2012.

[7] P. Moscato, C. Cotta, and A. Mendes. Memetic algorithms. In *New optimization techniques in engineering*, pages 53–85. Springer, 2004.

[8] J. Rice. The algorithm selection problem. *Advances in computers*, 15:65–118, 1976.