Hot-Pluggable Hyper-Heuristic Framework and Time Complexity Evaluation

Background:

To solve the NP-hard problem that exists in every corner in engineering, the approaches to get the result that is considered close enough to the global optima in polynomial time, heuristics, are getting more and more attention. However, there are still some difficulties to apply them to a newly encountered problem:

1. Select which particular search heuristic should be used.
2. What parameter setting should be applied.

As a result, hyper-heuristics found a place to overcome those difficulties. Seen as a high-level methodology, the hyper-heuristics automatically provides an adequate combination of the provided components to solve the given problem field efficiently.

Research Gaps:

Although many researches on the hyper-heuristics are conducted within recent years, there are still some gaps that could be investigated:

1. Although many hyper-heuristics selected or generated from simple low-level heuristics, generalized heuristic is seldom discussed and analyzed rigorously.
2. Population-based heuristic rule (breed multiple instances that accept different strategy and reproduce the next generation based on the result of the previous generation) is necessary to reproduce practical algorithm by hyper-heuristics like evolutionary algorithm, which is to be integrated to the current framework.
3. The commonly-used hyper-heuristic framework could be accelerated by distributed computing with the population-based heuristic rule or some particular kind of benchmark function.

Proposal:

The overall goal of this project is to develop a hyper-heuristics framework with distributed computing and evaluate the performance of each method. To be specific:

a) Integrate generalized heuristics and population-based heuristics into the hyper-heuristic framework.

b) Add distributed computing to the framework to optimize the framework especially where the population-based heuristic method is adopted.

c) Analyze the time complexity of the generated heuristics rigorously.

Methodology:

Drift Analysis

First proposed by Bruece Hajek (1982) [1], drift analysis serves as a convenient tool to estimate the performance of stochastic search algorithms by dividing the process and analyzing the expected

[1] Bruce Hajek. Hitting-time and occupation-time bounds implied by drift analysis with applications.Advances in Applied Probability, 13(3):502–525, 1982.

decrease in distance from the optimum in each step[2]. Therefore, we could use drift analysis to conduct a theoretical analysis of the time complexity of the algorithm generated by different combinations of parameters of the hyper-heuristics framework.

Hot-Pluggable Hyper-heuristic Framework with Visualization

We will present a composite hyper-heuristic framework that consists of two hot-pluggable modules:

1. Acceptance operators from simple AllMove, OnlyImprovement to generalized methods[3] and population-based method.
2. Benchmark function with the mutation operator such as LeadingOnes[4] with FlipOne and FlipTwo operators.

We will also build a visualization website that interacts with users to generate the heuristic algorithm with theoretical analysis and visualization of the optimization process to gain a deeper understanding of the effect of different configuration on benchmark functions.

Distributed Computing

Although the new population-based heuristic rule that would run multiple instances simultaneously at one step will also come with intensive computation challenges, it could also be divided into several smaller splits which then will be assigned to different nodes of a computing cluster. The previous researches[5][6][7] have already demonstrated the feasibility and potential improvement of the distributed genetic algorithm that is connected by LAN. In addition, current distributed computing frameworks such as Apache Hadoop and Spark[8], provide us with the possibility to scale up to thousands of nodes with local memory computation and storage. Since the genetic algorithm could also be considered as a population-based heuristic, we could accelerate our generated heuristic algorithm with population-based heuristic rule based on the distributed computing framework.

[2] Lengler, J., 2017. Drift analysis. *arXiv preprint arXiv:1712.00964*.
[3] Lehre, P.K. and Özcan, E., 2013, January. A runtime analysis of simple hyper-heuristics: to mix or not to mix operators. In *Proceedings of the twelfth workshop on Foundations of genetic algorithms XII* (pp. 97-104). ACM.
[4] Alanazi, F. and Lehre, P.K., 2014, July. Runtime analysis of selection hyper-heuristics with classical learning mechanisms. In *2014 IEEE congress on evolutionary computation (CEC)* (pp. 2515-2523). IEEE
[5] Tanese, R., 1989. Distributed genetic algorithms for function optimization.
[6] Belding, T.C., 1995. The distributed genetic algorithm revisited. *arXiv preprint adap-org/9504007*.
[7] Adeli, H. and Kumar, S., 1995. Distributed genetic algorithm for structural optimization. *Journal of Aerospace Engineering*, *8*(3), pp.156-163.
[8] Liu, P., Ye, S., Wang, C. and Zhu, Z., 2019. Spark-Based Parallel Genetic Algorithm for Simulating a Solution of Optimal Deployment of an Underwater Sensor Network. *Sensors*, *19*(12), p.2717.