NAHH: A Non-Adaptive Hyper-Heuristic for CHeSC 2011

Franco Mascia and Thomas Stützle

fmascia,stuetzle@ulb.ac.be
IRIDIA - Université Libre de Bruxelles

Abstract. In this extended abstract we describe the design of NAHH, our submission for the Cross-domain Heuristic Search Challenge (CHeSC) [3].

1 Algorithmic Schemata

We implemented in the HyFlex framework [2] a number of algorithmic schemata that use all low-level heuristics (except crossovers) as basic operators. Among the implemented schemata there are several algorithms well established in literature such as: Randomised Iterative Improvement (RII), Probabilistic Iterative Improvement (PII), Iterated Local Search (ILS), Variable Neighbourhood Descent (VND), Simulated Annealing (SA), and Iterated Greedy (IG). Of these algorithm we implemented a few variations both in the algorithm design and in the policies for selecting the low-level heuristics. For example we implemented the Iterated Local Search described in [1], an ILS that uses a VND as subsidiary local search, a VND that uses *Ruin and Recreate* instead of *Local Search* low-level heuristics, an IG with a probabilistic acceptance criterion, etc.

All the aforementioned algorithmic schemata are parametric. Moreover we implemented a further algorithm we called Tuneable Hyper Heuristic (THH), which is a juxtaposition of blocks of low-level heuristics. THH has been designed with the aim to see how good could perform an algorithm with less rationale in the design and a larger parameter space for automatic tuning.

2 Off-line Tuning of the Algorithmic Schemata

The off-line tuning is divided in two phases. In the first, we perform a parameter selection for each algorithmic schema on each problem domain. In the second, for each problem domain, we select the best tuned schemata. Both tunings have been done with Irace [4].

Table 1 shows the best performing algorithms for the single domains. The THHs selected for Max-SAT, Bin Packing and Personnel Scheduling differ by the parameter setting chosen. In order to have a more robust hyper-heuristic for the competition, we also tune and select the best performing algorithm across all problem domains.

Listing 1.1. Tuneable Hyper-Heuristic pseudo-code.

```
procedure THH(n_rr, p_a_rr, n_ls, p_a_ls, p_a, p_m, p_testart)
    s ← random initial solution
    while time has not expired:
        s' ← s
        for n_rr times:
            apply randomly selected heuristic of type Ruin and Recreate
            accept non−improving solution with probability p_a_rr
        for n_ls times:
            apply randomly selected heuristic of type Local Search
            accept non−improving solution with probability p_a_ls
        if f(s) > f(s') or with probability p_a:
            s ← s'
        with probability p_m:
            apply randomly selected heuristic of type Mutation
        with probability p_restart:
            s ← random initial solution
```

3 NAHH: A Non-Adaptive Hyper-Heuristic

NAHH is composed of the following three phases.

Phase 1: Analysis of the operators.
This phase is devoted to the analysis of the low-level heuristics available for the problem being optimised. The low-level heuristics are run repeatedly for a fraction of the total allocated runtime and summary statistics on their runtime and solution qualities are collected. Dominated low-level heuristics are then discarded and only the non-dominated ones will be available for the remaining runtime. At least two low-level heuristics are chosen for each type.

Phase 2: Algorithm selection.
In this phase, a fraction of the remaining time is allocated for selecting the best performing scheme for the problem at hand. The algorithms participating in the selection are the four best performing schemata for each domain and the best tuned algorithm across all domains. NAHH runs interleaved the five tuned algorithms in Table 1 discarding the worst performing ones, until only one algorithm remains.

Phase 3: Run.
The best-performing algorithm is executed for the remaining allocated time.

References

1. Edmund Burke, Tim Curtois, Matthew Hyde, Graham Kendall, Gabriela Ochoa, Sanja Petrovic, Jose A. Vazquez-Rodriguez, and Michel Gendreau. Iterated local

Problem domain	Algorithmic Schema
Max-SAT	THH, parameter setting Max-Sat
Bin Packing	THH, parameter setting Bin Packing
Personnel Scheduling	THH, parameter setting Personnel Scheduling
Permutation Flow Shop	IteratedGreedy with probabilistic acceptance criterion
All domains	Iterated Local Search

Table 1. Selected algorithmic schemata for the problem domains.

search vs. hyper-heuristics: Towards general-purpose search algorithms. In *IEEE Congress on Evolutionary Computation*, pages 1–8. IEEE, July 2010.

2. E.K. Burke, T. Curtois, M. Hyde, G. Kendall, G. Ochoa, S. Petrovic, and JA Vazquez-Rodriguez. Hyflex: A flexible framework for the design and analysis of hyper-heuristics. In *Multidisciplinary International Scheduling Conference (MISTA 2009), Dublin, Ireland, Dublin, Ireland*, page 790, 2009.
3. E.K. Burke, M. Gendreau, M. Hyde, G. Kendall, B. McCollum, G. Ochoa, A. Parkes, and S. Petrovic. The cross-domain heuristic search challenge–an international research competition. In *Proceedings of Learning and Intelligent Optimization Fifth International Conference, LION 5, Rome, Italy, January 17-21, 2011. To appear.*
4. Manuel López-Ibáñez, Jérémie Dubois-Lacoste, Thomas Stützle, and Mauro Birattari. The irace package, iterated race for automatic algorithm configuration. Technical Report TR/IRIDIA/2011-004, IRIDIA, Université Libre de Bruxelles, Belgium, 2011.