A Single Objective Variant of the Online Selective Markov chain Hyper-heuristic (MCHH-S)

Kent McClymont and Ed Keedwell

College of Engineering, Mathematics and Physical Sciences, University of Exeter, Exeter, EX4 4QF

{km314, e.c.keedwell}@exeter.ac.uk

1 Introduction

The Markov chain Hyper-heuristic algorithm (MCHH) [1] is an online selective hyper-heuristic designed to select sequences of perturbing heuristics for multi-objective continuous and mixed encoding problems. The method is designed to operate on problems where it is possible to achieve improvements in objective values by relatively small perturbations in decision space and so allow for fine tuning of solutions. More specifically, the MCHH was designed to operate on the Water Distribution Network (WDN) design problem [2] which exhibits this feature.

In this document we outline a modified single objective variant of the multi-objective MCHH approach, referred to as the single objective Markov chain hyper-heuristic (MCHH-S). The modifications were made to reflect both the change in objective space dimensionality and enable a more effective inclusion of local search heuristics in order to apply it to the single objective CHeSC [3] competition problems. In contrast to many continuous and mixed encoding test problems, the complex single objective operations research problems, such as the Flow Shop problems, used in the CHeSC competition required a more structured application of perturbing, constructive (or ruin and recreate) and local search heuristics.

2 Method

Similarly to the choice function method presented in [4][5], the Markov chain Hyper-heuristic (MCHH) attempts to learn good transitions between heuristics and model the best sequence of heuristics to apply in order to improve the search process. To achieve this, the MCHH constructs a fully connected finite Markov chain with one state for each heuristic, i.e., each state in the chain is connected to every other state and to itself (see Fig. 1). The weight of each edge out of a state represents the probability of moving from the current state (heuristic) to the destination state (heuristic), where all edges out of each state sum to one.

The MCHH traverses this Markov chain by stochastically selecting the next heuristic, biased by the outbound edge weights. Each weight is limited to a minimum of 0.01 to ensure each heuristic has a chance of being applied. The heuristic is applied

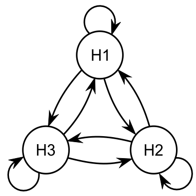

Figure 1. Example Markov chain with 3 states representing three heuristics.

to one solution from the population before selecting the next heuristic. The next heuristic is then applied to the next solution in the population before again selecting another heuristic, and so on. After each solution is evaluated, the quality score given in Equation 1 is calculated and the weight of the last edge traversed by the MCHH, the edge used to move to the current heuristic, is updated.

$$p = (\mu - \lambda) \times \left(1 - \frac{\Delta t}{T}\right) \quad (1)$$

The performance measure, given in Equation 1, returns a score based on the improvement in objective ($\mu - \lambda$) value scaled by the time taken (Δt) to execute proportional to the current maximum time taken to execute (T). The maximum time is updated after each heuristic is applied if the heuristic exceeds the current maximum time.

Once the weights have been updated, the MCHH selects the next heuristic and moves to it, using roulette wheel selection. The algorithm is outlined below in Fig 2.

---

1. Initialise parent population (μ)
2. Select random heuristic
3. **Repeat While** hasTimeExpired() **:**
   3.1. Vary a parent (μ) using current heuristic to generate a child (λ)
   3.2. Evaluate child (λ)
   3.3. Calculate performance (p) of current heuristic
   3.4. Increase the weight from last heuristic to current heuristic by p
   3.5. **If** $p > 0$ **Or** *rand*() $< \alpha / 5$ **Then:**
      3.5.1. Update parent so that $\mu = \lambda$
      3.5.2. **Set** $\alpha = 0$
   3.6. **Else:**
      3.6.1. **Set** $\alpha = \alpha + 1$
   3.7. **If** updated parent **Then:**
      3.7.1. Select next heuristic
      3.7.2. Select next parent
   3.8. **Else:**
      3.8.1. Select next heuristic (limited to local search)

Figure 2. Pseudocode for the Single objective Markov chain Hyper-heuristic (MCHH-S)

The original MCHH was implemented to operate independently of the underlying meta-heuristic or selection strategy in order to make the method transferable between existing meta-heuristics. For the CHeSC contest a basic elitist strategy was implemented with increasing chance of selecting worse solutions based on the number of operations since the last solution was accepted. The featured enabled a more explorative search of the landscape. If the parent was updated then the MCHH-S proceeds on to the next solution in the population, looping from the last solution to the first. If, however, the parent solution is not updated then the selection of the next solution is limited to local search heuristics and the parent is retained.

3 Leader board Results

Table 1 details the final leader board results, collated on 7th June 2011, prior to the final competition. The MCHH-S achieved 12th place out of 17 over the 4 test categories. However, in the four category breakdowns, the MCHH-S was placed 6th on the Max-SAT problem. The better performance on the Max-SAT problems is assumed to be related to the higher performance of the perturbing heuristics and a more stable performance of the heuristics over the search. Further analysis is required to better understand the interaction between the heuristics and the problems and how this affects the MCHH-S.

Table 1: Preliminary Leaderboard

	Algorithm	**Score**
1	PHunter4	204.28
2	ISEA2	177.28
3	HAHA1	166.78
4	TW4	130.2
5	basic-test	125.67
6	ADHS1	120.08
7	SALS	100.33
8	ML	88.67
9	HAEA	78.07
10	SEA2	69
11	AVEG_Nep	63.17
12	MCHH-S	58.5
13	XCJ	46.28
14	FPA1	43.15
15	SelfSearch	36
16	GISS0	34.53
17	Ant-Q	18

Clearly in this initial leaderboard the MCHH-S has demonstrated some specialism in problems with smoother landscapes and more stable performing low level heuristics. However, over the range of problems, and particularly the Bin Packing and Flow Shop problems, the MCHH-S performed less well. Future research should focus on indentifying and exploiting the features of the slower executing heuristics which are essential for solving these harder problems.

4 References

[1] McClymont, K.; Keedwell, E. C.; (2011) Markov chain Hyper-heuristic (MCHH): an Online Selective Hyper-heuristic for Multi-objective Continuous Problems, Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2011), Ireland (to appear).

[2] McClymont, K.; Keedwell, E. C.; Savic, D.; Randall-Smith, M.; (2011) A Hyper-heuristic Approach to Water Distribution Network Design, in Proceedings of Computing and Control for the Water Industry 2011 (CCWI 2011), Exeter (to appear).

[3] E. K. Burke, M. Gendreau, M. Hyde, G. Kendall, B. McCollum, G. Ochoa, A. J. Parkes, S. Petrovic (2011) The Cross-domain Heuristic Search Challenge- An International Research Competition. Proc. Fifth International Conference on Learning and Intelligent Optimization (LION5), Lecture Notes in Computer Science, Vol 6683, Springer.

[4] Cowling, P., Kendall, G., Soubeiga, E. A Hyperheuristic Approach to Scheduling a Sales Summit. In Practice and Theory of Automated Timetabling III : Third International Conference, PATAT 2000. Lecture Notes in Computer Science. Springer, 2079 (2000), 176-190.

[5] Kendall, G., Soubeiga, E., Cowling, P., Choice function and random hyperheuristics, in Proceedings of the fourth Asia-Pacific Conference on Simulated Evolution And Learning, SEAL (2002), pp. 667-671.