Automated construction of evolutionary algorithm operators for the bi-objective water distribution network design problem using a genetic programming based hyper-heuristic approach

Kent McClymont, Edward C. Keedwell, Dragan Savić and Mark Randall-Smith

ABSTRACT

The water distribution network (WDN) design problem is primarily concerned with finding the optimal pipe sizes that provide the best service for minimal cost; a problem of continuing importance both in the UK and internationally. Consequently, many methods for solving this problem have been proposed in the literature, often using tailored, hand-crafted approaches to more effectively optimise this difficult problem. In this paper we investigate a novel hyper-heuristic approach that uses genetic programming (GP) to evolve mutation operators for evolutionary algorithms (EAs) which are specialised for a bi-objective formulation of the WDN design problem (minimising WDN cost and head deficit). Once generated, the evolved operators can then be used *ad infinitum* in any EA on any WDN to improve performance. A novel multi-objective method is demonstrated that evolves a set of mutation operators for one training WDN. The best operators are evaluated in detail by applying them to three test networks of varying complexity. An experiment is conducted in which 83 operators are evolved. The best 10 are examined in detail. One operator, GP1, is shown to be especially effective and incorporates interesting domain-specific learning (pipe smoothing) while GP5 demonstrates the ability of the method to find known, well-used operators like a Gaussian.

Key words | evolutionary algorithm, genetic programming, hyper-heuristic, mutation, optimisation, water distribution network

Kent McClymont (corresponding author)
Edward C. Keedwell
Dragan Savić
College of Engineering,
Mathematics and Physical Sciences,
University of Exeter,
Exeter EX4 4QF,
UK
E-mail: *K.McClymont@exeter.ac.uk*

Mark Randall-Smith
Mouchel, Clyst Works,
Clyst Road, Topsham,
Exeter EX3 0DB,
UK

INTRODUCTION

The water distribution network (WDN) design problem is primarily concerned with optimising the size (diameters) of pipes in a network in order to satisfy customer demand while adhering to operational hydraulic constraints such as head and velocity requirements. Modification of pipe sizes affects the hydraulic conditions in a network and hence the quality of the network based on its ability to serve the various demand points. As such, the problem is complicated as the overall hydraulic conditions are affected by each pipe and so changes to one pipe will have a different effect on the overall conditions depending on the sizes of all the other pipes in the network, creating interdependencies between the relative sizes of different pipes in the network. As such, each pipe cannot be designed in isolation, but rather as a combination of sizes for all pipes in the network. This combinatorial effect means that even for relatively small networks, the number of possible combinations of pipes is very large and makes enumeration of all the possible designs impossible within reasonable time. If, for example, there were six potential sizes for each pipe in a network of just 30 pipes, there would be 2.21×10^{23} possible combinations – far more than is possible to evaluate within reasonable time – and so WDN design is therefore known as a NP-hard problem (Yates *et al.* 1984).

doi: 10.2166/hydro.2013.226

The quality of potential WDN designs (candidate solutions) can be evaluated against a range of criteria, such as the ability to satisfy demand, by building computational models of these networks in programs such as EPANET (Rossman 2000). Such models provide a means for automatically evaluating candidate network designs and therefore enables the use of optimisation techniques like genetic algorithms (GAs) (Goldberg 1989; Simpson *et al.* 1994; Savić & Walters 1997) to automatically search for approximately optimal network designs. GAs are a type of evolutionary algorithm (EA) which are nature inspired methods that mimic Darwinian evolution and use populations of candidate solutions (potential network designs) to explore the problem search space, looking for optimal network designs over a number of generations by iteratively mutating and proposing new designs. Although these traditional optimisation methods have been demonstrated numerous times in the literature to be effective at solving the WDN design problem, in recent years a new methodology called hyper-heuristics has been established which is more effective at solving a wide range of optimisation problems, including the WDN design problem. Hyper-heuristics are able to provide improved performance over traditional optimisers, like EAs, as they utilise machine learning techniques to tailor the optimiser (e.g., EA) to each problem, like the WDN design problem, through automated learning methods or, as is in this paper, construction of optimised heuristics (like a GA's mutation operator). The benefit of meta-optimisation methods like hyper-heuristics is that they are able to more efficiently solve optimisation problems by optimising the optimiser and tailoring them to the problem, reducing the resources required to obtain the same quality network designs which makes optimisation of large-scale problems more feasible within a reasonable time.

Generative hyper-heuristic approaches automate the process of creating tailored, more effective optimisation operators for a specific problem, such as the WDN design problem. By automating this process of optimising the optimiser, rather than hand-crafting new mutation operators, hyper-heuristics are able to consider a much larger set of mutation operators than a human expert and thus potentially able to find better mutation operators. Once the hyper-heuristic has evolved a tailored mutation operator (or collection of operators), the evolved mutation operator(s) is then fixed and thus reusable and can be easily incorporated into existing meta-heuristic optimisers like the well-known genetic algorithm NSGA-II (Deb *et al.* 2000) or any other EA of choice. The power of this approach is even more apparent when it can be conceived that a set of tailored mutation operators could be utilised by selective hyper-heuristics such as AMALGAM (Raad *et al.* 2010) or the MCHH (McClymont *et al.* 2012b), both of which have been successfully applied to the WDN problem, and so combine the tuning of the generative hyper-heuristic and the adaptive strength of the online selective hyper-heuristic.

This paper presents a hyper-heuristic approach for evolving mutation operators for the WDN design problem. The proposed approach extends the early, single-objective method presented in McClymont *et al.* (2012a) and presents a novel application of genetic programming (GP) based hyper-heuristics for the bi-objective WDN design problem. The paper studies the potential of evolving novel EA mutation operators tailored for the WDN design problem and for use in any EA. The evolved mutation operators are examined through an experiment which illustrates the potential of this method.

The remainder of this section is dedicated to a summary of the key relevant works in the areas of WDN design and hyper-heuristic research. The *Method* section describes the hyper-heuristic method used in this study which is applied to a bi-objective WDN design problem outlined in the *Water distribution network problem* sub-section of *Experimental setup*. The *Experimental setup* section describes an experiment which demonstrates the efficacy of the method which is shown in the *Results* section. In particular, one mutation operator is highlighted which has interesting properties that reflect useful, domain-specific behaviour. The method, results and findings are discussed in the *Conclusion*.

The water distribution network design problem

Traditionally, the WDN design problem has been formulated as a single-objective problem where the quality of the network is based solely on the economic impact of the design; i.e., given a fixed layout, the optimal network design is one which meets the hydraulic requirements with the least possible cost. The hydraulic constraints are usually

given as an acceptable range of node pressures or pipe velocities.

A range of methods has been proposed in the literature for solving the WDN problem. Perhaps the most common approach is the use of meta-heuristic EAs (Laumanns *et al.* 2000), such as GAs (Goldberg 1989; Simpson *et al.* 1994; Savić & Walters 1997). The methods use 'populations' of individual network designs and evolutionary operators, like crossover and mutation, to mimic the process of evolution and search for good network designs over a number of generations. While these methods have been shown in numerous studies to be effective at solving a variety of single-objective and multi-objective variants of the WDN, it is acknowledged that EA methods require a large number of evaluations of potential networks in order to locate good network designs. While this is acceptable for small networks, the expensive nature of EA search (in terms of time and computing resources) coupled with the complex and slow run times of many network simulation tools can be prohibitive when searching larger network designs.

In order to combat the problem of expensive EA searches, a number of fast methods have been explored in the literature that aim to either boost the initial EA generations or replace the EA search process altogether. For example, Keedwell & Khu (2005) proposed a cellular automata (CA) inspired approach to solving the WDN design problem which required significantly less evaluations. Furthermore, when coupled with GAs, the CA approach was shown to provide an efficient enhancement to the early stages of the GA search. This technique and others like them have led to the creation of algorithms for particular problems and problem types through the construction of specialised heuristics and GA operators. This has typically been undertaken as a manual process, utilising human expertise and incorporating this into the search process. However, recently, an automated approach to this problem has been developed in the field known as hyper-heuristics, effectively the automated construction of meta-heuristics.

Hyper-heuristics

In recent years, a new methodology has emerged in the field of optimisation called hyper-heuristics (Cowling *et al.* 2000; Burke *et al.* 2010). This new paradigm is dedicated to extracting key optimisation mechanics and in order to make them more generalised across many different sets of optimisation problems while utilising highly specialised domain-specific knowledge.

Two types of hyper-heuristics have been identified in the literature, called selective and generative hyper-heuristics (Burke *et al.* 2009, 2010). Selective hyper-heuristics are designed to optimise the selection and sequencing of existing 'low-level heuristics', such as mutation operators in an EA, to optimise both search speed and quality of results. Examples of selective hyper-heuristics in hydro-informatics include the MCHH (McClymont *et al.* 2012b), an online selective hyper-heuristic for embedding in meta-heuristics and AMALGAM (Raad *et al.* 2010), a multi-method online selective hyper-heuristic which controls population assignment for multiple meta-heuristics.

Generative hyper-heuristics, an example of which is studied in this paper, are designed to automate the creation of specialised, domain-specific 'low-level heuristics', e.g., mutation operators. For example, an EA uses two 'low-level heuristics' to create new network designs: crossover and mutation. While crossover and mutation are effective at solving a range of problems, specialised operators such as that proposed in Keedwell & Khu (2005) demonstrate the power of utilising knowledge of the domain to significantly improve the efficiency of the optimisation search process. Generative hyper-heuristics are able to automatically construct these domain-specific EA operators using techniques such as GP (Koza 1992).

By creating EA operators using GP, it is possible to search and compare a vast range of different mutation operators and select those that are most appropriate for a given problem. Furthermore, GP evolved mutation operators are able to represent a wider set of operational behaviour beyond normal mutation and crossover operators and, theoretically, could locate entirely new EA operators that are better suited to a specific problem. GP is particularly appropriate for this as the approach is not constrained to a specific type of operation (such as applying an additive single-point mutation) and rather than searching for better parameters for existing types of operation, GPs search the space of different operational behaviour and so have the potential to discover entirely novel EA operator behaviours. The method discussed below utilises this GP approach and

so is classed as a generative hyper-heuristic rather than simply a parameter tuning method.

METHOD

As highlighted in Keedwell & Khu (2005), the WDN design problem has a number of features that can be exploited to potentially improve the search process. First, the network layout is fixed and each pipe (the optimisation parameters) has a fixed relationship with every other pipe. Furthermore, through simulation, it is possible to associate specific conditions with each pipe. For example, while we assess the overall head conditions of the network to determine a design's validity, it is possible to associate the downstream node's head with each contributing pipe. For example, if a node has excessive head, it is reasonable to assume that the supplying pipes may be too large and so are eligible for diameter reduction. Likewise, if a node has head deficit, then the supplying pipe is likely to be too small. Using these principles, it is possible to create mutation operators that take these hydraulic factors into account when creating new network designs, i.e., building informed mutation operators. This section describes a novel multi-objective generative hyper-heuristic framework for building novel mutation operators for the WDN design problem.

A generative hyper-heuristic framework

Figure 1 depicts the general generative hyper-heuristic framework used in this study. The approach uses a training network, i.e., a simple WDN, to evolve '*optimal*' mutation operators for use on any WDN. The generative framework is split into three phases: initialise, generate and evaluate. The initialise phase generates the initial random population of mutation operators to seed the optimisation process. The initialise phase also generates the sample network designs to the underlying WDN which are used to evaluate the evolved mutation operators. The sample solutions (candidate WDN designs) are fixed and to ensure a fair as is possible

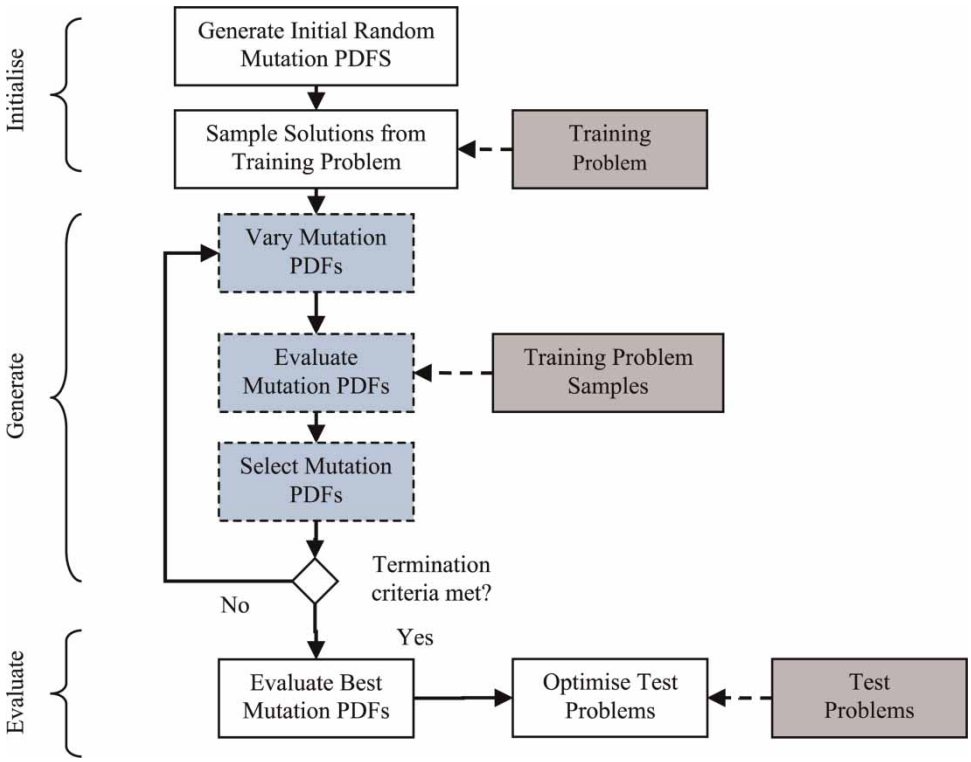

Figure 1 | General generative framework. Elements with dashed, shaded boxes indicate generative optimisation actions and grey shaded elements indicate interaction underlying problem class. The framework shows how a probability distribution function (PDF), in this case a specialised GP tree, can be evolved using samples from a training network in using the generative hyper-heuristic approach.

comparison between the evolving mutation operators. The generate phase is an optimisation loop where the current population of mutation operators are varied, evaluated using the network designs sampled from the underlying training network, and selected for propagation into the next generation. This optimisation loop is repeated until some termination criteria are met – such as a fixed number of generations. Once the generative optimisation phase is completed, the best evolved mutation operators are then evaluated in more detail by inserting them into identical EAs and applying them to a set of test networks (in this case the Anytown benchmark and two real-world WDNs). The evaluation phase is used to examine how well the evolved mutation operators perform across the whole search process and to what extent they are useful in practical applications. The evaluation phase is also used for removing mutation operators which are over-fit to the training network.

Evolutionary algorithm for testing

In this study, a $(\mu + \lambda)$ evolution strategy (ES) (Laumanns *et al*. 2000) is used to test and compare the best evolved mutation operators. ESs are similar to GAs, using similar population selection methods with only a few different features. GAs use both mutation and crossover operators to generate new network designs while ESs use only a mutation operator. ESs are therefore more appropriate in this study for comparing the evolved mutation operators as they remove the influence of the GA crossover operator. ESs also maintain an additional population, called an *archive*, which contains the best, non-dominated candidate network designs found so far in each optimisation run. In this case, the archive stores the best candidate networks generated by the ESs using the evolved mutation operators. The archives can then be used to calculate the hypervolume indicator and compare the performance of the difference evolved mutation operators. The terms μ and λ refer to the size of the parent and child populations, respectively.

Optimisation method

Any optimising method could be used to optimise the GP mutation operators in the *generate* phase of the framework given in Figure 1. In the following experiments the optimiser SPEA2 (Strength Pareto Evolutionary Algorithm 2) (Zitzler *et al*. 2002) was used to optimise the GP mutation operators. SPEA2 was given an unlimited passive archive. The network design encoding, evaluation functions, variation operators and selection methods are described below.

Genetic programming

GP was proposed by Koza (1992) as a method for utilising EAs for automating the creation of programs. GPs use trees to represent computer programs, such as the example GP tree shown in Figure 2. The trees can be manipulated by mutating

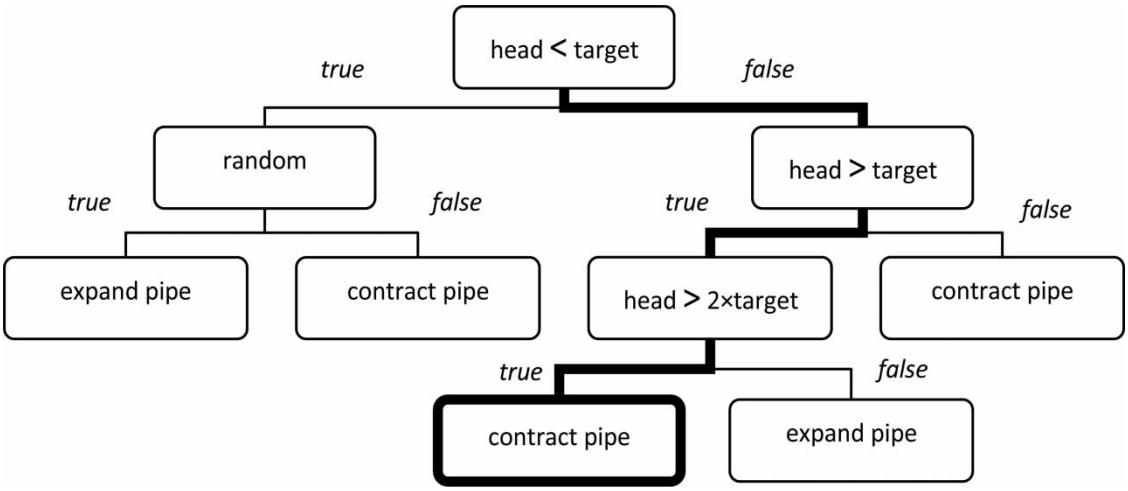

Figure 2 | Decision tree representation used in the generative hyper-heuristic to create GP evolved mutation operators for the WDN design problem with the illustrated path and action in thick bold lines.

nodes on the tree or rearranging branches of the tree or even swapping sections of different trees. These modifications act in much the same way as mutation and crossover in GAs and enables the automatic creation and search of small 'programs'. Usually the fitness of a program is assessed by testing it with a range of inputs and determining how close the output of the evolved program is to some target.

Traditionally, GP was used to represent functions and evolved to approximate some given target function. For example, in classification, the evolved programs could be used to label samples and associate them with a specific class. However, with the emergence of the field of hyper-heuristics, the power of GP was quickly realised and utilised to automatically generate new, novel heuristics that were specialised for a given problem (Burke *et al.* 2006). This method uses GPs to evolve new mutation operators, representing the mutation operators as program trees in order to evolve different mutation behaviours.

GP evolved mutation operators

All GP evolved mutation operators first selected a fixed number of pipes at random. Each of the selected pipes were parsed by the GP in turn and mutated depending on the tree's structure. In this study we used a simple decision tree structure constructed of branches and terminals (see example in Figure 2). All branches in the tree represent Boolean conditional statements and all terminals represent mutation operations. The Boolean branches compared the pipe's features or used random numbers to determine which terminal mutation operation would be applied. The branches were nested, allowing for a number of conditional statements in succession. For example, given a pipe with more than twice the target head at the downstream node, the features of the pipe would be used to navigate the tree and apply the terminal operation as illustrated in Figure 2. If a pipe with different attributes was parsed by the same tree, the output would potentially be different. The combination of the conditionals and fixed mutation operations enable the creation of '*expert*' mutation operators that determine the most appropriate form of mutation given the pipe characteristics.

The Boolean conditional statements either compared the selected pipe's downstream node's head to the target head (or some relative value) or compared a randomly drawn number with a given threshold. These two types of conditional statements allowed for domain-specific branching and, if desired, a random element. The Boolean branches are given in Table 1.

The mutation operations (terminals) determined what type of mutation action would be applied to the selected pipe. Two types of mutation were used: fixed mutation and random mutation. The fixed mutation always either increased or decreased the pipe by a fixed amount. The random mutation replaced the pipe diameter with a new randomly selected pipe diameter. All the mutation operations are given in Table 1.

Sampling training solutions (network designs)

To evaluate the evolved mutation operators, the proposed generative framework tests the operators on a set of sampled network designs from the underlying problem (in this case WDN designs) and determines whether the operator is likely to create better networks by mutating each sample multiple times and comparing the newly generated networks with the original sample. In this study, sample networks were obtained by optimising the test network and recording each of the network designs created during this optimisation search. This ensured that a range of samples (networks) of varying quality were produced; poor at the start of the search and good at the end of the search. The variety of quality allowed the GP mutation operators to be evaluated on both good and poor networks to assess whether it was useful at the start or end of an optimisation search.

A $(\mu + \lambda)$-ES (parent and child populations of size 10) with traditional uniform crossover and additive multi-point Gaussian mutation was used to optimise the test network and collect the sample network designs. The network designs generated by this optimiser were then used for training. A $(\mu + \lambda)$-ES was used instead of SPEA2 (which was used to evolve the GP mutation operators) for sampling networks as the selection mechanism gave minimal bias to the distribution of network generated by the meta-heuristic. SPEA2 is a faster, more efficient optimiser compared to the $(\mu + \lambda)$-ES and so would generate a larger quantity of good networks compared to the $(\mu + \lambda)$-ES which generated a more even distribution; the latter is preferable for training the evolved GP mutation operators.

Table 1 | Base mutation operations represented as GP branches (if-else statements) and terminals (conditional expressions and actions)

GP element	Description
***if-else* statements (branches)**	
if [condition] *then* [action] *else* [action]	*Evaluates a condition and, if true, executes the first action, otherwise the second action is executed.*
if [condition] *and* [condition] *then* [action] *else* [action]	*Evaluates both conditions **and** if both are true then executes the first action, otherwise the second action is executed.*
Conditional expressions (operands)	
rand > [0, 1]	Generates a new random real-valued number in the range [0, 1] (inclusive) and returns true if the random number is **greater than** a constant real-valued number in the range [0, 1] (fixed in the GP).
rand < [0, 1]	Generates a new random real-valued number in the range [0, 1] (inclusive) and returns true if the random number is **less than** a constant real-valued number in the range [0, 1] (fixed in the GP).
[downstream / upstream]_diameter < current_diameter	Compares the diameter of the current pipe with the diameter of either the downstream or upstream pipe and returns true if the current pipe is **larger**.
[downstream / upstream]_diameter > current_diameter	Compares the diameter of the current pipe with the diameter of either the downstream or upstream pipe and returns true if the current pipe is **smaller**.
[downstream / upstream]_head < [0, 90 m]	Compares the head of the current pipe's downstream or upstream node with a constant value in range [0, 90 m] returns true if the head is **less than** the constant (fixed in the GP).
[downstream / upstream]_head > [0, 90 m]	Compares the head of the current pipe's downstream or upstream node with a constant value in range [0, 90 m] returns true if the head is **greater than** the constant (fixed in the GP).
Actions (terminals)	
Increase diameter by [1, 3]	**Increases** the current pipe's diameter by 1, 2 or 3 pipe diameter sizes (fixed in the GP).
Decrease diameter by [1, 3]	**Decreases** the current pipe's diameter by 1, 2 or 3 pipe diameter sizes (fixed in the GP).

The $(\mu + \lambda)$-ES optimiser is run on the test WDN a set number of times to generate the desired number of sample network designs. The set of sample network designs are then sorted into three sets of equal size: random and early networks (referred to later as 'far'); mid optimisation networks (referred to later as 'mid'); and networks closest to the global optima (referred to later as 'close'). These three categories broadly define the general stages in the optimisation search. Again, once the sets are generated they are fixed for all evaluations of candidate GP mutation operators; i.e., these networks form the pool of initial networks which the mutation operators must then perturb. The deviation in fitness value (or Pareto domination) of the new heuristically derived networks (generated by the evolved mutation operator) from the original sampled networks informs the fitness of that particular mutation operator.

To create the tree sample sets of 'close', 'mid' and 'far', the sampled network designs from the multi-objective problems created by the $(\mu + \lambda)$-ES optimiser runs were initially combined and sorted into fronts using Pareto dominance. The network designs in each front (those that all mutually non-dominated one another) were then sorted again within the front by the sum of their objective values; e.g., the network designs in the first front were sorted by the sum of their objective values – producing an ordered front. The network designs in the next front were then sorted – producing a second ordered front – and so on until all the network designs were sorted first by front number and then by the sum of their objective values (in ascending order, giving preference to smaller summed objectives). The whole population of sorted network designs was then split equally into the categories as described above. Providing an ordering to the network designs enabled an even split of network designs across each of the categories. While the ordering introduces a small bias to the network designs in fronts split between two adjacent categories, the bias has little effect on the evolved distributions.

Evaluating GP mutation operators

Multi-objective problems are more difficult to evaluate than single-objective problems as network designs to these problems cannot be directly and fairly compared using a single scalar value, rather the difference between the network designs is described by a vector. This is a fundamental issue for multi-objective optimisation research and a variety of methods have been explored to overcome this problem, such as weighted average and more commonly Pareto dominance.

The Pareto dominance relation describes the relative quality of two network designs based on their objective vectors. If a network design, a, is shown to be equal or better in quality in all objectives and at least better in one when compared to another network design, b, it is said to dominate b; denoted as $a \prec b$. Likewise, if b is shown to be equal or better in all objectives and better in at least one when compared with a, then b is said to dominate a. If neither a dominates b nor b dominates a then they are said to be mutually non-dominating.

The Pareto dominance relationship provides a method for describing the relationship between two network designs and can be used as a proxy for the improvement of a new child network design compared to its parent. The calculation of difference between two network designs is represented by a scalar value representing the dominance relationship between the two network designs. If the new perturbed network design dominates the parent sampled network design then a difference score of -1 is given (better). If the new perturbed network design is dominated by the sampled network design then a difference score of 1 is given (worse). Otherwise, a difference of zero is given.

A GP evolved mutation operator is evaluated by applying the GP mutation operator to each of three sets of WDN solution samples. The mutation is applied a fixed number of times (q) to each sample in each set to generate q new perturbed network designs per sample. Each new perturbed network design is evaluated on the underlying benchmark WDN design problem used for training and compared to the original sample network design.

The dominance of the perturbed network designs over the original sampled network design is recorded and averaged over all q perturbations. The averaged variance (i.e., average dominance, mutual non-dominance or dominated score) is then averaged over all the sample network designs in each set and used to denote the quality of the mutation operator on that set of sampled network designs. The values are normalised in the range [0, 2]. The objective function used for the GP mutation evaluation is given in Equation (1) below. The term *var* refers to the average variation of the mutated objective values from the original sampled network design objective values. The term *len* (*samples*) is a function which returns the number of sample network designs used to evaluate the GP mutation operator. The term *avg*(*samples*) is a function which returns the average objective value from the sampled network designs.

$$\text{objective} = \begin{cases} \text{var} > 0, & 1 + \left| \dfrac{\text{var}}{\text{len(samples)} - \text{avg(samples)}} \right| \\ \text{var} < 0, & \dfrac{\text{avg(samples)} + \text{var}}{\text{avg(samples)}} \end{cases} \tag{1}$$

EXPERIMENTAL SETUP

An experiment is described in this section which demonstrates the application of the above hyper-heuristic method to the optimisation of EA mutation operators for the WDN design problem. The experiment was designed to demonstrate the feasibility of the proposed method in general terms and not specifically in relation to any one EA method. Rather, the proposed approach is designed to be intentionally agnostic of any one EA and can be used in conjunction with any specialised or a more advanced EA than the ES used herein. A simple EA, in this case an ES, was selected for this experiment as it had relatively few advanced features which may introduce additional dynamics into the results and obfuscate features pertinent to this study.

The experiment is conducted to allow for the comparison of evolved, specialised mutation operators for the WDN design problem against one another and also against a typical operator from the literature for reference, such as a Gaussian mutation. Comparisons with other more advanced optimisation techniques are not conducted as they fall outside of the scope of this study and could not be fairly compared against the evolved operators as many additional

factors, such as the selection strategy, will significantly bias the results. Furthermore, such a study is not necessary as the evolved operators do not 'compete' with other optimisers as they are only components within an EA, rather than an entire stand-alone optimisation method.

The water distribution network design problem

A traditional bi-objective formulation of the WDN design problem was used in this experiment similar to di Pierro *et al.* (2009). The problem was formulated as follows:

$$\text{Minimize cost where cost} = \sum_{i=0 \text{ to } k} (d \times l) \tag{2}$$

$$\text{Minimize head } (h) \text{ deficit } (hd) \text{ where } hd = \sum_{i=0 \text{ to } k} (\min(0,\ h - 30)) \tag{3}$$

The terms k, d, l and h in Equations (2) and (3) refer to the *number of pipes*, *diameter*, *length* and *downstream node head* respectively. The term hd represents the *head deficit* at a pipe's downstream node. The function min (…) returns the minimum value of the two given arguments.

All the networks used in the experiment were arranged as partial expansion problems, where only fixed pipes of the network could be adjusted. The layout and pump operations were fixed. Only pipe diameters were optimised using a fixed set of possible diameters with associated costs per kilometre. For simplicity, the same pipe diameter and associated costs were used which are given below given that the real-world network pipe choices and scaling of costs was similar to those of Hanoi and Anytown.

Training the GP evolved mutation operators

The GP evolved mutation operators were constructed as outlined in the *Method* section. The trees were limited to a depth of 4 – i.e., 3 conditional branches deep with terminals. The GPs were evolved using SPEA2 (Zitzler *et al.* 2002) with a passive archive. The passive archive stored the 100 best mutation operators found during the search. SPEA2 was run for 250 generations with a population of 50. The trees were encoded using a fixed length encoding scheme to enable the use of traditional uniform random mutation and uniform crossover to be applied.

The GP evolved mutation operators were evaluated by inserting them into a $(10 + 10)$-ES (without crossover) (Laumanns *et al.* 2000) and applying the $(10 + 10)$-ES to a training problem for 500 generations over 20 trial runs. The $(10 + 10)$ refers to a size of the parent and child populations. The quality of the GP evolved mutation operator was then evaluated using the method outlined in the *Evaluating GP mutation operators* sub-section of the *Method* section. The GP evolved mutation operators were evaluated on the same training network, Hanoi. The Hanoi network consists of 34 links which connects the 32 nodes and a reservoir. The cost tables for the Hanoi and Anytown benchmark networks are used from the original papers and are available online at http://centres.exeter.ac.uk/cws/.

Testing the GP evolved mutation operators

After evolving the GP evolved mutation operators with SPEA2, the 10 best GP evolved mutation operators stored in the passive archive were compared on a set of test WDN networks. In order to compare the automatically constructed EA mutation operators, they were each inserted into identical $(10 + 10)$-ESs with passive archives. As before, the $(10 + 10)$-ESs did not apply crossover and used elitist selection – basing the performance of the $(10 + 10)$-ESs solely on the efficacy of the mutation operators. Each of the $(10 + 10)$-ESs were run for 2,000 generations and applied for 20 trial runs on each test problem with the results at each generation recorded for every run.

Three networks were used for testing: one benchmark network (Anytown) and two real-world networks. Six pipes were able to be resized in Anytown while 27 and 81 pipes were able to be resized in the two industrial networks. The Anytown network consists of one reservoir, one pumping station, two tanks, 22 nodes and 42 links. For each of the two industrial networks all the pipes for resizing were located within the same area in a single group. We selected the pipes from sub-regions that were mostly self-contained but that were still reasonably well connected to a number of areas in the network. The real-world networks were sourced by one and two reservoirs, respectively. Each of the sub-regions being optimised contained no pumping

stations, other than the largest real-world network contained one tank and associated pump which operated during the two daily peak periods.

Performance measure for comparing mutation operators

Hypervolume (Bader *et al.* 2008) was used to evaluate and compare the selected evolved mutation operators. Hypervolume is a commonly used performance indicator in multi-objective optimisation research which provides a single scalar value for the quality of an optimiser's population (in this case, the ES archive) at each generation of a single run. Hypervolume evaluates a population both in terms of its spread and convergence by measuring the population's coverage of objective space. The scalar hypervolume measure is useful as it allows for information to be obtained about the method's average performance by completing multiple optimisation runs and averaging the hypervolume results from each run. Comparing Pareto front's alone is useful if comparing specific solutions to a specific problem (as is done when discussing the evolved GP operators, see Figure 3). However, when discussing the performance of the GP evolved mutation operators on the WDN design problem in general, the hypervolume measure is more appropriate as it allows the evolved operators to be compared in terms of their expected behaviour on any network, using the selected networks as examples (shown later).

The hypervolume indicator (Bader *et al.* 2008) (which was normalised to 1) was used to monitor the performance of each of the evolved GP evolved mutation operators over all generations during all test optimisation runs. The hypervolume indicator was calculated using random samples drawn from within the objective space as outlined in Bader *et al.* (2008). At each generation, the hypervolume was calculated by finding the number of points which were dominated by each GP evolved mutation operator's current population of candidate network designs – thus, giving an indication of the proportion of space covered by the population and hence quality of the population as a whole. As such, the hypervolume indicator gives a scalar representation of the ratio of objective space dominated by the population. Once a sample set had been generated it was kept and used for all hypervolume calculations on that problem for all algorithms and trials. Each of the GP evolved mutation operators were run 20 times and the hypervolume results averaged to ensure a fair comparison of performance.

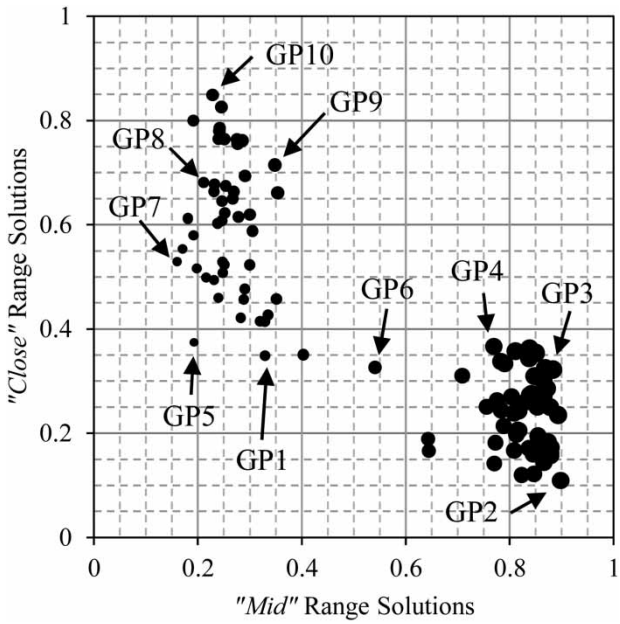

Figure 3 | Scatter plot showing the Pareto optimal GP evolved mutation operators for the bi-objective WDN problem evolved using SPEA2. The 'close' and 'mid' range objectives are shown on the (*x*, *y*) axes and the 'far' objective indicated by point size. All objectives are to be minimised, where smaller point sizes indicate a better objective value.

RESULTS

Evolved mutation operators

The GP evolved mutation operators evolved on the Hanoi training problem using SPEA2 are shown in Figure 3 as a scatter plot of their hyper-heuristic objective values and given in Table 2 for 20 of the evolved mutation operators, including the 10 selected mutation operators. The complete results for all 83 Pareto optimal evolved operators are given in Appendix 1, Table 3 (available online at http://www.iwaponline.com/jh/016/226.pdf).

Each of the evolved mutation operators were evaluated by applying them to three sets of sample network designs from a selected training network (in this case Hanoi) as outlined in the *Method* section. The overall performance of the

Table 2 | Objective values for 20 of the 83 best evolved mutation operators. The top 10 highlighted mutation operators show the objective values for the 10 selected mutation operators indicated in Figure 3 and explored in more detail below. The columns show the objective values for each GP operator on the 'close', 'mid' and 'far' objectives. Additional columns have been included which show the variability of the mutation operators' performance on each objective through the standard deviation of values obtained across the training set

GP evolved mutation operator	Close (mean)	Close (std. dev.)	Mid (mean)	Mid (std. dev.)	Far (mean)	Far (std. dev.)
GP1	0.349	±0.014	0.329	±0.021	0.352	±0.022
GP2	0.109	±0.005	0.898	±0.014	0.576	±0.109
GP3	0.324	±0.1	0.869	±0.013	0.615	±0.194
GP4	0.367	±0.049	0.769	±0.033	0.576	±0.117
GP5	0.374	±0.14	0.193	±0.03	0.29	±0.074
GP6	0.326	±0.093	0.541	±0.155	0.452	±0.135
GP7	0.529	±0.15	0.16	±0.021	0.312	±0.033
GP8	0.681	±0.164	0.211	±0.039	0.376	±0.052
GP9	0.715	±0.044	0.348	±0.1	0.453	±0.178
GP10	0.849	±0.111	0.228	±0.009	0.426	±0.073
GP11	0.358	±0.001	0.811	±0.111	0.595	±0.051
GP12	0.307	±0.061	0.858	±0.083	0.606	±0.146
GP13	0.623	±0.213	0.252	±0.058	0.382	±0.107
GP14	0.167	±0.012	0.809	±0.056	0.546	±0.028
GP15	0.694	±0.152	0.29	±0.002	0.419	±0.001
GP16	0.27	±0.071	0.865	±0.031	0.6	±0.116
GP17	0.494	±0.23	0.231	±0.019	0.339	±0.053
GP18	0.173	±0.009	0.88	±0.021	0.583	±0.107
GP19	0.608	±0.133	0.246	±0.051	0.375	±0.027
GP20	0.763	±0.112	0.275	±0.063	0.429	±0.158

operator on sample network designs from each sample set was used to determine the fitness, or objective quality, of the mutation operator. The performance on best network designs (the 'close' sample set) was used to evaluate the 'close' objective. Similarly, the average and worst quality network design sets were used to evaluate the 'mid' and 'far' objectives respectively.

Two of the objectives are shown on the (x, y) axes for the mutation operator quality on the 'close' and 'mid' range set of network designs used for training. The third objective, assessing mutation operators on network designs 'far' from the Pareto front, is indicated by the size of the points; where smaller point sizes are given for smaller, better objective values on that set of points. Generally, the mutation operators which perform well on the 'close' range network designs objective do not perform well on the 'far' objective, while those that are good in the 'mid' objective tend to perform well on the 'far' objective. The weak correlation between the 'mid' and 'far' objectives can be seen by the general increase in point sizes ('far' objective) as the 'mid' objective values increase.

The evolved mutation operators produce an interesting Pareto front where the GP evolved mutation operators are most commonly specialised for one of the three different objective values. This produces a higher density of evolved solutions at the extremities of the Pareto front with fewer mutation operators producing a good trade-off between all three objectives. Ten GP evolved mutation operators are highlighted on the plot (Figure 3) which represent a range of GP trees and objective values. Of specific interest are GP1, GP5 and GP10, which are shown later in the test WDN optimisation results to produce very different convergence behaviours. Of note are objective values of the GP1 and GP5 mutation operators which are both shown below

to perform well on the test WDN problems as well as obtaining potentially the most favourable trade-off between the three objectives on the training Hanoi problem.

The GP1, 5 and 10 mutation operators are shown in Figure 4. Each of the three mutation operators represent a different class of evolved mutation operator and were selected to illustrate the variety of mutation operators that can be constructed using the multi-objective generative hyper-heuristic method proposed in the *Method* section. The mutation operators range from entirely deterministic operations in GP10 through to the entirely random GP5. GP1 provides a mix of these two types of operation through a combination of random mutation and deterministic, domain-specific operations.

GP1:

```
if rand < 0 then
 if rand > 0.5 then
 increase diameter by 1
 else
 increase diameter by 3
 end if
else
 if rand > 0.5 then
 if rand > 0.3 then
 increase diameter by 1
 else
 decrease diameter by 1
 end if
 else
 if rand > 0.5 then
 if upstream_diameter >
     current_diameter and
     downstream_diameter =
     current_diameter then
increase diameter by 1
else
if downstream_diameter <
      current_diameter then
decrease diameter by 1
else
increase diameter by 2
end if
end if
else
if downstream_head > 40m then
decrease diameter by 1
else
if downstream_head < 30m then
increase diameter by 2
end if
end if
end if
end if
end if
```

GP5:

```
if rand > 0.5 then
 if rand > 0.5 then
 increase diameter by 1
 else
 decrease diameter by 1
 end if
else
 if rand > 0.5 then
 if rand > 0.5 then
 increase diameter by 3
 else
 increase diameter by 2
 end if
 else
 decrease diameter by 2
 end if
end if
```

GP10:

```
if upstream_diameter >
current_diameter and
downstream_diameter =
current_diameter then
 increase diameter by 1
else
 if downstream_diameter <
current_diameter then
 decrease diameter by 1
 else
 increase diameter by 2
 end if
end if
```

Figure 4 | Pseudo-code for the GP1, GP5 and GP10 mutation operators evolved using SPEA2 on the Hanoi training problem, where *rand* refers to a randomly generated real number in the range [0, 1].

GP5

One of the more common classes of mutation operators evolved by the generative hyper-heuristic method was that of entirely random mutations, such as GP5. This result suggests that even with the potential for including domain-specific information, such as pipe smoothing, into the GP evolved mutation operator operations the optimisation process of EAs can accommodate and promote the use of entirely random mutation in its stochastic search. Indeed, it is important to note that the GP5 mutation operator is the equivalent of a single-peaked mutation operator, in this case a Gaussian, and so provides a good representation for these more traditional mutation operators. The nesting of the larger mutations under subsequent 50:50 random choices reduces the likelihood of applying large perturbations compared to the smaller one pipe size step mutations which will be applied in approximately 50% of all mutations whereas the two pipe size steps will be applied to only 25% of mutations and so on.

It should be noted that the evolved GP5 operator is effectively a Gaussian mutation distribution and, as such, identical to a manually tuned mutation distribution which would normally be compared against. For this reason, GP5 is used below in Figure 5 as a suitable proxy for a typical operator for comparative purposes, rather than replicating results with an effectively identical Gaussian mutation operator. In particular, this mutation operator is of interest for three reasons: (1) it demonstrated the method could find existing well-used operators; (2) it showed that existing typical operators were very competitive; and (3) it provided a typical operator for benchmarking and comparison.

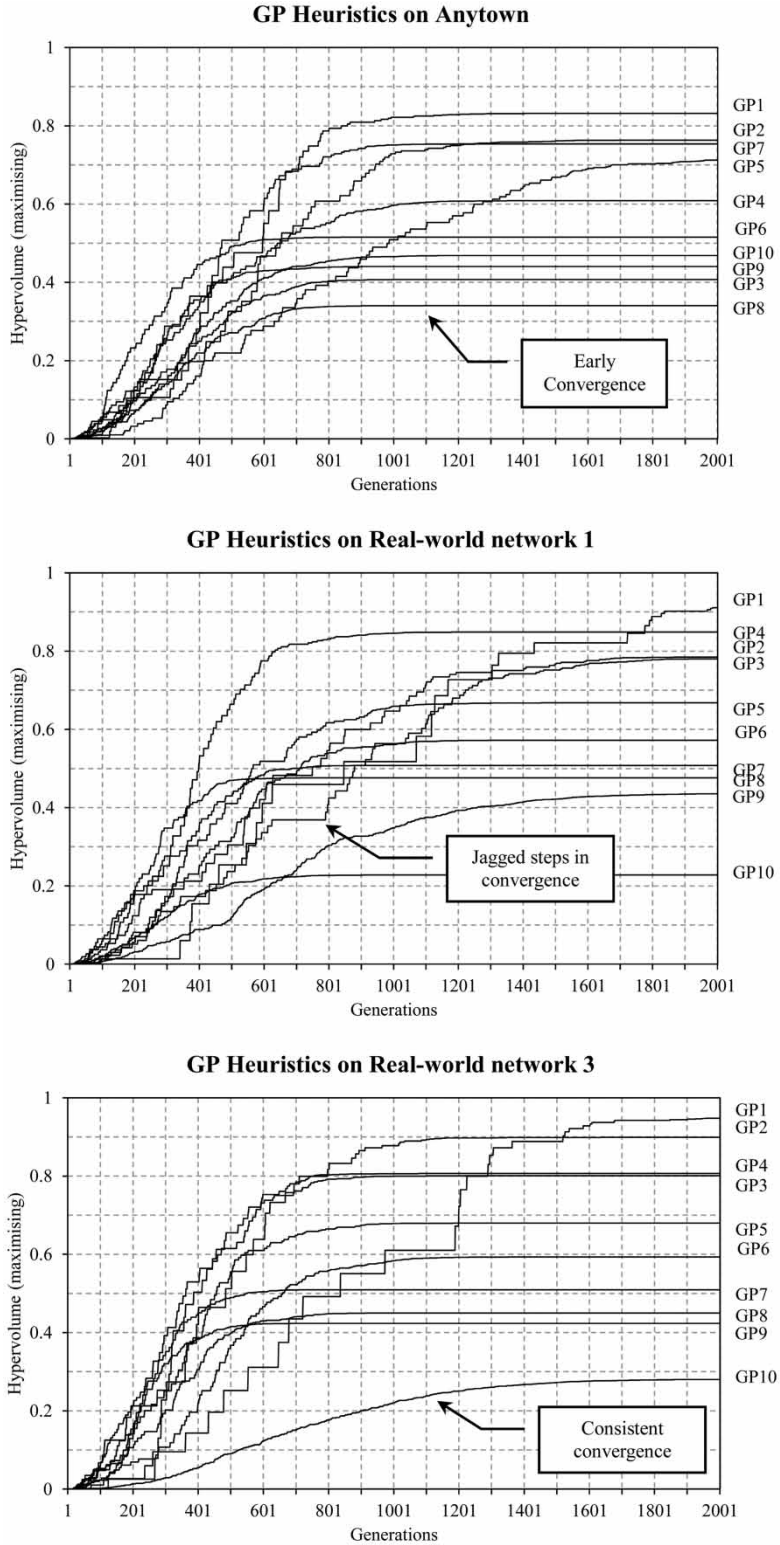

Figure 5 | Hypervolume results for 10 selected GP evolved mutation operators on the Anytown benchmark and real-world networks 1 and 2. The GP evolved mutation operators are labelled on each plot (in order) adjacent to their respective trend lines.

GP10

The GP10 mutation operator provides the clearest example of an entirely domain-specific mutation operator. The generative hyper-heuristic method proposed above was designed to evolve mutation operators which contained some domain-specific information learned in the search (such as that in GP1) but it was not expected that mutation operators, such as GP10, would be evolved that perform highly specialised tasks. The mutation operator effectively applies a pipe smoothing operation by averaging the pipe size between the upstream and downstream pipes, increasing the pipe size to match the upstream pipe (if it is larger), or increasing the pipe size above the downstream node (increasing the upstream capacity). The random application of the mutation operator to pipes in the network generates a seemingly random but overall smoothing effect after a number of applications, where the main supplying pipes are increased in size and the downstream nodes reduced in size. As will be shown later, the deterministic nature of this mutation operator means that its search capacity is significantly limited compared to those mutation operators with random mutation elements but could, in combination with random mutation operators, provide a useful function in producing sensible WDN designs with well-formed pipe diameter properties.

GP1

The GP1 mutation operator is an interesting example of random mutation that is biased by network design-specific features and so encodes some domain-specific knowledge – providing both pipe smoothing and demand deficit correction operations. This mutation operator is one of the most complex evolved in this study which accommodates the biased random search with the two specialised functions. As is shown later, the combination of these features enables the mutation operator to outperform many of the other mutation operators and consistently perform better than the more traditional mutation operator on all the test problems.

It should also be noted that part of the mutation operator is effectively a 'dead branch' which is redundant as it will never be used and should be trimmed if the mutation operator were to be coded for more permanent application and inclusion in a meta- or hyper-heuristic algorithm. The remainder of the GP is split by a random branching which either applies a random mutation or the 'specialist function' branch of the GP. This part of the GP is again split by a random branch which differentiates between the 'smoothing' operation and the 'excess/deficit correction' operation. It should be noted that the mutation operator has a greater tendency to increase pipe sizes as the random mutation is positively biased.

Comparing the evolved mutation operators

Of the evolved mutation operators on the Hanoi training problem, the 10 selected mutation operators (highlighted in Table 2) were each tested on the Anytown benchmark network and two real-world networks with theoretical expansion options. The results from these optimisation runs are given in Figure 5 which shows the average hypervolume (Bader *et al*. 2008) trends of each of the mutation operators on the bi-objective formulation of the WDN design problem. As the problem is bi-objective, the hypervolume indicator (to be maximised) was used to indicate the convergence of each of the optimisers; averaged over the 20 trial optimisation runs. As explained in the *Performance measures for comparison* sub-section, the hypervolume indicator measures the population's coverage of the objective space – the larger the hypervolume score the more of the objective space that is dominated by the population and the closer the population is to the Pareto front. Hypervolume is ideal for this type of experimental study as the true Pareto fronts for each of the instances of this problem are unknown and not needed by the indicator to provide a comparative scoring of each of the mutation operators. The hypervolume results are normalised in the range [0, 1] where 1 indicates complete coverage of the objective space and 0 indicates no coverage. The Anytown and real-world networks are shown in Figure 5.

Network 'difficulty'

The results from the GP evolved mutation operators, especially GP10 which represents the traditional, unbiased random mutation, indicate that the Anytown benchmark

network is easier to optimise than the two selected real-world networks with all the mutation operators obtaining reasonably good hypervolume results. Even the GP1 mutation operator plateaus on this problem and converges early in the search. The real-world network 2 stimulates the widest early convergence of all the problems with all the mutation operators (excluding GP1 and GP2) converging before 1,000 generations. This suggests the problem encourages convergence on local optima and that the network has a number of deceptive fronts which discourages the $(10 + 10)$-ESs from continuing to explore the optimisation search space.

Comparing mutation operators

A set of interesting features are shown by annotations on the plots illustrating the results in Figure 5. These features are described more fully below.

- **Final generation results (rankings):** Of all the mutation operators, GP1 is consistently the best performing mutation operator over all the test problems. The GP10 mutation operator produces average results on the Anytown network but obtains the worst results on the real-world networks – limited by its fixed mutation operations. It is also interesting to note that the mutation operators with better 'mid' and 'far' objective results from the training evaluations converge earlier than those which perform better on the 'close' objective which tend to converge more slowly but eventually achieve better final generation results. The more traditional mutation operator, GP5, consistently obtains the fourth or fifth best result and is a good average performing mutation operator on these test networks. This is to be expected as the mutation operator enables a reasonable guided random search through the standard ES selection mechanism but fails to take advantage of the domain-specific learning which is encapsulated in the GP1, 2, 10 and other mutation operators.
- **Early convergence (flat-lining):** One of the most apparent problems with the mutation operators' performance results is the GP evolved mutation operators' tendency to converge early on sub-optimal results. This is shown by a flat-line in hypervolume results, which is most evident on the Anytown network. Early convergence is a significant problem in meta-heuristic optimisers and so the more robust GP1 and GP2 mutation operators are very favourable mutation operators as they both appear to continue to converge for a longer period in the search. The behavioural tendency to increase the pipe diameters in the GP1 mutation operator means that it converges more slowly than the other mutation operators but, importantly, allows it to continue exploring different configurations throughout the search and potentially accounts for its superior results compared to the other algorithms. However, the early convergence of the more deterministic mutation operators, like GP10, could be beneficial in cases where reasonable network designs to a problem are desired at a minimal cost; i.e., with a minimal number of evaluations. The mix of behavioural traits is also beneficial to meta-optimising methods like selective hyper-heuristics which can 'pick and choose' the mutation operators and apply both the slower converging, more explorative mutation operators in combination with the faster converging exploitative mutation operators to a greater effect that applying them individually (McClymont *et al.* 2012b).
- **Noise (jagged steps):** Both GP1 and GP2 produce 'jagged' convergence trends. This feature is produced as a result of the mutation operators' variable performance on the optimisation problem and sudden advances in their populations. This feature also indicates (which was confirmed in the results data) that there is a higher variance in the optimisation runs compared to mutation operators with more consistent performance, such as GP10, which produce smoother trend lines. It is interesting that these two mutation operators, which both have the largest GP trees, are the most variable in their optimisation performance, also achieve the highest average hypervolume results.
- **Over-fitting:** One concern when using machine learning techniques to optimise the performance of a system, such as an EA's mutation operator for the WDN design problem is over-fitting; the effect by which the results are highly tuned to the training data but not general enough to perform well on test or practical data. The results from the experiment described above show how some of the evolved mutation operators were more robust on the larger test networks than others and indicated that some of the evolved mutation operators were overly

tuned to the training networks. Indeed, the GP10 mutation operator illustrates how evolved mutation operators can 'over-fit' to training problems, performing well on the smaller networks but not scaling well on the larger 81 pipe industrial network. The GP10 mutation operator therefore would not be a suitable candidate for reuse in practical optimisation studies. This study reinforces the point that tuned, tailored or optimised search algorithms must be qualified on test networks prior to application to ensure such over-fitting does not occur or is not carried through to practical use.

CONCLUSION

This paper presents a novel GP evolved decision tree generative hyper-heuristic method which is used to automatically build novel mutation operators for the bi-objective WDN design problem. Many of the GP decision tree-based mutation operators utilise domain knowledge in the form of features like downstream node head conditions to inform the type of mutation to apply to each selected pipe. The method is applied to and trained on the Hanoi benchmark problem with the GP evolved mutation operators evolved using SPEA2. The 10 varied GP evolved mutation operators from the best evolved mutation operators were compared on the Anytown benchmark and two real-world networks. The results demonstrated how the mutation operators varied in behaviour and produced different convergence characteristics. Furthermore, the results also showed how some of the evolved mutation operators were more robust on the larger test networks. Indeed, the GP10 mutation operator illustrates how evolved mutation operators can 'over-fit' to training problems, performing well on the smaller networks but not scaling well on the larger 81 pipe industrial network. However, the results also demonstrated the potential of the method with one mutation operator (GP1) outperforming consistently, obtaining the best final generation result on all the test networks. Interestingly, GP1 converges less quickly that many of the GP evolved mutation operators which suggests it has a better exploration capacity, and thus better results, which is supported by the analysis of the GP tree.

ACKNOWLEDGEMENTS

This research was supported by an EPSRC CASE Studentship award (Grant No. CASE/CNA/07/100) and EPSRC project (Grant No. EP/K000519/1) in conjunction with Mouchel Ltd.

REFERENCES

Bader, J., Deb, K. & Zitzler, E. 2008 Faster hypervolume-based search using Monte Carlo sampling. *Mult. Criteria Decis. Mak. Sust. Energy Transp. Syst.* **634**, 313–326.

Burke, E. K., Hyde, M. R. & Kendall, G. 2006 Evolving bin packing heuristics with genetic programming. In: *Parallel Problem Solving from Nature - PPSN IX, Springer Lecture Notes in Computer Science* (T. P. Runarsson, H.-G. Beyer, E. Burke, J. J. Merelo-Gurevos, L. D. Whitley & X. Yao, eds). vol. 4193, Springer-Verlag, Reykjavik, Iceland, pp. 860–869.

Burke, E. K., Hyde, M., Kendall, G., Ochoa, G., Ozcan, E. & Woodward, J. 2009 A classification of hyper-heuristics approaches. In: *Handbook of Metaheuristics* (M. Gendreau & J.-Y. Potvin, eds). International Series in Operations Research & Management Science: Springer, Berlin.

Burke, E. K., Hyde, M., Kendall, G., Ochoa, G., Ozcan, E. & Qu, R. 2010 Hyper-heuristics: A Survey of the State of the Art. Computer Science Tech. Rep., University of Nottingham, Nottingham.

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

Deb, K., Agrawal, S., Pratab, A. & Meyarivan, T. 2000 A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In: *Proceedings of the Parallel Problem Solving from Nature VI Conference* (M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo & H.-P. Schwefel, eds) Lecture Notes in Computer Science No. 1917. Springer, Paris, France, pp. 849–858.

di Pierro, F., Khu, S.-T., Savić, D. & Berardi, L. 2009 Efficient multi-objective optimal design of water distribution networks on a budget of simulations using hybrid algorithms. *Environ. Modell. Softw.* **24** (2), 202–213.

Goldberg, D. E. 1989 *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Publishing Company, Reading, MA.

Keedwell, E. & Khu, S.-T. 2005 A hybrid genetic algorithm for the design of water distribution networks. *Eng. Appl. Artif. Intell.* **18** (4), 461–472.

Koza, J. 1992 *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA.

Laumanns, M., Zitzler, E. & Thiele, L. 2000 A unified model for multi-objective evolutionary algorithms with elitism. *Proceedings of the 2000 Congress on Evolutionary Computation*, La Jolla, CA, 1, pp. 46–53.

McClymont, K., Keedwell, E., Savić, D. & Randall-Smith, M. 2012a Automated construction of fast heuristics for the water distribution network design problem. *Proceedings of the 10th International Conference on Hydroinformatics (HIC 2012)*, Hamburg, Germany.

McClymont, K., Keedwell, E., Savić, D. & Randall-Smith, M. 2012b A general multi-objective hyper-heuristic for water distribution network design with discolouration risk. *J. Hydroinform.* (in press).

Raad, D., Sinske, A. & van Vuuren, J. 2010 Multiobjective optimization for water distribution system design using a hyperheuristic. *J. Water Resour. Plann. Manage.* **136** (5), 592–596.

Rossman, L. A. 2000 EPANET 2 Users Manual, United States Environment Protection Agency, Technical Report EPA/600/R-00/057.

Savić, D. A. & Walters, G. A. 1997 Genetic algorithms for least-cost design of water distribution networks. *J. Water Res. Pl.-ASCE* **123** (2), 67–77.

Simpson, A. R., Dandy, G. C. & Murphy, L. 1994 Genetic algorithms techniques for pipe optimization. *J. Water Resour. Plann. Manage.* **120** (4), 423–443.

Yates, D. E., Templeman, A. B. & Boffey, T. B. 1984 The computational complexity of the problem of determining least capital cost designs for water supply networks. *Eng. Optim.* **7** (2), 142–155.

Zitzler, E., Laumanns, M. & Thiele, L. 2002 SPEA2: Improving the Strength Pareto Evolutionary Algorithm For Multiobjective Optimization. IEEE Congress on Evolutionary Computation, Honolulu, HI, USA.

First received 21 November 2012; accepted in revised form 7 May 2013. Available online 7 June 2013