A Simulated Evolution-Tabu Search Hybrid Metaheuristic for Routing in Computer Networks

Salman A. Khan, Member, IEEE, Zubair A. Baig, Member, IEEE

Abstract—Routing in computer networks is a nonlinear combinatorial optimization problem with numerous constraints and is classified as an NP-complete problem. There are certain important QoS metrics which affect the performance of a network. One of these metrics is the average network delay, which should be minimized. In this paper, a routing strategy based on Simulated Evolution algorithm to find suboptimal routing solution for computer networks while optimizing the above metric is presented. To intensify the search, a hybrid variant of the proposed algorithm has also been implemented. This variant incorporates Tabu Search characteristics into the Simulated Evolution algorithm. Performance evaluation of the two approaches is done via simulation. Empirical results suggest that the hybrid variant performs better than the original Simulated Evolution algorithm.

I. INTRODUCTION

Design of computer communication networks is a complex problem. One aspect of this problem is to implement a routing scheme between nodes such that an acceptable performance level is attained. One important factor that affects this performance level is the average network delay per packet in the network. An efficient routing scheme should allow the selection of best possible path (route) between two nodes. For a single source-destination pair of nodes, selection of an optimum route out of the available routes is not a complex task. However, if all source-destination pairs in the network are considered simultaneously and attempts are made to optimize some network parameters, the task becomes very complicated. There are many network configurations that can be achieved by changing the selected route of one source-destination (SD) pair while fixing the remaining SD pairs. The search space for such optimization problem is large even for small networks. For instance, consider the network shown in Figure 1(a) with 6 nodes and 11 links. There are at least 8 possible routes between each SD pair. The total number of SD pairs in the network is M(M-1), where M is the number of nodes. This results in 30 SD pairs. Therefore, there are at least $8^{30}$ different combinations (or network configurations) which is a large number for such a small network. Thus, an effective method is needed for finding quasi-optimal routing solution.

A category of algorithms that have been found to be effective for complex problems such as routing are iterative metaheuristics. These metaheuristics traverse the state space of solutions while evaluating each solution against any desirable set of properties. These metaheuristics are characterized by hill climbing property that allows occasional acceptance of inferior solutions [1]. Heuristics like genetic algorithm [2], simulated annealing [3], tabu search [4], simulated evolution [5], stochastic evolution [6], ant colony optimization [7], and particle swarm optimization [8] are examples of stochastic iterative heuristics.

A number of iterative heuristics have been applied to computer networks routing. In [9], Sim et al. proposed a routing algorithm for computer networks based on ant colony optimization. In [10], Cassetti et al. presented a routing algorithm for MPLS-based IP networks using tabu search algorithm. Wang et al. [11] proposed a simulated annealing algorithm for QoS-based multicast routing in computer networks while considering the requirements of two QoS parameters, i.e. end-to-end delay and error rate and network resources in multimedia group communication. In [12], Bhattacharya et al. presented a new multicast routing scheme based on genetic algorithms. Apart from the above examples, several other studies have addressed the issue of routing using iterative heuristics. However, application of simulated evolution algorithm has not been reported in the literature for routing in computer networks.

In this work, a hybrid metaheuristic based on simulated evolution (SimE) algorithm for routing in computer networks is proposed. SimE is a memoryless metaheuristic, in which the search in the state space is heavily influenced by the allocation operator (discussed below). Due to the memoryless nature of the heuristic, the search usually results in partial revisiting of areas of the state space. To minimize the effect of such undesirable behavior, tabu search approach is implemented in the allocation step of SimE.

The remainder of this paper is organized as follows. In Section II, a brief background of the SimE algorithm is given. This is followed by the description of the objective function and the network model in Section III. In Section IV, the proposed SimE algorithm, and its hybrid variant using tabu search algorithm is presented. Section V presents and discusses the results. A conclusion is given in Section VI.

II. BACKGROUND OF SIMULATED EVOLUTION ALGORITHM

Simulated Evolution (SimE) is a general iterative heuristic proposed by Ralph Kling [13]. It has been applied to a variety of complex problems such as driver scheduling problem [14], set covering problem [15], operand data type problem [16], VLSI cell placement [17], and network topology design [18].
SimE falls in the category of algorithms which emphasize the behavioral link between parents and offspring, or between reproductive populations, rather than the genetic link [19]. This scheme combines iterative improvement and constructive perturbation and saves itself from getting trapped in local minima by following a stochastic perturbation approach.

Starting from a given initial solution, SimE repetitively executes the evaluation, selection, and allocation steps in sequence on a single solution, until certain stopping conditions are met. The pseudo-code of the SimE algorithm is given in Figure 2. The selection and allocation steps constitute a compound move from current solution to another feasible solution of the state space. In the evaluation step, goodness of each element in its current location is estimated. The goodness of an element is a ratio of its optimum cost to its actual cost estimate, and therefore falls in the interval [0,1]. The goodness of an element \( e_i \) is computed using the following formula:

\[
  g_i = \frac{O_i}{C_i}
\]

where \( O_i \) is the estimate of the optimal cost of the element \( e_i \) and \( C_i \) is the actual cost of \( e_i \) in its current location. The individual \( e_i \) is analogous to a gene of a chromosome in genetic algorithms.

The goodness is a measure of how near each element is to its optimum position. The higher the goodness of an element, the closer is that element to its optimum position with respect to the current position. In selection step, the algorithm probabilistically selects elements for relocation. Elements having low goodness values have higher probabilities of getting selected. A selection bias (\( B \)) is used to compensate for errors made in the estimation of goodness. The purpose of bias is to inflate or deflate the goodness of elements. A high positive value of bias decreases the probability of selection and vice versa. A very low bias value degrades the solution quality, since low bias value results in large selection sets. Consequently, large perturbations take place which create uncertainties. Similarly, for high bias values the size of the selection set is small, which degrades the quality of solution due to limitations of the algorithm to escape local minima. A carefully tuned bias value results in good solution quality and reduced execution time [5].

Elements selected during the selection step are assigned to new locations in the allocation step with the hope of improving their goodness values, and thereby reducing the overall cost of the solution. Allocation is the step that has the highest impact on the quality of the search performed by the SimE algorithm. A completely random allocation makes the SimE algorithm behave like a random walk. Therefore, this operator should be carefully engineered to the problem instance and must include domain-specific knowledge. Dif-


**III. OBJECTIVE FUNCTION AND NETWORK MODEL**

A queue associated with a link having a high delay will build up fast, and hence the probability of packet loss will increase [20], [21]. Thus, packet delay can be considered as an indirect measure of packet loss probability and consequently the average packet delay can be considered as the objective function to be optimized [20], [22], [21]. Hence the theoretical form of the objective function is [20], [22], [21]:

\[
D = \frac{1}{\gamma} \sum_{i=1}^{L} \frac{f_i}{B_i - f_i}
\]

subject to

\[f_i \leq B_i, \quad \forall \text{ link}_{i} \in M\]

where \(f_i\) represents the flow on link \(i\) in bps, \(B_i\) is the transmission capacity in bps, \(\gamma\) is the total external load on the network, \(L\) is the total number of links in the network, and \(M\) is the total number of nodes in the network. A dynamic routing problem has been considered in this paper. In dynamic routing, the network model is constant and only the traffic demands are variable [20]. However in this paper, only networks carrying the same type of traffic with the same QoS requirements are considered, similar to the model in [21]. Also, it is assumed that the network is fixed during the time scale of interest. A route (i.e. path) may be made up of one or more links. Thus, routes can be classified into two types: direct and alternate. A direct route consists of one physical link connecting a pair of nodes directly. An alternate route consists of more than one physical links in sequence. For example in Figure 1(a), a direct route between nodes 1 and 2 includes link \(L_1\), while an alternate route might be through node 6 and consisting of links \(L_2\) and \(L_6\).

**IV. PROPOSED ALGORITHM AND IMPLEMENTATION DETAILS**

This section describes the encoding mechanism for a network configuration as well as the details of different steps of the proposed SimE algorithm and its hybrid variant.

**A. Encoding mechanism**

In the proposed SimE, each route is identified by a route number, which is its row number in the routing table. For large-scale networks, the number of possible routes for each SD pair may become very large. To make the routing tables small in this situation, the number of candidate routes is limited to \(K\) shortest routes [20], [21]. The proposed encoding method is presented in Figure 3. In Figure 3(a), a routing table for each SD pair in the network in Figure 1(a) is given (a total of 30 in this case). Each routing table consists of 8 “best” paths from each source to each destination, and route number (given in column 1) is assigned to each route. The term “best” refers to the length of the route; the shorter the length, the better is the route. It is possible to add more routes, but this will increase the complexity of the problem as well as the search space. Besides, having longer routes with a large number of hops between an SD pair is undesirable as it might negatively affect the overall performance of the network. In Figure 3(b), a possible configuration for the network under consideration is given, where a selected route out of the available routes is given for each SD pair. Finally in Figure 3(c), a string is given which represents the selected network configuration (i.e. a routing solution). In this string, each value represents the chosen route for each SD pair in sequence. For example for SD pair 1-2, route number 3 is selected; for SD pair 1-3, route 5 is selected, and so on.

Below, the proposed SimE algorithm is presented. In this SimE, a solution represents a routing solution, whereas each route in this string, designated by an integer, is called an individual.

**B. Initialization**

The initial routing solution is generated randomly, while keeping into account the feasibility constraint mentioned earlier.

**C. Evaluation Scheme**

The goodness of each individual is computed as follows. In our case, an individual is a route between two nodes. In the evaluation scheme, we need to find out the optimum (minimum) possible delay between two nodes, and the current delay that exist between the two nodes due to the current route between them. The minimum bound on the delay can be taken as if there was a direct path between the two nodes (i.e. if the two nodes were directly connected.
TS, Tabu Search characteristics have been in-
removed (i.e. not used) in the routing solution. Bias
routes are removed from the network one at a
time. For E. A
control the size of the set of paths selected for removal.

B
then route
∈ RANDOM
network configuration, where
D. Se
existing configuration.
be compared to the current traffic on this direct route in the
the mutual traffic generated by these two nodes, which could
pairs where a direct “physical” link exists (such as between
SD pairs can be found using the same approach. For SD
delay for this SD pair. Similarly, optimum delays for other
virtual traffic could give us the minimum possible (optimum)
there was no traffic on this direct route except for the traffic
(best route is of size 2 (through node 6). However, an ideal
example, consider the network in Figure 1(a). In this figure,
there is no direct route between nodes 4 and 5, and the
there is no direct route between nodes 4 and 5, and the
resulting configuration string with length 30 for Fig. 1(a).

F. T
introduced, details of which follow.

G. S
cess, preventing it from moving in certain directions to drive
search for the best move in the neighborhood of the current
solution.

In this stage of the algorithm, for each route \( e_i \) in current
network configuration, where \( i = 1,2, \ldots, M \), a random number
\( \text{RANDOM} \in [0,1] \) is generated and compared with \( g_i + B \), where \( B \) is the selection bias. If \( \text{RANDOM} > g_i + B \),
then route \( e_i \) is selected for allocation and considered as
removed (i.e. not used) in the routing solution. Bias \( B \) is used
to control the size of the set of paths selected for removal.

E. Allocation Scheme

During the allocation stage of the algorithm, the selected
routes are removed from the network one at a time. For
each removed route, new routes are tried in such a way that
they result in overall better solution. Before the allocation
step starts, the selected routes are sorted according to their
goodness values in ascending order.

In detail, the proposed allocation scheme works as follows:
all selected routes (i.e. routes in the selected set) are removed
one at a time and trial routes are placed for each removed
route. We start with the head-of-line route, i.e. the route with
the worst goodness. This route is removed from the network.
This is followed by the process of placing trial routes. In this
work, the approach to place trial routes is as follows. For each
removed route, three routes from the same routing table is
tried. This action is termed as making “moves”. The valid
moves among the three moves are evaluated based on the
formula in Section II and the best move (i.e. the one which
results in the highest goodness value) among the three moves
is made permanent. If all the three moves are invalid (i.e. if
they violate the constraint), they are rejected and the original
route is restored back. This procedure is repeated for all the
routes that are present in the set of selected routes.

Two variations of allocation schemes were implemented
in this work. The first one is the same as has been described
above, which is named as SimE. In the second variation,
called SimE_TS, Tabu Search characteristics have been in-
trduced, details of which follow.

F. Tabu Search based Allocation

Tabu Search (TS) is a general iterative heuristic that is
used for solving combinatorial optimization problems. The
algorithm was first presented by F. Glover [4]. A key charac-
teristic of TS is that it imposes restrictions on the search pro-
cess, preventing it from moving in certain directions to drive
the process through regions desired for investigation [4]. It
searches for the best move in the neighborhood of the current
solution.

In this work, the SimE algorithm has been modified by
introducing Tabu Search characteristics in the allocation
phase. Recall that in the allocation phase, certain number
of moves are made for each route in the selection set and
the best move is accepted, making the move (i.e., route)
permanent. This newly accepted route is saved in a tabu list.
Thus our attribute is the route itself. The aspiration criterion
adopted is that if the route that had been made tabu produces
a higher goodness value than the current one, then we will
override the tabu status of the route and make it permanent.
This strategy prevents the selection and allocation operators
from repetitively removing the same route and replacing it
with a route of equal or worse goodness.

G. Stopping Criterion

In all experiments conducted, the stopping criterion is that
if there is no improvement seen in the quality of solution for
10 % of the total number of iterations, then the execution of
the algorithm stops.

Fig. 3. Encoding for network configuration (a) For each SD pair in Fig.
1(a), the K best routes (b) Selected routes for each SD pair (c) Resulting
configuration string with length 30 for Fig. 1(a).

with each other), and there was no traffic on this direct
link except the one that exists for these two nodes. This
optimum value should then be compared with the delay
on the current route that exist between the two nodes. For
example, consider the network in Figure 1(a). In this figure,
there is no direct route between nodes 4 and 5, and the
best route is of size 2 (through node 6). However, an ideal
(optimum) case would be if there were a direct link between
nodes 4 and 5 (illustrated with dotted line in the figure) and
there was no traffic on this direct route except for the traffic
between these nodes. Thus, this “imaginary” link with its
virtual traffic could give us the minimum possible (optimum)
delay for this SD pair. Similarly, optimum delays for other
SD pairs can be found using the same approach. For SD
pairs where a direct “physical” link exists (such as between
nodes 1 and 4), the optimum would be to have delays due to
the mutual traffic generated by these two nodes, which could
be compared to the current traffic on this direct route in the
existing configuration.

D. Selection

In this stage of the algorithm, for each route \( e_i \) in current
network configuration, where \( i = 1,2, \ldots, M \), a random number
\( \text{RANDOM} \in [0,1] \) is generated and compared with \( g_i + B \), where \( B \) is the selection bias. If \( \text{RANDOM} > g_i + B \),
then route \( e_i \) is selected for allocation and considered as
removed (i.e. not used) in the routing solution. Bias \( B \) is used
to control the size of the set of paths selected for removal.
TABLE I

CHARACTERISTICS OF TEST CASES USED IN EXPERIMENTS.

<table>
<thead>
<tr>
<th>Test Case</th>
<th># of nodes</th>
<th># of links</th>
<th># of routes per SD pair (K)</th>
</tr>
</thead>
<tbody>
<tr>
<td>n6</td>
<td>6</td>
<td>11</td>
<td>8</td>
</tr>
<tr>
<td>n11</td>
<td>11</td>
<td>11</td>
<td>8</td>
</tr>
<tr>
<td>n16</td>
<td>16</td>
<td>29</td>
<td>10</td>
</tr>
<tr>
<td>n20</td>
<td>20</td>
<td>30</td>
<td>10</td>
</tr>
</tbody>
</table>

TABLE II

AVERAGE DELAY FOR BEST SOLUTION FOR DIFFERENT TABU LIST SIZES. DELAY IS IN MILLISECONDS PER PACKET

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Tabu list size</th>
<th>Average Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>n6</td>
<td>1</td>
<td>2.976</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2.768</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>3.394</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>3.872</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>4.939</td>
</tr>
<tr>
<td>n11</td>
<td>1</td>
<td>3.765</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>4.353</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>4.939</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>5.765</td>
</tr>
<tr>
<td>n16</td>
<td>1</td>
<td>3.765</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>4.385</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>4.939</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>5.590</td>
</tr>
<tr>
<td>n20</td>
<td>1</td>
<td>3.765</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>4.385</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>4.939</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>5.387</td>
</tr>
</tbody>
</table>

V. RESULTS AND DISCUSSION

The SimE and SimE,TS algorithms described in this paper were tested on four networks shown in Figure 1. Characteristics of these networks are given in Table I. Moreover, for each test case, capacity of a link is assumed to be 100 Mbps.

A. Effect of Tabu Search Based Allocation and Tabu List Size

Table II shows the results obtained for the test cases using different tabu list sizes. In this table, average delay is reported along with the respective tabu list size. It is noticed in this table that as the test case size increases, the tabu list that gives the best solution also increases. For example, in n6, tabu list size of 2 gives the best solution. Similarly, best solutions are achieved by tabu list sizes of 4, 5, and 5 in n11, n16, and n20 respectively.

Table III gives the results for the four test cases considering the frequency of tabu moves, and the respective tabu list size that gave the best solutions with their execution times. Frequency of tabu moves represents the number of times a route was found tabu. This is recorded through a counter called tabu counter. The tabu counter only includes the number of tabu routes which could not pass the aspiration criteria. It does not count the frequency of routes which were actually tabu but managed to pass the aspiration criteria. From this table, it is observed that the percentage of tabu moves varies between 4% and 8%.

TABLE III

RESULTS FOR BEST TABU LIST SIZE. EXECUTION TIME IS IN SECONDS.

<table>
<thead>
<tr>
<th>Test case</th>
<th>Tabu list size for best solution</th>
<th>Total moves</th>
<th>Tabu moves</th>
<th>% Tabu</th>
<th>Exec time</th>
</tr>
</thead>
<tbody>
<tr>
<td>n6</td>
<td>2</td>
<td>1434</td>
<td>67</td>
<td>4.67</td>
<td>65</td>
</tr>
<tr>
<td>n11</td>
<td>4</td>
<td>2255</td>
<td>94</td>
<td>4.12</td>
<td>170</td>
</tr>
<tr>
<td>n16</td>
<td>5</td>
<td>2575</td>
<td>147</td>
<td>5.71</td>
<td>313</td>
</tr>
<tr>
<td>n20</td>
<td>5</td>
<td>2717</td>
<td>212</td>
<td>7.80</td>
<td>554</td>
</tr>
</tbody>
</table>

TABLE IV

COMPARISON OF SIM E AND SIM E,TS. D = DELAY IN MILLISECONDS PER PACKET, T = EXECUTION TIME IN SECONDS, TL = TABU LIST SIZE. PERCENTAGE GAIN SHOWS IMPROVEMENT ACHIEVED BY SIM E,TS COMPARED TO SIM E.

<table>
<thead>
<tr>
<th>Case</th>
<th>SimE</th>
<th>SimE,TS</th>
<th>% Gain</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>T</td>
<td>TL</td>
<td>D</td>
</tr>
<tr>
<td>n6</td>
<td>3.531</td>
<td>84</td>
<td>2</td>
</tr>
<tr>
<td>n11</td>
<td>4.072</td>
<td>115</td>
<td>4</td>
</tr>
<tr>
<td>n16</td>
<td>3.013</td>
<td>93</td>
<td>5</td>
</tr>
<tr>
<td>n20</td>
<td>3.529</td>
<td>415</td>
<td>5</td>
</tr>
</tbody>
</table>

B. Comparison of SimE and SimE,TS

In this section, a comparison of SimE,TS with SimE is presented. Table IV shows the results for SimE and best tabu list size SimE,TS. The percentage gain shows the improvement achieved by SimE,TS when compared to SimE. From this table, it is seen that SimE,TS performs better than SimE. In all the test cases, a gain is achieved by SimE,TS. For example, a gain of 20.17 % is achieved in case of n20. As far as the execution time is concerned, it is also comparable.

A comparatively better performance of SimE,TS with respect to SimE can be attributed to the fact that the search space covered by SimE,TS is more than that of SimE. In SimE, it may happen that after some iterations, same moves are repeated and thus the algorithm keeps searching in the same search space most of the time, while in SimE,TS, more search space is covered because previous moves remain tabu for some time, causing the algorithm to diversify the search into another subarea. Recall that in the allocation stage, three moves are evaluated for each route in selection set and the best move (route) is made permanent. This new route is also saved simultaneously in the tabu list. However, it might happen that this new route may become “bad” (in terms of evaluation function) in the following iterations, upon which it is replaced with a new route. However, it is also possible that this route may become good again after one or more iterations, but since it is in the tabu list, it will not be chosen to be placed again, thus giving room to other routes to be chosen.

VI. CONCLUSION

In this paper a novel strategy for routing in computer networks has been presented. This approach is based on the simulated evolution algorithm with two variations in its allocation phase. Results obtained for the test cases suggest that the simulated evolution algorithm with tabu search allocation is a robust approach for this problem, and was always able...
to find good quality feasible solutions. Current and future work will focus on proposing routing algorithms based on ant colony optimization as well as particle swarm optimization algorithm hybridized with tabu search characteristics.

REFERENCES
