Hyper-Criticism: A critical reflection on todays Hyper-heuristics.

Willem Van Onsem
KU Leuven, Departement of Computer Science
Willem.VanOnsem@cs.kuleuven.be

Bart Demoen
KU Leuven, Departement of Computer Science
Bart.Demoen@cs.kuleuven.be

Patrick De Causmaecker
KU Leuven, Departement of Computer Science
Patrick.DeCausmaecker@cs.kuleuven.be

Abstract

The results of a comparative study among 16 implementations of the *Cross-domain Heuristic Search Challenge (CHeSC'11)* are presented. The studied hyper-heuristics were reimplemented and diagnostic results were generated. Issues with several implementations are addressed and explanations are given why some hyper-heuristics perform better than others.

Introduction A *hyper-heuristic* is defined as a heuristic for selecting heuristics where the latter try to solve a given optimization problem. The selection is done without any knowledge of the problem. *HyFlex* is a framework that aims to ease the implementation and benchmarking of hyper-heuristics. In 2011, a competition called *CHeSC* was organized and 20 hyper-heuristics were submitted, 16 of which were studied for this survey.

Choice based on the progress of the fitness value In order to decide which heuristic will be executed next, hyper-heuristics aim to learn from the sequence of applied heuristics and the corresponding difference in fitness-value of the obtained solutions. Most hyper-heuristics that use the absolute difference between two fitness-values ended in the lower parts of the competition results.

This can be explained since the behavior of such hyper-heuristics depends on the unit of the objective function. It can even be shown that using the relative difference of the fitness-value does not always resolves this issue.

Furthermore, heuristics that are executed in the beginning of the process are favored because making progress in an early stage is more likely.

A (partial) solution to this issue is implemented in *AdapHH* and *VNS-TW* where only the number of improvements are considered, and not the difference. We propose another solution where the distribution the fitness-function is fuzzily constrained with respect to the search space.

The semantics of heuristics *HyFlex* groups heuristics into *diversification operators* and *local search*. Diversification operators are not designed to come up with a better solution per se, but can get the system out of a local optimum. Well performing hyper-heuristics encode this knowledge. Other systems aim to learn the difference using for instance machine learning techniques.

For some implementations this is even impossible since it is not included in the inductive bias of the learning system. Such systems punish a heuristic that comes up with a worse solution resulting in the fact that local search is applied too often without yielding any better solutions while it becomes less likely a diversification operator is applied.

Even if a system has a notion of diversification, it is hard to classify if the heuristic performed well. After a diversification is applied a sequence of local search heuristics should aim to optimize the solution before the original and new solution can be compared. This is the idea behind *Iterative Local Search*, a methodology that performs well in general.

Eliminating non-improvements The lower 50% of the *CHeSC'11* competition often lacks a component that eliminates time consuming heuristics. For instance if a local search heuristic yields the same solution as the original one, applying the heuristic a second time won't yield another solution.

The issue becomes even more severe when sequences of heuristics are considered: some of the sequences may contain redundant parts that are not eliminated.

Tabu Search versus probabilistic learning We conclude – based on the implementations of the *CHeSC'11* competition – that hyper-heuristics that implement the tabu search methodology for heuristic selection perform in general superior to probabilistic learning systems. This is mainly due to the forgive-and-forget policy of the algorithm: bad performance at one phase in the process does not result in a complete exclusion of these heuristic in later parts of the process.

We designed a mathematical formalism that shows that in some cases, even if one heuristic consistently performs better than another one, a difference in the initial probability often results in the fact that the latter will substantially be more applied than the former. A solution for probabilistic learning systems might be to consider a time-window where only the last k results determine the probabilities.

Conclusion Although a large number of submitted hyper-heuristics are reported to work well on specific optimization problems, hyper-heuristics needs a higher level of abstraction. This extended abstract proposes some directions for the development of new hyper-heuristics.

Three important findings are: (1) the fitness value of the solutions should be treated with care; (2) some semantics regarding the underlying heuristics can be encoded or should at least be representable by a learning algorithm; and (3) algorithms should be able to forget what they have learned after a certain amount of time.

Proving the reported issues is hard, however by mathematical formalisms and altering hyper-heuristic behavior, at least some decisions can be argued.