Markov Chain Hyper-heuristic (MCHH): an Online Selective Hyper-heuristic for Multi-objective Continuous Problems

Kent McClymont
University of Exeter
Harrison Building,
Exeter EX4 4QJ
+44 (0) 1392 722524
km314@exeter.ac.uk

Ed C. Keedwell
University of Exeter
Harrison Building,
Exeter EX4 4QJ
+44 (0) 1392 722524
e.c.keedwell@exeter.ac.uk

ABSTRACT
In this paper we present the Markov chain Hyper-heuristic (MCHH), a novel online selective hyper-heuristic which employs reinforcement learning and Markov chains to provide an adaptive heuristic selection method. Experiments are conducted to demonstrate the efficacy of the method and comparisons are made with standard heuristics, a random hyper-heuristic and a multi-objective hyper-heuristic from the literature. The approaches are compared on a small number of evaluations of the multi-objective DTLZ test problems to reflect the computational limitations of expensive optimisation problems. The results demonstrate the MCHH robust and reliable performance on these problems.

Categories and Subject Descriptors
I.2.8 [Problem Solving, Control Methods, and Search (F.2.2)]

General Terms
Algorithms, Theory.

Keywords
Multi-objective, Continuous, Optimisation, Hyper-heuristics.

1. INTRODUCTION
Real-world multi-objective optimisation problems are often computationally hard to solve, requiring sophisticated heuristic methods to generate acceptable approximations of the Pareto front. Furthermore, the computation models used to simulate these problems and evaluate the quality of solutions are commonly expensive to execute, limiting the number of evaluations one can feasibly make when searching for optimal solutions. This class of problem, with a restricted number of evaluations, presents a dilemma for optimisation experts – is it better to provide one solution close to some unknown front or provide a set of solutions that are likely to be further away from the optimum but provide a variety of options for the decision makers? Clearly, with few evaluations, it is highly unlikely that the true Pareto front will be located and so this trade-off becomes an important factor when deciding which type of heuristic to apply.

GECCO'11, July 12–16, 2011, Dublin, Ireland.

A wide range of meta-heuristics have been presented in the literature designed specifically for expensive problems. Generally these techniques use surrogate models to approximate the problem function and reduce the number of actual evaluations [1] [2]. Another classical approach is to employ Kriging-based methods to control the sequential generation of candidate solutions, such as Efficient Global Optimization [3] and Informational Approach to Global Optimization [4][5]. However, many of these approaches are better at solving specific types of problem and can be costly to tune to each new problem instance. Evolutionary Algorithms (EAs), for example, operate through a known set of specific heuristics that provide a balance of different operations. These heuristics, such as mutation and crossover, are vital to the overall performance of the meta-heuristic EAs but require parameter tuning for each problem to achieve the best results.

Hyper-heuristics are designed to mitigate this problem and are aimed at providing methods to automate the process of selecting from the set of heuristics and operators to better improve the search [6]. By the careful selection of heuristics during the search process a wider range of heuristics can be supplied to the hyper-heuristic which removes much of the burden associated with parameter tuning and heuristic selection.

A hyper-heuristic algorithm searches the space and combinations of heuristics independently of the problem [7], in theory creating new sequences of heuristics for each problem they are applied to. This higher level of abstraction allows hyper-heuristics to be applied to each new problem without any adjustment to the algorithm. Instead, the specialisation is introduced by supplying a set of heuristics that can be applied to the new problem.

However, whilst hyper-heuristics have been shown to be very effective at solving a range of single-objective combinatorial optimisation problems [8], few examples of multi-objective approaches exist in the literature. Indeed, much of the hyper-heuristic literature is focused on demonstrating the efficacy of new methods on single- or bi-objective combinatorial problems.

In this paper we present a novel selective hyper-heuristic for online use in hybrid meta-heuristics, such as Evolution Strategies (ESs). Heuristic selection is informed by the use of a Markov chain [9] that models the probability of moving between heuristics. The approach uses online reinforcement learning to adapt heuristic selection in response to each heuristic's performance on the current problem by updating the transition weights between heuristics. The combination of weight adaption through reinforcement learning and the Markov chain representation is used by the hyper-heuristic to learn the best sequence of heuristics to apply as the search progresses.

The presented method is designed in an encapsulated, modular approach which can easily be integrated into any iterative optimiser and is therefore well suited to many Evolutionary Algorithm architectures. Experiments are conducted to demonstrate the efficacy of the method which is shown to compare well with standard heuristics, a random hyper-heuristic and a multi-objective hyper-heuristic from the literature. The approaches are compared on a small number of evaluations of the multi-objective DTLZ test problems [10] to simulate the restricted number of evaluations available in expensive real-world optimisation problems.

2. METHOD

In this section we outline the method used to evaluate the performance of heuristics in a multi-objective context and present a novel online selective hyper-heuristic applicable to problems with any number of objectives.

2.1 Heuristic Performance

Although a large number of studies have been conducted on hyper-heuristics, the majority of these are focused on single-objective problems. As such, no one framework exists that aptly evaluates heuristic performance on multi-objective problems. Examples from the literature include objective specific learning methods, such as that used in the TSRoulWheel selective hyper-heuristic [11]. In TSRoulWheel, each heuristic is rated on its performance on each objective and selection weights are updated using reinforcement learning. I.e., for a problem with m objectives and h heuristics, a weight matrix of $m \times h$ is produced. The algorithm first selects an objective to use as a basis from which it then selects heuristics, based on each heuristic's weight for that specific objective. Whilst this method is shown to be the most effective examined in [11], and to be a reasonably good method in this study, it relies on improvements in one objective resulting in improvements in another as it evaluates each objective in turn, formulating the problem as a set of single-objective sub-problems.

2.1.1 Pareto Dominance

The use of Pareto optimality has been a cornerstone technique for the majority of multi-objective EAs since it was first suggested by Goldberg in 1989 [12]. In this study we use the Pareto dominance relationship to assign a quality performance measure to each heuristic. After a number of generations it is possible to calculate an estimate of the probability of each heuristic producing dominating solutions when applied in future iterations. This is estimated by calculating, for each solution in the new child population, the ratio of solutions it dominates in the parent population and then averages these ratios to produce a single score. This is shown in equation 1. Theoretically, good heuristics (for moving towards the Pareto front) will have a high probability of generating dominating solutions.

$$p(h,\mu,\lambda) = \sum \forall_{a\in\lambda} \forall_{b\in\mu} dom(a,b)$$
$$where\ dom(a,b) = \begin{cases} 1, & a \prec b \\ 0, & a \nprec b \end{cases} \quad (1)$$

The function $p(h,\mu,\lambda)$ shown in Equation 1 returns the average ratio of parent solutions μ dominated by each child solution in λ produced by heuristic h. The terms a and b refer to an individual child and parent respectively whilst the function $dom(a, b)$ returns an integer (0 or 1) value indicating whether a dominates b. The approach assumes that a heuristic operates on a parent population of undefined size and produces a child population of undefined size, both of which could be a population of one.

2.2 Markov chain Hyper-heuristic (MCHH)

This section outlines a novel selective hyper-heuristic for single- and multi-objective optimisation that uses the performance measure defined in section 2.1. The hyper-heuristic uses a Markov chain to guide the selection of heuristics and applies online reinforcement learning to adapt the transition weights in the Markov chain. By using a Markov chain to control the selection of heuristics and adapting the transition weights from one heuristic to another the MCHH is able to not only learn which heuristics are effective, but what sequence of heuristics are most effective. For example, a heuristic may be good in general but a combination of two other heuristics, when applied in a specific sequence, may perform even better. The approach is designed to try and learn these transition sequences to further improve the optimisation process.

[13] presents a similar single-objective approach, which relies on learning effective transition rules for each state. Although this is demonstrated as performing well, the set of rules are human designed and rely on knowledge about the behaviour of each heuristic. Additional constraints are also applied to the heuristic set, forcing heuristics to be allocated to one of two categories: spreading and converging. Whilst for single objective problems this might appear rational, for multi-objective problems there is no clear boundary between the two types of heuristic. It is theoretically plausible that the behaviour of a heuristic will change as the optimiser moves through the search space, exhibiting spreading behaviour in one area and converging behaviour in another, as is demonstrated later by the Transposition heuristic on DTLZ2. The method proposed in this section does not require this prior information and learns which heuristics are effective at converging during the search.

2.2.1 Markov chain

The proposed Markov chain Hyper-heuristic (MCHH) constructs a fully connected 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 Figure 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.

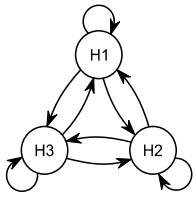

Figure 1. Example Markov chain with 3 states representing three heuristics.

The MCHH traverses this Markov chain by stochastically selecting the next heuristic, biased by the outbound edge weights. Each heuristic is applied ϵ times (set to 5 in the experiments below) before selecting the next heuristic. The next heuristic is then applied ϵ times before again selecting another heuristic, and so on. At the end of each episode (ϵ applications of a heuristic) the quality score 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. Once the weights have been updated, the MCHH selects the next heuristic and moves to it. This process is given as psuedocode in Figure 3, showing how each generation of the (μ+λ)-ES represents a single application of the heuristic. A description of the online weight learning is given in section 2.2.2.

The transition weights in Markov chain can also be represented as a matrix of $m \times m$, shown in Figure 2, where m is the number of heuristics. In this figure, the row represents the current state and the columns the potential state to move to. The sum of weights for edges leaving a state (a row) is normalised to 1. I.e., if the row label represents the current heuristic and the columns the probability of transitioning to each heuristic then the values in the row will sum to 1 (once normalised). The sum of each column represents the "influence" of each heuristic where higher values indicate an overall higher likelihood of applying that heuristic. This summed weight vector is similar to a single vector credit assignment formulation of heuristic selection. Whilst the credit assignment approach is good at learning which operators are effective in general it cannot discover and utilise the most beneficial transitions between heuristics.

		Transition State		
		H1	**H2**	**H3**
Current State	**H1**	1/3	1/2	1/6
	H2	1/6	1/3	1/2
	H3	1/3	1/3	1/3
Overall Weight:		*5/18*	*7/18*	*6/18*

Figure 2. Transition Weight Matrix for Example Markov chain with 3 states representing three heuristics.

2.2.2 Online Reinforcement Learning

In its simplest form, each edge in the Markov chain could represent the count of dominating solutions generated by the target heuristic following the preceding heuristic. In essence, this sets the weight of applying one heuristic and then the other to the performance of the second target heuristic after applying the first heuristic. In this study we applied the measure in Equation 1 to calculate the performance of each heuristic, replacing the count of dominating solutions with reinforcement weights based on the probability of generating dominating solutions.

After applying the heuristics for one episode of ϵ applications, the probability of dominance is calculated using Equation 1. If the resulting score is greater than some threshold γ, the weight corresponding to the last transition (made to get to the current operator) is increased by α. Otherwise, the weight is degraded by β. Once the weight has been adjusted, the sum of outflow edges from the previous state are normalised to 1 to maintain the fidelity of the matrix.

After normalising, the effect of increasing or decreasing an edge in the Markov chain will decrease or increase the other edges respectively. The repetition of this process should allow the matrix to converge on a set of probabilities for transitioning between individuals in the set of heuristics. This process identifies the good links between heuristics, with sequencing controlled by the edge direction, giving probabilistic information about combinations of heuristics.

2.2.3 Markov chain Hyper-heuristic (MCHH)

The MCHH is an online selective hyper-heuristic and is designed to operate only on the selection of heuristics in an encapsulated way. As such, the MCHH can be incorporated in any meta-heuristic and used as a hybridised hyper-heuristic. In this study we modified a (1+1) Evolution Strategy (ES) based on the UMMEA [14] to allow for fair comparisons with single heuristics.

Figure 3 outlines the MCHH incorporated in a (μ+λ) Evolution Strategy. The steps in 2.1 relate to the selective hyper-heuristic operations, operating on abstracted information about the heuristics. This step is introduced into the (μ+λ)-ES meta-heuristic as an online selection strategy. All the weights for the Markov chain transition matrix are initialised to 1 and adapted during the search process.

1. Initialise parent population (μ)
2. Repeat:
   - 2.1. If ϵ generations since last episode then
     - 2.1.1. Calculate performance (p) of current heuristic
     - 2.1.2. If $p > \gamma$ then increase the weight from last heuristic to current heuristic by α
     - 2.1.3. Else if $p < \gamma$ then decrease the weight from last heuristic to current heuristic by β
     - 2.1.4. Select next heuristic
   - 2.2. Vary parents (μ) using current heuristic to generate children (λ)
   - 2.3. Evaluate children (λ)
   - 2.4. Select parents for the next generation (μ') from union of parents (μ) and children (λ)
   - 2.5. Update the archive (ω) with children (λ)

Figure 3. Pseudocode for the Markov chain Hyper-heuristic (MCHH) algorithm incorporated in a (μ+λ)-ES

3. EXPERIMENT SETUP

In order to examine the performance of the Markov chain Hyper-heuristic (MCHH) an experiment was conducted to compare the MCHH with (1+1)-ES meta-heuristics employing 1 heuristic each, a random hyper-heuristic, and the TSRoulWheel multi-objective hyper-heuristics from the literature. The aim of this experiment was to examine the general performance of the MCHH in addition to testing each hyper-heuristic's ability to cope with a set of heuristics that vary in behaviour and performance; including a completely ineffective heuristic.

3.1 Test Problems

In this experiment the DTLZ test problems [10], a well known test problem suite from the literature, was used to compare the meta- and hyper-heuristics. The heuristics were applied to DTLZ1 to 7. Two test problems are examined in detail (DTLZ1 and DTLZ2) as they provided different levels of difficulty (hard and easy) and varied problem features such as Pareto front geometry. Both problems were formulated for 3 objectives with DTLZ1 taking 7 parameters and DTLZ2 taking 12 parameters, all within the same domain.

The purpose of this experiment was to test the hyper-heuristic's ability to optimise problems with a fixed set of heuristics. As such basic (1+1)-ES meta-heuristics were used as a basis for all algorithms to ensure a fair comparison between the hyper-heuristics and against the performance of the heuristics applied individually. The (1+1)-ESs were all limited in ability as the DTLZ problems are best solved with advanced population selection strategies and population based heuristics. In an ES, the absence of a selection operator enhances the impact of the heuristic variation operators on the search performance. This was

done in these experiments to minimise the influence of factors external to the hyper-heuristic heuristic selection techniques.

A fixed seed population of solutions with random parameters were generated for all problems and used for every run on the respective problems. Each meta- and hyper-heuristic was run for 1000 evaluations on both test problems and trialled 30 times on both problems to ensure a fair comparison and to examine the consistency in performances.

3.2 Heuristics

Four heuristics were created for this experiment: 3 perturbative heuristics and 1 ineffective heuristic. The three perturbative heuristics were mutation, replication and transposition. The mutation heuristic applies single point additive mutation with random values drawn from a Gaussian distribution with standard deviation of $\sigma = 0.01$. The replication heuristic copies the value from one randomly selected parameter and replaces another different randomly selected parameter with the first value. The transposition heuristic swaps the values of a pair of different randomly selected parameters twice, swapping two different pairs. All three heuristics operate on one solution, affecting only that solution's parameter values. The ineffective heuristic returns a clone (exact copy of the original parameter values) of any given solution, essentially performing no operation and facilitating a test of the hyper-heuristics' ability to discard poor performing heuristics.

3.3 (1+1) Evolution Strategies

Four identical (1+1) Evolution Strategies (ESs) were created following the UMMEA framework to examine the performance of each heuristic applied individually. Each (1+1)-ESs employed one of the four heuristics from section 3.2 to vary the population. Each (1+1)-ES was given an unlimited archive of non-dominated solutions and used an elitist solution selection policy. In the selection policy, the parent is only replaced if the child dominates the parent.

3.4 Hyper-heuristics

3.4.1 Markov chain Hyper-heuristic

The hyper-heuristic selection method outlined in section 2 was embedded in a hybrid (1+1) Evolution Strategy; the same as in section 3.3. The hybrid (1+1)-ES was given an unlimited archive and used the same elitist solution selection policy. The MCHH learning parameters were set to $\gamma = 0.25$, $\alpha = 0.1$ and $\beta = 0.1$ with $\epsilon = 5$ generations per episode. The MCHH was given all four heuristics at the start of the optimisation process.

3.4.2 Random Hyper-heuristic

A random heuristic selection method was used for comparison with the MCHH. Again, the selection method was embedded in a hybrid (1+1)-ES with an unlimited archive and the same elitist solution selection policy. After each generation, the random heuristic selection method chose at random a new heuristic to apply in the next generation. The Random hyper-heuristic was given all four heuristics at the start of the optimisation process.

3.4.3 TSRoulWheel

The TSRoulWheel hyper-heuristic presented in [11] was implemented with a learning rate of 1. This optimiser was used to compare the performance of the MCHH learning method with a suitable example from the literature. As with MCHH and the Random hyper-heuristic, TSRoulWheel strategy was embedded in a (1+1)-ES and given all four heuristics at the start of the optimisation process.

3.5 Quality Measures for Comparison

Generational distance and Hypervolume were used to compare the performance of the meta- and hyper-heuristics. The generational distance [15] was used to examine the convergence of each method, calculating the average distance from the front for each solution in the archive at each generation. The distance measure was calculated using the minimum distance to a fixed sample set on the Pareto front. The hypervolume was used to examine coverage and population diversity. The hypervolume was calculated by sampling in the range [0, 100] for all objectives [16]. The generational distance of the final population was also used to compare the consistency in performance of each optimiser.

3.5.1 MCHH Learning

The sum of transition weights (averaged over the 30 trials) to each heuristic from MCHH weight matrix was calculated for each generation to examine the behaviour of the learning method used in the MCHH. The final MCHH weight matrix (averaged over the 30 trials) was also recorded to examine the final population weight matrices produced by the MCHH.

4. RESULTS

Sections 4.1 and 4.2 compare the algorithms on test problems DTLZ1 and DTLZ2 respectively. Section 4.3 shows the final generational distance results on DTLZ problems 1 to 7.

4.1 DTLZ1

Figure 4 and 5 show the generational distance for each heuristic and hyper-heuristic on DTLZ1, averaged over 30 trials. As expected, each of the heuristics perform with vary degrees of quality, from clone (that does nothing) to the mutation heuristic which outperforms both the random hyperheuristic and TSRoulWheel. The MCHH works well, outperforming mutation and getting closest to the true Pareto front. Interestingly, the random hyper-heuristic is comparable to TSRoulWheel in terms of averaged generational distance but actually achieves better coverage and diversity, as indicated by the hypervolume. Although, on average, the TSRoulWheel is less effective than the random hyper-heuristic, the results are more reliable. This is shown in Figure 6 which displays each algorithm's distribution of final generation distances for each trial on DTLZ1.

The replication heuristic was shown in the experiment to be highly exploitative, efficiently copying good parameter values across the solution parameter vector. However, as with all greedy heuristics, the algorithm quickly stagnates and reaches the best possible result with the limited parameter values available to it. Interestingly, although transposition did poorly in terms of generational distance, it continued to improve the hypervolume suggesting the heuristic increases diversity.

It is this range of behaviours that allows the hyper-heuristics, like MCHH, to perform better than any one heuristic applied alone. Furthermore, the results for the random hyper-heuristic give support to the theory that any combination of heuristics, even applied without learning, is better than the average of each used

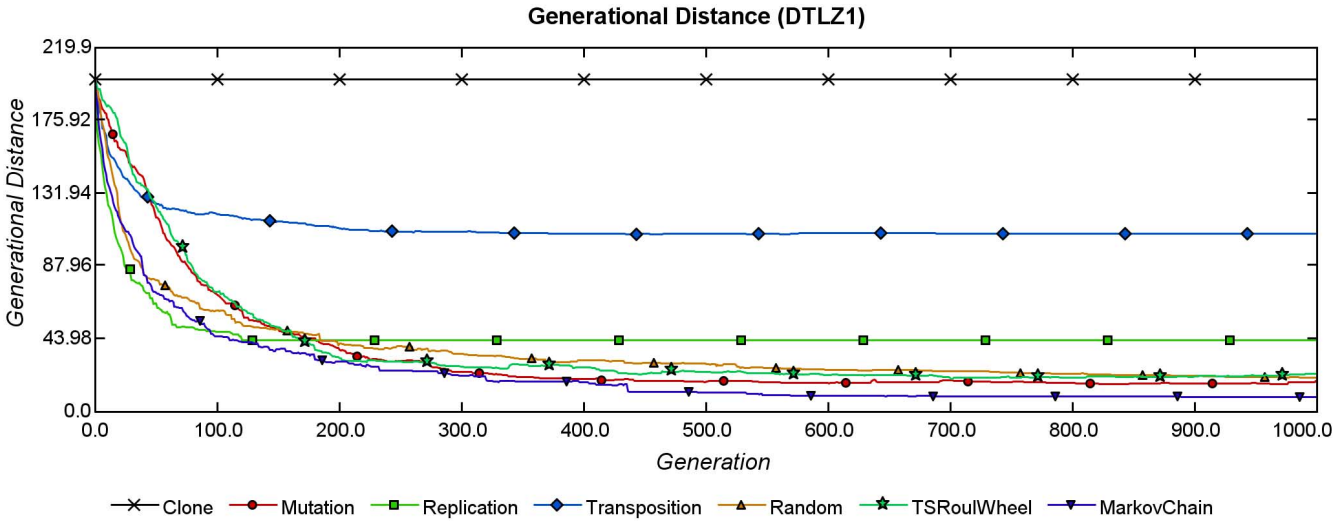

Figure 4. Generational distance over generations which was averaged over 30 trials of 1000 evaluations of DLTZ1.

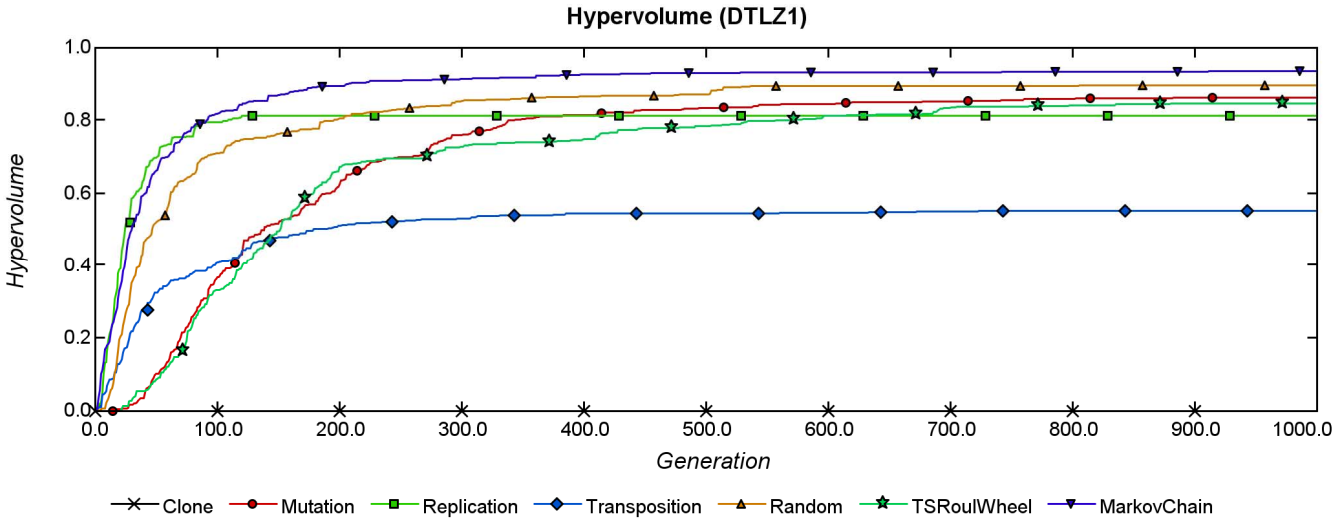

Figure 5. Hypervolume over generations which was averaged over 30 trials of 1000 evaluations of DLTZ1.

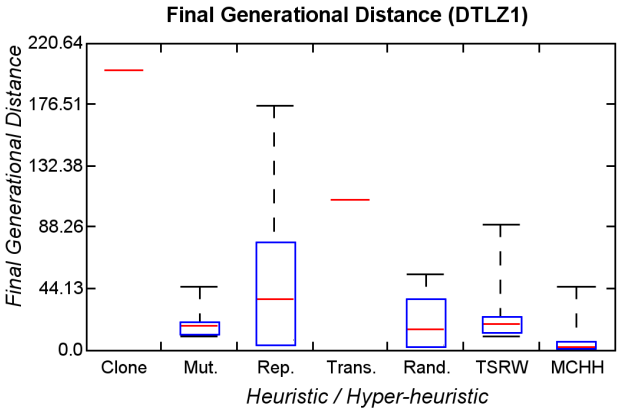

Figure 6. Boxplot of the generational distance of the final population from all 30 trials for all optimisers on DLTZ1.

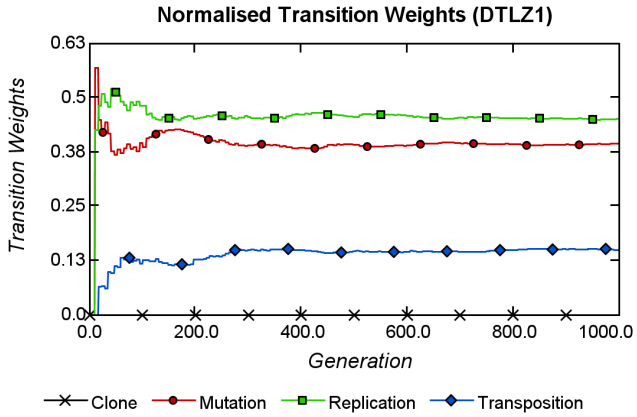

Figure 7. Sum of normalised transition weights for moves to each heuristic over generations for MCHH on DLTZ1.

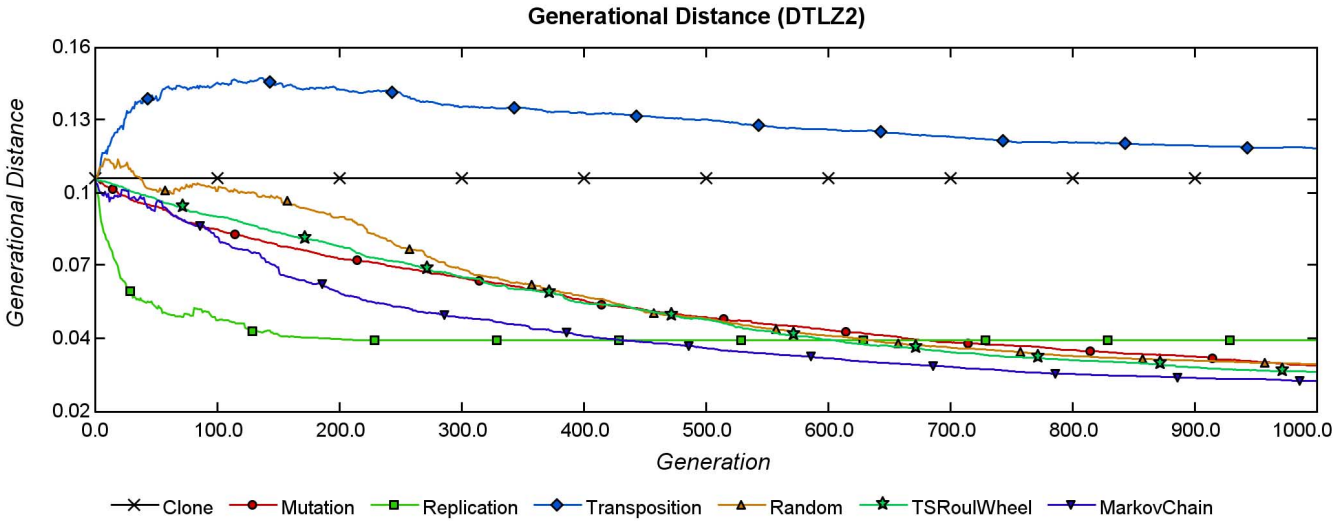

Figure 8. Generational distance over generations which was averaged over 30 trials of 1000 evaluations of DLTZ2.

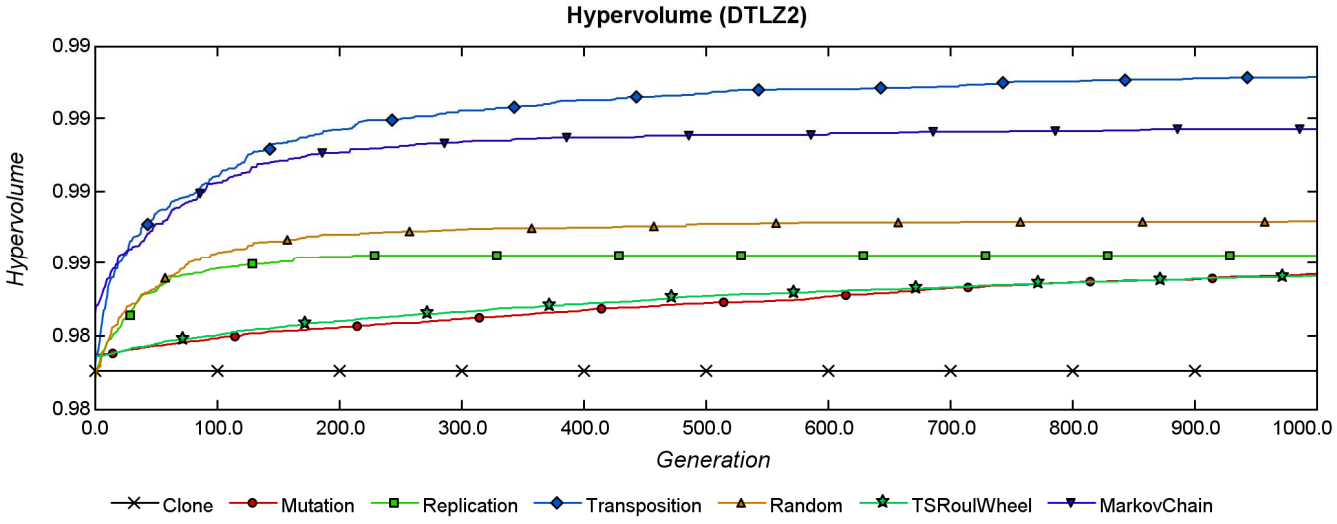

Figure 9. Hypervolume over generations which was averaged over 30 trials of 1000 evaluations of DLTZ2.

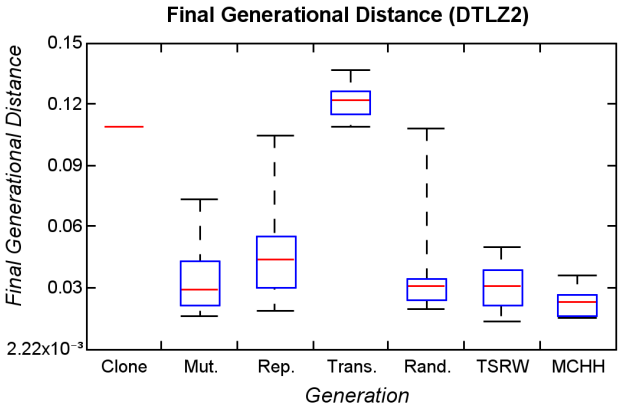

Figure 10. Boxplot of the generational distance of the final population from all 30 trials for all optimisers on DLTZ2.

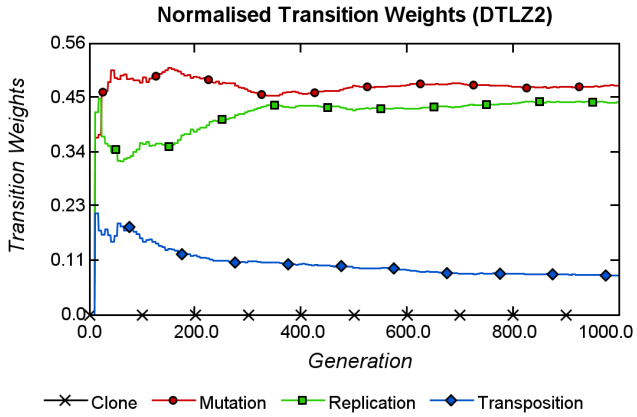

Figure 11. Sum of normalised transition weights for each heuristic over generations for MCHH on DLTZ2.

alone. In this example, the continual mix of heuristics results in an averaging of performance, improving on the worst heuristics but not attaining the same performance as the very best.

One of the aims of the experiment was to examine how the learning mechanism in the MCHH performs on a range of heuristics, specifically how it copes with poor heuristics like clone. Figure 7 and 12 display the weights generated by the MCHH on DTLZ1. Figure 7 shows for each heuristic the sum of income weights for DTLZ1 over generations. Interestingly, in the first 250 generations the weighting of the mutation and replication heuristics appear to oscillate. This could be a result of the MCHH overweighting good performance of heuristics in the early stages or a reflection of the actual performance of the individual heuristics. The latter does match the results in 4 and 5, where the replication heuristic is shown to perform very well in the early stages of the optimisation process and then plateau. This is also reflected in Figure 12 which shows the transition weight matrix after optimising DTLZ1.

Although the clone heuristic was intentionally introduced as a poor performer to examine the affect of ineffective heuristics on the hyper-heuristics it doesn't appear to have a significant effect. As is shown in Figure 12, the MCHH eliminates any weighting for the clone heuristic which might be contributing to the improved performance over random and TSRoulWheel.

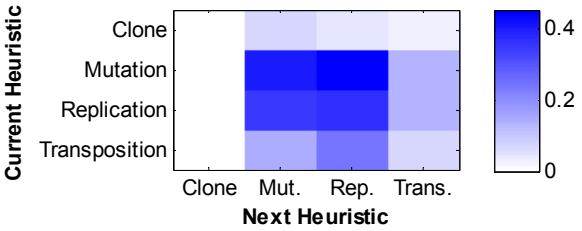

Figure 12. Colour map of the final matrix of transition weights between states in the MCHH averaged over 30 trials of 1000 evaluations of DTLZ1.

4.2 DTLZ2

Figure 8 and 9 show the generational distance for each heuristic and hyper-heuristic on DTLZ2, averaged over 30 trials. The majority of heuristics maintain a similar performance on this easier problem, which is shown in Figure 10. Although MCHH doesn't perform as well as in DTLZ1 (relative to the other heuristics), it still obtains the best final generational distance and performs better than both the random and TSRoulWheel hyper-heuristics throughout the search. In contrast, the random hyper-heuristic initially performs poorly, reflecting the behaviour of the transposition heuristic. However, after the first 100 generations, the random hyper-heuristic's performance recovers and quickly matches TSRoulWheel and the mutation heuristic by generation 500. In these circumstances, where a heuristic is adversely affecting convergence, the positive effects of the learning applied by TSRoulWheel and the MCHH are best demonstrated.

Other than the clone heuristic, the three other heuristics were designed to operate with varying degrees of performance, whilst still expecting dominating solutions to be produced occasionally. However, the transposition heuristic appears to diverge, moving away from the front for the first 150 generations before starting to converge again. The final generational distance of the transposition heuristic is in fact worse than the initial random seed. However, whilst the generational distance is degrading, it was noted that the transposition heuristic still generates new non-dominated solutions and inserted them into the archive (which will allow for the generational distance to diverge). The hypervolume trend also indicates an increasing diversity in the population, covering a larger proportion of objective space.

The behaviour of transposition can be explained by the natural shape formed by Pareto fronts (given a random distribution of points) and the geometry of the DTLZ2 problem. Pareto fronts tend to form a rounded surface with a knee towards the intersection of the axes as a result of the dominance function. In 3 dimensions, this creates a shape roughly similar to an octant of a sphere with the centroid about the nadir (worst possible solution). The DTLZ2 problem is artificially designed with the true Pareto covering the surface of an octant of a sphere with its centroid at the intersection of the axes – the inverse of the natural formation of a Pareto front. Therefore, as solutions are found towards the extremes of the current Pareto front (which are non-dominated and inserted into the archive) they could, in fact, be further away from the true Pareto front, degrading the generational distance whilst improving the hypervolume. This unintentional behaviour is a good demonstration of the problems that are be encountered by meta-heuristics with fixed heuristic strategies. By employing selective hyper-heuristics, like the MCHH, it is possible to mitigate these problems and further improve performance.

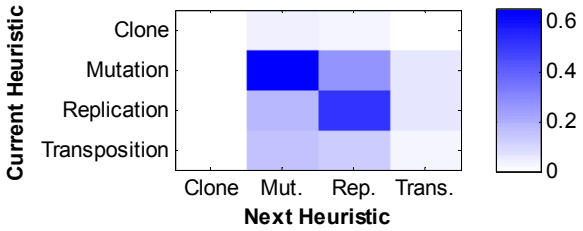

Figure 13. Colour map of the final matrix of transition weights between states in the MCHH averaged over 30 trials of 1000 evaluations of DTLZ2.

Figure 13 shows the transition weight matrix after optimising DTLZ2. In these weights, the MCHH has identified the poor performance of the transposition heuristic and reduced weights that might allow for moves to this heuristic, unlike on DTLZ1 where there was a chance of applying transposition. For the easier problem, the MCHH gave a higher weighting to the mutation heuristic, preferring to remain in that state for the majority of the search. However, a small weighting is still given to the replication heuristic, suggesting occasional use.

Figure 11 shows, for each heuristic, the sum of income weights over generations for DTLZ2. As on DTLZ1, the replication heuristic does well at the start of the search, with a higher weighting than the mutation heuristic. However, as the performance of mutation heuristic improves the relative weighting is reversed. This demonstrates the ability of the MCHH in adapting the weights for each heuristic as their relative performances change and one heuristic becomes more optimal at different stages in the search process.

4.3 Summary of DTLZ1-7

The final generational distance results for all algorithms on the DTLZ test problems 1 to 7 are given in Table 1. As with the results shown in Section 4.1 and 4.2, the results in Table 1 are averaged over 30 trials with each heuristic starting with the same

seed population. In these experiments, none of the heuristics or hyper-heuristics outperformed the MCHH during both the short 1,000 generation runs and longer runs for each problem. The MCHH achieved the best generational distance, even after 50,000 generations on DTLZ1.

Table 1. Final generational distance averaged over 30 trials over 1,000 generations of (1+1)-ES. The best results are shown in bold italics and the second best results in bold.

	Clone	Mut.	Rep.	Trans.	Rand.	TSRW	MCHH
DTLZ1	189.80	29.87	53.22	186.59	51.06	**29.36**	***22.05***
DTLZ2	0.13	0.025	0.023	0.072	**0.013**	0.026	***0.012***
DTLZ3	819.95	170.41	469.98	840.74	**158.19**	176.80	***90.78***
DTLZ4	27.19	5.78	15.28	27.49	**5.21**	5.95	***3.08***
DTLZ5	0.92	0.33	0.27	0.90	**0.13**	0.37	***0.06***
DTLZ6	1.05	1.46	1.42	**0.64**	2.36	1.38	***0.63***
DTLZ7	14.57	12.95	6.04	**5.47**	5.59	12.96	***5.04***

5. CONCLUSION

In this paper we presented a novel selective hyper-heuristic, the Markov chain Hyper-heuristic (MCHH), and applied it to multi-objective continuous problems. The hyper-heuristic applies a reinforcement learning technique to update the transition weights in a Markov chain [9]. An experiment was conducted to compare the performance of the proposed technique against a set of meta-heuristic (1+1) Evolution Strategies, a random selective hyper-heuristic, and a multi-objective hyper-heuristic from the literature [11].

The results demonstrate the efficacy of the method in terms of its ability to learn good heuristic combinations, outperforming the tuned heuristics. The MCHH is shown to effectively penalise poor heuristics and learn good heuristic sequences. The TSRoulWheel hyper-heuristic from the literature is shown to match the performance of the best heuristic, but does not surpass it. The MCHH outperforms the random hyper-heuristic - a surprisingly good optimiser despite having no intelligent selection strategy.

In addition to applying the MCHH to the DTLZ problems (of which two are shown in detail above), we applied the MCHH to the more complex Walking-Fish Group (WFG) toolkit problems [17]. The results also demonstrated the efficacy of the method but highlighted the need for more advanced solution acceptance strategies to further improve the search. Whilst intelligent heuristic selection can greatly improve the optimisation process, an optimiser is limited by the worst operating element and so it is important to incorporate higher-level techniques like the MCHH in effective meta-heuristics to achieve the best performance.

Although the MCHH is shown to be good at converging to the front, there is no additional mechanism to encourage diversity. Future work will look to explore diversity preserving mechanisms and how they may be introduced into the MCHH. In addition, the MCHH should be compared with single-objective hyper-heuristics on traditional single-objective combinatorial hyper-heuristic problems from the literature to better demonstrate the generality of the method.

6. ACKNOWLEDGEMENTS

This work was supported by an EPSRC CASE award (Grant No. CASE/CNA/07/100) and Mouchel Limited.

7. REFERENCES

[1] Booker, A. J., Dennis, J. E., Frank. P. D., Serafini, D. B., Torczon, V., and Trosset, M. W. A rigorous framework for optimization of expensive functions by surrogates. *Structural and Multidisciplinary Optimization*. *17* (1999), 1-13.

[2] Kazemi, G. M., Wang, G., Rahnamayan, S., and Gupta, K., Metamodel-Based Optimization for Problems With Expensive Objective and Constraint Functions. *J. Mech. Des. 133*, 014505 (2011).

[3] Jones, D. R., Schonlau, M., and Welch, W. J.. Efficient global optimization of expensive black-box functions. *J. of Global Optimization*. *13*, 4 (1998), 455-492.

[4] Vazquez, E., Villemonteix, J., Sidorkiewicz, M., and Walter, E. Global optimization based on noisy evaluations: an empirical study of two statistical approaches. *J. of Global Optimization*. (2008).

[5] Villemonteix, J., Vazquez, E., and Walter, E. An informational approach to the global optimization of expensive-to-evaluate functions. *J. of Global Optimization*. (2008), 26-34.

[6] 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.

[7] Burke, E. K., Hyde, M., Kendall, G., Ochoa, G., Ozcan, E., and Woodward, J. A Classification of Hyper-heuristics Approaches. *Handbook of Metaheuristics*, International Series in Operations Research & Management Science. Springer, 2009.

[8] Burke, E. K., Hyde, M., Kendall, G., Ochoa, G., Ozcan, E., and Qu, R. *Hyper-heuristics: A Survey of the State of the Art*. Computer Science Tech. Rep. NOTTCS-TR-SUB-0906241418-2747, University of Nottingham, 2010.

[9] J. G. Kemeny, and J. L. Snell. *Finite Markov Chains*. Springer Verlag, Princeton, NJ, 1976.

[10] Deb, K., Thiele, L., Laumanns, M., and Zitzler, E. *Scalable Test Problems for Evolutionary Multi-Objective Optimization*. Zurich, Switzerland, Tech. Rep. 112, 2001.

[11] Burke, E. K., Landa Silva, J. D., Soubeiga, E. Multi-objective Hyper-heuristic Approaches for Space Allocation and Timetabling, *In Meta-heuristics: Progress as Real Problem Solvers, Selected Papers from the 5th Metaheuristics International Conference (MIC 2003)*, Springer, 2005, 129-158.

[12] Goldberg, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, 1989.

[13] Meignan, D., Koukam, A., Creput, J. C. Coalition-based metaheuristic: A self-adaptive metaheuristic using reinforcement learning and mimetism. *J. Heuristics*. *16*, (2010), 859-879.

[14] Laumanns, M., Zitzler, E., Thiele, L. A unified model for multi-objective evolutionary algorithms with elitism. *In Proceedings of the 2000 Congress on Evolutionary Computation, 2000*, (La Jolla, CA), *1*, (2000), 46-53.

[15] Van Veldhuizen, D. A. and Lamont, G. B. Evolutionary Computation and Convergence to a Pareto Front. *In Late Breaking Papers at the Genetic Programming 1998 Conference*. (Stanford University), 1998, 221-228.

[16] Bader, J., Deb, K., and Zitzler, E. Faster Hypervolume-based Search using Monte Carlo Sampling. *In Conference on Multiple Criteria Decision Making (MCDM 2008)*. Springer, 2008, 313-326.

[17] Huband, S., Hingston, P., Barone, L., While, L. A Review of Multiobjective Test Problems and a Scalable Test Problem Toolkit. *In IEEE Trans. on Evolutionary Computation*. *10*, 5 (2006), 477 - 506.