Ann Oper Res (2013) 206:527–555
DOI 10.1007/s10479-013-1322-7

A simulated annealing based genetic local search algorithm for multi-objective multicast routing problems

Ying Xu · Rong Qu · Renfa Li

Published online: 13 February 2013

Abstract This paper presents a new hybrid evolutionary algorithm to solve multi-objective multicast routing problems in telecommunication networks. The algorithm combines simulated annealing based strategies and a genetic local search, aiming at a more flexible and effective exploration and exploitation in the search space of the complex problem to find more non-dominated solutions in the Pareto Front. Due to the complex structure of the multicast tree, crossover and mutation operators have been specifically devised concerning the features and constraints in the problem. A new adaptive mutation probability based on simulated annealing is proposed in the hybrid algorithm to adaptively adjust the mutation rate according to the fitness of the new solution against the average quality of the current population during the evolution procedure. Two simulated annealing based search direction tuning strategies are applied to improve the efficiency and effectiveness of the hybrid evolutionary algorithm. Simulations have been carried out on some benchmark multi-objective multicast routing instances and a large amount of random networks with five real world objectives including cost, delay, link utilisations, average delay and delay variation in telecommunication networks. Experimental results demonstrate that both the simulated annealing based strategies and the genetic local search within the proposed multi-objective algorithm, compared with other multi-objective evolutionary algorithms, can efficiently identify high quality non-dominated solution set for multi-objective multicast routing problems and outperform other conventional multi-objective evolutionary algorithms in the literature.

Keywords Multi-objective genetic local search · Simulated annealing · Multicast routing

Y. Xu (✉) · R. Li
College of Information Science and Engineering, Hunan University, Changsha, Hunan, 410082, China
e-mail: hnxy@hnu.edu.cn

R. Qu
The Automated Scheduling, Optimisation and Planning (ASAP) Group, School of Computer Science, The University of Nottingham, Nottingham, NG8 1BB, UK
e-mail: rxq@cs.nott.ac.uk

1 Introduction

1.1 The multicast routing problem (MRP)

Multicast is a telecommunication technique that simultaneously transfers information (IP datagrams) from a source to a group of destinations in communication networks. Compared to unicast, which relies on the point-to-point transmission, multicast is a more efficient solution which utilises the parallelism in networks. In this work, we consider the Multicast Routing Problem (MRP), which concerns finding the spanning tree while optimising the resource usage within the network. Due to the increasing development of numerous multicast network applications including distance learning, E-commerce and video/audio conferencing, the MRP has become one of the key problems in multimedia telecommunications and received increasing research attention in operational research.

Real world multicast applications generally have some Quality of Service (QoS) parameters or constraints and objectives. For example, an important and common QoS constraint in multicast routing applications is the bounded end-to-end delay. That is, messages must be transmitted from the source to destinations via the multicast tree within a certain limited time; otherwise most customers would cancel their requests. The efficient allocation of network resources to satisfy different QoS requirements, for example, minimising the cost of transmission via the multicast tree, is the primary goal of QoS-based multicast routing.

It is well known that the Steiner tree problem (Hwang and Richards 1992), the underlying model of MRPs, is a NP-hard combinatorial optimisation problem (Garey and Johnson 1979). It has also been proved that finding a feasible multicast tree with two independent path constraints is NP-hard (Chen and Nahrestedt 1998). The constrained Steiner tree problem under various QoS constraints is thus also NP-hard (Kompella et al. 1993). This makes the complex QoS based MRPs one of the challenging optimisation problems. Over the past decade, the problem has attracted increasing attention from the meta-heuristic research community in both computer communications and operational research (Diot et al. 1997; Yeo et al. 2004; Oliveira and Pardalos 2005). A large amount of investigations on meta-heuristic algorithms exist in the literature (Haghighat et al. 2004; Kun et al. 2005; Skorin-Kapov and Kos 2006; Zahrani et al. 2008; Qu et al. 2009). However, at the early stage, the MRPs have been mainly defined and solved as a single-objective optimisation problem subject to certain QoS constraints, i.e. to minimise the tree cost subject to a maximum end-to-end delay restriction.

With a range of inter-dependent and conflicting multiple QoS objectives and constraints (e.g. cost, delay, bandwidth, link utilisation, delay variation, packet loss ratio and hop count) in real world applications, the QoS-based MRPs can be more appropriately defined as multi-objective optimisation problems. Recent multi-objective optimisation algorithms for MRPs have been investigated concerning more realistic constraints and objectives.

1.2 Related work

A recent survey in Fabregat et al. (2005) has reviewed a variety of multi-objective multicast routing algorithms. In Table 1, we categorise meta-heuristic algorithms in the literature according to the objectives and constraints considered in problems, where a single multicast tree is constructed. It can be seen that different meta-heuristics, e.g. genetic algorithm, ant colony algorithm, artificial immune algorithm and particle swarm optimisation, have been investigated for multi-objective MRPs with various objectives. Due to the nature of multi-objective optimisation, where a set of alternative solutions is considered, it is not surprising

to see that genetic algorithms, one of the mostly studied population based algorithms, have been adapted in most multi-objective multicast routing algorithms.

Roy et al. (2002) adapt the widely studied multi-objective NSGA (Non-dominated Sorting based Genetic Algorithm) (Srinivas and Deb 1994) to simultaneously optimise end-to-end delay, bandwidth and residential bandwidth utilisation rather than combining them into a single weighted sum objective function for wireless network routing problems. Due to the user mobility and uncertainties in wireless cellular networks, Roy and Das (2004) employ a fast and efficient QoS-based mobile multicast routing protocol based on multi-objective genetic algorithms for dynamic MRPs. In Crichigno and Baran (2004a, 2004b), two multi-objective evolutionary algorithms (MOEA) with an external population of Pareto optimal solutions have been proposed based on the strength Pareto evolutionary algorithm (Zitzler and Thiele 1999). Experimental analysis shows that MOEA1 with a binary tournament selection outperforms MOEA2 with a roulette wheel selection. Other multi-objective genetic algorithms include Koyama et al. (2004), which optimise the cost and delay of the multicast tree and Cui et al. (2003), which develop the algorithm based on Pareto dominance.

A variety of other population based meta-heuristics also appear in the multi-objective multicast routing literature. Two ant colony optimisation algorithms in Diego and Baran (2005) have shown to find more non-dominated solutions than the MOEA2 algorithm in Crichigno and Baran (2004b) on benchmark problems with different features using same computational expenses. Wang et al. (2006) propose a QoS multicast routing model based on an artificial immune system with a gene library and a clone search operator to search for better solutions. The algorithm can effectively identify a set of Pareto optimisation solutions compromising multiple QoS objectives. Particle swarm optimisation has also been investigated in Li et al. (2007) to enhance selected elite individuals before generating the next generation within a hybrid multi-objective genetic algorithm.

In the recent multi-objective optimisation research, various simulated annealing (SA) approaches (Czyzzak and Jaszkiewicz 1998; Ehrgott and Gandibleux 2000; Landa-Silva et al. 2004; Li and Landa-Silva 2008; Martins and Costa 2010; Xu and Qu 2011) have been successfully applied for different multi-objective optimisation problems. Annealing is known as a thermal process, where a solid is melted by increasing its temperature and then followed by a slow progressive temperature decrease aiming at recovering a solid state of lower energy. The SA algorithm simulates the physical annealing process to solve optimisation problems, where a solution corresponds to a state of the physical system and the fitness value of a solution corresponds to the energy of a state. It has the ability in this process to escape from local optima by visiting worse neighbouring solutions, and shows to be very effective when exploring the search space of complex multi-objective optimisation problems. Meanwhile, genetic local search algorithms (Ishibuchi and Murata 1998; Jaszkiewicz 2002; Mendoza et al. 2010) have been investigated for different multi-objective optimisation problems. Due to the ability of local search to find local optima effectively over a relatively small part of the search space, genetic local search algorithms have shown to be very suitable for solving complex multi-objective optimisation problems. Refer to Beume et al. (2007), Zhang and Li (2007), Li and Zhang (2009), Bader and Zitzler (2011), Ishibuchi et al. (2011), etc., for some recent multi-objective optimisation algorithms.

To our knowledge, there is no investigation on hybridising SA with genetic local search algorithms to multi-objective MRPs. The only two recent relevant algorithms that we are aware of are applied to single objective MRPs. In Zahrani et al. (2008), a genetic local search utilises a logarithmic simulated annealing in a pre-processing step to analyse the landscape of a single objective MRP subject to multiple constraints in a group multicast scenario. Another genetic simulated annealing algorithm has been proposed in Zhang et al. (2009)

Table 1 Multi-objective multicast routing meta-heuristics, categorised by objectives and constraints considered, ordered by the year of publication (MOEA: multi-objective evolutionary algorithm; MOGA: multi-objective genetic algorithm)

Meta-heuristic algorithms	Objectives							Constraints	
	Cost	Delay	Average delay	Link utilisation	Bandwidth	Packet loss	Delay jitter	Link capacity	Delay
MOEA based on NSGA (Roy et al. 2002)		x		x	x				
MOGA (Cui et al. 2003)		x			x	x			x
QoS-based mobile multicast routing protocol based on MOGA (Roy and Das 2004)		x		x	x				
MOEA1 (Crichigno and Baran 2004a) and MOEA2 (Crichigno and Baran 2004b) based on the strength Pareto evolutionary algorithm	x	x		x				x	
MOGA (Koyama et al. 2004)	x	x							
Multi-objective Ant colony optimisation systems (Diego and Baran 2005)	x	x		x				x	
Multi-objective immune algorithm (Wang et al. 2006)	x	x			x			x	x
Hybrid genetic algorithm and particle swarm optimisation (Li et al. 2007)	x	x		x				x	
MOGA with Fuzzy based parameter setting for QoS multicasting in wireless ad hoc networks (Rai et al. 2010)		x				x	x		
MOGA for QoS multicasting in wireless ad hoc networks (Huang and Liu 2010)		x			x	x	x		
Evolutionary multi-objective simulated annealing (Xu and Qu 2011)	x	x	x	x				x	

for delay jitter bounded least-cost MRP with bandwidth and delay constraints. Simulated annealing is used to compute the probability of accepting newly generated solutions. These two methods have shown to be effective in solving single objective MRPs.

In our recent work (Xu and Qu 2011), simulated annealing strategies have shown to be effective in driving a population of solutions towards the Pareto front of MRPs with four objectives. However, together with the variable neighbourhoods specially designed for MRPs, they have shown to have much less impact on the algorithm performance. In solving complex problems such as MRPs with special solution structure, specifically designed neighbourhood operators with regard to problem features have shown to be highly effective on improving algorithm performance.

In this work, motivated by the efficiency of both the simulated annealing strategies and genetic local search, we develop the first multi-objective simulated annealing based genetic local search (MOSAGLS) algorithm to solve the multi-objective MRPs. The hybrid MOSAGLS algorithm aims to combine the strengths of both Simulated Annealing and Genetic Algorithm. On the one hand, genetic algorithms have been widely used for solving multi-objective optimisation problems in the literature due to their population-based nature and the ability to simultaneously search different regions of a solution space. On the other hand, Simulated Annealing has the character of escaping from local optima by intelligently accepting worse solutions thus addressing the issue of premature convergence of GAs. In our proposed MOSAGLS, a new genetic local search with genetic operators which are specially designed for MRPs has been developed to simultaneously minimise five real life objectives, namely (1) the cost, (2) the maximum end-to-end delay, (3) the maximum link utilisation, (4) the average delay and (5) the delay variation of the multicast tree. MOSAGLS evolves by using SA-based strategies within the genetic evolutionary process to generate non-dominated solutions. A new SA-based adaptive mutation probability is also used to improve the performance of the hybrid algorithm. The impact of the SA based strategies and the local search within the genetic evolution has been investigated within this new hybrid algorithm.

The rest of the paper is organised as follows. In Sect. 2, the multi-objective MRP is formally defined. Sections 3 and 4 present the proposed hybrid algorithm and evaluate its performance by experimental results. Finally, Sect. 5 concludes the paper.

2 The multi-objective MRP

The multi-objective optimisation problem with n decision variables, k objective functions and q restrictions can be defined as follows (Deb 2005):

$$\begin{aligned} &\text{Optimise} && F(x) = \big(f_1(x), f_2(x), \ldots, f_k(x)\big) \\ &\text{s.t.} && e(x) = \big(e_1(x), e_2(x), \ldots, e_q(x)\big) \geq 0 \end{aligned} \tag{1}$$

where

X: the decision space of feasible regions in the solution space.
x: a vector of decision variables or a solution, $x = (x_1, x_2, \ldots, x_n) \in X$.
$f_i(x)$ $(i = 1, \ldots, k)$: objective functions with k objectives to be optimised.
$F(x)$: the image of x in the k-objective space given by the vector of k objective functions $f_i(x)$.
$e_i(x)$ $(i = 1, \ldots, q)$: the set of restrictions which determines the set of feasible solutions.

Multi-objective optimisation generally concerns a set of trade-off optimal solutions, none of which can be considered superior to the others in the search space when all objectives

are taken into consideration. The set of all these Pareto-optimal solutions in X is called the ***Pareto-optimal Set***.

To model the general MRP, we denote a communication network as a directed graph $G = (V, E)$ with $|V| = n$ nodes and $|E| = l$ links. The following notations are used in the rest of the paper:

$(i, j) \in E$: the link from node i to node j, $i, j \in V$.
$c_{ij} \in R^+$: the cost of link (i, j).
$d_{ij} \in R^+$: the delay of link (i, j).
$z_{ij} \in R^+$: the capacity of link (i, j), measured in Mbps.
$t_{ij} \in R^+$: the current traffic of link (i, j), measured in Mbps.
$s \in V$: the source node of a multicast group.
$R \subseteq V - \{s\}$: the set of destinations of a multicast group.
$r_d \in R$: the destinations in a multicast group.
$|R|$: the cardinality of R, i.e. the number of destinations, also called group size.
$\phi \in R^+$: the traffic demand (bandwidth requirement) of a multicast request, measured in Mbps.
$T(s, R)$: the multicast tree with the source node s spanning all destinations $r_d \in R$.
$p_T(s, r_d) \subseteq T(s, R)$: the path connecting the source s and a destination $r_d \in R$ in the multicast tree T.
$d(p_T(s, r_d))$: the delay of path $p_T(s, r_d)$, given by $d(p_T(s, r_d)) = \sum_{(i,j)\in p_T(s,r_d)} d_{ij}$, $r_d \in R$.

Based on the above definitions, a multi-objective MRP can then be formulated as a multi-objective optimisation problem. In this paper, we consider the multi-objective MRP with more objectives than those defined in our previous work Xu and Qu (2011) and in Crichigno and Baran (2004a). The problem is to find a multicast tree while minimising the values of the following five objectives:

- The cost of the multicast tree:

$$C(T) = \phi \cdot \sum_{(i,j)\in T} c_{ij} \tag{2}$$

- The maximal end-to-end delay of the multicast tree:

$$DM(T) = \mathrm{Max}\big\{d\big(p_T(s, r_d)\big)\big\}, \quad r_d \in R \tag{3}$$

- The maximal link utilisation:

$$\alpha(T) = \mathrm{Max}\left\{\frac{\phi + t_{ij}}{z_{ij}}\right\}, \quad (i, j) \in T \tag{4}$$

- The average delay of the multicast tree:

$$DA(T) = \frac{1}{|R|}\sum_{r_d\in R} d\big(p_T(s, r_d)\big) \tag{5}$$

- Delay variation of the multicast tree:

$$DV(T) = \mathrm{Max}\big\{d\big(p_T(s, r_d)\big)\big\} - \mathrm{Min}\big\{d\big(p_T(s, r_j)\big)\big\}, \quad r_d,\ r_j \in R \tag{6}$$

Objective (2) aims to minimise the cost occurred as the multicast tree T occupies certain required bandwidth on links in the network. Objective (3) minimises the maximal delay time of sending the data via the multicast tree so that they arrive all destinations within a shortest bounded time. Objective (4) tries to minimise the maximal link utilisation, i.e. traffic demand

over the available bandwidth on the links. Objective (5) minimises the average delay time of sending the data so they arrive all $|R|$ destinations in the shortest average time. Objective (6) minimises the delay variation of the multicast tree, which is defined as the difference between the maximum and minimum delays among all the path delays from the source to all destinations. Note that objective (3) concerns the maximal delay within the multicast tree, while objectives (5) and (6) minimise the average delay and the delay variation, respectively, thus concerning the delay to all destinations in the network. These five objectives have some correlations. For example, the delay-related objectives (2), (4) and (5) (Eqs. (3), (5) and (6)) which are dependent on the delays from the source to destinations in the tree are strongly correlated. The cost of the multicast tree, i.e., Objective (1) conflicts with these delay-related objective (2), (4) and (5), since the decrease of the tree cost normally brings the increment of delays. The link utilisation conflicts with the tree cost and the delay-related objectives, since the decrease of the link utilisation causes the increase of the cost and delays. As indicated by the literature, these objectives represent the most common requirements in communications. It remains interesting future work to formulate a wider range of various objectives based on the above defined problem for different applications with specific requirements.

In communication networks, the total bandwidth of datagrams on a link must not exceed the limited bandwidth available. Hence, the total traffic on link (i, j), i.e. the traffic demand ϕ of a multicast request plus the current traffic t_{ij} is subject to the link capacity z_{ij}:

$$\phi + t_{ij} \leq z_{ij}, \quad \forall (i, j) \in T(s, R) \tag{7}$$

Due to the complex real world constraints in multi-objective MRPs, the search space of such problems becomes highly restricted and unpredictable (Xu and Qu 2012). This demands more efficient and effective optimisation techniques to traverse the search space of such problems with many local optimal solutions and disconnected regions of feasible solutions.

3 The simulated annealing based multi-objective genetic local search (MOSAGLS) algorithm

The proposed multi-objective simulated annealing based genetic local search (MOSAGLS) evolves by using simulated annealing based strategies within the genetic evolutionary process concerning non-dominated solutions with regard to the five objectives defined. Figure 1 shows the flowchart of our proposed MOSAGLS algorithm, details presented in the following subsections.

In MOSAGLS, the initial population of multicast trees is randomly generated. During the evolution, parent solutions are chosen to produce child trees by using the defined crossover and mutation operators. A local search is then applied to the generated child tree to produce a new improved tree. An external solution set *NDS* is maintained to record the non-dominated solutions obtained during the evolution. The MOSAGLS stops after a certain computational time, or the temperature in the SA drops to the final temperature. The *NDS* after the evolution is finished is output as the final results. More details of the genetic local search algorithm are given in Sect. 3.2.

The proposed MOSAGLS evolves by using the SA strategies to adaptively set the mutation rate, to guide the search directions and to make decision of solution acceptance. A temperature is defined and decreased through generations. Firstly, after crossover, mutation is carried out based on an adaptive rate according to both the current temperature and the fitness of the offspring and current population. Secondly, each solution in the population is associated with a random weight vector. This vector, together with the temperature, takes

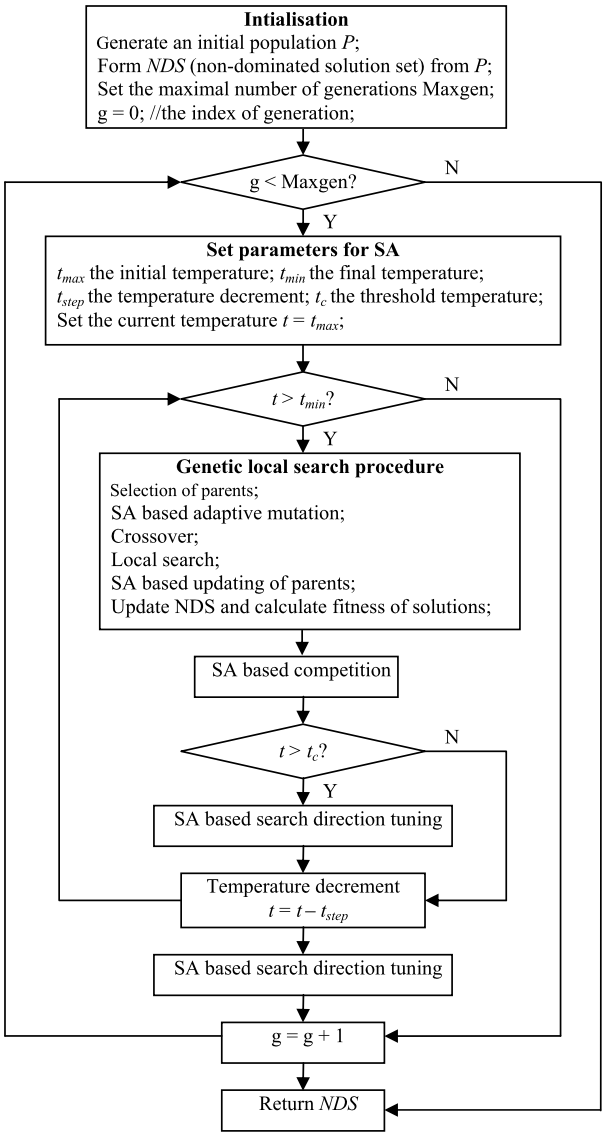


Fig. 1 The flowchart of the proposed MOSAGLS algorithm

part in the solution acceptance and the tuning of search directions. That is, the newly generated trees replace the selected parent based on a probability calculated using the weight vector and the current temperature. When the temperature is decreased to below a threshold, the weight vector is modified to tune the search directions. More details of the SA strategies are given in Sect. 3.3.

3.1 The representation of the multicast tree

In the proposed MOSAGLS algorithm, we adopt the encoding method in Fabregat et al. (2005) to represent the solutions (multicast trees) for MRPs in both the genetic local search

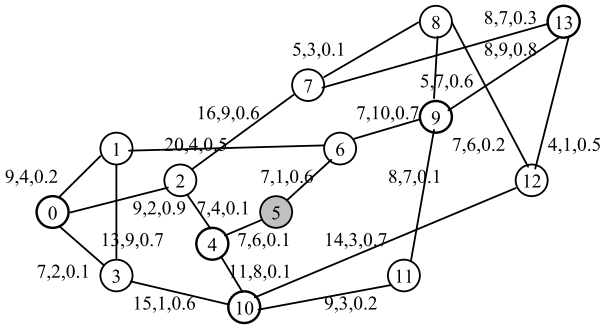
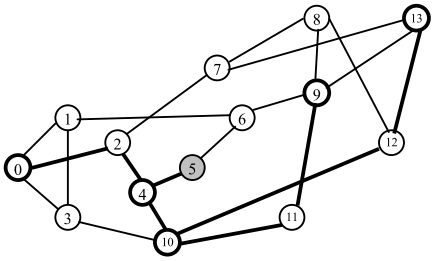


Fig. 2 The NSF (National Science Foundation) network. On each link, d_{ij}, c_{ij} and t_{ij} denote the delay, the cost and the current traffic. The traffic demand $\phi = 0.2$ Mbps, the capacity $z_{ij} = 1.5$ Mbps. The source node $s = 5$, the destinations $R = \{0, 4, 9, 10, 13\}$



(a) A multicast tree for the NSF network in Figure 2

The Solution Encoding	
Gene	Path
g_1	5-4-2-0
g_2	5-4
g_3	5-4-10-11-9
g_4	5-4-10
g_5	5-4-10-12-13

(b) The representation of the solution in (a)

Fig. 3 An example multicast tree and its representation for the NSF network in Fig. 2

and SA process. In this simple yet effective representation, a multicast tree is represented by an ordered set of $|R|$ paths from the source node s to each destination $r_d \in R$, $|R|$ is the group size. That is, each solution contains $|R|$ components $\{g_1, g_2, \ldots, g_{|R|}\}$, where g_i represents a path between the source node s and the d-th destination node r_d, $d = 1, \ldots |R|$.

Given the benchmark NSF (National Science Foundation) network (Cui et al. 2003) in Fig. 2, an example multicast tree and its representation in MOSAGLS are shown in Fig. 3. The NSF network is a major part of the early 1990s Internet backbone for mainly academic uses. It has been tested as a benchmark problem in the existing literature by a number of researchers (Crichigno and Baran 2004a, 2004b; Xu and Qu 2011).

3.2 The genetic local search in MOSAGLS

Genetic algorithms represent one of the mostly investigated evolutionary algorithms in the literature. It simulates the evolutionary process of the nature to evolve from a population of individuals by using genetic operators (Goldberg 1989). Better individuals of higher fitness have more chance to evolve individuals which inherit good building blocks.

In the evolutionary process of our proposed MOSAGLS, the initial population consists of a fixed number of random multicast trees. They are generated by starting from the source node and randomly selecting the next connected node until all the destination nodes have been added to the tree. In each generation, crossover and mutation operations are carried out on two randomly selected parents from the current population. A local search is used to further explore better neighbouring solutions of the generated tree. A non-dominated set *NDS* is maintained during the evolution. It stores the newly generated tree if it is not

dominated by any tree in the current *NDS*, and removes the trees which are dominated by the generated tree.

For the multi-objective MRPs being concerned, the strength Pareto based evaluation in Zitzler and Thiele (1999) is adopted in the genetic algorithm in MOSAGLS to calculate the fitness of individuals. It is used to maintain and update the *NDS* set as well as used in the selection, crossover and mutation operations.

3.2.1 The strength Pareto based evaluation

The value of the five objective functions (2)–(6) defined in Sect. 2 is calculated for each individual. Based on these values, the fitness of each individual is evaluated by using the evaluation method of the Strength Pareto EA in Zitzler and Thiele (1999) as follows:

(1) For each non-dominated solution $T_i \in NDS$, a strength $q_i \in [0, 1]$ is calculated. It is the proportion of the number of solutions T_j which are dominated by T_i to the population size, i.e. T_j is dominated by T_i, denoted by $T_i \triangleright T_j$:

$$q_i = |T_j | T_j \in P \wedge T_i \triangleright T_j | / |P| \tag{8}$$

(2) For each individual T_j in the population, the strength $q_j \in [1, 1 + |NDS|]$ is calculated by summing the strength of all non-dominated solutions $T_i \in NDS$, where $T_i \triangleright T_j$, plus one:

$$q_j = 1 + \sum_{T_i \in NDS, T_i \triangleright T_j} q_i \tag{9}$$

(3) Finally, the fitness of each individual T_j in the population $F(T_j)$ is calculated as the inverse of its strength q_j:

$$F(T_j) = q_j^{-1} \tag{10}$$

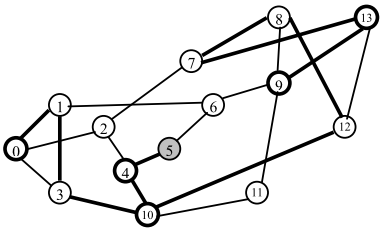
The strength Pareto based evaluation in our algorithm is similar to that is used in MOEA algorithms in Crichigno and Baran (2004a, 2004b). In this work, we focus on the investigation of genetic local search with SA strategies, so this simple and effective strength Pareto based evaluation method in the literature has been adopted. It also enables us to carry out fair comparisons to evaluate the performance of our proposed hybrid algorithm against the MOEA algorithms in the experiments in Sect. 4.

3.2.2 The selection method

We employ the binary tournament selection method, also used in the MOEA algorithm in Crichigno and Baran (2004a), to select parents. Each time two individuals from the population are randomly selected. The individual with a higher fitness value defined by the strength Pareto based evaluation in (10) wins the tournament and is selected as a parent. Two parents are chosen by applying the tournament selection twice.

3.2.3 The crossover operation

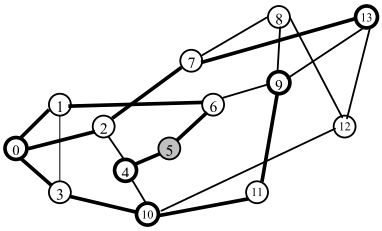
A two-point crossover operator, with a crossover rate of 1, is applied to each selected pair of parents. Based on the representation of the ordered set of paths in Sect. 3.1, the paths between two randomly generated points in one parent are selected and replaced by the corresponding paths in the other parent. Note that some selected paths may share the same links with some remaining paths. Such links will not be removed to ensure the tree is connected.



Crossover points

Parent 1	
Gene	Path
g_1	5-4-10-3-1-0
g_2	5-4
g_3	**5-4-10-12-8-7-13-9**
g_4	5-4-10
g_5	5-4-10-12-8-7-13

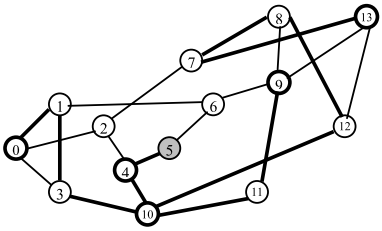
(a) Parent 1, objective values: $C(T) = 11.2$, $DM(T) = 60$, $\alpha(T) = 0.67$, $DA(T) = 38.4$, $DV(T) = 53$



Crossover points

Parent 2	
Gene	Path
g_1	5-6-1-0
g_2	5-4
g_3	**5-6-1-0-3-10-11-9**
g_4	5-6-1-0-3-10
g_5	5-6-1-0-2-7-13

(b) Parent 2, objective values: $C(T) = 9.2$, $DM(T) = 75$, $\alpha(T) = 0.73$, $DA(T) = 49$, $DV(T) = 68$



Individual	
Gene	Path
g_1	5-4-10-3-1-0
g_2	5-4
g_3	**5-4-10-11-9**
g_4	5-4-10
g_5	5-4-10-12-8-7-13

(c) New offspring tree after the crossover, objective values:

$C(T) = 11.4$, $DM(T) = 55$, $\alpha(T) = 0.6$, $DA(T) = 33.4$, $DV(T) = 48$

Fig. 4 An example of the two-point crossover operation

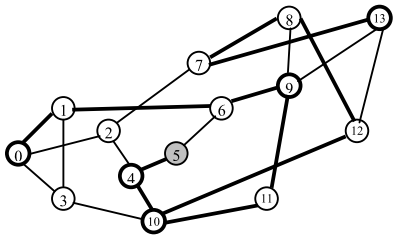
To avoid loops in generating the new multicast tree, the selected path will be replaced by adding the new path from its destination node until it connects to an on-tree node.

An example of the crossover operation is shown in Fig. 4, where path $g_3 = (5–4–10–12–8–7–13–9)$ between the selected two crossover points in parent 1 is replaced by the new corresponding path (5–6–1–0–3–10–11–9) of g_3 in parent 2. To avoid loops in the generated multicast tree, the new path (5–6–1–0–3–10–11–9) is added to the tree by starting from the destination node 9, and adding only the path 9–11–10 until it connects to the on-tree node 10. A new tree is then generated as shown in Fig. 4(c).

The simple representation of multicast trees (see Sect. 3.1) facilitates an easy implementation of crossover operations. By adding the selected path(s) in parent 2 from the destination node, the newly generated offspring is guaranteed to be feasible (if the link capacity constraint (7) is satisfied). Note that while $g_3 = (5–4–10–12–8–7–13–9)$ in parent 1 is being replaced, the links along the path (5–4–10–12–8–7–13) still remains in the multicast tree as they also appear in g_5 in the original tree of parent 1. In multicast trees, some paths share common links, especially those near the root of the tree, i.e. links near the source node appear multiple times in the solution. This is due to the nature of the multicast tree that all

ID	Least cost paths	ID	Least delay paths	ID	Least used paths
1	5-6-1-0	1	5-4-2-0	1	5-4-10-3-0
2	5-4-2-0	2	5-6-1-0	2	5-6-1-0
3	5-6-9-11-10-3-0	3	5-4-10-3-0	3	5-4-2-0
4	5-6-1-3-0	4	5-4-2-7-8-9-6-1-0	4	**5-4-10-11-9-6-1-0**
5	5-4-10-3-0	5	5-6-9-8-7-2-0	5	5-4-10-3-1-0

(a) Routing table of destination node $r_1 = 0$ for the multicast tree in Figure 4(c)



Solution	
Gene	Path
g_1	**5-4-10-11-9-6-1-0**
g_2	5-4
g_3	5-4-10-11-9
g_4	5-4-10
g_5	5-4-10-12-8-7-13

(b) The multicast tree after mutation to the tree in Figure 4(c).
Objective values: $C(T) = 11.4$, $DM(T) = 71$, $\alpha(T) = 0.6$, $DA(T) = 36.6$, $DV(T) = 64$

Fig. 5 An example of the mutation operation

paths must pass a subset of the d_s links from the source node, where d_s is the degree of the source node. These (partial) paths in the tree are adaptively selected through the evolution process of MOSAGLS if they present to be good building blocks in the individuals.

3.2.4 The mutation operation

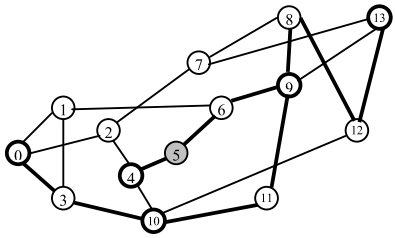
In our MOSAGLS algorithm, mutation is carried out with an adaptive probability by using both the evolutionary information and SA strategies. The adaptive probability is calculated based on not only the fitness of the individual, but also the current temperature. More details of the SA strategies are given in Sect. 3.3.1.

For a selected individual, the mutation operation randomly replaces a path by using an alternative path stored in a routing table, which is the same as that is devised in Crichigno and Baran (2004a). The routing table for the destination $r_d \in R$ of the selected path g_d is consists of m least cost, m least delay and m least used paths (least utilisation path) generated by using the k-shortest path algorithm (Eppstein 1998). As objectives (2), (3) and (6) defined in Sect. 2 are all related to the delay objective, they share the same least delay paths in the routing table. A new randomly selected path $p \in \{path_1, \ldots, path_{3m}\}$ in the routing table then replaces the original path g_d from the source to the destination r_d.

An illustration of the mutation operation is given in Fig. 5 for the offspring generated in Fig. 4(c). Figure 5(a) presents the routing table of destination $r_1 = 0$, listing 5 paths for each objective, i.e. $m = 5$ in this example. If a random path (5–4–10–11–9–6–1–0) is selected from the routing table, a new solution is generated in Fig. 5(b) by replacing the original path (5–4–10–3–1–0) of g_1 in Fig. 4(c).

3.2.5 The local search in MOSAGLS

Instead of reproducing the offspring directly to the next generation, a local search is applied to further enhance the offspring, simulating the maturing phenomenon in the nature. To apply the local search, each solution (a multicast tree) is firstly represented by a binary array of $|V| = n$ bits, each corresponding to a node in the multicast tree. Each bit is assigned a



Solution	
Gene	Path
g_1	5-6-9-11-10-3-0
g_2	5-4
g_3	5-6-9
g_4	5-6-9-11-10
g_5	5-6-9-8-12-13

Fig. 6 An example solution after the local search on the solution in Fig. 5(b). Objective values: $C(T) = 8.8$, $DM(T) = 53$, $\alpha(T) = 0.6$, $DA(T) = 27$, $DV(T) = 46$

value 1 if the corresponding node is on the tree; 0 otherwise. This representation has been widely used in the literature and shows to be effective for local search and genetic operations for MRPs (Skorin-Kapov and Kos 2003, 2006).

In our MOSAGLS, the neighbourhood operator of the local search is based on the well-known Prim's minimum spanning tree algorithm (Betsekas and Gallager 1992) which finds a tree with the minimal total weights of the links spanning a subset of nodes in the graph. The local search repeatedly flips a bit in the binary array which represents a solution until a new better multicast tree is found or a fixed maximum number of nodes have been flipped. This local search method can greatly improve the solutions with regard to the five objectives by using the strength Pareto based evaluation (10). After the local search is applied to the above solution in Fig. 5(b), a new solution is shown in Fig. 6.

This node-based local search has been applied in our previous work (Qu et al. 2009). The selection of the neighbourhood operator is based on our previous observation that the node-based neighbourhood operator is easy to implement and effective for searching better neighbourhood solutions. In this paper, we just investigate this simple yet effective local search operator. More efficient and effective local search methods and the choice of starting solutions for local search (Ishibuchi et al. 2010) may be investigated to reduce the computational time of the hybrid algorithm in our future work.

Based on the procedure described above, Fig. 7 presents the pseudo-code of the hybrid MOSAGLS. In order to improve the performance of MOSAGLS, as shown in Fig. 7, several SA strategies have been applied. We illustrate these strategies in the following subsections.

3.3 Simulated annealing strategies in MOSAGLS

Simulated annealing (Kirkpatrick et al. 1983) is one of the mostly studied probabilistic meta-heuristics for global optimisation. The basic idea is inspired from the physical annealing process where the heated material is gradually cooled to reduce the defects and form large size crystals. Based on this physical process, during the search the SA algorithm accepts a non-improved state/solution with a probability depending on the temperature in the cooling schedule at that time. As a result, it is able to escape from local optima by intelligently accepting worse solutions and effectively explore the search space of complex multi-objective optimisation problems (Czyzzak and Jaszkiewicz 1998; Li and Landa-Silva 2008).

In our MOSAGLS, SA strategies have been adopted to (1) set the adaptive mutation rate, (2) make decision of the acceptance of the new offspring in the genetic local search, and (3) guide the search directions at the later stage of the evolution.

MOSAGLS $(G = (V, E), s, R, k)$
{ // G: the network topology; s: the source node; R: the destination set; k: number of objectives
Generate the random initial population P with $T_1, \ldots, T_{|P|}$ multicast trees;
For $T_1, \ldots, T_{|P|}$, produce $|P|$ distinct random weight vectors $\lambda^1, \ldots, \lambda^{|P|}$, each with a uniform spread;
Select non-dominated solutions with respect to multiple objectives in the population P to form *NDS*;
Calculate the fitness value of all the solutions in *NDS* and P by using the strength Pareto based evaluation in Eq. (10);

$g = 0$;
while ($g <$ Maxgen) **do** {
Current temperature $t = t_{max}$.
while $t \geq t_{min}$ **do** {
Randomly select a pair of parent solutions T_i and T_j from P;
Crossover operation over parents T_i and T_j;
Mutation with adaptive probability p_m to the new solution from crossover; // simulated annealing strategy
Generate T_i' by applying local search to the new solution after mutation; // Sect. 3.2.5
if T_i' is not dominated by parents T_i or T_j **then**
Update *NDS*;
Replace a selected parent T_c by T_i' with the probability $P(T_c, T_i', \lambda^i, t)$; // simulated annealing strategy
Re-calculate the fitness for all solutions in *NDS* and P
Find the closest solution $T_k \in P$ of T_i'
if (T_k is worse than T_i') **then** // simulated annealing strategy
$T_k = T_i'$ // replace the closest member T_k in P
$t = t - t_{step}$ // temperature decrement
if ($t < t_c$) **then**
TuningSearchDirections(t) // simulated annealing strategy: see Fig. 8
} **end of while loop**
g++;
}
return *NDS*;
}

Notes: Maxgen is the maximum of generation; t_{max}/t_{min}: the starting/final temperatures; t_c: the temperature threshold for tuning the weight vector λ; t_{step}: the temperature decrement; P: the current population; *NDS*: the non-dominated solution set

Fig. 7 The pseudo-code of the MOSAGLS algorithm

3.3.1 Simulated annealing based adaptive mutation probability

In MOSAGLS, mutation is always applied to an offspring if it is better than the average of the current population (see more details of the mutation operation in Sect. 3.2.4). For a worse new offspring, mutation is applied with an adaptive probability p_m based on not only the current temperature but also the difference between the fitness of the new offspring and the average fitness of the current population. The adaptive mutation probability p_m is defined as follows:

$$P_m = \begin{cases} 1 & F_{new} > F_{avg} \\ e^{-|F_{avg} - F_{new}|/t} & F_{new} \leq F_{avg} \end{cases} \tag{11}$$

where the average fitness of the current population F_{avg} and the fitness of the new offspring F_{new} are calculated by using the strength Pareto based evaluation function defined in (10); t: the current temperature.

On the one hand, better offspring near the Pareto front will be given higher chance to be evolved by the mutation operation, making the evolution more effective by investing computational time on more promising offspring. On the other hand, the mutation rate is

higher if the current temperature t is higher. This means at the early stage of the evolution, worse offspring still has the chance to be mutated, giving the whole population the chance to explore more areas of the search space. At the later stage when the mutation rate decreases to a small value along with the temperature decrement, worse solutions are rarely evolved to encourage the convergence of the algorithm.

3.3.2 Simulated annealing based acceptance probability

To assist the decision making of accepting a new generated offspring, and to tune the search directions in the later stage (see Sect. 3.3.3), a weighted sum of a convex combination of different objectives is used. A Pareto-optimal solution is obtained if it is the unique global minimum of the following scalar optimisation problem:

$$\text{Minimise } g^{(ws)}(x,\lambda) = \sum_{i=1}^{k} \lambda_i f_i(x), \quad x \in X \tag{12}$$

where λ is a weight vector, $\lambda_i \in [0,1]$, $i = 1,\ldots,k$, k is the number of objectives and $\sum_{i=1}^{k} \lambda_i = 1$; each λ_i is associated with an objective function $f_i(x)$.

Equation (12) is called the weighted scalarising function, one of the mostly used scalarising functions in multi-objective algorithms in the literature (see more details of other functions in Miettinen 1999). In our MOSAGLS, the probability of a new solution x' to replace a solution x is adopted based on the weighted scalarising function $g^{(ws)}$ in Eq. (12):

$$P\big(x,x',\lambda,t\big) = \begin{cases} 1 & \text{if } \Delta g(x,x',\lambda) < 0 \\ e^{\frac{-\Delta g(x,x',\lambda)}{t}} & \text{otherwise} \end{cases} \tag{13}$$

where $t > 0$ is the current temperature; $\Delta g(x,x',\lambda) = g^{(ws)}(x',\lambda) - g^{(ws)}(x,\lambda)$ is the difference between x and x' by using the weighted scalarising function in (12).

In the initialisation of our MOSAGLS algorithm, a normalised weight vector λ^i is randomly generated for each solution T_i. At each generation, a new solution T_i' is generated after the local search upon the offspring produced from the chosen pair of parent solutions T_i and T_j. The parent of lower quality is then replaced by the newly generated tree T_i' based on the weight vector and the current temperature, i.e. parent with a higher probability by comparing $P(T_i, T_i', \lambda^i, t)$ and $P(T_j, T_i', \lambda^j, t)$ in Eq. (13) is replaced.

3.3.3 Simulated annealing based competition and search direction tuning

Simulated annealing strategies have shown to be effective to tune search directions in solving multi-objective travelling salesman problems (Li and Landa-Silva 2008). To tune the search directions in our proposed MOSAGLS, two similar adaptation strategies to Li and Landa-Silva (2008) have been adopted. They are designed to diversify the search directions and avoid solutions being trapped in local optima during the evolutionary process.

(1) The first strategy is defined concerning the competition between close solutions in the current population. Euclidean distance, which is widely used in the multi-objective optimisation literature, has been used in MOSAGLS to measure the distance between two solutions. This is calculated upon the differences between all the objectives values in the two solutions for the problem. After a new solution T_i' is generated, the closest solution T_j measured by the Euclidean distance in the current population is chosen as a neighbour of T_i'. MOSAGLS then replaces T_j with T_i' if T_j is worse than T_i' by comparing their weighted scalarising function values using Eq. (12), i.e. if $g^{(ws)}(T_i', \lambda^j) < g(T_j, \lambda^j)$.

```
TuningSearchDirections (t)
{ // t: current temperature; μ: the constant for tuning the search direction
    foreach T_i ∈ P do {
      Find the closest non-dominated solution T'_i of T_i in the current population
      by using the Euclidean distance upon λ^{i'} and λ^i
      foreach obj ∈ {1, ..., k} do // k: number of objectives {
        if f_obj(T'_i) < f_obj(T_i) then λ^i_obj = μλ^i_obj;
        else λ^i_obj = λ^i_obj/μ;
      } end of foreach
    Normalise λ^i by setting λ^i_obj = λ^i_obj / Σ_{p=1}^{k} λ^i_p }
}
```

Fig. 8 The pseudo-code of search directions tuning by using simulated annealing strategies

(2) The second strategy aims to tune the search direction when the search is getting closer to the Pareto front at the later stage of the evolution. This strategy is similar to that of multi-objective SA algorithms in Czyzzak and Jaszkiewicz (1998) and Li and Landa-Silva (2008). It adaptively tunes the weight vector according to the closest non-dominated neighbouring solution in the current population when the current temperature is decreased to below a threshold. The pseudo-code of the search direction tuning is presented in Fig. 8.

The latter two strategies in Sects. 3.3.2 and 3.3.3 on the solution acceptance and the tuning of search directions have also been investigated in the simulated annealing based multi-objective algorithm named EMOSA in Xu and Qu (2011) for solving the multi-objective MRPs, where only four objectives (i.e. Eqs. (2), (3), (4) and (5), except Eq. (6)) have shown effective to find better non-dominated solutions than other multi-objective evolutionary algorithms (Crichigno and Baran 2004a, 2004b) in the literature. Based on our previous work, these two SA based strategies have been adopted to in our proposed MOSAGLS hybridised with the local search operator as the post improvement optimisation strategy for solving the multi-objective MRPs with more objectives in this paper.

4 Performance evaluation

4.1 Simulation environment and test instances

We have carried out a large amount simulations to test our algorithms on both the benchmark and random networks with different objectives. Two variants of our multi-objective multicast routing algorithms have been evaluated in the following experiments:

(1) MOSAGA, the proposed algorithm without local search, and
(2) MOSAGLS, the proposed algorithm with local search.

Both of these algorithms are new in solving multi-objective MRPs. We firstly evaluate different components of these algorithms on a set of random networks with two objectives (cost and delay, see Sect. 2). Based on the observations obtained, we then compare our algorithms with the existing algorithms in the literature on two benchmark networks with different number of objectives. Finally, we demonstrate the effectiveness and efficiency of our algorithms upon a set of random networks with all the five objectives defined in Sect. 2.

All simulations have been run on a Windows XP computer with PVI 3.4 GHz, 1 GB RAM, and within a multicast routing simulator developed based on Salama's generator (Salama et al. 1997). The multicast routing simulator generates random network

topologies by using the Waxman's graph generation algorithm (Waxman 1988). The nodes of the network are located within a simulated rectangle of size 4000×4000 km^2. The Euclidean metric is used to determine the distance $l(i, j)$ between pairs of nodes i and j. Links connect nodes (u, v), with a probability

$$P(i, j) = \beta e^{-l(i,j)/\alpha L}, \quad \alpha, \beta \in (0, 1] \tag{14}$$

where L is the maximum distance between two nodes; α and β are parameters which can be set to different values to create desired characteristics in the network. Therefore, a large value to β creates a high average degree to nodes, and a small value to α creates long connections between nodes. More details of the simulator can be found in Salama et al. (1997) and Qu et al. (2009).

In all our simulations, we set $\alpha = 0.25$ and $\beta = 0.40$ to generate random network topologies. Within the network, the link cost c_{ij} is assigned a random value between $(0, 100]$ to define the current total bandwidth reserved on the link. The link delay d_{ij} is defined as the propagation delay depending on the length of the link. The link capacity z_{ij} is set as 1.5 Mbps. The current traffic t_{ij} is randomly loaded around 50 % of its total link capacity, which is set the same as that in Diego and Baran 2005. To encourage scientific comparisons, the detailed information of all the problem instances and the experimental results has been provided at http://www.cs.nott.ac.uk/~rxq/benchmarks.htm.

4.2 Parameter settings

In the evolutionary process of MOSAGA, the population size is set to 50, the number of generations is set to 500, and the crossover rate is set to 1. These are the same as those in the other two evolutionary algorithms MOEA1 (Crichigno and Baran 2004a) and MOEA2 (Crichigno and Baran 2004b) for multi-objective MRPs. Instead of using a fixed mutation rate 0.3 in MOEA1 and MOEA2, we employ an adaptive mutation rate in MOSAGA. Based on initial tests, we set $m = 25$ to generate the routing table of $3 \times m$ alternative paths in the mutation operation. In the SA process of our MOSAGA, we set the initial temperature $t_{max} = 50$, the final temperature $t_{min} = 5$, the temperature decrement $t_{step} = 5$, and the temperature threshold $t_c = 25$. The parameters of the SA process are kept the same as those in the SA based multi-objective algorithm for solving travelling salesman problems (Li and Landa-Silva 2008). For simplicity and fair comparisons, we keep the same parameter settings for the proposed algorithms on all the tested instances in this paper unless otherwise stated.

All the parameter values have been also determined after a set of tests to find the balance between the quality of solutions and the running time. For example, to obtain a proper range of values for the population size and the starting temperature, a set of initial tests have been carried on the NSF network shown in Fig. 2 to compare the performance of MOSAGA with different parameter settings. The optimal Pareto Front (PF) of the NSF benchmark problem that composes of 16 solutions has been found by an exhaustive search method in Crichigno and Baran (2004b), where four objectives (2), (3), (4) and (5) have been considered in the algorithms. We thus considered the same four objectives in our MOSAGA algorithms in this group of tests. Table 2 presents the maximum, minimum and average number of non-dominated solutions found by each variant of MOSAGA in 50 runs. The setting of population size $pop = 50$ and the starting temperature $t_{max} = 50$ provides the best solutions and requires less computing time, is thus selected in our algorithms.

4.3 Results on random networks with two objectives

In order to visually analyse the impact of SA strategies, the adaptive mutation and the local search on the performance of the proposed algorithms, we carry out a series of experiments

Table 2 Comparison of MOSAGA with different parameter settings for solving the NSF network in Fig. 2. $|NDS|$: the number of non-dominated solutions; Running time = 60 seconds in each run

pop	t_{max}	Max $\|NDS\|$	Min $\|NDS\|$	Average $\|NDS\|$
50	**50**	**16**	**16**	**16**
50	100	16	13	14
100	50	16	12	15
100	100	16	13	14

on random networks with two objectives, namely the cost and the delay as defined in Sect. 2. Four random networks have been generated with different network sizes ($|V| = 50$ and $|V| = 100$) and group sizes (number of destination nodes $|R| = 20\,\% * |V|$ and $|R| = 30\,\% * |V|$).

4.3.1 The impact of simulated annealing strategies

In the first group of experiments, we compare MOSAGA with two evolutionary algorithms, MOEA1 (Crichigno and Baran 2004a) and MOEA2 (Crichigno and Baran 2004b), to identify the impact of SA strategies to our multi-objective genetic local search. For a fair comparison, the same parameter setting in MOEA1 and MOEA2 has been used in the evolution process in our MOSAGA. The non-dominated solutions found by the three algorithms after 50 runs are shown in Fig. 9, with their average computing time in Table 3.

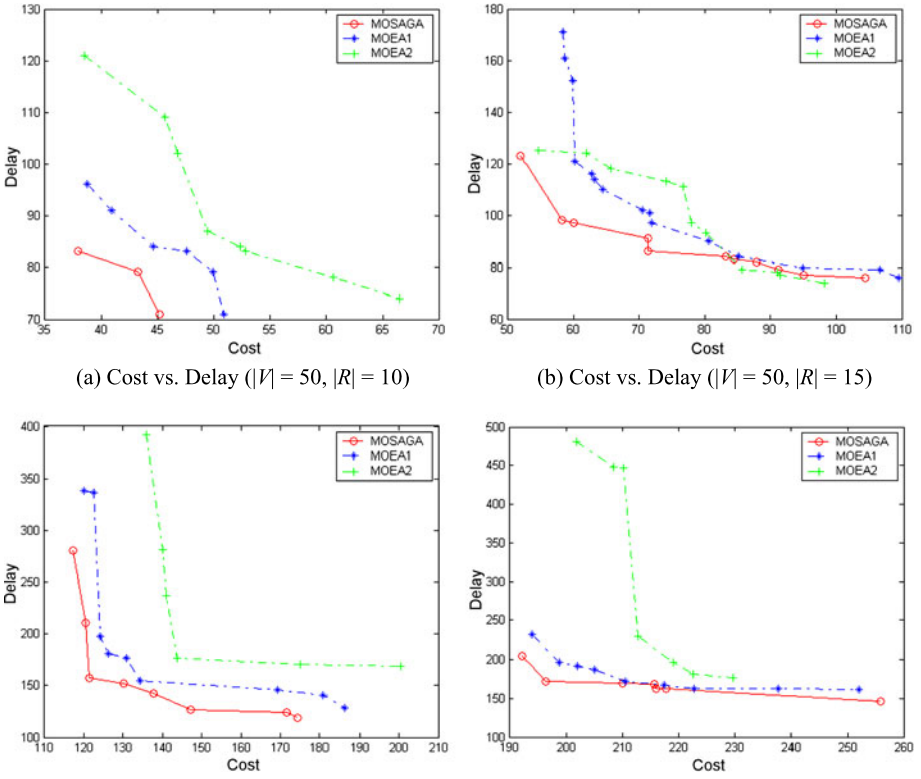
The experimental results clearly show that the MOSAGA algorithm found a set of better non-dominated solutions compared with those from the two MOEA algorithms on 3 out of 4 problems, i.e. Fig. 9(a), (c) and (d). For the problem in Fig. 9(b), MOSAGA is able to find better non-dominated solutions except those three solutions found by MOEA2. However, for all other three problems, MOEA2 has performed the worst. With regard to the computing time, MOSAGA requires the least computing time. This demonstrates that the SA strategies in MOSAGA can guide the search direction of genetic algorithm to find better solutions in less computing time in comparison with the MOEA algorithms.

4.3.2 The impact of the adaptive mutation

In the second group of experiments, we test the effect of the adaptive mutation by comparing it with four fixed mutation rates, i.e. $P_m = 0.3, 0.6, 0.9, 1$ in MOSAGA. The non-dominated solutions found by MOSAGA with different mutation rates after 50 runs are shown in Fig. 10. In general, MOSAGA with the adaptive mutation rate has the best overall performance among all the other variants with fixed mutation rates, although the Pareto fronts found by variants of the algorithm interweave at certain part of the front for some networks, i.e. Fig. 10(b) and Fig. 10(d). With the fixed mutation rate, MOSAGA occasionally finds several better non-dominated solutions, but fails to obtain the majority of solutions at the Pareto front found by MOSAGA with adaptive mutation rate for all four networks. The adaptive mutation rate makes the search more effective based on both the temperature and the fitness of the new solution and the current population.

4.3.3 The impact of the local search

In the third group of experiments, we compare the MOSAGA and MOSAGLS algorithms to demonstrate the effect of the local search in our proposed algorithms. The local search in MOSAGLS stops after a number of nodes (set as 10 here) has been flipped.



(a) Cost vs. Delay ($|V| = 50$, $|R| = 10$)

(b) Cost vs. Delay ($|V| = 50$, $|R| = 15$)

(c) Cost vs. Delay ($|V| = 100$, $|R| = 20$)

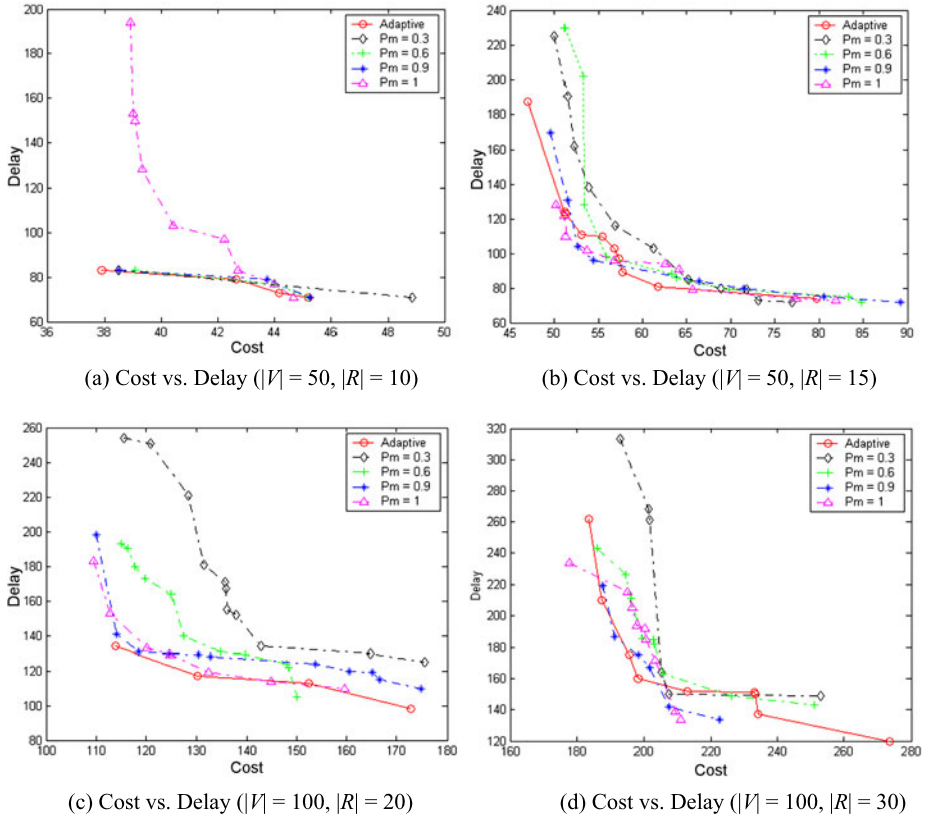
(d) Cost vs. Delay ($|V| = 100$, $|R| = 30$)

Fig. 9 The non-dominated solutions found by MOSAGA, MOEA1 and MOEA2 on random networks with two objectives in 50 runs

Table 3 Average computing time of MOSAGA, MOEA1 and MOEA2 on random networks with two objectives

Network size	Group size	Computing time (sec)		
		MOSAGA	MOEA1	MOEA2
50	10	**7.068**	25.211	25.544
50	15	**6.208**	34.901	36.703
100	20	**13.342**	88.411	96.489
100	30	**18.338**	126.517	132.015

The non-dominated solutions found by both algorithms for the four random networks in 50 runs are shown in Fig. 11, clearly showing the improvement made by the local search in MOSAGLS. For the small network in Fig. 11(a), MOSAGLS found four non-dominated solutions, while MOSAGA found only two non-dominated solutions. For the same networks with larger group size and for networks of large size, it is obvious that MOSAGLA outperforms MOSAGA by finding much better non-dominated solutions. This demonstrates



(a) Cost vs. Delay ($|V| = 50$, $|R| = 10$)

(b) Cost vs. Delay ($|V| = 50$, $|R| = 15$)

(c) Cost vs. Delay ($|V| = 100$, $|R| = 20$)

(d) Cost vs. Delay ($|V| = 100$, $|R| = 30$)

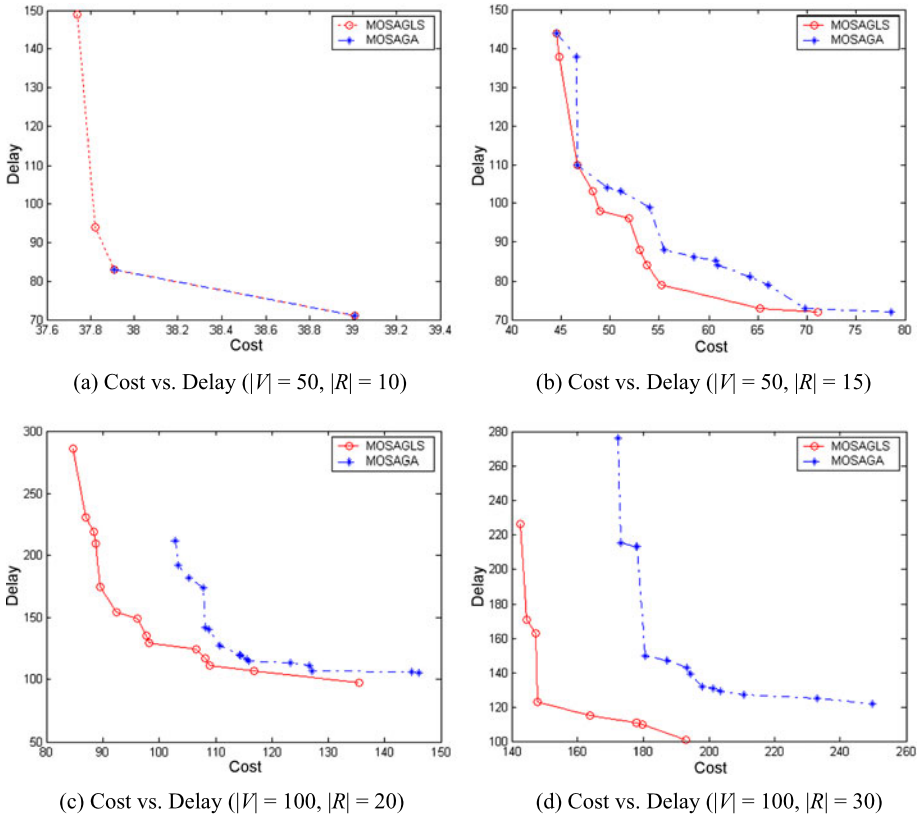
Fig. 10 The non-dominated solutions found by MOSAGA with different mutation rates on random networks with two objectives in 50 runs

Table 4 Average computing time of MOSAGA and MOSAGLS on random networks with two objectives

Network size	Group size	Computing time (sec)	
		MOSAGA	MOSAGLS
50	10	0.573	4.076
50	15	0.684	5.331
100	20	1.62	22.198
100	30	1.616	36.274

the efficiency of the local search in our proposed MOSAGLS algorithm. Please note the non-dominated solutions obtained by MOSAGA in Fig. 11 are different from the results of MOSAGA in Fig. 10 due to the independent runs of experiments on the random networks.

Table 4 presents the average computing time of MOSAGA and MOSAGLS on the four random networks, showing (not surprisingly) that MOSAGLS finds better solutions at the expenses of longer computing time compared with the MOSAGA algorithm. In real world



(a) Cost vs. Delay ($|V| = 50$, $|R| = 10$)

(b) Cost vs. Delay ($|V| = 50$, $|R| = 15$)

(c) Cost vs. Delay ($|V| = 100$, $|R| = 20$)

(d) Cost vs. Delay ($|V| = 100$, $|R| = 30$)

Fig. 11 The non-dominated solutions found by MOSAGLS and MOSAGA on random networks with two objectives in 50 runs

applications, the balance of solution quality and the computational time should be concerned while designing the local search in MOSAGLS. This demonstrates that our proposed local search is able to improve the search results of MOSAGLS while consuming longer computational time.

4.4 Results on benchmark and random problems with different objectives

4.4.1 Results on the NSF network with four objectives

Based on the above observations, we compare the performance of the MOSAGLS and MOSAGA algorithms on the benchmark NSF network with that of MOEA algorithms. As mentioned in Sect. 4.2, the optimal Pareto front of 16 solutions for the NSF problem has been found in Crichigno and Baran (2004b), concerning the four objectives (2), (3), (4) and (5) defined in Sect. 2. We thus consider the same four objectives in this group of experiments.

The adaptive mutation rate has been applied in MOSAGLS and MOSAGA. To ensure a fair comparison between algorithms, we have re-implemented the MOEA algorithms, and compared the results obtained by the four algorithms within 60 seconds instead of defining a fixed number of generations. Table 5 presents the maximum, minimum and average number of non-dominated solutions found by each algorithm in 50 runs.

Table 5 The number of non-dominated solutions found by different algorithms on the NSF network in Fig. 2 with four objectives. Computing time = 60 seconds

Algorithms	Max $\|NDS\|$	Min $\|NDS\|$	Average $\|NDS\|$
MOSAGLS	**16**	**16**	**16**
MOSAGA	**16**	**16**	**16**
MOEA1	16	14	15
MOEA2	16	10	15

Table 6 Two different multicast groups used for the experiments on the NTT network in Fig. 12

Multicast group	Source	Destination set R	$\|R\|$
Group 1 (small)	5	{0, 1, 8, 10, 22, 32, 38, 43, 53}	9
Group 2 (large)	4	{0, 1, 3, 5, 6, 9, 10, 11, 12, 17, 19, 21, 22, 23, 25, 33, 34, 37, 41, 44, 46, 47, 52, 54}	24

For this small NSF problem, our proposed MOSAGLS and MOSAGA algorithms outperform the two MOEA algorithms, i.e. always find all 16 solutions in the optimal Pareto front in 50 runs. This, together with our above observations, indicates that SA based MOSAGA and MOSAGLS with adaptive mutation rate are more effective than the two conventional MOEA algorithms for multi-objective MRPs.

4.4.2 Results on the NTT network with five objectives

We also test our algorithms on the NTT (Nippon Telephone Telegraph) network in Fig. 12. Two different multicast groups have been tested, as shown in Table 6. We set the current traffic t_{ij} of each link as randomly loaded with around 50 % of its total link capacity, which is the same as that in Diego and Baran (2005). All algorithms have been evaluated with respect to all the five objectives defined in Sect. 2.

As the optimal Pareto front for the NTT network with all the five objectives defined in Sect. 2 is not known, we use a six-step procedure devised in Diego and Baran (2005) to obtain an approximation of the Pareto front for each problem. The six-step procedure in Diego and Baran (2005) is presented as follows:

(1) Each of the algorithms tested is run 10 times.
(2) For each algorithm, 10 sets of non-dominated solutions $Y_1, Y_2, \ldots, Y_{10}$ are obtained, one from each run.
(3) For each algorithm, an aggregate collection of solutions Y_T is obtained, i.e. $Y_T = \bigcup_{i=1}^{10} Y_i$.
(4) The non-dominated solutions in Y_T are stored for each algorithm, denoted by Y_{alg}.
(5) A set of solutions Y' is obtained by combining all Y_{alg}, i.e. $Y' = \bigcup Y_{alg}$.
(6) The non-dominated solutions in Y' are used as an approximation of the true optimal Pareto solution set, called Y_{PF}.

Based on this approximation of the optimal Pareto solution set Y_{PF}, in order to have an insight of the quality of solutions obtained by each algorithm with regard to Y_{PF}, the following notations are used:

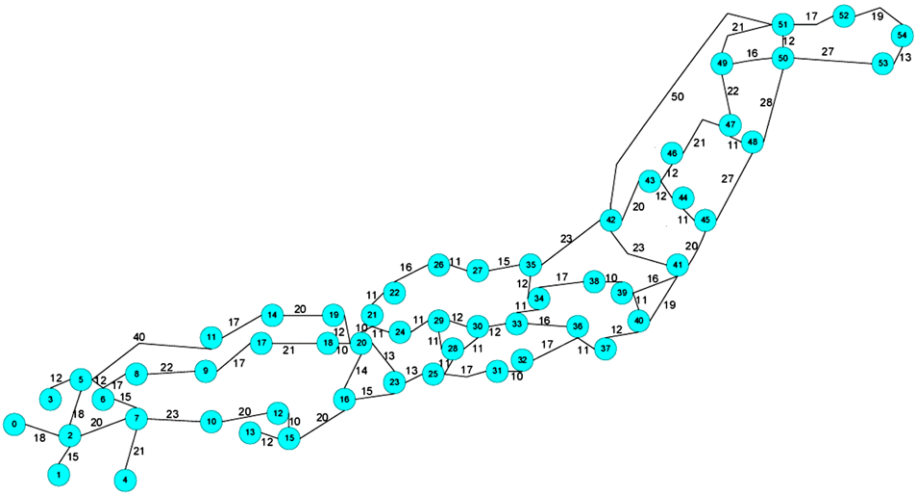


Fig. 12 The Nippon Telephone Telegraph (NTT) network of Japan with 55 nodes and 142 links. Costs are shown on each link

– $\in Y_{PF}$: the average number of solutions that are in Y_{PF} found by each algorithm in all runs;
– $Y_{PF} \succ$: the average number of solutions that are dominated by Y_{PF} found by each algorithm in all runs;
– $|\bar{Y}|$: the average number of any solutions found by each algorithm in all runs;
– % Y_{PF}: the average percentage of solutions in Y_{PF} found by each algorithm in all runs, i.e. $\in Y_{PF}/|Y_{PF}|$.

(1) Comparisons of MOSAGLS with MOSAGA on the NTT network

Firstly, to investigate the effectiveness and efficiency of the local search operator in our proposed MOSAGLS, we compare MOSAGLS with MOSAGA in which no local search is applied. Table 7 presents the total number non-dominated solutions in Y_{PF} obtained by MOSAGA and MOSAGLS for the NTT network in Fig. 12 with two different multicast groups in Table 6 by using the above six-step procedure. For a fair comparison, the computational time for both algorithms for each run is 60 seconds.

Results in both Table 8 and Table 9 show that MOSAGLS found more non-dominated solutions $\in Y_{PF}$ than that of MOSAGA, and thus gave a better approximation to the Pareto Front. This becomes more obvious for the NTT network with the large group size (see group 2 in Table 6) in Table 9, where MOSAGLS found 17.27 % of the non-dominated solutions in Y_{PF}, compared with 7.27 % of solutions in Y_{PF} found by MOSAGA. Better experimental results obtained by MOSAGLS demonstrate that the local search operator can improve the performance of MOSAGLS by finding more non-dominated Pareto optimal solutions for solving the multi-objective MRP with five objectives.

(2) Comparisons of MOSAGLS with MOEA1 and MOEA2 on the NTT network

Secondly, we compare MOSAGLS with other two multi-objective evolutionary algorithms MOEA1 and MOEA2 on the NTT benchmark problem with five objectives. Each algorithm has been run 10 times on each instance.

Table 7 Based on the six-step procedure, the total number of non-dominated solutions Y_{PF} obtained by MOSAGLS and MOSAGA for the NTT network with two different multicast groups with respect to all five objectives

Computational time	60 seconds each run	
Multicast group	Group 1	Group 2
$\|Y_{PF}\|$	6	11

Table 8 Comparisons of MOSAGLS and MOSAGA for the NTT network with the small multicast group 1. Computational time for each run is 60 seconds. The best results are in bold

Algorithms	$\in Y_{PF}$	$Y_{PF} \succ$	$\|Y_{alg}\|$	% Y_{PF}
MOSAGLS	**3.86**	**6.1**	9	**64.33** %
MOSAGA	3.02	7.52	10	50.33 %

Table 9 Comparisons of MOSAGLS and MOSAGA for solving the NTT network with the large multicast group 2. Computation time for each run is 60 seconds. The best results are in bold

Algorithms	$\in Y_{PF}$	$Y_{PF} \succ$	$\|Y_{alg}\|$	% Y_{PF}
MOSAGLS	**1.9**	**9.06**	10.96	**17.27 %**
MOSAGA	0.8	10.42	11.22	7.27 %

Table 10 Based on the six-step procedure, the total number of non-dominated solutions in Y_{PF} obtained by MOSAGLS, MOEA1 and MOEA2 for the NTT network with two different multicast groups with respect to all five objectives

Running time	160 seconds		320 seconds	
Multicast group	Group 1	Group 2	Group 1	Group 2
$\|Y_{PF}\|$	44	16	42	17

Table 10 presents the total number non-dominated solutions in Y_{PF} obtained by the three algorithms for the two NTT instances by using the six-step procedure. For a fair and comprehensive comparison, all algorithms tested have been given the same computational time, i.e. 160 seconds and 320 seconds, respectively, for each run.

Tables 11 and 12 present the results obtained for the NTT network with two different multicast groups. Both tables show that MOSAGLS found more non-dominated solutions $\in Y_{PF}$ than that of MOEA algorithms. A measurement used in Zitzler and Thiele (1999) is used here to calculate the coverage of the non-dominated solution set obtained by each algorithm. This is represented by the ratio of the number of non-dominated solutions obtained by each algorithm to that of another algorithm. Tables 11 and 12 show that the non-dominated solutions found by MOSAGLS cover the majority of the non-dominated solutions found by MOEA algorithms. This becomes more obvious for the NTT network with larger group size, where MOSAGLS found almost all non-dominated solutions, while the MOEA algorithms found just one or no solution that belongs to MOSAGLS.

Table 11 Results of different algorithms for solving the NTT network with small multicast group 1 in Table 6

Algorithms	$\in Y_{PF}$	$Y_{PF} \succ$	$\lvert\bar{Y}\rvert$	% Y_{PF}	Coverage	Y_{MOSAGLS}	Y_{MOEA1}	Y_{MOEA2}
Running time = 160 seconds								
MOSAGLS	4.1	14.6	18.7	9.32 %	Y_{MOSAGLS}	\	0.57	0.8
MOEA1	1.4	25.7	27.1	3.18 %	Y_{MOEA1}	0.33	\	0.32
MOEA2	1.8	14.1	15.9	4.09 %	Y_{MOEA2}	0.56	0.5	\
Running time = 320 seconds								
MOSAGLS	5	17.1	22.1	12.9 %	Y_{MOSAGLS}	\	0.57	0.82
MOEA1	4.8	18.6	23.4	11.4 %	Y_{MOEA1}	0.71	\	0.96
MOEA2	1.9	13	14.9	4.75 %	Y_{MOEA2}	0.57	0.49	\

Table 12 Results of different algorithms for solving the NTT network with large multicast group 2 in Table 6

Algorithms	$\in Y_{PF}$	$Y_{PF} \succ$	$\lvert\bar{Y}\rvert$	% Y_{PF}	Coverage	Y_{MOSAGLS}	Y_{MOEA1}	Y_{MOEA2}
Running time = 160 seconds								
MOSAGLS	7.3	11.2	18.5	45.63 %	Y_{MOSAGLS}	\	1	1
MOEA1	0	17.8	17.8	0	Y_{MOEA1}	0	\	0.64
MOEA2	0	10.9	10.9	0	Y_{MOEA2}	0	0	\
Running time = 320 seconds								
MOSAGLS	6.8	13.5	20.3	40 %	Y_{MOSAGLS}	\	0.95	1
MOEA1	0.1	16	16.1	0.59 %	Y_{MOEA1}	0.06	\	1
MOEA2	0	12.1	12.1	0	Y_{MOEA2}	0	0	\

Table 13 The average number of non-dominated solutions on the random networks

Network size	$\lvert V\rvert = 50$		$\lvert V\rvert = 100$	
Group size	$\lvert R\rvert = 10$	$\lvert R\rvert = 15$	$\lvert R\rvert = 20$	$\lvert R\rvert = 30$
$\lvert\bar{Y}_{PF}\rvert$	264.4	196.2	91.4	105.2

4.4.3 Results on random networks with five objectives

Finally, we test the effectiveness and efficiency of our algorithm upon a set of random networks of different characteristics with respect to all five objectives defined in Sect. 2. We generate 10 random graphs for each network size of $\lvert V\rvert = 50$ and $\lvert V\rvert = 100$, and with different group sizes ($\lvert R\rvert = 20\ \% * \lvert V\rvert$ and $\lvert R\rvert = 30\ \% * \lvert V\rvert$). Therefore, there are in total 40 instances (10 for each of the four types of networks) tested.

We set the same running time, i.e. 320 seconds for all algorithms in each run. Table 13 presents the average number of non-dominated solutions ($\lvert\bar{Y}_{PF}\rvert$) obtained from 10 runs of the three algorithms on each of the four types of random networks by using the six-step procedure in Sect. 4.4.2.

Tables 14 and 15 present results of the three algorithms on the random networks with different characteristics. Average results from 10 runs on 10 different instances for each type of networks show that MOSAGLS significantly outperforms the two MOEA algorithms by

Table 14 Comparison of different algorithms on random networks of $|V| = 50$ with different group sizes. Computational time for each run is 320 seconds

Algorithms	$\in Y_{PF}$	$Y_{PF} \succ$	$\lvert\bar{Y}\rvert$	% Y_{PF}	Coverage	Y_{MOSAGLS}	Y_{MOEA1}	Y_{MOEA2}
Group size $\lvert R\rvert = 10$								
MOSAGLS	37.72	66.08	103.8	14.69 %	Y_{MOSAGLS}	\	0.64	0.82
MOEA1	6.48	110.46	116.94	2.91 %	Y_{MOEA1}	0.01	\	0.55
MOEA2	2.76	94.84	97.6	1.16 %	Y_{MOEA2}	0.01	0.19	\
Group size $\lvert R\rvert = 15$								
MOSAGLS	30.3	61.9	92.2	15.91 %	Y_{MOSAGLS}	\	0.7	0.84
MOEA1	3.26	95.7	98.96	1.67 %	Y_{MOEA1}	0.04	\	0.76
MOEA2	1.38	85.76	87.14	0.54 %	Y_{MOEA2}	0.02	0.14	\

Table 15 Comparison of different algorithms on random networks of $|V| = 100$ with different group sizes. Computational time for each run is 320 seconds

Algorithms	$\in Y_{PF}$	$Y_{PF} \succ$	$\lvert\bar{Y}\rvert$	% Y_{PF}	Coverage	Y_{MOSAGLS}	Y_{MOEA1}	Y_{MOEA2}
Group size $\lvert R\rvert = 20$								
MOSAGLS	2.15	16.36	18.51	2.9 %	Y_{MOSAGLS}	\	0.9	0.97
MOEA1	0.05	17.60	17.66	0.05 %	Y_{MOEA1}	0.004	\	0.71
MOEA2	0.004	14.72	14.72	0.004 %	Y_{MOEA2}	0.008	0.13	\
Group size $\lvert R\rvert = 30$								
MOSAGLS	0.928	4.146	5.074	0.93 %	Y_{MOSAGLS}	\	0.9	1
MOEA1	0.13	5.62	5.75	0.08 %	Y_{MOEA1}	0.002	\	0.7
MOEA2	0	4.59	4.59	0	Y_{MOEA2}	0	0.17	\

finding more non-dominated solutions in Y_{PF}. The non-dominated solution set obtained by MOSAGLS covers most of the non-dominated solutions found by MOEA1 and MOEA2. This again demonstrates the effectiveness of our proposed MOSAGLS algorithm on the random networks with all five objectives defined.

To summarise, the large amount of experiments on a range of multi-objective MRPs with different features demonstrate the efficiency and effectiveness of our proposed simulated annealing based multi-objective genetic local search algorithm MOSAGLS. The SA strategies improve the performance of the algorithm by finding better non-dominated solutions in less computing time compared with conventional multi-objective evolutionary algorithms. The adaptive mutation contributes a better performance than that of fixed mutation rates. The local search further improves the performance of MOSAGLS, however at a higher computational time. Comparisons on both the benchmark problems and random networks demonstrate that the proposed MOSAGLS has the best performance among variants of algorithms, and show that MOSAGLS is able to find high quality solutions for multi-objective MRPs with different features.

5 Conclusions

In this paper, we investigate the first simulated annealing based multi-objective genetic local search (MOSAGLS) for solving multi-objective MRPs. A new simulated annealing based adaptive mutation probability is proposed in MOSAGLS, which can adaptively adjust the mutation rate according to the fitness of the new solution against the average quality of the current population during the genetic evolution procedure. In addition, two adaptation strategies based on simulated annealing are adopted to tune the search directions in MOSAGLS. One is the competition between similar members in the current population, another one is the two-phase strategy for tuning weight vectors. Integrated with these simulated annealing strategies, the hybrid multi-objective genetic local search is able to efficiently search towards the Pareto front and diversify the population with regard to the multiple objectives in the problem.

Due to the complex structure of the multicast tree, an ordered set of paths has been used to represent solutions. Based on this simple solution representation, the crossover and mutation operators have been specifically designed concerning the network structure to facilitate efficient and effective operations and improve the objective values of the tree while satisfying the constraint. Finally, the local search further intensifies the search by exploring more promising neighbouring solutions based on a binary string solution representation.

Through a large amount of extensive simulations, we evaluate our proposed MOSAGLS on both benchmarks and a series of random networks with different objectives and different features. Experimental results show that the simulated annealing strategies significantly improve our proposed algorithm compared against another two conventional MOEA algorithms in the literature with respect to both the solution quality and computing time for the multi-objective MRPs. This demonstrates that better control of the evolution by using the simulated annealing strategies significantly improves the efficiency and effectiveness of MOSAGLS. Compared with the simulated annealing based multi-objective genetic algorithm without local search, MOSAGLS obtained significantly better results but at larger computational expenses.

The proposed MOSAGLS algorithm has shown to be an ideal approach for solving the multi-objective MRPs, and is flexible to be extended to solve other multi-objective optimisation problems. In our future work, we also intend to investigate the influence of different selection strategies and adaptive crossover in the hybrid algorithm. More efficient and effective local search methods and the choice of starting solutions for local search may be investigated to reduce the computational time of the hybrid algorithm. Other Pareto dominance methods may also be applied to further improve the search towards the Pareto front in the MOSAGLS algorithm. In addition, some recent powerful multi-objective optimisation algorithms can be investigated in our future work.

Acknowledgements This research is supported by Natural Science Foundation of China (NSFC project No. 61202289), the science and technology plan of Hunan Province (No. 2012FJ4263) and the project of the support plan for young teachers in Hunan University, China (No. 531107021137).

References

Bader, J., & Zitzler, E. (2011). HypE: an algorithm for fast hypervolume-based many-objective optimization. *Evolutionary Computation*, *19*(1), 45–76.

Betsekas, D., & Gallager, R. (1992). *Data networks* (2nd ed.). Englewood Cliffs: Prentice-Hall.

Beume, N., Naujoks, B., & Emmerich, M. (2007). SMS-EMOA: multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, *181*(3), 1653–1669.

Chen, S., & Nahrestedt, K. (1998). An overview of quality of service routing for next-generation high-speed networks: problems and solutions. *IEEE Network Magazine*, *12*(6), 64–79. Special issue on transmission and distribution of digital video.

Cui, X., Lin, C., & Wei, Y. (2003). A multiobjective model for QoS multicast routing based on genetic algorithm. In *Proceedings of 2003 international conference on computer networks and mobile computing (ICCNMC'03)* (pp. 49–53).

Crichigno, J., & Baran, B. (2004a). Multiobjective multicast routing algorithm for traffic engineering. In *Proceedings of the 13th international conference on computer communications and networks* (pp. 301–306). Heidelberg: Springer.

Crichigno, J., & Baran, B. (2004b). Multiobjective multicast routing algorithm. In *Lecture notes in computer science* (Vol. 3124, pp. 1029–1034).

Czyzzak, P., & Jaszkiewicz, A. (1998). Pareto simulated annealing—a meta-heuristic technique for multiobjective combinatorial optimization. *Journal of Multi-Criteria Decision Analysis*, *7*(1), 34–37.

Diot, C., Dabbous, W., & Crowcroft, J. (1997). Multipoint communication: a survey of protocols, functions, and mechanisms. *IEEE Journal on Selected Areas in Communications*, *15*, 277–290.

Diego, P., & Baran, B. (2005). Solving multiobjective MRP with a new ant colony optimization approach. In *Proceedings of the 3rd international IFIP/ACM Latin American conference on networking*, Columbia (pp. 11–19).

Deb, K. (2005). Multi-objective optimisation. In E. K. Burke & G. Kendall (Eds.), *Search methodologies: introductory tutorials in optimisation and decision support methodologies* (pp. 273–316). Berlin: Springer.

Eppstein, D. (1998). Finding the *k* shortest paths. *SIAM Journal on Computing*, *28*, 652–673.

Ehrgott, M., & Gandibleux, X. (2000). A survey and annotated bibliography of multiobjective combinatorial optimization. *OR Spektrum*, *22*, 425–460.

Fabregat, R., Donoso, Y., Baran, B., Solano, F., & Marzo, J. L. (2005). Multi-objective optimization scheme for multicast flows: a survey, a model and a MOEA solution. In *Proceedings of the 3rd international IFIP/ACM Latin American conference on networking (LANC'05)*. New York: ACM.

Goldberg, D. (1989). *Genetic algorithm in search, optimization & machine learning*. Reading: Addison-Wesley.

Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability: a guide to the theory of NP-completeness*. New York: Freeman.

Haghighat, A. T., Faez, K., Dehghan, M., Mowlaei, A., & Ghahremani, Y. (2004). GA-based heuristic algorithms for bandwidth-delay-constrained least-cost multicast routing. *Computer Communications*, *27*, 111–127.

Huang, J., & Liu, Y. (2010). MOEAQ: a QoS-aware multicast routing algorithm for MANET. *Expert Systems with Applications*, *37*, 1391–1399.

Hwang, F. K., & Richards, D. S. (1992). Steiner tree problems. *Networks*, *22*, 55–89.

Ishibuchi, H., & Murata, T. (1998). A Multi-objective genetic local search algorithm and its application to flowshop scheduling. *IEEE Transactions on Systems, Man and Cybernetics. Part C, Applications and Reviews*, *28*(3), 392–403.

Ishibuchi, H., Hitotsuyanagi, Y., Wakamatsu, Y., & Nojima, Y. (2010). How to choose solutions for local search in multiobjective combinatorial memetic algorithms. In *Proc. of 11th international conference on parallel problem solving from nature* (pp. 516–525).

Ishibuchi, H., Akedo, N., Ohyanagi, H., & Nojima, Y. (2011). Behavior of EMO algorithms on many-objective optimization problems with correlated objectives. In *Proc. of 2011 IEEE congress on evolutionary computation*, New Orleans, USA, June 5–8, 2011 (pp. 1465–1472).

Jaszkiewicz, A. (2002). Genetic local search for multi-objective combinatorial optimization. *European Journal of Operational Research*, *137*, 50–71.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, *220*(4598), 671–680.

Kompella, V. P., Pasquale, J. C., & Polyzos, G. C. (1993). Multicast routing for multimedia communication. *IEEE/ACM Transactions on Networking*, *1*(3), 286–292.

Koyama, A., Barolli, L., Matsumoto, K., & Apduhan, B. O. (2004). A GA-based multi-purpose optimization algorithm for QoS routing. In *Proceedings of the 18th international conference on advanced information networking and applications (AINA 2004)* (Vol. 1(1), pp. 23–28).

Kun, Z., Heng, W., & Feng-Yu, L. (2005). Distributed multicast routing for delay variation-bounded Steiner tree using simulated annealing. *Computer Communications*, *28*, 1356–1370.

Landa-Silva, J. D., Burke, E. K., & Petrovic, S. (2004). An introduction to multiobjective metaheuristics for scheduling and timetabling. In X. Gandibleux, M. Sevaux, K. Sorensen, & V. T'kindt (Eds.), *Lecture notes in economics and mathematical systems: Vol. 535. Metaheuristic for multiobjective optimisation* (pp. 91–129). Berlin: Springer.

Li, C., Cao, C., Li, Y., & Yu, Y. (2007). Hybrid of genetic algorithm and particle swarm optimization for multicast QoS routing. In *IEEE international conference on control and automation*, Guangzhou, China.

Li, H., & Landa-Silva, J. D. (2008). Evolutionary multi-objective simulated annealing with adaptive and competitive search direction. In *Proceedings of the 2008 IEEE congress on evolutionary computation (CEC 2008)* (pp. 3310–3317). Hong Kong: IEEE Press.

Li, H., & Zhang, Q. (2009). Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II. *IEEE Transactions on Evolutionary Computation*, *13*(2), 284–302.

Martins, F., & Costa, C. A. V. (2010). Multiobjective optimization with economic and environmental objective functions using modified simulated annealing. *Computer-Aided Chemical Engineering*, *28*, 919–924.

Mendoza, J. E., Castanier, B., Guéret, C., Medaglia, A. L., & Velasco, N. (2010). A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands. *Computers & Operations Research*, *37*(11), 1886–1898.

Miettinen, K. (1999). *Nonlinear multiobjective optimization*. Boston: Kluwer Academic.

Oliveira, C. A. S., & Pardalos, P. M. (2005). A Survey of combinatorial optimization problems in multicast routing. *Computers & Operations Research*, *32*(8), 1953–1981.

Qu, R., Xu, Y., & Kendall, G. (2009). *A variable descent search algorithm for delay-constrained least-cost multicast routing*. In *Lecture notes in computer science* (Vol. 5851, pp. 15–29).

Rai, S. C., Misra, B. B., Nayak, A. K., Mall, R., & Pradhan, S. K. (2010). A multi-objective QoS optimization with fuzzy based parameter setting for real time multicasting. *International Journal of Communications, Network and System Sciences*, *3*, 530–539.

Roy, A., & Das, S. K. (2004). QM2RP: a QoS-based mobile multicast routing protocol using multi-objective genetic algorithm. *Wireless Networks*, *10*(3), 271–286.

Roy, A., Banerjee, N., & Das, S. K. (2002). An efficient multi-objective QoS-routing algorithm for wireless-multicasting. In *IEEE 55th vehicular technology conference* (pp. 1160–1164).

Salama, H. F., Reeves, D. S., & Viniotis, Y. (1997). Evaluation of multicast routing algorithms for real-time communication on high-speed networks. *IEEE Journal on Selected Areas in Communications*, *15*, 332–345.

Skorin-Kapov, N., & Kos, M. (2003). The application of Steiner trees to delay constrained multicast routing: a tabu search approach. In *Proceedings of the seventh international conference on telecommunications*, Zagreb, Croatia.

Skorin-Kapov, N., & Kos, M. (2006). A GRASP heuristic for the delay-constrained MRP. *Telecommunications Systems*, *32*(1), 55–69.

Srinivas, N., & Deb, K. (1994). Multiobjective optimization using non-dominated sorting in genetic algorithms. *Evolutionary Computation*, *2*(3), 221–248.

Wang, J. Q., Qin, J., & Kang, L. S. (2006). A new QoS mutlicast routing model and its immune optimization algorithm. In J. Ma et al. (Eds.), *Lecture notes in computer science: Vol. 4159. Ubiquitous intelligence and computing* (pp. 369–378).

Waxman, B. M. (1988). Routing of multipoint connections. *IEEE Journal on Selected Areas in Communications*, *6*, 1617–1622.

Xu, Y., & Qu, R. (2011). Solving multi-objective multicast routing problems by evolutionary multi-objective simulated annealing algorithms with variable neighbourhoods. *Journal of the Operational Research Society*, *62*, 313–325.

Xu, Y., & Qu, R. (2012). An iterative local search approach based on fitness landscapes analysis for the delay-constrained multicast routing problem. *Computer Communications*, *35*, 352–365.

Yeo, C. K., Lee, B. S., & Er, M. H. (2004). A survey of application level multicast techniques. *Computer Communications*, *27*(15), 1547–1568.

Zahrani, M. S., Loomes, M. J., Malcolm, J. A., Dayem Ullah, A. Z. M., Steinhofel, K., & Albrecht, A. A. (2008). Genetic local search for multicast routing with pre-processing by logarithmic simulated annealing. *Computers & Operations Research*, *35*, 2049–2070.

Zitzler, E., & Thiele, L. (1999). Multiobjective evolutionary algorithms: a comprehensive case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation*, *3*(4), 257–271.

Zhang, Q., & Li, H. (2007). MOEA/D: a multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, *11*(6), 712–731.

Zhang, L., Cai, L. B., Li, M., & Wang, F. H. (2009). A method for least-cost QoS multicast routing based on genetic simulated annealing algorithm. *Computer Communications*, *32*, 105–110.