A multipopulation parallel genetic simulated annealing based QoS routing and wavelength assignment integration algorithm for multicast in optical networks

Hui Cheng[a,*], Xingwei Wang[b], Shengxiang Yang[a], Min Huang[b]

[a]Department of Computer Science, University of Leicester, University Road, Leicester LE1 7RH, UK
[b]College of Information Science and Engineering, Northeastern University, Shenyang 110004, China

Abstract

In this paper, we propose an integrated Quality of Service (QoS) routing algorithm for optical networks. Given a QoS multicast request and the delay interval specified by users, the proposed algorithm can find a flexible-QoS-based cost suboptimal routing tree. The algorithm first constructs the multicast tree based on the multipopulation-parallel-genetic-simulated-annealing algorithm, and then assigns wavelengths to the tree based on the wavelength graph. In the algorithm, routing and wavelength assignment are integrated into a single process. For routing, the objective is to find a cost suboptimal multicast tree. For wavelength assignment, the objective is to minimize the delay of the multicast tree, which is achieved by minimizing the number of wavelength conversion. Thus both the cost of multicast tree and the user QoS satisfaction degree can approach the optimal. Our algorithm also considers load balance. Simulation results show that the proposed algorithm is feasible and effective. We also discuss the practical realization mechanisms of the algorithm.

Keywords: Optical network; Multicast; Wavelength assignment; Multipopulation genetic simulated annealing algorithm

1. Introduction

Optical networks [1] have emerged as a promising candidate for next-generation networks providing high channel bandwidth and low communication latency. It is the essential requirement for next-generation networks to provide Quality of Service (QoS) [2] and multicast [3] capabilities. Hence, we need to address the issue of QoS multicast in optical networks. It means to develop efficient multicast routing algorithms, which can find the cost suboptimal multicast tree and assign wavelengths to it. It has been proved that finding such a tree is NP-hard [4].

A single population genetic algorithm [5] is powerful and performs well on a broad class of problems. However, better results can be achieved by introducing multiple populations (i.e., subpopulations). Every subpopulation evolves for a few generations independently (just like the single population genetic algorithm), and then one or more chromosomes are exchanged between these subpopulations. The Multipopulation parallel genetic algorithm [6] models the evolution of a species in a way more similar to nature than the single population genetic algorithm. There are three different models for parallel genetic algorithms, i.e., the global model, the diffusion model and the migration model.

In this paper, the proposed algorithm is based on the migration model. The migration model divides the population into multiple subpopulations. These subpopulations evolve independently from each other for a certain number of generations (isolation time). After the isolation time a number of chromosomes are exchanged between the subpopulations (migration). The number of exchanged chromosomes (migration rate), the selection method of the chromosomes for migration and the scheme of migration determine how much genetic diversity can occur in the subpopulations and the exchange of information between subpopulations.

Multipopulation parallel genetic algorithm and simulated annealing algorithm [7] are two standard techniques for hard combinatorial optimization problems. A new algorithm is developed by combining them together, which is named multipopulation parallel genetic simulated annealing algorithm (MPGSAA) [8-11]. Our proposed algorithm generates the cost suboptimal multicast tree based on MPGSAA, and then assigns wavelengths to the tree. The wavelength assignment algorithm is based on the basic idea of the wavelength graph proposed by Chlamtac [12]. The objective of wavelength assignment is to minimize the delay of the multicast tree, which is an important QoS parameter and decides the user QoS satisfaction degree. We integrate the algorithm for wavelength assignment into the process of the construction of the multicast tree. Thus we avoid that no wavelength can be assigned or the assignment result leads to a multicast tree with poor QoS performance. Therefore, the cost of the multicast tree can approach the optimal, and the user QoS requirement is also satisfied simultaneously.

The rest of this paper is organized as follows. Section 2 introduces the related work. Section 3 describes the network model and mathematical model. Section 4 describes the

* Corresponding author. Tel.: +44 116 2525295; fax: +44 116 2523915.

E-mail addresses: hc118@le.ac.uk (H. Cheng).

proposed algorithm and Section 5 discusses its practical implementation. Simulation results are presented in Section 6. We conclude the paper in Section 7.

2. Related work

In recent years, there are a few papers published in the area of multicast in WDM optical network. They can be divided into two types. The first type reports deterministic algorithms [13-17] and the second type reports GA-based non-deterministic algorithms [18-20]. Our proposed algorithm belongs to the non-deterministic algorithm. In the following, we review both deterministic and non-deterministic algorithms.

In [13], two integrated QoS multicast algorithms for routing and wavelength assignment were proposed. Both algorithms utilize Minimum Spanning Tree (MST) to construct low cost multicast trees. During the tree construction process, the case that the multicast end-to-end delay from the source to a destination exceeds the pre-specified upper bound is dealt with. The wavelength assignment is based on the greedy strategy, i.e., trying the best to assign a currently used wavelength to the multicast tree.

In [14], the objective of the QoS multicast algorithms is to minimize the number of used wavelengths. For a given set of multicast requests with bounded delay, the algorithms can construct trees and assign wavelengths. Two basic algorithms A and B were firstly proposed. Then two optimization algorithms C and D were proposed to further minimize the number of wavelengths over the results produced by A and B. Algorithm C and D integrate routing and wavelength assignment by using rerouting and reassigning techniques.

In [15], an algorithm was proposed, which consists of a heuristic multicast algorithm and an optimal wavelength assignment algorithm. It defines four kinds of costs related with the WDM multicast. The multicast tree is generated by combining the optimal unicast lightpaths and aims at minimizing the total cost of the multicast session. The objective of the wavelength assignment algorithm is to minimize the wavelength conversion cost of the multicast trees.

In addition, in [16], three low-cost, delay-bounded heuristic multicast algorithms LDR, ILDR and LDF were proposed. In [17], a distributed and sender-initiative routing and wavelength assignment algorithm was proposed for the establishment of a real-time multicast connection in WDM networks.

In [18], it considers the optimal multiple multicast problem on WDM ring networks without wavelength conversion. Given a set of multicast requests, it proposed several genetic algorithms to select a suitable path(s) and wavelength(s) for each request to minimize the used wavelengths. Since there is no wavelength conversion, there is a constraint that not any paths using the same wavelength pass through the same link. In [19], the multicast routing under delay constraint problem was considered in a WDM network with different light splitting. It firstly reduces the problem to the MST problem. Then it solves the problem by well-designed genetic algorithms.

In [13-17, 19, 20], the delay requirement is bounded by a fixed value and in [18] the delay is not considered. However, it is not enough for multicast applications where the users have flexible QoS requirements. The algorithms in [13, 14, 18] are only applicable to single-hop WDM networks, i.e., there is no wavelength conversion in networks. Hence, they pose a limitation that all the links in a tree can only use the same wavelength. The algorithm in [15] separates routing and wavelength assignment. As a result, it is possible that there are no available wavelengths for the multicast tree or the wavelength assignment result leads to poor QoS performance.

3. Model description

3.1 Network model

An optical network can be modeled by a directed and connected graph $G(V, E)$, where V is the set of nodes representing optical nodes and E is the set of edges representing optical fibers that connect the nodes. Each edge carries two oppositely-directed fibers for data transmission in the two directions of the edge. Each directed fiber is called a link.

Every node $v_i \in V$ has multicast capability by equipping an optical splitter [21]. We assume an optical signal can be split into an arbitrary number of optical signals at a splitter. Since the all-optical wavelength converter is still in its early development stage and the optoelectronic conversion not only is very expensive but also has limited performance, we assume only partial nodes are equipped with full-range wavelength converter [21] in the network. The full-range wavelength converter is able to convert optical signal on one wavelength into any other wavelengths. The wavelength conversion also introduces additional processing and control delay called wavelength conversion delay. Without loss of generality, we assume the conversion between any two different wavelengths has the same delay at any optical node with the wavelength converter, i.e., $t(v_i) \equiv t$. If there is no wavelength conversion at an intermediate node v_i, we set $t(v_i) = 0$.

Each link $e_{ij} = (v_i, v_j) \in E$ is associated with three parameters:

- $\Lambda(e_{ij})$, the set of available wavelengths. $\Lambda(e_{ij}) \subseteq \Lambda = \{\lambda_1, \lambda_2, \cdots, \lambda_w\}$, Λ is the set of wavelengths supported by each link in the network.
- $\delta(e_{ij})$, the transmission delay. Here, $\delta(e_{ij}) = \delta(e_{ji})$.

- $c(e_{ij})$, the link cost.

3.2 Mathematical model

In graph $G(V, E)$, we consider a multicast request for multicast connection setup, $R(s, D, \Delta)$, where s is the source node, D is the set of destinations. Different from the previous literatures [13-17, 19, 20], we define Δ as the delay requirement interval specified by the user. It is more practical to represent the delay requirement by an interval than by a single value because in practice the network information is inaccurate and the user QoS requirement is often flexible [22]. The lower bound and the upper bound of the delay interval are determined by the user and the application.

The route of the multicast connection is represented by a tree $T = (X_T, F_T)$, $X_T \subseteq V$, $F_T \subseteq E$. The total cost of T is defined as

$$Cost(T) = \sum_{e_{ij} \in F_T} c(e_{ij}) \cdot \tag{1}$$

The communication delay on a path consists of two components, i.e., link transmission delay and wavelength conversion delay. Let $P(s, d_i)$ denote the path from source node s to any destination node d_i in T and let D_{sd_i} denote the path delay. We have

$$D_{sd_i} = \left[\sum_{v_i \in P(s,d_i)} t(v_i) + \sum_{e_{ij} \in P(s,d_i)} \delta(e_{ij}) \right]. \tag{2}$$

The delay of T is defined as

$$Delay(T) = \max\{D_{sd_i},\ \forall d_i \in D\}, \tag{3}$$

which is the maximum delay between the source node and all the destination nodes. We set $\Delta = [\Delta_{low}, \Delta_{high}]$ and then the user QoS satisfaction degree is defined as

$$Degree(QoS) = \begin{cases} 100\% & Delay(T) \le \Delta_{low} \\ \dfrac{\Delta_{high} - Delay(T)}{\Delta_{high} - \Delta_{low}} & \Delta_{low} < Delay(T) < \Delta_{high} \\ 0\% & Delay(T) \ge \Delta_{high} \end{cases}. \tag{4}$$

The algorithm should select the links with more available wavelengths to balance the network load and thereby reduce the call blocking probability. The load on a link is defined as the number of channels over that link. We can adjust it by defining proper link cost functions. For example, by defining heuristic cost functions, for the link with more available wavelengths, the cost takes smaller value. In the proposed algorithm, we define

$$c(e_{ij}) = w - |\Lambda(e_{ij})| \cdot \tag{5}$$

The key optimization objective considered in this paper is to minimize the tree cost while the user QoS satisfaction degree is still high. In addition, the end-to-end delay of tree T^* should not exceed the upper bound of the delay interval. Otherwise the user cannot accept it due to the poor QoS performance. Furthermore, for any link on tree T^*, there should exit at least one available wavelength. Otherwise, the multicast connection cannot be set up. We use T to denote any multicast tree spanning s and D in $G(V, E)$. Therefore, we solve the problem of QoS multicast in the optical network by finding an optimal multicast tree $T^*(X_{T^*}, F_{T^*}), \{s\} \cup D \subseteq X_{T^*}, F_{T^*} \subseteq E$, which minimizes

$$Cost(T^*) = \min_T\{Cost(T)\}, \tag{6}$$

subject to

$$Delay(T^*) \le \Delta_{high}; \tag{7}$$

$$\forall e_{ij} \in F_{T^*},\quad |\Lambda(e_{ij})| \ge 1 \cdot \tag{8}$$

4. The proposed algorithm

4.1 Expression of the solution

We denote the solution by binary coding. Each bit of binary string corresponds to a different network node. The graph corresponding to the solution S is $G'(V', E')$. Let the function $bit(S, i)$ denotes the ith bit of S. If and only if $bit(S, i)$=1, then $v_i \in V'$. For our problem, every solution S corresponds to a tree $T_i'(X_i', F_i')$, which is the minimum cost spanning tree of G'. T_i' spans the source node and all the destination nodes.

Another problem is that G' may be unconnected. Thus, every subgraph of G' has a minimum cost spanning tree, the solution S corresponds to a minimum cost spanning forest, which is also denoted by $T_i'(X_i', F_i')$. If G' is unconnected, we add penalty value to the cost and give smaller $Degree(QoS)$ to the solution. Thus, every solution S corresponds to a graph G', which corresponds to a minimum cost spanning forest T_i' (a forest can have only one tree). After pruning, we obtain the forest T_i', which corresponds to solution S.

4.2 The algorithm for wavelength assignment

If T_i is a tree, we assign wavelengths to it. The objective of the proposed wavelength assignment algorithm is to minimize the delay of the tree by minimizing the number of wavelength conversion. Thus the user can get a high QoS satisfaction degree.

The proposed algorithm is based on the idea of wavelength graph [12]. First we construct wavelength graph WG for the tree T_i (X_i, F_i). The construction method is stated as follows.

1) $N = |X_i|$, $w = \left| \bigcup_{e_{ij} \in F_i} \Lambda(e_{ij}) \right|$. In WG, we create $N * w$ number of nodes, namely v_{ij}, for $i = 1, 2, \cdots, w$ and $j = 1, 2, \cdots N$. All the nodes are arranged into a matrix with w rows and N columns. Row i represents the corresponding wavelength λ_i' and each column j represents a node v_j' in

T_i. A mapping table is created to record the corresponding relationship between i and λ'_i, and another is created to record the relationship between j and v'_j. The two tables will help reversely map the paths in *WG* back to the paths and wavelengths in T_i.

2) For $i = 1, 2, \cdots, w$, in the ith row, we add a horizontal directional link (v_{ij}, v_{ih}) between column j and column h if there exists a link $e'_{jh} = (v'_j, v'_h)$ in T_i from node v'_j to node v'_h and the wavelength λ'_i is available on this link. We assign the transmission delay $\delta(e'_{jh})$ as its weight.

3) For $j = 1, 2, \cdots N$, in the jth column, for $\forall i_1, i_2, i_1 \neq i_2$, we add a vertical bidirectional link $(v_{i_1 j}, v_{i_2 j})$ between row i_1 and row i_2 if node v'_j in T_i has the wavelength conversion capability. We assign the wavelength conversion delay t as its weight.

Using the above steps the wavelength graph *WG* is constructed. A vertical link in *WG* represents a wavelength conversion at a node and a horizontal link in *WG* represents an actual link in T_i. For convenience, we denote the nodes in *WG* by sequential node number $1 \sim N * w$. The sequential node number for the node in the ith row and jth column in *WG* is

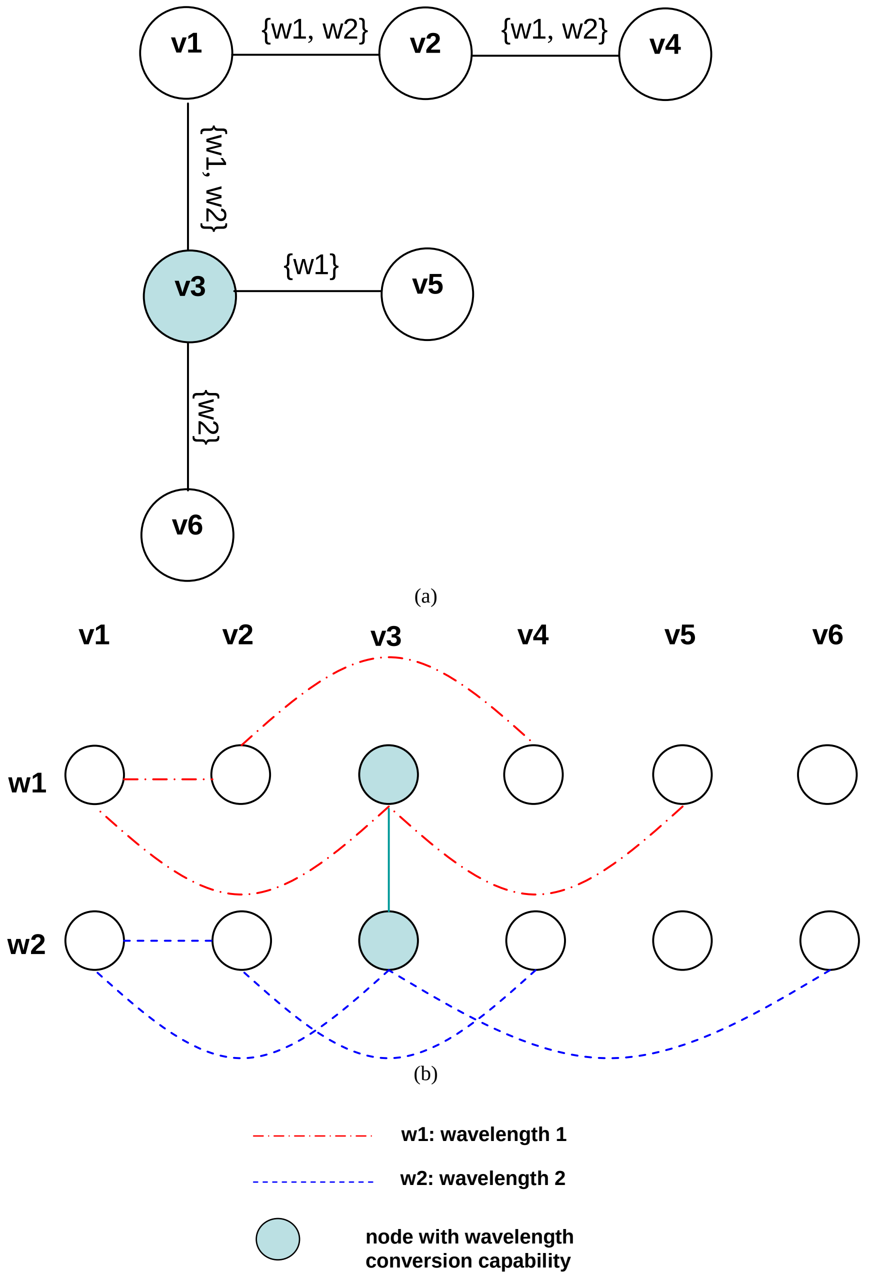

Fig. 1. The illustration of the construction of a wavelength graph: (a) physical network topology, (b) the corresponding wavelength graph.

$x = (i-1) * N + j$. (9)

We treat the wavelength graph *WG* as an ordinary network topology graph and run the following wavelength assignment algorithm.

Fig. 1 illustrates an example of constructing the wavelength graph. Fig. 1(a) is the physical network topology *G* where v1 to v6 represent the optical nodes. In the bracket near a link, w1 and/or w2 represent that wavelength 1 and/or wavelength 2 are available on that link. Node v3 is an optical node with wavelength conversion capability. Fig. 1(b) is the generated wavelength graph corresponding to the physical network topology.

Wavelength Assignment Algorithm

Input: the wavelength graph *WG* where the source node and all the destination nodes correspond to the column numbers in the matrix, i.e., $j_s, j_{d_1}, j_{d_2} \cdots, j_{d_m}$.

Output: the wavelength assignment result for tree T_i.

begin

for $(k = 1, k \le m, k++)$

{

for $(i = 1, i \le w, i++)$

{

$x_i = (i-1) * N + j_s$;

for $(j = 1, j \le w, j++)$

{

$y_{jk} = (j-1) * N + j_{d_k}$;

Apply the Dijkstra's shortest path algorithm to find the shortest path $P(x_i, y_{jk})$ from node x_i to node y_{jk};

}

$P(x_i, y_k) = \min\{P(x_i, y_{jk}), 1 \le j \le w\}$;

}

$P(x, y_k) = \min\{P(x_i, y_k), 1 \le i \le w\}$;

}

end

$P(x, y_k)$ is the shortest path from source node *s* to destination node d_k in *WG*. We have

$i = (x-1)/N + 1$, (10)

$j = (x-1)\%N + 1$. (11)

Using the above two expressions and the two mapping tables created in step 1, we can reversely map the paths consisting of the sequential node numbers back to the links and wavelengths in T_i conveniently. Thus the wavelength assignment is completed.

The time complexity of the above wavelength assignment algorithm is $O(mN^2w^4)$, where *m* is the number of destination nodes, *N* is the number of nodes in T_i, *w* is the number of wavelengths which are available on at least one link in T_i. We can see that they all take small integer values. In addition, all the wavelength assignments for solutions except the final solution will not be used as the final wavelength assignment result. Hence, the algorithm need not store lots of data and has a low space complexity.

4.3 Fitness function

After assigning wavelengths to T_i, the delay of T_i is determined and thereby *Degree*(*QoS*) is determined. The fitness of solution *S* is obtained by computing the following fitness function

$$f(S) = \frac{Cost(T_i) + [count(T_i) - 1] * \rho}{Degree(QoS)} = \frac{\sum_{e_{ij} \in F_{T_i}} c(e_{ij}) + [count(T_i) - 1] * \rho}{Degree(QoS)}, \quad (12)$$

where *count*(T_i) is the number of trees in the forest T_i, ρ is a constant. We can see that a smaller *f*(*S*) corresponds to a better solution.

4.4 Setting the initial temperature

We set the initial temperature $t_0 = K\delta$, where *K* is a sufficiently large number, and

$$\delta = \max\{f(j) \mid j \in Sp\} - \min\{f(j) \mid j \in Sp\}, \quad (13)$$

where *Sp* denotes the solution space. δ can be estimated simply as follows. Since $\max\{f(j) \mid j \in Sp\} \le C_{graph}$ (i.e., the total cost of the current network topology), and $\min\{f(j) \mid j \in Sp\} \ge C_{s \cup D}$ (i.e., the cost of the minimum spanning tree covering *s* and *D*), we have

$$\delta = C_{graph} - C_{s \cup D}. \quad (14)$$

Due to the use of the penalty value, the cost of the solution may be larger than C_{graph} after the penalty value is added. To satisfy $\max\{f(j) \mid j \in Sp\} \le C_{graph}$, we let $f(S) = C_{graph}$ when $f(S) > C_{graph}$.

4.5 Formal description of the algorithm

We first initialize the control parameters including the subpopulations number *M*, the size for every subpopulation n_p, the predefined maximum generation number *MAX_GN*, the individual generation number n_G, the crossover probability $\rho_c(i)$ for subpopulation *i* ($1 \le i \le M$), the mutation probability $\rho_m(i)$ for subpopulation *i* ($1 \le i \le M$), the coefficient of decreasing temperature α, the initial temperature $t_0(i)$ for subpopulation *i* ($1 \le i \le M$).

1) initialize *M* random subpopulations. Set *GN*=0, where *GN* denotes the generation number that the subpopulation has evolved so far. Set *k*=0, where *k* denotes the number of temperature decrease. Set $f(Sop) = \infty$, where *Sop* denotes the global optimal solution. Set *Loop*=0, where *Loop* is a counter variant.

2) If $Loop < n_G$, go to step 3; otherwise, go to step 5.

3) For subpopulation i ($1 \le i \le M$), perform the following operations to generate an offspring subpopulation.

a) Evaluate the fitness of every chromosome: $f(S_j),\ j = 1,2,\cdots n_p$;

b) Select the chromosomes $S_j, S_k\ (j \ne k)$ randomly and generate a random number $num \in [0,1]$. If $num > \rho_c(i)$, S_j, S_k are accepted for offspring subpopulation directly; otherwise, perform the crossover operation to generate two new chromosomes S'_j, S'_k.

c) Evaluate the fitness $f(S'_j), f(S'_k)$. We define $\Delta f' = f(S'_j) - f(S_j)$. If $\Delta f' < 0$, accept S'_j for offspring subpopulation; if $\Delta f' > 0$, then accept S'_j for offspring subpopulation at the probability $\exp(-\Delta f' / t_k(i))$. We have $\Delta f' = f(S'_k) - f(S_k)$. If $\Delta f' < 0$, accept S'_k for offspring subpopulation; if $\Delta f' > 0$, then accept S'_k for offspring subpopulation at the probability $\exp(-\Delta f' / t_k(i))$. If S'_j, S'_k are not accepted, S_j, S_k are accepted for offspring subpopulation directly. Repeat b) and c) $n_p / 2$ times, and get the offspring subpopulation i'.

d) For every chromosome S_j in i', generate a random number $num \in [0,1]$. If $num > \rho_m(i)$, S_j is accepted for offspring subpopulation directly; otherwise, perform the mutation operation to generate a new chromosome S'_j. Using the above method mentioned in c) to decide whether or not to accept S'_j for offspring subpopulation. If not, S_j is accepted for offspring subpopulation directly. After this operation, denote the offspring subpopulation as subpopulation i.

4) GN=GN+1, $Loop$=$Loop$+1, go to step 2.

5) First find the optimal chromosome in each subpopulation, and we get M chromosomes. Then find the optimal one S among the M chromosomes. Replace the worst chromosome of every subpopulation using S. If $f(S) < f(Sop)$, $Sop \leftarrow S$ (i.e., replace Sop using S).

6) If GN=MAX_GN, the algorithm stops; otherwise, modify the annealing temperature for each subpopulation, i.e., $t_{k+1}(i) = \alpha t_k(i)\ (k \ge 0, 0 < \alpha < 1, 1 \le i \le M)$. k=k+1, $Loop$=0. Go to step 2.

When the algorithm terminates, Sop is output as the final solution.

5. Discussion on the algorithm implementation

Parallel algorithms are developed to speed up the computation by harnessing the power of parallel computers or multiple processors computer. During the parallel evolution process of the multiple subpopulations, each subpopulation evolves independently from each other for a certain number of generations (isolation time). After the isolation time the optimal solution (chromosome) is exchanged between all the subpopulations.

We assume that the population size of each subpopulation is the same and that the crossover probability, mutation probability and temperature control parameters of each subpopulation may be different. This is a synchronous parallel algorithm. The implementation of the algorithm should adopt the Multiple Instruction stream Multiple Data stream (MIMD) computer architecture [23]. The number of processors should be the same as the number of subpopulations, and each processor processes the evolution of a subpopulation independently.

The synchronization mechanism is needed among different processes operating on different processors, i.e., after one processor finishes its isolation time, it stops to judge if the other ones have finished their isolation time. If there exists one processor which has not finished yet, all the others which have finished must wait till all the processors finish their isolation time.

There are two kinds of realization mechanisms for MPGSAA. One is to establish the shared memory and the other is to designate the control processor. The first method will establish a shared memory for all the subpopulations. Thus all the subpopulations communicate through a global shared variant. The present global optimal solution is also exchanged among all the subpopulations through the global shared variant. Since the global shared variant is a type of critical resource, the lock mechanism should apply to it. Each processor should create its own critical region for the global shared variant to realize the synchronization among all the processors. Fig. 2(a) illustrates the shared memory method. The second method designates a new processor as the control processor. The control processor can also be designated by election from all the processors used to process the subpopulations. The control processor is responsible for the distribution of the present global optimal solution and the synchronization among all the processors. Fig. 2(b) illustrates the control processor method.

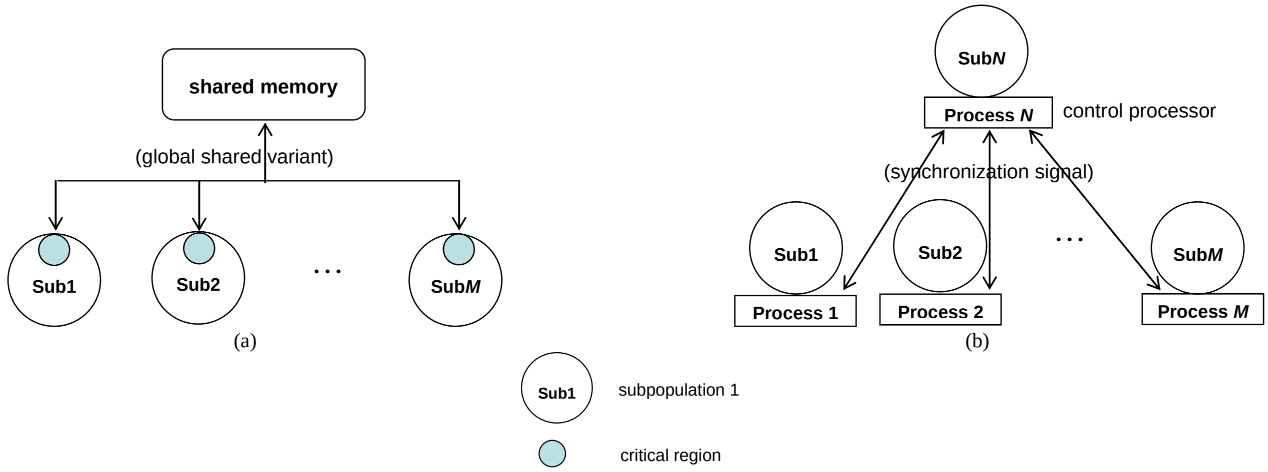

Fig. 2. Two possible realization mechanisms for MPGSAA: (a) the shared memory mechanism, (b) the control processor mechanism.

6. Performance evaluation

Due to hardware constraint, our simulation experiments are conducted on a single processor computer, and the parallel algorithm is implemented in a serial manner (pseudo-parallel). The following performance evaluation is based on NSFNET network topology [24]. Since the optimization objective of the proposed algorithm is to minimize the tree cost while the user QoS satisfaction degree is high, there is a tradeoff between the tree cost and delay. Hence, we evaluate the algorithm in two aspects, i.e., the

Table 1
The cost comparison results between the final solutions obtained by GA and the corresponding optimal solutions

Ratio of multicast nodes in the network	Delay interval	Running times	Optimal fitness value	<=1%	<=2%	<=5%	<=10%	<=20%	>20%
21.4%	(15, 30)	100	33	0.78	0.00	0.08	0.00	0.04	0.10
28.6%	(15, 30)	100	39	0.90	0.00	0.00	0.00	0.08	0.02
35.7%	(15, 30)	100	38	0.86	0.00	0.00	0.00	0.00	0.14
42.9%	(15, 30)	100	48	0.33	0.00	0.00	0.54	0.13	0.00
50.0%	(15, 30)	100	46	0.80	0.00	0.00	0.07	0.09	0.04
57.1%	(15, 30)	100	50	0.66	0.10	0.00	0.00	0.00	0.24
64.3%	(20, 40)	100	50	1.00	0.00	0.00	0.00	0.00	0.00
71.4%	(20, 40)	100	64	0.53	0.16	0.00	0.25	0.06	0.00
78.6%	(20, 40)	100	69	1.00	0.00	0.00	0.00	0.00	0.00
92.9%	(20, 40)	100	73	1.00	0.00	0.00	0.00	0.00	0.00

Table 2
The cost comparison results between the final solutions obtained by the proposed MPGSAA algorithm and the corresponding optimal solutions

Ratio of multicast nodes in the network	Delay interval	Running times	Optimal fitness value	<=1%	<=2%	<=5%	<=10%	<=20%	>20%
21.4%	(15, 30)	100	33	0.88	0.00	0.06	0.00	0.02	0.04
28.6%	(15, 30)	100	39	0.89	0.00	0.00	0.00	0.11	0.00
35.7%	(15, 30)	100	38	0.96	0.00	0.00	0.00	0.00	0.04
42.9%	(15, 30)	100	48	0.80	0.00	0.00	0.13	0.07	0.00
50.0%	(15, 30)	100	46	0.98	0.00	0.00	0.00	0.02	0.00
57.1%	(15, 30)	100	50	0.96	0.02	0.00	0.00	0.00	0.02
64.3%	(20, 40)	100	50	1.00	0.00	0.00	0.00	0.00	0.00
71.4%	(20, 40)	100	64	0.85	0.05	0.00	0.10	0.00	0.00
78.6%	(20, 40)	100	69	1.00	0.00	0.00	0.00	0.00	0.00
92.9%	(20, 40)	100	73	1.00	0.00	0.00	0.00	0.00	0.00

Table 3
The delay comparison results between GA considering QoS and GA without considering QoS

Ratio of multicast nodes in the network	Delay interval	Running times	Maximum average end-to-end delay when QoS is not considered	Maximum average end-to-end delay when QoS is considered
21.4%	(15, 30)	5	18.6	16.2
28.6%	(15, 30)	5	16.8	15.8
35.7%	(15, 30)	5	22.6	16.6
42.9%	(15, 30)	5	26.4	19.4
50.0%	(15, 30)	5	22.2	19.2
57.1%	(15, 30)	5	25.4	20.0
64.3%	(20, 40)	5	33.0	33.0
71.4%	(20, 40)	5	26.0	23.2
78.6%	(20, 40)	5	31.0	23.0
92.9%	(20, 40)	5	29.0	29.0

Table 4
The delay comparison results between MPGSAA considering QoS and MPGSAA without considering QoS

Ratio of multicast nodes in the network	Delay interval	Running times	Maximum average end-to-end delay when QoS is not considered	Maximum average end-to-end delay when QoS is considered
21.4%	(15, 30)	5	19.0	16.0
28.6%	(15, 30)	5	17.4	15.0
35.7%	(15, 30)	5	27.0	15.0
42.9%	(15, 30)	5	25.2	19.2
50.0%	(15, 30)	5	25.0	19.2
57.1%	(15, 30)	5	26.4	20.0
64.3%	(20, 40)	5	33.0	33.0
71.4%	(20, 40)	5	25.0	22.0
78.6%	(20, 40)	5	33.0	23.0
92.9%	(20, 40)	5	29.0	29.0

cost and the delay of the final multicast tree. Since single population GA has been widely used to solve the QoS multicast problem in WDM optical network [18-20], we compare our algorithm with it to show the performance improvements.

Referring to the simulation model established in [17], we set the transmission delay on each link to be a small integer in [1, 10], which is also in direct proportion to the length of the link. We set the wavelength conversion delay to be a constant integer in [1, 10]. We choose the 50% of all the nodes which have higher node degrees to be equipped with wavelength converters. We set $|\Lambda| = 20$ and $10 \le |\Lambda(e_{ij})| \le 15$.

If the fitness values of some chromosomes are too large, the difference between other chromosomes will be shielded. To avoid it, when *Degree*(*QoS*) is less than a small value *val*, we take *Degree*(*QoS*)=*val* in the fitness calculation. If the solution corresponding to the chromosome is unfeasible, we also take *Degree*(*QoS*)=*val*. By running extensive simulation experiments, we have chosen the appropriate values for the parameters of MPGSAA.

6.1 The evaluation on the tree cost

Both single population GA and the proposed MPGSAA algorithm are run to obtain the multicast trees. We run each algorithm 100 times and get 100 final solutions for each multicast session. We compare them with the optimal multicast tree, which is obtained by exhaustive search. The results are shown in Table 1 and Table 2, respectively.

In both Table 1 and Table 2, <=1% means that the ratio of the cost deviation of the final solution (i.e., the difference between the cost of the final solution and the cost of the optimal solution) to the cost of the optimal solution is <=1%. <=2% means that the ratio is >1% and <=2%. Similar meanings apply to other ratio intervals. The value under each ratio interval means the percentage of the final solutions whose cost deviation ratios fall into this interval.

These multicast session nodes are chosen randomly from sparse mode [25] to dense mode [26]. From these two tables we can get that for the actual topology like NSFNET, the quality of the final solutions obtained by the proposed MPGSAA algorithm is very good in terms of the cost. To show the performance improvement of the proposed MPGSAA algorithm over the single population GA, we plot

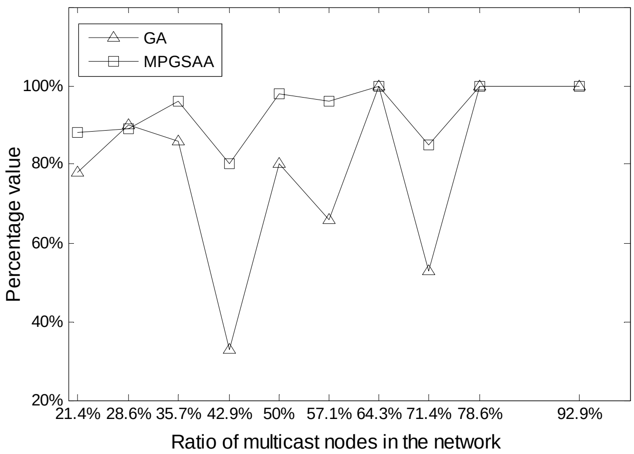

Fig. 3. The comparison between GA and MPGSAA in terms of their percentage values of the solutions falling into the ratio interval <=1%.

Fig. 3 to compare their percentage values of the solutions falling into the ratio interval <=1%.

6.2 The evaluation on the delay

We define the concept of the user QoS satisfaction degree and consider the QoS performance of chromosomes when the fitness values are calculated in MPGSAA. Hence, we evaluate both the cost and the maximum end-to-end delay when choosing chromosomes. The use of the user QoS satisfaction degree helps to make an ideal tradeoff between the cost and the delay of the multicast trees.

To evaluate the performance improvement made by using the user QoS satisfaction degree, we also run both GA and MPGSAA under the scenario that QoS (i.e., the user QoS satisfaction degree) is not considered. Then we compare the delay of the multicast trees obtained by the algorithms considering QoS and the one obtained by the algorithms without considering QoS. The results are shown in Table 3 and Table 4, respectively.

From Table 3 and Table 4, we can see that the delay of the multicast trees obtained by the algorithm considering QoS is less than the one without considering QoS. It proves that with the use of the user QoS satisfaction degree, we can achieve the multicast trees with a better QoS performance. We plot Fig. 4 to compare the maximum average end-to-end delay of the multicast trees obtained by the proposed MPGSAA algorithm and the single population GA.

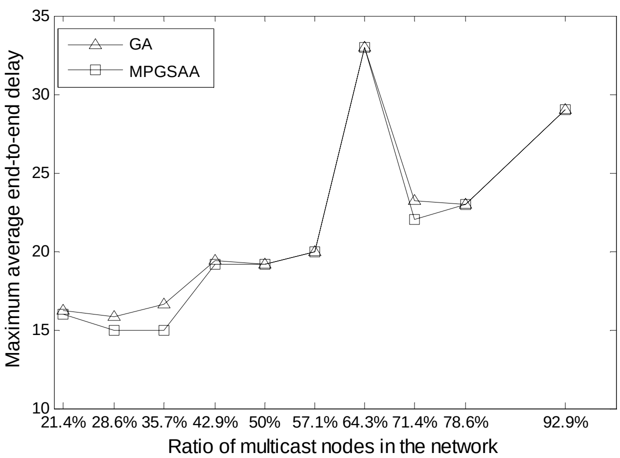

Fig. 4. The comparison between GA and MPGSAA in terms of the maximum average end-to-end delay.

From Fig. 3 and Fig. 4, we can see that the proposed MPGSAA algorithm performs better than GA in terms of both the tree cost and the end-to-end delay. Furthermore, the MPGSAA algorithm overcomes the drawback of premature convergence of the traditional GA and has better stability. What the MPGSAA has paid for the performance improvements is larger memory storage space and more powerful hardware.

6.3 The theoretical comparison on the time consumption

We now theoretically compare the time consumption of single population GA with that of MPGSAA. Since the predefined maximum generation number of MPGSAA is *MAX_GN* and the individual generation number is n_G, the times that the global optimal solution needs to be exchanged are $\left\lceil MAX\_GN / n_G \right\rceil$. Since we have M subpopulations in MPGSAA, the maximum generation number of GA is M*MAX_GN. We assume that the average time consumption, the average maximum time consumption, and the average minimum time consumption of each isolated evolution are $\overline{T_G}$, $\overline{T_G^{Max}}$, and $\overline{T_G^{Min}}$, separately. We have

$$\overline{T_G^{Min}} < \overline{T_G} < \overline{T_G^{Max}} . \quad (15)$$

We assume that in MPGSAA the average time consumption to determine the global optimal solution is $\overline{T_{Deter}}$, and the average time consumption to exchange the global optimal solution is $\overline{T_{Excha}}$. We use $\overline{T_{sync}}$ to denote the average synchronization time after the isolated evolution of all the subpopulations. We have

$$\overline{T_{sync}} = \overline{T_{Deter}} + \overline{T_{Excha}} . \quad (16)$$

We use T_{GA} and T_{MPGSAA} to denote the total time consumption of GA and MPGSAA, respectively. We have

$$T_{GA} = \left\lceil M * MAX\_GN / n_G \right\rceil * \overline{T_G} , \quad (17)$$

Since within each isolated evolution, the subpopulation which has the maximum time consumption determines the ending time, the average time of each isolated evolution is $\overline{T_G^{Max}}$ in MPGSAA. Therefore, we have

$$T_{MPGSAA} = \left\lceil MAX\_GN / n_G \right\rceil * (\overline{T_G^{Max}} + \overline{T_{sync}}) . \quad (18)$$

The difference of the time consumption between GA and MPGSAA is

$$\Delta T = T_{GA} - T_{MPGSAA} = \left\lceil MAX\_GN / n_G \right\rceil * (M * \overline{T_G} - \overline{T_G^{Max}} - \overline{T_{sync}}) \quad (19)$$

If we fix MAX_GN and take $\overline{T_G}$, $\overline{T_G^{Max}}$ and $\overline{T_{sync}}$ as constants, we can see that ΔT is mostly related to M and n_G. It means that more subpopulations and less individual generation number will lead to more time savings in MPGSAA.

7. Conclusions

In this paper, we first analyze the actual optical networks to abstract the network model, and then define the mathematical model for the QoS multicast routing problem in optical networks. Due to the problem complexity and network dynamics, the network state information cannot be accurate inherently. Hence, it is more practical for the user to propose the QoS requirements in a flexible way, e.g., by the delay interval. So we define a new concept- the user QoS satisfaction degree.

Based on the MPGSAA and the idea of wavelength graph, we propose a QoS multicast routing algorithm for optical networks. By the elaborate design of MPGSAA, the proposed algorithm can find a cost suboptimal routing tree. Each time a feasible multicast tree is found, we assign wavelengths to it with the goal of minimizing the end-to-end delay. Thus, we integrate the wavelength pre-assigning into the routing tree construction. A better tradeoff between the cost and the end-to-end delay is achieved for evaluating the quality of a multicast tree. By simulations, we evaluate the performance of the proposed algorithm in terms of the multicast tree cost and the multicast end-to-end delay, respectively. The results show that the proposed algorithm has a better performance than a single population GA.

Acknowledgements

The authors would like to thank anonymous reviewers for their valuable comments to improve the paper. This work was supported in part by the Engineering and Physical Sciences Research Council (EPSRC) of UK under Grant EP/E060722/1, the National Natural Science Foundation of China under Grant No. 60673159 and 70671020, the National High-Tech Research and Development Plan of China under Grant No. 2006AA01Z214, Program for New Century Excellent Talents in University, and the Key Project of Chinese Ministry of Education under Grant No. 108040.

References

[1] M.J. O'Mahony, C. Politi, D. Klonidis, R. Nejabati, D. Simeonidou, Future optical networks, J. Lightw. Technol. 24 (12) (2006) 4684-4696.

[2] X.P. Xiao, L.M. Ni, Internet QoS: the big picture, IEEE Netw. 13 (2) (1999) 8-18.

[3] R. Wittmann, M. Zitterbart, Multicast Communication: Protocols, Programming, and Applications, Morgan Kaufman, 2000, San Francisco, USA.

[4] J.P. Wang, X.T. Qi, B. Chen, Wavelength assignment for multicast in all-optical WDM networks with splitting constraints, IEEE/ACM Trans. Netw. 14 (1) (2006) 169-182.

[5] M. Gen, R.W. Cheng, Genetic Algorithms and Engineering Design, John Wiley & Sons, 1996, New York, USA.

[6] P. Hartmut, Development and Engineering Application of Evolutionary Algorithms, PhD thesis, 1998, Technical University Ilmenau, Germany.

[7] F. Mendivil, R. Shonkwiler, M.C. Spruill, Restarting search algorithms with applications to simulated annealing, Adv. in Appl. Probab. 33 (1) (2001) 242-259.

[8] H. Chen, N.S. Flann, D.W. Watson, Parallel genetic simulated annealing: a massively parallel SIMD algorithm, IEEE Trans. Parallel Distrib. Syst. 9 (2) (1998) 126-136.

[9] Z.G. Wang, M. Rahman, Y.S. Wong, K.S. Neo, Development of heterogeneous parallel genetic simulated annealing using multi-niche crowding, Int. J. Comput. Intell. 3 (1) (2007) 55-62.

[10] G. Rudolph, Massively parallel simulated annealing and its relation to evolutionary algorithms, Evol. Comput. 1 (4) (1993) 361-383.

[11] X. Liu, H.M. He, O. Sýkora, Parallel genetic algorithm and parallel simulated annealing algorithm for the closest string problem, Lecture Notes in Comput. Sci. 3584 (2005) 591-597.

[12] I. Chlamtac, A. Farago, T. Zhang, Lightpath (Wavelength) routing in large WDM networks, IEEE J. Sel. Areas Commun. 14 (5) (1996) 909-913.

[13] X.H. Jia, D.Z. Du, X.D. Hu, Integrated algorithms for delay bounded multicast routing and wavelength assignment in all optical networks, Comput. Commun. 24 (14) (2001) 1390-1399.

[14] X.H. Jia, D.Z. Du, X.D. Hu, M.K. Lee, J. Gu, Optimization of wavelength assignment for QoS multicast in WDM networks, IEEE Trans. Commun. 49 (2) (2001) 341-350.

[15] B. Chen, J.P. Wang, Efficient routing and wavelength assignment for multicast in WDM networks, IEEE J. Sel. Areas Commun. 20 (1) (2002) 97-109.

[16] T.F. Znati, T. Alrabiah, R. Melhem, Low-cost, delay-bounded point-to-multipoint communication to support multicasting over WDM networks, Comput. Netw. 38 (4) (2002) 423-445.

[17] C.H. Huang, X.H. Jia, Y. Zhang, A distributed routing and wavelength assignment algorithm for real-time multicast in WDM networks, Comput. Commun. 25 (17) (2002) 1527-1535.

[18] D.-R. Din, Genetic algorithms for multiple multicast on WDM ring network, Comput. Commun. 27 (9) (2004) 840-856.

[19] M.T. Chen, S.S. Tseng, A genetic algorithm for multicast routing under delay constraint in WDM network with different light splitting, J. Inf. Sci. Eng. 21 (1) (2005) 85-108.

[20] A.E. Jeyakumar, K. Baskaran, V. Sumathy, Genetic algorithm for optimal design of delay bounded WDM multicast networks, in: Proceedings of Conference on Convergent Technologies for Asia-Pacific Region (TENCON), 2003, vol. 3, pp. 1224-1228.

[21] A.J. Ding, G.S. Poo, A survey of optical multicast over WDM networks, Comput. Commun. 26 (2) (2003) 193-200.

[22] H.L. Dean, O. Ariel, QoS routing in networks with uncertain parameters, IEEE/ACM Trans. Netw. 6 (6) (1998) 768-778.

[23] E.L. Lafferty, M.J. Prelle, M.C. Michaud, J.B. Goethert, Parallel Computing: An Introduction, William Andrew Inc., 1993, New York, USA.

[24] D. Saha, M.D. Purkayastha, A. Mukherjee, An approach to wide area WDM optical network design using genetic algorithm, Comput. Commun. 22 (2) (1999) 156-172.

[25] D. Estrin, et al, Protocol independent multicast sparse-mode (PIM-SM): protocol specification, RFC 2362, June 1998.

[26] A. Andrew, et al, Protocol independent multicast-dense mode (PIM-DM): protocol specification, RFC 3973, January 2005.