

## Chapter 7

# Fuzzy Simulated Annealing for DLAN Topology Design

A fuzzy simulated annealing algorithm (FSA) is proposed in this chapter for DLAN network topology design. Two variants of FSA are also proposed, as described later in the chapter. These variants incorporate characteristics of the tabu search and simulated evolution algorithms into the FSA algorithm. Another modification of FSA is proposed later in this chapter, with the purpose of reducing the user's dependency in setting the value of the Markov chain factor of FSA. The performance of FSA and its variants are empirically evaluated and mutually compared.

### 7.1 Fuzzy Simulated Annealing Algorithm

The studies in Section 2.4.4 focussed on the use of SA for network design problems with a single objective. Therefore, more research was considered necessary to evaluate the performance of SA in a multi-objective environment. Another aspect of

SA which deserves more attention is its hybridization with other iterative heuristics. Considering the above two issues, this section proposes a fuzzy simulated annealing algorithm for the DLAN topology design problem.

In the fuzzy SA algorithm for the DLAN network topology design, the neighborhood state of a current solution is generated by randomly selecting a link  $l_i$  from the topology and replacing  $l_i$  with another link  $l_j$  not in the topology. The FSA algorithm executes different steps similar to the basic SA algorithm discussed in Section 2.4.4. These steps as adapted for FSA are discussed below in detail.

### 7.1.1 Initialization

The initial topology (a spanning tree) is generated randomly, while keeping in mind the constraints mentioned earlier. The FSA control parameters are also initialized, i.e. the initial temperature  $T_0$ , the cooling rate  $\alpha_{SA}$ , the constant  $\beta_{SA}$ , maximum time for the annealing process  $MaxTime$  (in terms of number of iterations), and  $M$  which represents the time until the next parameter update.

The control parameters have an impact on the convergence of the FSA algorithm. As mentioned in Chapter 2, a suitable value of  $T_0$  should be chosen. A number of approaches have been reported in the literature to find a suitable value of  $T_0$  [33, 126, 145, 167, 202]. In this thesis, the approach presented by Kirkpatrick *et al.* [145] has been adopted. According to this method, the value of  $T_0$  is chosen such that the initial acceptance ratio,  $\chi(T_0)$ , is close to unity, where

$$\chi(T_0) = \frac{\text{Number of moves accepted at } T_0}{\text{Total number of moves attempted at } T_0} \quad (7.1)$$

In Equation (7.1), a ratio close to unity will allow a high majority of initial moves to be accepted, thus allowing the algorithm to escape local minima. If the ratio is low, then the FSA algorithm will mainly accept good moves and will behave as a greedy algorithm, thus reducing the chances of escaping local minima, and resulting in premature convergence.

As for the other parameters of FSA, a discussion has already been given in Section 2.4.4 with regard to the importance of these parameters as well as the selection of appropriate values for them.

## 7.1.2 Metropolis Algorithm

The metropolis procedure uses a sub-procedure *neighbor* to perform a move. A move involves removing a randomly selected link between two nodes from the current solution (i.e. topology) and introducing another link between two nodes. A move is represented by a local neighbor *NewZ* of any given solution *Z*. Removal of a link and introduction of a new link is exactly the same as described in Section 5.1. This new link is accepted if all constraints are satisfied and the new *overall goodness* fulfils the metropolis criterion. The overall goodness represents the fuzzy value of a solution given by Equation (3.11) or (4.1) and is calculated by function *OG* in the pseudo-code of the simulated annealing algorithm given in Figure 2.10. If the overall goodness of a new solution, *NewZ* (where *NewZ* is the restructured tree), is higher than the overall cost of the current solution *Z*, then *NewZ* is definitely accepted. However, if the overall goodness of *NewZ* is less than the overall goodness of the current solution *Z*, then *NewZ* is accepted probabilistically based on the *metropolis criterion* given by  $P(\text{random} < e^{-\Delta h/T})$ . If any of the constraints are violated, or if

the new solution does not pass the metropolis criterion, then the new link is rejected and the original link is restored.

### 7.1.3 Evaluation of a solution

During the *Metropolis* stage of the algorithm, a move is made which might result in a new solution (if the constraints are satisfied and the new solution is definitely or probabilistically accepted). The overall goodness of the resulting solution is computed by combining the four criteria to be optimized using fuzzy logic, as described in Section 3.4. To form the membership functions of individual objectives, the minimum and maximum values are required for each objective, which have been discussed earlier in Section 3.1.

### 7.1.4 Stopping Criterion

The algorithm is stopped when a maximum number of iterations is reached. For the purposes of this thesis, the maximum number of iterations was set to 20,000.

## 7.2 Hybrid Simulated Annealing Algorithms

Apart from the FSA algorithm for DLAN topology design that was presented in Section 7.1, two hybrid variants of the FSA algorithm are proposed in this section. Section 7.2.1 presents a hybrid algorithm incorporating tabu search into the FSA algorithm, while a hybrid algorithm incorporating tabu search and simulated evolution characteristics into the FSA algorithm is presented in Section 7.2.2.

### 7.2.1 Tabu Fuzzy Simulated Annealing

This thesis proposes a new hybrid iterative search technique, tabu simulated annealing (TFSA), which combines features of TS within the metropolis procedure of FSA. Recall that, in the metropolis procedure, a move is made in which a link is randomly removed from the current solution (i.e. the current topology), and a new feasible link is introduced in the solution. This newly accepted link is saved in a *tabu list*. Thus, the *attribute* is the link itself. The *aspiration criterion* is that, if the link that had been made tabu produces a higher membership value than the current link in the membership function “good topology”, then the tabu status of the link is overridden, making the link permanent. This strategy prevents the algorithm from repetitively removing the same link and replacing it with a link of equal or worse goodness.

### 7.2.2 Evolutionary Tabu Fuzzy Simulated Annealing

The hybrid tabu search FSA algorithm, TFSA, as described in Section 7.2.1 is modified to incorporate the evaluation phase of the SimE algorithm. The resulting algorithm is referred to as TEFSA. In TEFSA, the evaluation phase considers a solution as a set of movable elements, where a movable element is a link. Each link,  $l_i$ , has an associated *goodness* (fitness) measure,  $g_i$ , in the interval  $[0,1]$ , defined as

$$g_i = \frac{O_i}{C_i} \quad (7.2)$$

where  $O_i$  is the estimate of the optimal cost of the link  $l_i$  and  $C_i$  is the actual cost of  $l_i$  in its current location. The fuzzy evaluation measure proposed in Rule 2 in

Section 6.1.2 is used to evaluate the goodness of a link according to Equation (6.1).

The main objective of developing TEFSA is to obtain a better solution than FSA and TFSA, with the same runtime for all algorithms. Recall that the metropolis procedure of TFSA randomly removes a link, and a new link is introduced to maintain the topology. There is a non-zero probability that this removed link was already placed in its optimum (or near-optimum) position. If this process continues blindly for other links with near optimum position, then it will take a significant amount of time for the algorithm to converge. Rather than having a ‘blind move’, an ‘intelligent move’ would be more appropriate, where a link is removed based on its quality. Thus, a link having low quality will have a higher chance of getting removed from its current position, while a link with a high quality (i.e. in a near-optimum position) will have a lower probability of being removed. On the other hand, with the ‘blind move’ approach, each link in the topology has an equal probability of being removed from its current position, irrespective of its quality. The ‘intelligent move’ approach has been incorporated in TEFSA, with all links being evaluated using Equation (6.1). After evaluation of all links, a set of links is randomly chosen. The size of this set can vary from one link to all links evaluated. A large-sized set will allow a higher number of low-quality links, but at the expense of more runtime. On the other hand, a small-sized set will have less runtime, but many of the low-quality links may not be a part of the set. In any case, the worst link from this set is chosen and removed from the topology. The new link is then inserted as described earlier, subject to constraints and the tabu criteria.

## 7.3 Results and Discussion

The FSA, TFSA, and TEFSA algorithms proposed in this thesis were experimentally tested using the five test cases. A variety of experiments were conducted to evaluate the performance of the algorithms. The first set focussed on comparisons between the three variants of SA. In the second set, a comparative analysis of OWA versus UAO was done. Details of these experiments are discussed below.

The control parameters of FSA have an impact on the performance of the algorithm, as discussed in Section 2.4.4. Inappropriate values for these parameters may result in non-optimal solutions. After trials with several values for the initial temperature, a value of  $T_0=1000$  was found to be the most appropriate since the value satisfied the ratio in Equation (7.1) to be near unity. To find a good value for  $\alpha_{SA}$ , the following values were considered: 0.6, 0.75, 0.85, 0.95, and 0.99. Two values of the Markov chain with  $M = 10$  and  $M = 30$  were tried, with the annealing constant set at  $\beta = 1.1$ . This value of  $\beta$  was taken since a gradual change in the value of  $M$  was desired. As shown in Tables 7.1 and 7.2,  $M = 30$  and  $M = 10$  produced results of equal quality, as validated by the t-test. In some cases,  $M = 10$  produced slightly better results (marked in italics). Therefore, a value of  $M = 10$  was used for all experimentation. For each variant of SA, 30 runs were executed for each test case, and the average and standard deviation of overall goodness of best solutions were calculated. For TFSA, different tabu list sizes as described in Section 7.3.1 were tried. The best tabu list size obtained for TFSA was also used for TEFSA. For fair comparison of the performance, the same initial solution was used for all runs of the three variants for each test case. The three algorithms were

run for the same amount of absolute time.

Table 7.1: Summary of best overall goodness with Markov chain size  $M = 10$  and  $M = 30$  using the OWA operator for FSA. % improvement shows improvement achieved by  $M = 10$  with reference to  $M = 30$ . Statistically significant improvement is in italics.

Case	$\alpha_{SA}$	$M = 10$ Avg. Goodness	$M = 30$ Avg. Goodness	% improvement
n15	0.95	0.206 $\pm$ 0.035	0.203 $\pm$ 0.034	1.33
n25	0.75	0.414 $\pm$ 0.115	0.346 $\pm$ 0.101	<i>19.92</i>
n33	0.85	0.623 $\pm$ 0.369	0.543 $\pm$ 0.380	<i>14.70</i>
n40	0.75	0.244 $\pm$ 0.051	0.253 $\pm$ 0.040	-3.65
n50	0.75	0.238 $\pm$ 0.066	0.237 $\pm$ 0.044	0.42

Table 7.2: Summary of best overall goodness with Markov chain size  $M = 10$  and  $M = 30$  using the UAO operator for FSA. % improvement shows improvement achieved by  $M = 10$  with reference to  $M = 30$ . Statistically significant improvement is in italics.

Case	$\alpha_{SA}$	$M = 10$ Avg. Goodness	$M = 30$ Avg. Goodness	% improvement
n15	0.99	0.335 $\pm$ 0.003	0.335 $\pm$ 0.003	-0.03
n25	0.75	0.345 $\pm$ 0.034	0.335 $\pm$ 0.034	2.86
n33	0.75	0.351 $\pm$ 0.092	0.394 $\pm$ 0.087	<i>-12.43</i>
n40	0.85	0.382 $\pm$ 0.068	0.368 $\pm$ 0.061	3.69
n50	0.85	0.350 $\pm$ 0.053	0.343 $\pm$ 0.036	1.97

### 7.3.1 Effect of Tabu List size

The effect of the tabu list size was investigated for the TFSA algorithm with tabu list sizes of 3, 5, 7, 9, 11, and 13. The effect was evaluated for both the OWA and UAO operators. As for the previous chapters, the purpose of this investigation is to see if the statement made by Glover [100] (mentioned in Section 5.3.1) can be confirmed. Tables 7.3 and 7.4 summarize the average overall goodness for the



best solutions for each test case with different tabu list sizes for OWA and UAO respectively.

It is observed in Table 7.3 that the size of tabu list that is related to best overall goodness varies for each case when OWA is used. A graphic illustration of the variation in average overall goodness with respect to different sizes for OWA is given in Figure 7.1. Table 7.3 also provides the percentage difference in the average overall goodness of the given size when compared with size 7. For example, for case *n33*, the average overall goodness obtained with size 9 was 19.35% better than that obtained with size 7. It appears from the results in Table 7.3 that, in general, Glover's first observation that tabu list size of 7 does not always produce the best results seem to be proven true. This observation is confirmed through validation of results using the t-test. The t-test was performed (at 5% significance level) to test the hypothesis whether the two averages (i.e. the average overall goodness obtained with a tabu list size of XYZ and that of 7) were significantly different from each other. Percentage improvements which are statistically significant are shown in italics. The t-test results signify two important issues. First, in general there is no hard evidence that the results produced by a tabu list size of 7 were statistically more significant than the results produced by other sizes. This observation is obvious from the results of *n15*, *n25*, *n33*, and *n40*, where size 7 failed to produce statistically better results than any other size. Even for test case *n50*, size 7 did not result in significant improvement as a whole; there is only one instance where size 7 produced better results than size 3. Second, there are instances where sizes other than 7 achieved statistically more significant results than results obtained with size 7. These instances include a tabu list size of 9 for *n25*, and the sizes 3, 9, and 13 for *n33*. Furthermore, Glover's

observation that size of the best tabu list increases as the test case size increases was not found to be true.

As for UAO, a trend similar to that of OWA was observed, where the size for best results varied with each test case. As observed from Table 7.4, there was only one instance of  $n15$  where size 7 produced statistically better results than size 3, with an improvement of 0.93%. For other test cases, size 7 failed to produce statistically significant results compared to other sizes. In some instances, tabu list sizes other than 7 produced statistically better results. Such instances include  $n25$ , where tabu list sizes of 3 and 9 respectively produced 4.71% and 6.7% better overall goodness than size 7. Another instance is  $n40$  where size 9 produced 11.40% better results than size 7. A graphic illustration of the variation in average overall goodness with respect to different tabu list sizes for UAO is given in Figure 7.2.

The conclusion from the above discussion and results is that it is not true that a tabu list size of 7 will always produce the best results for any problem; the size of the tabu list is problem-dependent. Moreover, the best tabu list size also does not necessarily increase with increasing the size of test case.

### 7.3.2 Comparison of FSA, TFSA, and TEFSA

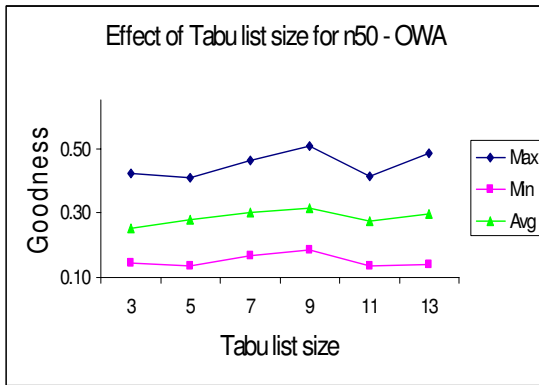
Table 7.5 summarizes the results of 30 runs obtained by TEFSA, TFSA, and FSA for the OWA operator. The corresponding run time is given in Table 7.6. For each test case, different values of  $\alpha_{SA}$  were tried, and the values which gave the best results are reported in Table 7.5. The results show that the value of  $\alpha_{SA}$  is not proportional to the size of the test case. For example, the best value of  $\alpha_{SA}$  for  $n15$  is 0.95, which is a small test case. A value of  $\alpha_{SA} = 0.85$  gave best results for

Table 7.3: Effect of tabu list size on the quality of overall goodness for TFSA using OWA. Run time is in seconds. Statistically significant improvement is in italics. NA = Not Applicable (since size 7 was used as the reference for comparison).

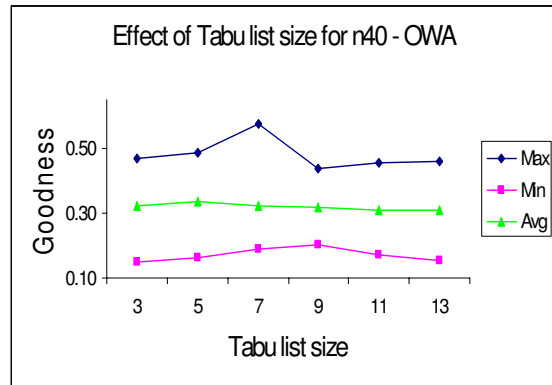
Test Case	Tabu list size	Avg. Overall goodness	Run time	% Improvement
n15	3	0.257 ± 0.041	53.9	-2.67
	5	0.257 ± 0.041	74.1	-2.37
	<b>7</b>	0.264 ± 0.042	75.0	NA
	<b>9</b>	0.265 ± 0.048	77.2	0.43
	11	0.265 ± 0.047	61.1	0.69
	13	0.250 ± 0.045	62.7	-5.31
n25	3	0.502 ± 0.114	240.8	5.09
	5	0.464 ± 0.134	242.1	-2.88
	7	0.478 ± 0.129	241.8	NA
	<b>9</b>	0.532 ± 0.106	246.1	<i>11.36</i>
	11	0.444 ± 0.141	247.1	-7.00
	13	0.485 ± 0.114	265.0	1.44
n33	3	0.526 ± 0.352	478.3	<i>11.38</i>
	5	0.490 ± 0.358	475.7	3.87
	7	0.472 ± 0.359	492.2	NA
	<b>9</b>	0.563 ± 0.362	509.2	<i>19.35</i>
	11	0.501 ± 0.354	525.3	6.11
	13	0.523 ± 0.350	567.1	<i>10.78</i>
n40	3	0.321 ± 0.078	1457.4	-0.42
	5	0.334 ± 0.099	1462.6	3.72
	<b>7</b>	0.322 ± 0.102	1491.4	NA
	9	0.316 ± 0.076	1509.0	-1.81
	11	0.309 ± 0.085	1542.4	-3.98
	13	0.310 ± 0.089	1544.6	-3.90
n50	3	0.254 ± 0.088	3208.1	<i>-15.38</i>
	5	0.279 ± 0.076	3310.8	-6.85
	7	0.300 ± 0.068	3308.9	NA
	<b>9</b>	0.316 ± 0.080	3289.6	5.52
	11	0.275 ± 0.082	3342.0	-8.34
	13	0.296 ± 0.092	3361.9	-1.41

Table 7.4: Effect of tabu list size on the quality of overall goodness for TFSA using UAO. Statistically significant improvement is in italics. NA = Not Applicable (since size 7 was used as the reference for comparison).

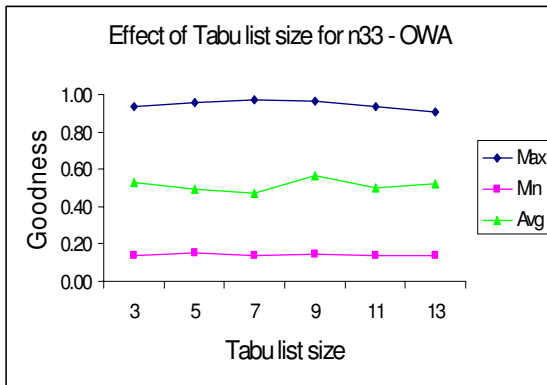
Test Case	Tabu list size	Avg. Overall goodness	Run time	% Improvement
n15	3	0.340 ± 0.008	53.3	0.15
	5	0.336 ± 0.004	73.7	<i>-0.93</i>
	<b>7</b>	0.339 ± 0.008	76.3	NA
	9	0.338 ± 0.006	77.2	-0.46
	11	0.339 ± 0.006	61.3	-0.14
	13	0.338 ± 0.005	65.4	-0.49
n25	3	0.368 ± 0.032	594.0	<i>4.71</i>
	5	0.360 ± 0.040	227.2	2.58
	7	0.351 ± 0.043	229.6	NA
	<b>9</b>	0.375 ± 0.048	239.1	<i>6.70</i>
	11	0.352 ± 0.039	234.1	0.23
	13	0.352 ± 0.032	234.8	0.25
n33	3	0.321 ± 0.068	772.3	-5.78
	5	0.328 ± 0.080	754.2	-3.86
	<b>7</b>	0.341 ± 0.094	759.5	NA
	9	0.339 ± 0.084	760.0	-0.59
	11	0.316 ± 0.074	848.2	-7.31
	13	0.337 ± 0.099	850.1	-1.05
n40	3	0.450 ± 0.057	944.8	<i>6.88</i>
	5	0.421 ± 0.070	944.9	0.04
	7	0.421 ± 0.070	1004.0	NA
	<b>9</b>	0.469 ± 0.036	1032.6	<i>11.40</i>
	11	0.431 ± 0.065	1061.9	2.44
	13	0.440 ± 0.063	1088.9	4.54
n50	3	0.356 ± 0.048	2050.9	-3.83
	5	0.369 ± 0.062	2104.9	-0.30
	7	0.370 ± 0.061	2162.1	NA
	<b>9</b>	0.393 ± 0.070	2214.6	6.13
	11	0.379 ± 0.065	2270.0	2.30
	13	0.370 ± 0.062	2322.5	-0.11



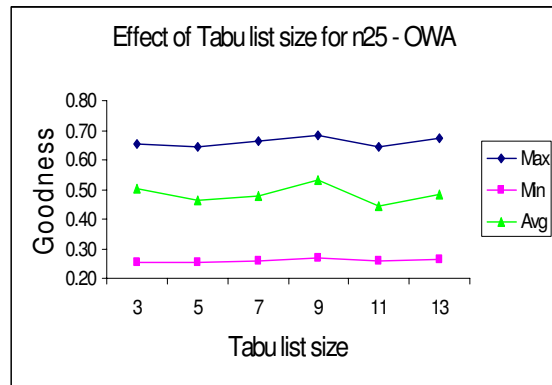
(a)



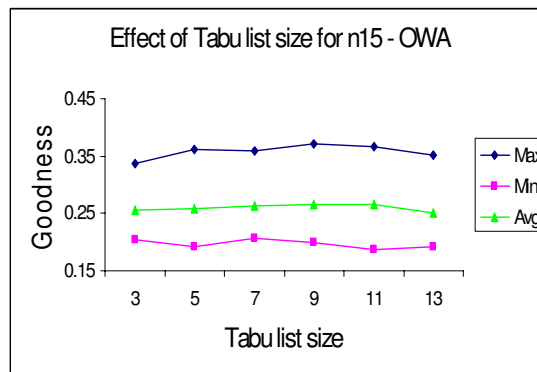
(b)



(c)

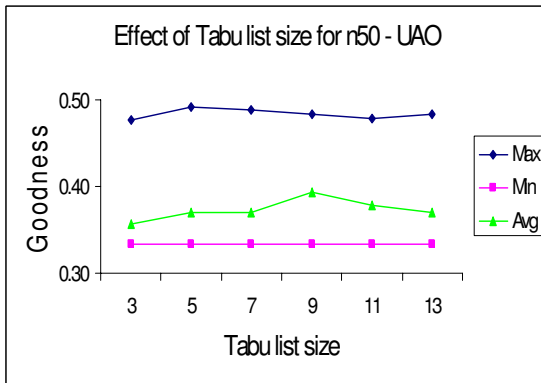


(d)

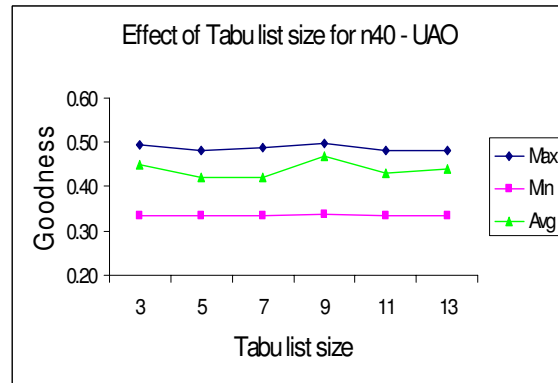


(e)

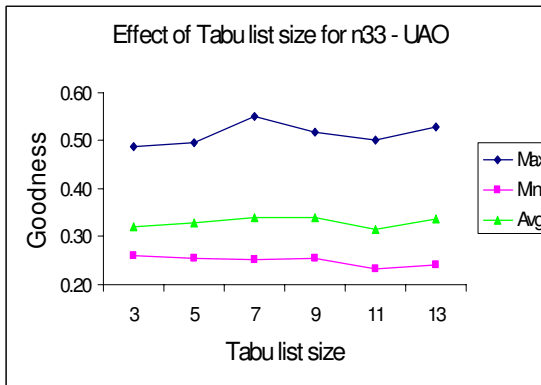
Figure 7.1: Plots of maximum, minimum, and average values of membership function “Good topology” versus tabu list size using the OWA operator for (a) n50 (b) n40 (c) n33 (d) n25 (e) n15



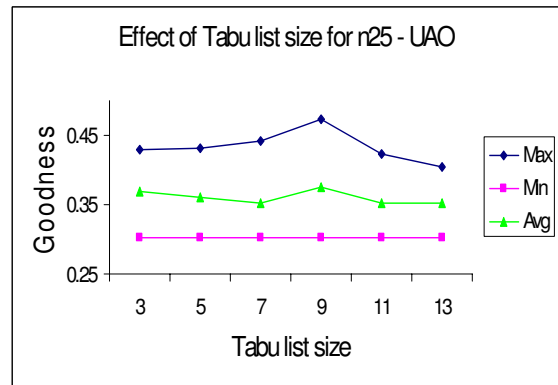
(a)



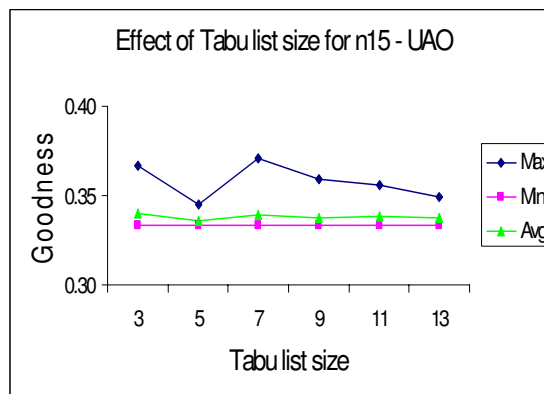
(b)



(c)



(d)



(e)

Figure 7.2: Plots of maximum, minimum, and average values of membership function “Good topology” versus tabu list size using the UAO operator for (a) n50 (b) n40 (c) n33 (d) n25 (e) n15

Table 7.5: Summary of overall goodness and percentage improvement with OWA for FSA, TFSA, and TEFSA. TL = Tabu list size, imp = percentage improvement. Statistically significant improvement is in italics.

Case	$\alpha_{SA}$	FSA Goodness	TFSA		TEFSA Goodness	% imp TEFSA vs FSA	% imp TEFSA vs TFSA	%imp TFSA vs FSA
			TL	Goodness				
n15	0.95	0.206±0.04	9	0.265±0.05	0.389±0.06	<i>89.14</i>	<i>47.13</i>	<i>28.56</i>
n25	0.75	0.391±0.12	9	0.53±0.106	0.500±0.13	<i>27.81</i>	-6.11	<i>36.12</i>
n33	0.85	0.646±0.37	9	0.563±0.36	0.429±0.19	<i>-33.56</i>	-23.79	-12.82
n40	0.75	0.231±0.05	7	0.322±0.10	0.489±0.08	<i>111.56</i>	<i>51.68</i>	<i>39.48</i>
n50	0.75	0.198±0.07	9	0.316±0.08	0.329±0.12	<i>66.39</i>	4.10	<i>59.83</i>

Table 7.6: Average run time (in seconds) of algorithms in Table 7.5.

Test Case	Run time		
	FSA	TFSA	TEFSA
n15	89.0	89.2	89.5
n25	314.4	315.1	314.8
n33	765.1	765.2	764.7
n40	1498.8	1498.4	1499.5
n50	4295.8	4296.6	4295.4

*n33* which is a medium size case. However, the best value of  $\alpha_{SA}$  was 0.75 for *n25*, *n40*, and *n50*. Notice that *n25* is a small size test case, while *n40* and *n50* are large size test cases. As for the UAO operator, Table 7.7 reports similar trends as that of OWA with respect to  $\alpha_{SA}$ . There is no clear trend which would suggest that the test case size is directly or inversely proportional to  $\alpha_{SA}$ . Therefore, it can be concluded that the best value of  $\alpha_{SA}$  is problem-dependent.

As far as the comparison of TEFSA, TFSA, and FSA with respect to OWA is concerned, it is observed from Table 7.5 that TEFSA obtained the best results among the three variants for most test cases. As per the table, TEFSA had statistically better overall goodness than FSA for test cases *n50*, *n40*, *n25*, and *n15*, as validated

Table 7.7: Summary of overall goodness and percentage improvement with UAO for FSA, TFSA, and TEFSA. TL = Tabu list size, imp = percentage improvement. Statistically significant improvement is in italics.

Case	$\alpha_{SA}$	FSA Goodness	TFSA		TEFSA Goodness	% imp TEFSA vs FSA	% imp TEFSA vs TFSA	%imp TFSA vs FSA
			TL	Goodness				
n15	0.99	0.335±0.0	7	0.339±0.01	0.365±0.02	<i>9.16</i>	<i>7.68</i>	<i>1.37</i>
n25	0.75	0.345±0.03	9	0.375±0.05	0.412±0.06	<i>19.30</i>	<i>9.82</i>	<i>8.63</i>
n33	0.75	0.339±0.09	7	0.341±0.09	0.411±0.07	<i>21.14</i>	<i>20.55</i>	0.49
n40	0.85	0.374±0.07	9	0.469±0.04	0.470±0.08	<i>25.72</i>	0.30	<i>25.34</i>
n50	0.85	0.350±0.05	9	0.388±0.07	0.374±0.05	<i>6.99</i>	-3.48	<i>10.85</i>

Table 7.8: Average run time (in seconds) of algorithms in Table 7.7.

Test Case	Run time		
	FSA	TFSA	TEFSA
n15	88.0	88.3	88.5
n25	321.3	322.1	322.4
n33	757.6	758.5	757.0
n40	1564.5	1565.6	1564.4
n50	3485.3	3484.6	3485.6

by the t-test (performed at 5% significance level). However, for *n33*, FSA obtained statistically better results than TEFSA. As for comparison of TEFSA and TFSA, results in Table 7.5 suggest that TEFSA produced results that were statistically significantly better for *n15* and *n40*. For *n25*, *n33*, and *n50*, TEFSA and TFSA had results of equal quality, as validated by the t-test. All in all, it can be fairly claimed that TEFSA performed better than the other two variants when OWA was used.

The above observations are further supported by Figure 7.3. In this figure, the average performance of the three algorithms is compared by plotting the frequency of solutions in different membership (i.e. overall goodness) range. The plots for *n50*, *n40* and *n15* show that TEFSA has more solutions in the higher membership



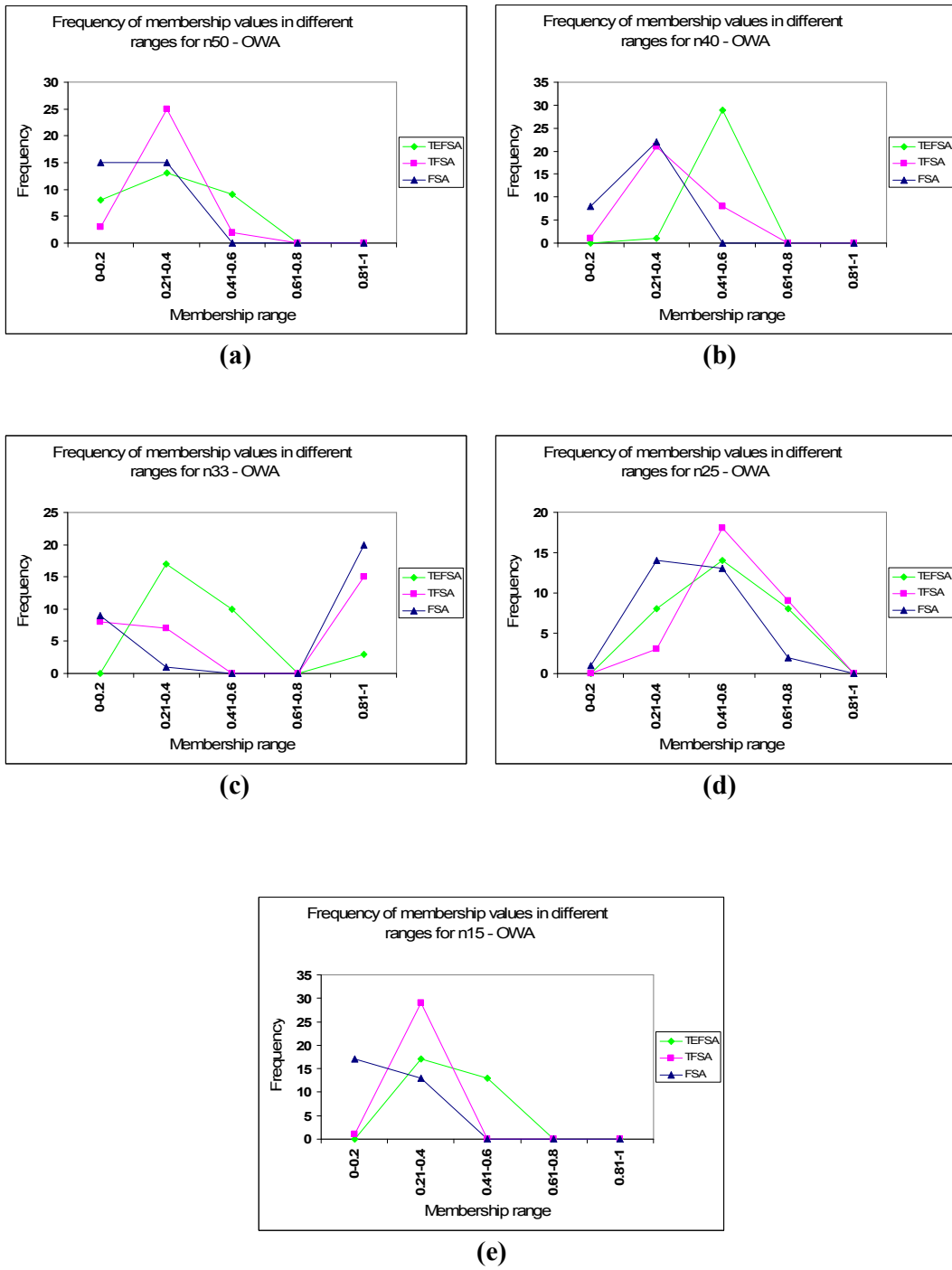


Figure 7.3: Frequency of solution in different membership ranges for function “Good topology” using the OWA operator for FSA, TFSA, and TEFSa for (a) n50 (b) n40 (c) n33 (d) n25 (e) n15

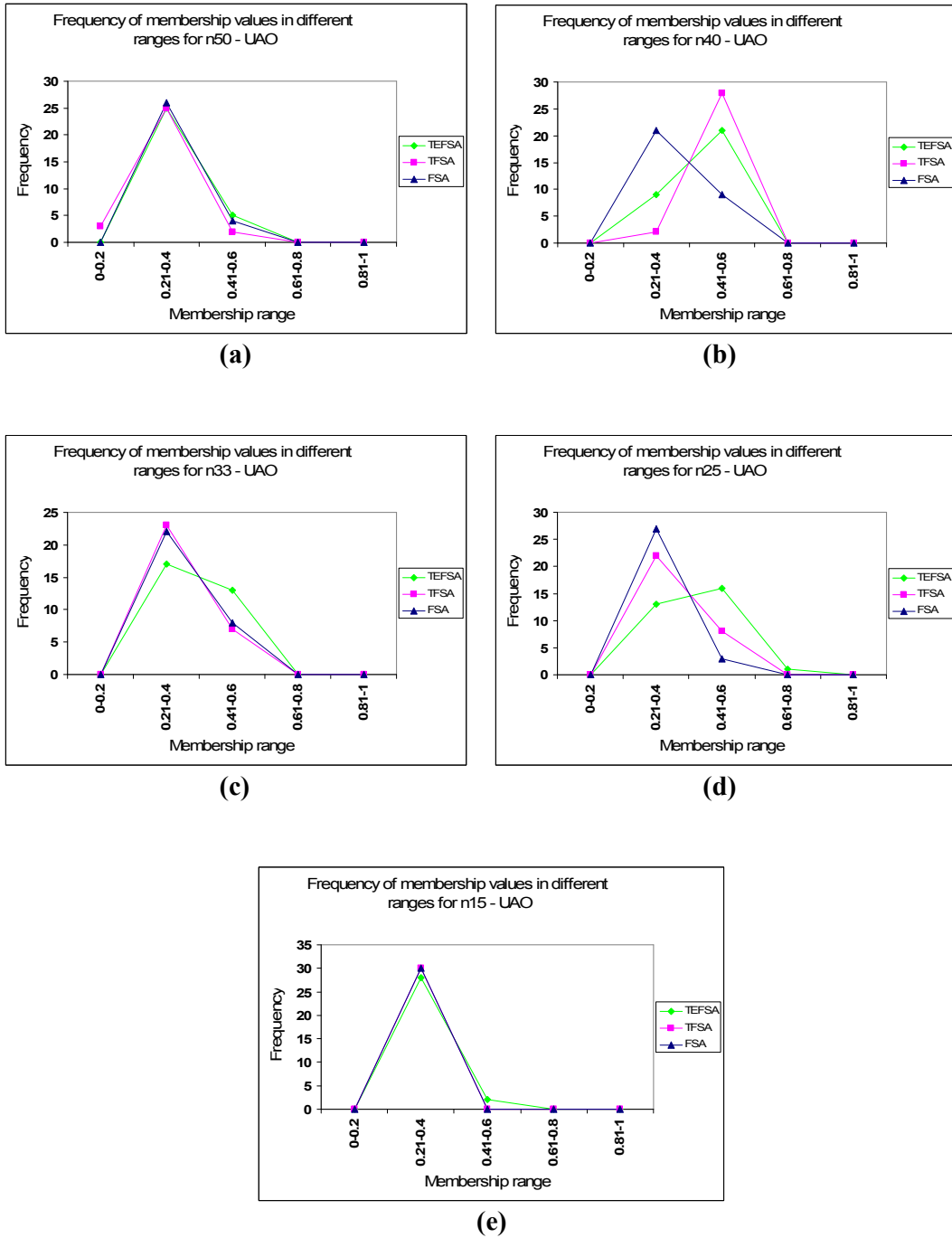


Figure 7.4: Frequency of solution in different membership ranges for function “Good topology” using the UAO operator for FSA, TFSA, and TEFSA for (a) n50 (b) n40 (c) n33 (d) n25 (e) n15

range than TFSA and FSA. For example, in Figure 7.3(a), TEFSA has around 10 solutions in the goodness range  $0.41 - 0.6$ , whereas TFSA and FSA have less than 2 solutions in the same range. A similar pattern is apparent in Figure 7.3(b) where TEFSA has almost all solutions in the range  $0.41 - 0.6$ . In contrast, TFSA has less than 10 solutions in the same range, while FSA has none. For  $n15$  (Figure 7.3(e)), the pattern is somewhat similar to Figure 7.3(a). For  $n25$ , TFSA shows the best performance, which is apparent from plots in Figure 7.3(d), where it is observed that TFSA had more solutions than TEFSA in the ranges  $0.41 - 0.6$  and  $0.61 - 0.8$ . In this plot, FSA has the worst performance since most of its solutions fall in ranges  $0.21 - 0.4$  and  $0.41 - 0.6$ . Finally, for  $n33$ , the plots show that FSA has more solutions than TEFSA and TFSA in the range  $0.81 - 1$ , with TEFSA showing the least number of solutions than FSA and TFSA.

With respect to the UAO operator, a mutual comparison of performance of TEFSA, TFSA, and FSA is given in Table 7.7. The table clearly shows that TEFSA was the best for all test cases except  $n50$ . For this test case, TEFSA and TFSA had almost the same level of performance. This particular observation is also supported by the graphs in Figure 7.4, where it is observed that, for  $n50$ , TFSA had more solutions in the higher overall range ( $0.41 - 0.6$ ) than TEFSA and FSA. For other test cases, TEFSA has more solutions than FSA and TFSA in higher goodness ranges, which is clearly visible in test cases  $n33$ ,  $n25$  and  $n40$ . The results and observations presented above suggest that TEFSA performed better than the other two variants for UAO.

It is evident from the above discussion that, in general, TEFSA was able to produce results of higher quality than TFSA and FSA, for both OWA and UAO.

This superior performance of TEFSA can be attributed to the fact that TEFSA performs a more efficient exploration of the search space than TFSA and FSA (refer to Section 7.2.2). Recall that, in TEFSA, links with bad goodness have a higher probability of being removed from the topology, while links which are already in “good” positions have a lower probability of being removed. This characteristic of TEFSA makes it different from TFSA and FSA, where all links, whether good or bad, have equal probability of being removed. This explanation is also supported by the results in Table 7.9, where the average goodness of links (AGL) for FSA, TFSA, and TEFSA is given for test cases  $n33$  and  $n40$ . The AGL is calculated as the average of 30 runs, and, for each run, the value of AGL associated with the best solution in that run is taken. As observed from Table 7.9, there is clear evidence that the AGL for TEFSA was higher than that for TFSA and FSA. In other words, TEFSA was able to retain links of higher goodness, while TFSA and FSA were not so efficient in retaining links of higher goodness.

As far as a comparison of TFSA and TSA is concerned, it is observed from Tables 7.5 and 7.7 that TFSA produced statistically better results than TSA for all cases, with the exception of  $n33$ , as depicted in the last columns of Tables 7.5 and 7.7. A possible explanation for the better performance of TFSA in comparison to FSA is that there might be certain moves which are recursively repeated. For example, a link is removed from the topology and placed back in its position. It is quite possible that, in the next iteration, the same link is chosen, removed, and placed back in its location. If this process continues for a certain amount of time, then the algorithm will not improve the solution and execution time will be wasted. However, TFSA prevents this situation from occurring, thus allowing the algorithm to explore more

of the search space and obtain a better quality solution than FSA. This explanation is also supported by the subsequent example where two typical runs for FSA and TFSA for  $n40$  (while using OWA) were compared in terms of number of repeated moves. The results suggest that out of the 20000 moves performed for each run, FSA had 6842 moves repeated, while TFSA had 4103 moves repeated. That is, FSA repeated 2739 additional moves compared to TFSA. In other words, TFSA had 2739 new moves compared to FSA. These additional new moves allowed TFSA to explore a bigger part of the search space than did FSA, thus resulting in higher quality solutions.

Table 7.9: Average goodness of links for FSA, TFSA, and TEFSA using the OWA operator. AGL represents the average goodness of links. Statistically significant percentage difference is given in italics.

Case	AGL			% Difference	% Difference
	FSA	TFSA	TEFSA	TEFSA vs FSA	TEFSA vs TFSA
n33	0.491	0.597	0.733	<i>17.71</i>	<i>18.61</i>
n40	0.401	0.484	0.646	<i>17.12</i>	<i>25.09</i>

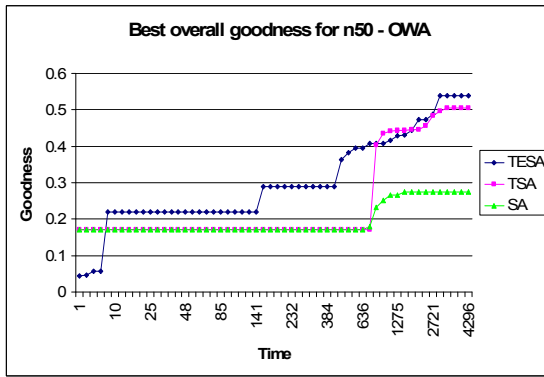
Figure 7.5 illustrates a typical performance pattern over time for TEFSA, TFSA, and FSA, with respect to the best value of membership function “Good topology” (i.e. best overall goodness) using the OWA operator. It is observed that, for all test cases, the overall goodness of TESA generally stays higher than that of FSA and TFSA. For example, for  $n50$ , the overall goodness of FSA, TFSA, and TEFSA is the same at time 0 (since all three algorithms start with the same initial solution). However, as time progresses, the overall goodness of TEFSA increases at a faster rate than FSA and TFSA. This pattern is also prominent in cases  $n25$  and  $n15$ . However, for  $n40$  and  $n33$ , there are some time instances where TESA attains higher overall

goodness than TEFSA. As for UAO, Figure 7.6 shows patterns similar to that of OWA. In all test cases, TEFSA either maintains a higher overall goodness or the same level of goodness, when compared with TESA and TSA.

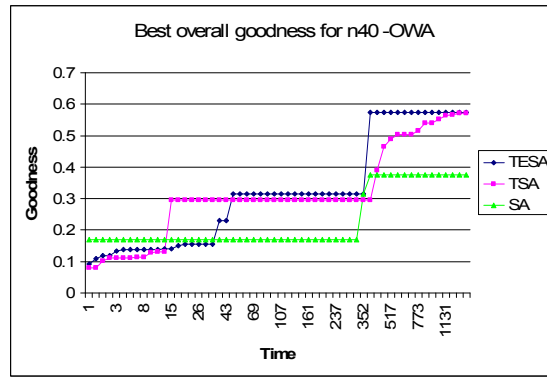
With respect to the performance patterns of TESA and TSA, Figures 7.5 and 7.6 show that TESA generally maintains a higher goodness level than TSA for both OWA and UAO. However, there are a few instances, such as *n50* in Figure 7.5(a) and *n15* in Figure 7.5(e), where both TESA and TSA maintain the same level of overall goodness for a major portion of the runtime.

## 7.4 Dynamic Markov chain size

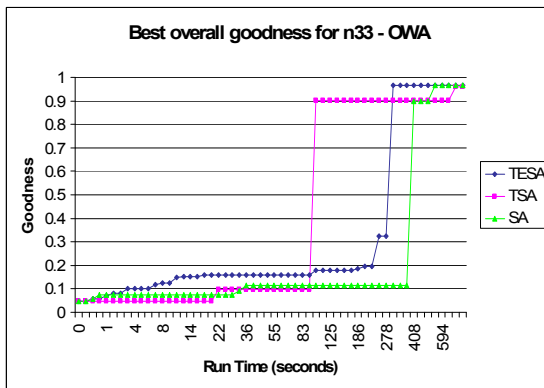
The effect of Markov chain,  $M$ , on the convergence of SA algorithm was discussed in detail in Section 2.4.4. A too high value of  $M$  increases the algorithm runtime unnecessarily, while a too low value may produce results of low quality. Therefore, it is necessary to find an appropriate value of  $M$ . It will be more efficient if the value of the Markov chain is dynamically adjusted. The motivation behind a dynamic value of the Markov chain is that it is computationally expensive to find the best value of  $M$  by a process of trial-and-error. A dynamic value of  $M$  can be achieved by enhancing the capabilities of the SA algorithm such that the algorithm itself adjusts the value of  $M$ , rather than using a pre-defined static value of  $M$ . Considering the above points, it is proposed that at each iteration,  $t$ , the value of the Markov chain be calculated as:



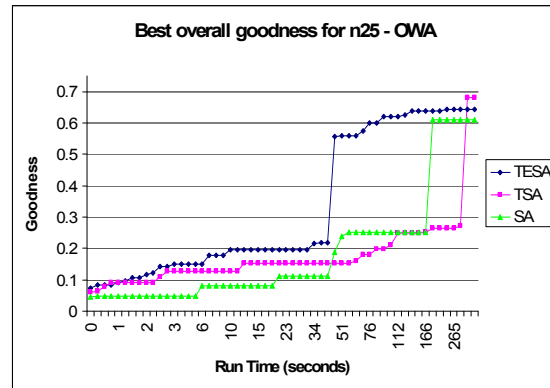
(a)



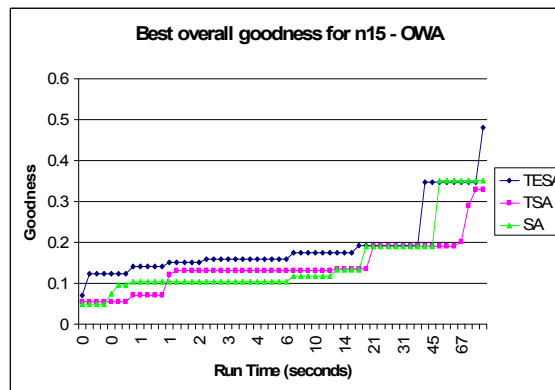
(b)



(c)

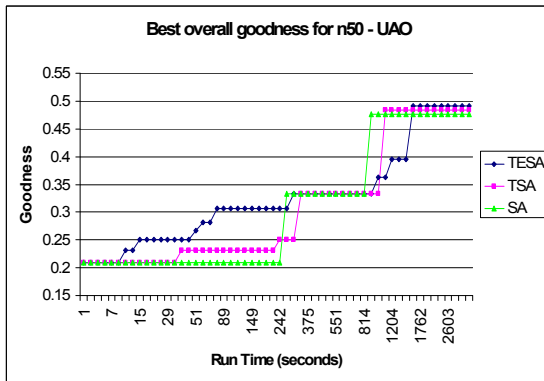


(d)

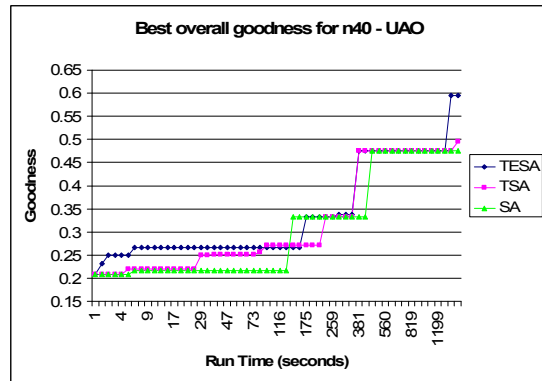


(e)

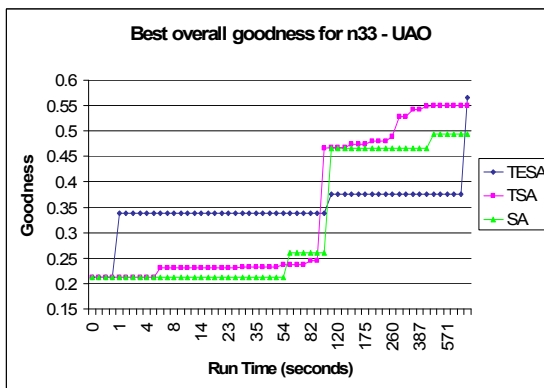
Figure 7.5: Plots of best value of membership function “Good topology” versus execution time using the OWA operator for FSA, TFSA, and TEFSA for (a) n50 (b) n40 (c) n33 (d) n25 (e) n15



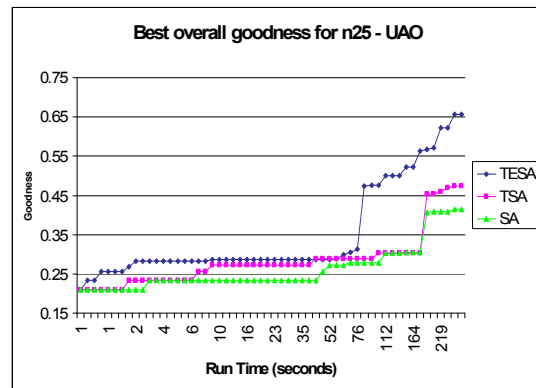
(a)



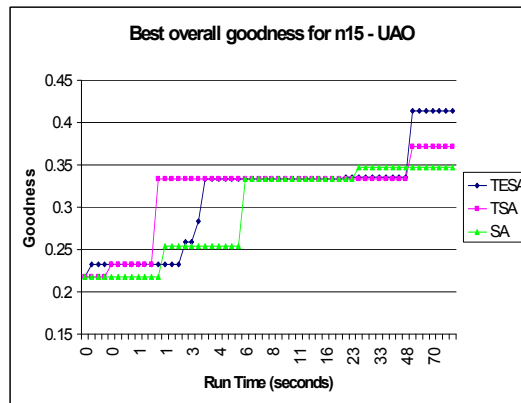
(b)



(c)



(d)



(e)

Figure 7.6: Plots of best value of membership function “Good topology” versus execution time using the UAO operator for FSA, TFSA, and TEFSA for (a) n50 (b) n40 (c) n33 (d) n25 (e) n15



$$M(t) = \begin{cases} M(t-1) + (\frac{1}{L} \sum_{i=1}^L g_{l_i}(t))/\mu & \text{if } \mu \neq 0 \\ M(t-1) + 1 & \text{if } \mu = 0 \end{cases} \quad (7.3)$$

where  $M(t)$  is the bias at iteration  $t$ ,  $g_{l_i}(t)$  is the goodness of all links in the solution at iteration  $t$ ,  $L$  is the total number of links in the solution at iteration  $t$ , and  $\mu$  is the overall goodness of the solution at time  $t$ . From Equation (7.3), the value of the Markov chain length is updated by a factor which is a function of the average goodness of links present in the current solution divided by the overall goodness (solution membership)  $\mu$  of the solution, provided that the goodness of solution is not zero. However, the goodness of solution can be zero in an iteration, resulting in  $M = \infty$ . To avoid this,  $M$  is incremented by 1 in the case where the goodness of solution is zero. The above strategy to dynamically adjust the value of  $M$  was incorporated into the TEFSA algorithm. The resulting algorithm is referred to as the DTEFSA algorithm.

Tables 7.10 and 7.11 respectively present the results obtained for dynamic  $M$  for both the OWA and UAO operators. It is obvious from these tables that the computation of  $M$  based on Equation (7.3) did not produce better results than FSA. For the OWA operator, a degradation was observed in the quality of overall goodness produced by DTEFSA in most of the cases, as shown in Table 7.10. This degradation was mainly in the range of 14% to almost 72%. There was an exception to the above trend, where an improvement of 2.13% was observed for DTEFSA with test case *n15*. As for the UAO, a trend similar to that of OWA is observed, where DTEFSA produced results of lesser quality for all cases. The degradation was in the range of 4% to 19.5%. Statistical testing with the t-test also confirmed that the

degradation in results reported in Tables 7.10 and 7.11 was statistically significant.

From the above discussion, a general observation is that DTEFSA was not able to produce better results than FSA. However, the extent of degradation of performance by DTEFSA for UAO was far less than that of OWA. This information gives motivation for improving the proposed dynamic  $M$  measure such that the gap observed in the quality of solution is reduced or completely eliminated. A modified approach for DTEFSA can be proposed where different combinations of parameters in Equation (7.3) can be tried.

Table 7.10: Comparison of FSA and DTEFSA for OWA. Time = Run time (in seconds). % imp shows percentage improvement achieved by DTEFSA compared to FSA. Statistically significant results are in italics.

Case	$\alpha_{SA}$	FSA		DTEFSA			% imp
		Avg. Overall Goodness	Time	$TL$	Avg. Overall Goodness	Time	
n15	0.95	0.206 $\pm$ 0.035	89.0	9	0.210 $\pm$ 0.057	93.9	2.13
n25	0.75	0.391 $\pm$ 0.145	314.4	9	0.257 $\pm$ 0.008	300.1	<i>-34.31</i>
n33	0.85	0.646 $\pm$ 0.363	765.1	9	0.184 $\pm$ 0.012	791.5	<i>-71.56</i>
n40	0.75	0.231 $\pm$ 0.067	1498.8	7	0.199 $\pm$ 0.007	1911.6	<i>-14.24</i>
n50	0.75	0.198 $\pm$ 0.046	4295.8	9	0.173 $\pm$ 0.019	4161.5	<i>-12.80</i>

#### 7.4.1 Comparison of OWA and UAO

The relative performance of the OWA and UAO operators was also compared with respect to the four individual objectives, by taking the averages of 30 runs for the three algorithms. With respect to the *Cost* objective, OWA performed slightly better than UAO for FSA, since OWA generated statistically better results for  $n25$  and  $n50$ , while UAO generated better results for  $n33$ , as shown in Table 7.12. With regard to TFSA, again OWA was better than UAO, since OWA generated better

Table 7.11: Comparison of FSA and DTEFSA for UAO. Time = Run time (in seconds). % imp shows percentage improvement achieved by DTEFSA compared to FSA. Statistically significant results are in italics.

Case	$\alpha_{SA}$	FSA		DTEFSA			% imp
		Avg. Overall Goodness	Time	$TL$	Avg. Overall Goodness	Time	
n15	0.99	0.335 $\pm$ 0.003	88.0	7	0.321 $\pm$ 0.028	98.9	<i>-3.93</i>
n25	0.75	0.345 $\pm$ 0.034	321.3	9	0.303 $\pm$ 0.007	225.8	<i>-12.08</i>
n33	0.75	0.339 $\pm$ 0.088	757.6	7	0.313 $\pm$ 0.017	739.1	<i>-7.65</i>
n40	0.85	0.377 $\pm$ 0.066	1564.5	9	0.304 $\pm$ 0.006	1950.9	<i>-19.49</i>
n50	0.85	0.350 $\pm$ 0.053	3485.3	9	0.285 $\pm$ 0.013	2496.9	<i>-18.65</i>

results for test cases  $n50$ ,  $n40$  and  $n25$  as shown in Table 7.13. For TEFSa, UAO performed better than OWA, since UAO produced solutions with lesser costs than OWA for  $n50$  and  $n25$ , while for  $n15$ ,  $n33$ , and  $n40$ , UAO and OWA produced results of equal quality, as statistically tested and reported in Table 7.14.

For the *Delay* objective, OWA performed relatively better than UAO for all three algorithms. For FSA, OWA produced lesser delays for two cases ( $n15$  and  $n40$ ) than UAO as shown in Table 7.15. Similarly, for TFSA, OWA showed more improvement for  $n50$ ,  $n40$ , and  $n15$ , and equal improvement for  $n25$  and  $n33$  as shown in Table 7.16. As for TEFSa, OWA had statistically better results for  $n50$ ,  $n40$ , and  $n25$ , and equal performance for  $n15$ , as shown in Table 7.17.

As far as *number of hops* is concerned, OWA generally performed better than UAO. For example, OWA performed better than UAO for all test cases except  $n50$ ,  $n40$ , and  $n15$  for FSA as shown in Table 7.18. For TFSA (Table 7.19), OWA had more improvement for  $n50$ ,  $n40$ , and  $n15$ . For TEFSa, OWA again performed better than UAO for test cases  $n50$ ,  $n33$ , and  $n25$ , as shown in Table 7.20.

Finally, for the *Reliability* objective, UAO and OWA performed equally well.

Table 7.12: Comparison of OWA and UAO for monetary cost of best solutions of 30 runs for FSA. % imp = percentage improvement achieved by UAO compared to OWA. Statistically significant results are in italics.

Case	OWA	UAO	% imp	Remarks
n15	11422.0 ±1266.5	10850.3 ±1475.9	5.0	UAO and OWA statistically equal
n25	12346.0 ±2549.7	15453.3 ±1831.8	<i>-25.2</i>	OWA statistically better
n33	27645.6 ±2730.4	22355.5 ±6419.9	<i>19.1</i>	UAO statistically better
n40	34932.9 ±4002.5	35566.0 ±2526.4	-1.8	UAO and OWA statistically equal
n50	45508.6 ±3772.7	50443.8 ±3405.3	<i>-10.8</i>	OWA statistically better

Table 7.13: Comparison of OWA and UAO for monetary cost of best solutions of 30 runs for TFSA. % imp = percentage improvement achieved by UAO compared to OWA. Statistically significant results are in italics.

Case	OWA	UAO	% imp	Remarks
n15	9647.7 ±910.3	9494.3 ±1108.6	1.6	UAO and OWA statistically equal
n25	10788.1 ±1448.1	13288.5 ±1747.4	<i>-23.2</i>	OWA statistically better
n33	26969.2 ±2641.5	26396.6 ±2622.1	2.1	UAO and OWA statistically equal
n40	31599.9 ±4728.8	35608.5 ±2250.0	<i>-12.7</i>	OWA statistically better
n50	45241.5 ±3835.8	49910.9 ±3329.4	<i>-10.3</i>	OWA statistically better

Table 7.14: Comparison of OWA and UAO for monetary cost of best solutions of 30 runs for TEFSA. % imp = percentage improvement achieved by UAO compared to OWA. Statistically significant results are in italics.

Case	OWA	UAO	% imp	Remarks
n15	7968.7 ±563.3	7716.7 ±701.6	3.2	UAO and OWA statistically equal
n25	9792.2 ±1255.2	8580.2 ±1736.7	<i>12.4</i>	UAO statistically better
n33	17423.3 ±1514.4	18067.1 ±2595.9	-3.7	UAO and OWA statistically equal
n40	21851.2 ±1888.8	21701.9 ±1401.0	0.7	UAO and OWA statistically equal
n50	31642.0 ±6766.2	28239.2 ±3657.3	<i>10.8</i>	UAO statistically better

Table 7.15: Comparison of OWA and UAO for delay of best solutions of 30 runs for FSA. % imp = percentage improvement achieved by UAO compared to OWA. Statistically significant results are in italics.

Case	OWA	UAO	% imp	Remarks
n15	0.0031 ±0.0002	0.0036 ±0.0004	<i>-15.8</i>	OWA statistically better
n25	0.0036 ±0.0004	0.0037 ±0.0007	-2.8	UAO and OWA statistically equal
n33	0.0076 ±0.0038	0.0083 ±0.0145	-9.7	UAO and OWA statistically equal
n40	0.0061 ±0.0008	0.0095 ±0.0109	<i>-55.4</i>	OWA statistically better
n50	0.0075 ±0.0073	0.0083 ±0.0038	-10.0	UAO and OWA statistically equal

Table 7.16: Comparison of OWA and UAO for delay of best solutions of 30 runs for TFSA. % imp = percentage improvement achieved by UAO compared to OWA. Statistically significant results are in italics.

Case	OWA	UAO	% imp	Remarks
n15	0.0030 ±0.0002	0.0032 ±0.0003	<i>-5.24</i>	OWA statistically better
n25	0.0034 ±0.0002	0.0034 ±0.0004	0.36	UAO and OWA statistically equal
n33	0.0066 ±0.0059	0.0064 ±0.0063	4.19	UAO and OWA statistically equal
n40	0.0055 ±0.0006	0.0071 ±0.0024	<i>-29.85</i>	OWA statistically better
n50	0.0056 ±0.0023	0.0093 ±0.0134	<i>-65.37</i>	OWA statistically better

Table 7.17: Comparison of OWA and UAO for delay of best solutions of 30 runs for TEFSA. % imp = percentage improvement achieved by UAO compared to OWA. Statistically significant results are in italics.

Case	OWA	UAO	% imp	Remarks
n15	0.0030 ±0.0002	0.0030 ±0.0002	0.0	UAO and OWA statistically equal
n25	0.0034 ±0.0002	0.0039 ±0.0008	<i>-15.1</i>	OWA statistically better
n33	0.0073 ±0.0127	0.0058 ±0.0014	<i>20.6</i>	UAO statistically better
n40	0.0056 ±0.0005	0.0066 ±0.0016	<i>-17.4</i>	OWA statistically better
n50	0.0055 ±0.0032	0.0066 ±0.0039	<i>-19.7</i>	OWA statistically better

Table 7.18: Comparison of OWA and UAO for number of hops of best solutions of 30 runs for FSA. % imp = percentage improvement achieved by UAO compared to OWA. Statistically significant results are in italics.

Case	OWA	UAO	% imp	Remarks
n15	5.9 ±0.7	7.2 ±1.3	<i>-20.8</i>	OWA statistically better
n25	6.3 ±1.4	6.5 ±1.9	-4.3	UAO and OWA statistically equal
n33	11.2 ±2.9	9.7 ±3.2	<i>13.9</i>	UAO statistically better
n40	9.5 ±1.2	11.7 ±1.5	<i>-23.2</i>	OWA statistically better
n50	9.9 ±1.5	12.0 ±1.9	<i>-20.5</i>	OWA statistically better

Table 7.19: Comparison of OWA and UAO for number of hops of best solutions of 30 runs for TFSA. % imp = percentage improvement achieved by UAO compared to OWA. Statistically significant results are in italics.

Case	OWA	UAO	% imp	Remarks
n15	5.6 ±0.8	6.3 ±1.0	<i>-13.10</i>	OWA statistically better
n25	5.3 ±0.9	5.1 ±1.3	3.77	UAO and OWA statistically equal
n33	9.7 ±2.5	8.6 ±2.9	11.38	UAO and OWA statistically equal
n40	8.5 ±1.2	11.0 ±1.4	<i>-29.80</i>	OWA statistically better
n50	8.8 ±1.3	10.8 ±1.9	<i>-22.64</i>	OWA statistically better

Table 7.20: Comparison of OWA and UAO for number of hops of best solutions of 30 runs for TEFSA. % imp = percentage improvement achieved by UAO compared to OWA. Statistically significant results are in italics.

Case	OWA	UAO	% imp	Remarks
n15	5.2 ±0.7	5.5 ±0.6	-5.1	UAO and OWA statistically equal
n25	5.5 ±1.2	6.8 ±2.5	<i>-23.5</i>	OWA statistically better
n33	7.8 ±1.0	8.7 ±1.9	<i>-10.6</i>	OWA statistically better
n40	9.1 ±1.0	9.2 ±1.2	-0.7	UAO and OWA statistically equal
n50	7.9 ±1.1	8.8 ±1.2	<i>-11.8</i>	OWA statistically better

Table 7.21: Comparison of OWA and UAO for reliability of best solutions of 30 runs for FSA. % imp = percentage improvement achieved by UAO compared to OWA. Statistically significant results are in italics.

Case	OWA	UAO	% imp	Remarks
n15	0.273 ±0.000	0.273 ±0.000	0.0	UAO and OWA statistically equal
n25	0.091 ±0.019	0.089 ±0.018	2.9	UAO and OWA statistically equal
n33	0.064 ±0.022	0.059 ±0.011	8.5	UAO and OWA statistically equal
n40	0.036 ±0.005	0.037 ±0.000	-3.6	UAO and OWA statistically equal
n50	0.023 ±0.004	0.024 ±0.003	-4.4	UAO and OWA statistically equal

Table 7.22: Comparison of OWA and UAO for reliability of best solutions of 30 runs for TFSA. % imp = percentage improvement achieved by UAO compared to OWA. Statistically significant results are in italics.

Case	OWA	UAO	% imp	Remarks
n15	0.273 ±0.000	0.272 ±0.000	0.04	UAO and OWA statistically equal
n25	0.092 ±0.038	0.087 ±0.015	5.86	UAO and OWA statistically equal
n33	0.061 ±0.003	0.063 ±0.017	-3.32	UAO and OWA statistically equal
n40	0.037 ±0.000	0.037 ±0.000	0.02	UAO and OWA statistically equal
n50	0.025 ±0.000	0.025 ±0.000	0.14	UAO and OWA statistically equal

Table 7.23: Comparison of OWA and UAO for reliability of best solutions of 30 runs for TEFSA. % imp = percentage improvement achieved by UAO compared to OWA. Statistically significant results are in italics.

Case	OWA	UAO	% imp	Remarks
n15	0.273 ±0.001	0.272 ±0.000	0.1	UAO and OWA statistically equal
n25	0.095 ±0.031	0.088 ±0.014	<i>7.9</i>	UAO statistically better
n33	0.060 ±0.000	0.060 ±0.000	0.2	UAO and OWA statistically equal
n40	0.037 ±0.000	0.037 ±0.000	0.0	UAO and OWA statistically equal
n50	0.020 ±0.007	0.024 ±0.004	<i>-19.0</i>	OWA statistically better

For example, both UAO and OWA produced statistically equal results for FSA as observed in Table 7.21. The same situation was observed in Table 7.22 for TFSA where OWA and UAO showed statistically equal improvement for all test cases. Finally, for TEFSA, a trend similar to FSA and TFSA was observed (refer to Table 7.23), where UAO and OWA had equal improvement for four test cases. The only exception was  $n50$ , where OWA showed statistically better improvement than UAO.

## 7.5 Conclusion

This chapter presented a novel approach for topology design of DLANs based on a fuzzy simulated annealing (FSA) algorithm. Two variants of the FSA algorithm were proposed, namely tabu FSA (TFSA) and evolutionary tabu FSA (TEFSA). TFSA combines FSA with tabu search, while TEFSA integrated features of tabu search and simulated evolution with FSA. The performance of the three algorithms was evaluated using both the OWA and UAO operators. Results suggest that TEFSA generally performed better than the other two variants. Moreover, the performance of UAO and OWA were found to be comparable. Another issue discussed in the chapter was the proposal of dynamic assignment of the Markov chain,  $M$ . Although the proposed computation of dynamic  $M$  produced lower quality results than FSA, degradation in quality was not significant.

The next chapter proposes and discusses a swarm intelligence technique, namely ant colony optimization, in the context of its application to the multi-objective DLAN topology design problem.



## Chapter 8

# Fuzzy Ant Colony Optimization

## Algorithm for DLAN Topology

### Design

This chapter presents a fuzzy multi-objective ant colony optimization algorithm for the DLAN topology design problem. The chapter first discusses the main features of the algorithm, which include the discussion on the fuzzy heuristic value, and the fuzzy multi-objective function to assess the quality of solution generated by an ant. This is followed by empirical results to evaluate the performance of the fuzzy ACO algorithm. This evaluation is based on a number of ACO parameters, which include pheromone deposit and evaporation rate and the number of ants. The performance of the OWA and UAO operators is also evaluated through their usage with the ant colony algorithm.

## 8.1 Fuzzy Ant Colony Optimization Algorithm

As discussed earlier in Section 2.4.6, Ant Colony Optimization (ACO) is a relatively new optimization algorithm. The ACO maintains a population of ants, where each ant is responsible for building a feasible network topology. The ant starts with the root node and incrementally builds a topology. The guiding factors in the process of decision and selection of a particular path are the heuristic value, pheromone trail, and evaporation. A complete tour by an ant results in a complete feasible network topology. Below, each step of the proposed fuzzy ACO is presented.

### 8.1.1 Initialization (Generation of Ants)

Since ACO is a population-based algorithm, initialization consists of generating a set of candidate solutions. These initial solutions are generated randomly. That is, each ant produces a solution without any feedback or previous information about the paths. However, while an ant is generating a solution, the constraints are checked at each step to ensure feasible initial solutions. Algorithm parameters (such as pheromone deposit and evaporation rates, as well as values of  $\alpha_{ant}$  and  $\beta_{ant}$ ) are also initialized in this phase.

### 8.1.2 Ants Activity

Once the initial set of solutions is generated, the next step is to find the probability of selecting link  $l$  between nodes  $i$  and  $j$  for the  $k^{th}$  ant, and to apply pheromone deposit and pheromone evaporation rules. The approach adopted in this work is to update pheromone values based on the solution generated by the “elitist” ant

only (discussed in Section 2.4.6). More specifically, the iteration-based elitist ant approach [10, 117, 170] has been used in this thesis. In every iteration following initialization, each ant constructs a solution using the link selection probability function given in the following equation:

$$p_{ij}^k(t) = \frac{[\tau_{ij}(t)]^{\alpha_{ant}} [\eta_{ij}]^{\beta_{ant}}}{\sum_{l \in N_i} [\tau_{ij}(t)]^{\alpha_{ant}} [\eta_{ij}]^{\beta_{ant}}} \quad (8.1)$$

where  $N_i$  is the set of neighbors of node  $i$ ,  $p_{ij}^k(t)$ , is the probability of selecting a connection  $l$  between nodes  $i$  and  $j$  for the  $k^{th}$  ant,  $\tau_{ij}$  is the pheromone on connection  $l$ , and  $\eta_{ij}$  is the heuristic value of connection  $l$ .  $\alpha_{ant}$  and  $\beta_{ant}$  represent the influence of pheromone content and heuristic respectively.

Once each ant has finished its tour and found a valid solution, the ant which generates the highest value for Equation (3.11) (or Equation (4.1)) is taken as the elitist ant. This ant is allowed to update the pheromone value. For each link that occurs in the solution of the elitist ant, pheromone is updated by adding  $\Delta\tau_{ij}$  to the current pheromone concentration of that link. Pheromone evaporation takes place on all links, irrespective of whether the links are part of the elitist ant or not. The updated values of pheromone are then passed to following ants, and the above cycle is repeated until a predefined stopping condition is met.

The prime advantage of the above elitist ant approach is that the approach forces the ants to search more rigorously near the best solution that has been found so far. However, the approach can have a disadvantage of exploiting too much, and exploring less.

### 8.1.3 Fuzzy Heuristic Value

Another novelty of the proposed fuzzy ACO is the introduction of a fuzzy heuristic value in Equation (8.1) above. The heuristic value  $\eta_{ij}$  basically provides an insight into the structure of the individual elements of the solution. In the context of DLAN topology design, each element is a link between any two nodes  $i$  and  $j$ . The heuristic value of a link is calculated exactly as given in Equation (6.1). Details of calculating the upper and lower bounds for each objective in Equation (6.1) have already been discussed in Chapter 6.

## 8.2 Results and Discussion

The fuzzy ACO was applied to the five test cases. To give equal importance to pheromone level on a link and the heuristic value of the link, both  $\alpha_{ant}$  and  $\beta_{ant}$  were taken as 0.5. The main focus of the experiments was to study the effect of the colony size (i.e. number of ants), and the effect of pheromone deposit and evaporation, and also to evaluate the performance of the ACO using the OWA and UAO operators. Table 8.1 shows the combinations of pheromone deposit and evaporation rates used in the experiments. The colony was taken as 10, 15, 20, 25, and 30 ants. Thirty independent runs were made for each parameter setup, and the average of best solutions found in each run was reported, with the standard deviation. Each test instance was run for 100 iterations.

Table 8.1: Parameter settings for fuzzy ACO used in experiments. DEP = difference between pheromone deposit and evaporation rates.

Parameter set	Pheromone rates		DEP
	Deposit	Evaporation	
Set 1	0.2	0.0	0.2
Set 2	0.4	0.1	0.3
Set 3	0.6	0.2	0.4
Set 4	0.8	0.3	0.5

### 8.2.1 Effect of Pheromone Deposit and Evaporation

An objective of the work was to study the effect of pheromone deposit and evaporation. In the original ACO algorithm, the trail is updated proportional to the quality of solution as given in Equation (2.24). This does not provide enough insight into the effect of high and low pheromone deposit and evaporation rates. Therefore, pheromone update and evaporation were statically assigned to links. The parameters in Table 8.1 were used to study the effect of pheromone deposit and evaporation. These parameters depict different rates of pheromone deposit and evaporation, as given in columns 2 and 3 respectively. Column 4 of the table depicts the difference between the pheromone deposit and evaporation rate, denoted by DEP. Note that in Table 8.1, the gap between pheromone deposit and evaporation rates (i.e. DEP) increases with each set. For example, in Set 1, the difference between deposit and evaporation was 0.2. DEP gradually increased in each set and reached a maximum of 0.5 in Set 4. Experiments were done using 20 ants.

Table 8.2 depicts the best and worst overall goodness achieved using OWA. The last column depicts the percentage improvement obtained when the two situations were compared. As an example, consider the case of  $n25$ , where the overall goodness

of 0.421 was obtained with  $DEP = 0.4$  (i.e. deposit = 0.6 and evaporation = 0.2), while for  $DEP = 0.2$  (representing deposit = 0.2, and evaporation = 0.0), the overall goodness was 0.277. This resulted in an improvement of 34.17% which was achieved when the gap between deposit and evaporation was high. As is obvious from the table, moderate to high improvements were achieved for almost all cases. An exception was case *n15*, where the best results were obtained when the deposit rate was low and no evaporation was allowed ( $DEP = 0.2$ ), giving an improvement of 17.01% over the lowest goodness levels obtained with  $DEP = 0.5$ . A general trend in this table suggests that the worst results were obtained when the difference between the deposit and evaporation was the lowest (i.e. deposit = 0.2, evaporation = 0.0). Nevertheless, all percentage improvements in Table 8.2 were statistically significant, as validated by the t-test.

Table 8.2: Results for best and worst average overall goodness and their respective pheromone deposit and evaporation rate setup using OWA. Time = Run time (in seconds), % imp = percentage improvement. Statistically significant improvement is in italics.

Case	DEP	Max. Goodness	Time	DEP	Min. Goodness	Time	% imp
n15	0.2	0.294 ±0.033	39.3	0.5	0.244 ±0.024	25.4	<i>17.01</i>
n25	0.4	0.421 ±0.022	222.9	0.2	0.277 ±0.021	230.7	<i>34.17</i>
n33	0.3	0.342 ±0.030	452.5	0.2	0.254 ±0.023	458.2	<i>25.79</i>
n40	0.5	0.331 ±0.021	1228.6	0.2	0.279 ±0.014	2044.7	<i>15.62</i>
n50	0.5	0.258 ±0.022	4050.3	0.2	0.217 ±0.014	12572.9	<i>15.86</i>

The results presented in Table 8.3 for UAO followed the same pattern as that of OWA. In this table, it is observed that most of these improvements were of moderate level, generally greater than 15%. Again, an exception was in the case of *n15*, where a slight improvement of 6.21% was observed. It is also obvious from the table that

Table 8.3: Results for best and worst average overall goodness and their respective pheromone deposit and evaporation rate setup using UAO. Time = Run time (in seconds), % imp = percentage improvement. Statistically significant improvement is in italics.

Case	DEP	Max. Goodness	Time	DEP	Min. Goodness	Time	% imp
n15	0.5	0.333 ±0.000	25.4	0.2	0.312 ±0.009	25.3	<i>6.21</i>
n25	0.3	0.360 ±0.009	246.0	0.2	0.289 ±0.004	245.4	<i>19.81</i>
n33	0.5	0.347 ±0.004	501.9	0.2	0.268 ±0.007	516.3	<i>22.75</i>
n40	0.4	0.346 ±0.006	1143.0	0.2	0.290 ±0.003	1741.2	<i>16.30</i>
n50	0.4	0.334 ±0.002	4239.4	0.2	0.264 ±0.006	12461.4	<i>20.93</i>

for all test cases, the worst results were obtained when  $DEP = 0.2$ . However, all percentage improvements were found to be statistically significant, as validated by the t-test.

The results and observations presented in Tables 8.2 and 8.3 can be attributed to the following explanation. When the pheromone deposit rate is low, and no evaporation takes place (a situation associated with  $DEP = 0.2$ ), then the pheromone contents on links present in the elitist ant are not much different from the pheromone contents on links not present in the elitist ant. Consequently, there will be a little difference between the probabilities,  $p_{ij}^k(t)$ , which makes selection of links more random. In general, this situation would persist as time elapses. There might be very few links which would have accumulated a noticeable amount of pheromone, and therefore would be selected again and again. On the other hand, if the pheromone deposit rate is high, and pheromone evaporation also takes place on links (a situation associated with  $DEP = 0.3, 0.4, \text{ and } 0.5$ ), then the amount of pheromone deposited on links in the elitist ant will be substantially higher than the pheromone on links not present in the elitist ant. Thus, there will be a significant difference between

the probabilities,  $p_{ij}^k(t)$ . Therefore, links with high pheromone content would become strong candidates for selection, resulting in the search to be more directional. Therefore, for the same amount of time, FACO will converge faster towards a sub-optimal solution for situations where DEP is high as compared to instances where DEP is low. The above claim is supported by the plots in Figure 8.1. This figure shows a typical behavior of FACO with  $DEP = 0.2$  and  $DEP = 0.5$  for test case  $n50$  using the UAO operator. It is observed in this figure that with  $DEP = 0.5$ , the improvement in the overall goodness is much higher than that obtained with  $DEP = 0.2$  for the same number of iterations.

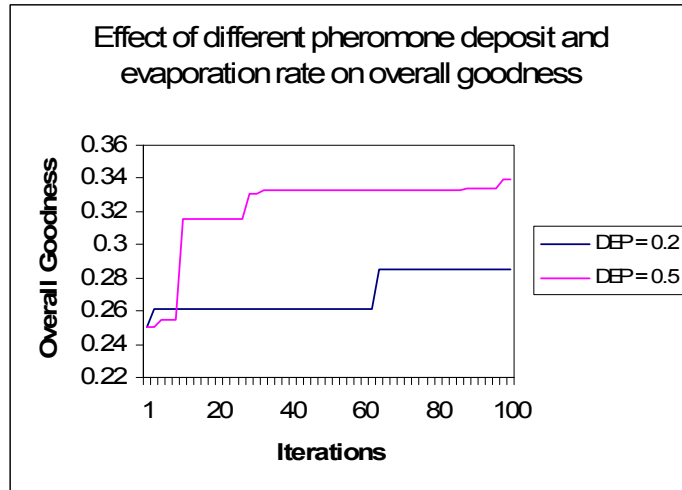


Figure 8.1: Plot for overall goodness for test case  $n50$  using FACO with  $DEP = 0.2$  and  $DEP = 0.5$

### 8.2.2 Effect of Number of Ants

Tables 8.4 to 8.8 show the effect of the number of ants on the overall goodness of the solution using OWA. The tables consist of nine columns. The first column shows the number of ants, ranging from 10 to 30. Columns 2 to 9 display the results



and run time obtained using the four parameter setups according to Table 8.1. It is observed in these tables that, collectively, an increasing number of ants had a positive impact on the quality of the final solution. The trend was observed for all four parameter sets. For example, in column 8 of Table 8.4, it is observed for case  $n50$  with  $DEP = 0.5$  that the overall goodness of solution was 0.239 for 10 ants, which increased with each step as the number of ants increased. An overall goodness value of 0.270 was achieved with 30 ants at the end. This improvement is given as a percentage in Table 8.14 with a value of 11.21% when OWA was used. In a similar manner, percentage improvements for other cases are also given the table. A validation through t-test showed that, in general, all improvements were statistically significant. The percentage improvements are also illustrated in Figure 8.2(a). Another interesting observation in Tables 8.4 to 8.8 is that the best results (marked in boldface) were obtained with the number of ants being 30 in most cases. An exception is case  $n40$ , where the best overall goodness was achieved with 25 ants.

Tables 8.9 to 8.13 depict the behavior of the ACO algorithm with UAO when the number of ants was increased from 10 to 30. For UAO, the pattern was similar to that of OWA, where it is observed that, when the number of ants were increased from 10 to 30 in progression, the overall goodness of solution also increased proportionally. The percentage improvements are given in Table 8.14 and depicted in Figure 8.2(b). The results in Table 8.14 show that the improvements were statistically significant, as validated by the t-test. As with the case of OWA, the best results were obtained when 30 ants were used, with the exception of  $n33$  where the best solution was obtained with the number of ants equal to 25.

The above trends appear to be logical. Note that each ant is independent in

generating its solution, although the elitist ant influences the pheromone concentrations on the links of the solution represented by the ant. Increasing the number of ants initially enhances diversity in solutions, thus increasing the probability of finding a new elite ant of better quality. This in turn allows the elitist ant to pass information of links having higher quality to ants in the next iteration.

Another interesting observation in Tables 8.4 to 8.8 is about the quality of solutions obtained with different pheromone deposit and evaporation rates. A careful analysis of these tables shows that the level of overall goodness achieved with DEP values of 0.3, 0.4, and 0.5 was almost in the same range, while with  $DEP = 0.2$  the quality of overall goodness was considerably less. As an example, consider Table 8.5, where the overall goodness for  $DEP = 0.3$  varied between 0.310 and 0.333, for  $DEP = 0.4$  between 0.312 and 0.331, and for  $DEP = 0.5$  between 0.314 and 0.331. These ranges are very close to each other. However, when  $DEP = 0.2$  was considered, the goodness ranged from 0.268 to 0.283. This range is relatively much less than the others mentioned above. The trend was more or less true for all test cases, with the exception of *n15*.

The same trend was also very visible for UAO for all cases. One possible reason for this behavior is that for  $DEP = 0.2$  (with no evaporation), there is no feedback from ‘bad’ links. In other words, although the good links are rewarded by pheromone deposit, bad links are not penalized since no evaporation takes place on these links. Therefore, their quality does not deteriorate, in contrast to other cases with DEP from 0.3 to 0.5, where bad links are penalized. This penalty has a positive effect on the faster convergence of the algorithm, since good quality links will be passed to later generations again and again, while bad quality links will be annihilated from

Table 8.4: Results for n50 with OWA for different population size, pheromone deposit rate, and evaporation rate. OG = average overall goodness with standard deviation.

Number of Ants	DEP = 0.2		DEP = 0.3		DEP = 0.4		DEP = 0.5	
	OG	Time	OG	Time	OG	Time	OG	Time
10	0.208 ±0.015	6399.1	0.244 ±0.028	2666.7	0.243 ±0.022	2166.7	0.239 ±0.028	1995.4
15	0.217 ±0.015	8246.5	0.247 ±0.028	3925.3	0.257 ±0.041	3248.2	0.238 ±0.022	3011.6
20	0.217 ±0.014	12572.9	0.251 ±0.028	5299.8	0.250 ±0.017	4250.3	0.258 ±0.022	4050.3
25	0.224 ±0.014	15465.0	0.252 ±0.019	6660.7	0.254 ±0.021	5498.5	0.267 ±0.030	4970.7
30	0.219 ±0.015	18884.8	0.262 ±0.021	7838.6	0.253 ±0.022	6632.7	<b>0.270</b> ±0.032	5923.3

Table 8.5: Results for n40 with OWA for different population size, pheromone deposit rate, and evaporation rate. Goodness = average overall goodness with standard deviation.

Number of Ants	DEP = 0.2		DEP = 0.3		DEP = 0.4		DEP = 0.5	
	OG	Time	OG	Time	OG	Time	OG	Time
10	0.268 ±0.012	1213.1	0.310 ±0.019	849.1	0.312 ±0.020	817.8	0.326 ±0.025	811.0
15	0.271 ±0.012	1681.9	0.324 ±0.026	1066.0	0.323 ±0.022	1034.1	0.314 ±0.019	1019.2
20	0.279 ±0.014	2044.7	0.322 ±0.020	1286.6	0.323 ±0.020	1246.1	0.331 ±0.021	1228.6
25	0.283 ±0.021	2450.1	<b>0.333</b> ±0.028	1521.2	0.331 ±0.021	1464.9	0.330 ±0.019	1437.2
30	0.283 ±0.016	2906.9	0.323 ±0.022	1129.3	0.315 ±0.020	1068.7	0.324 ±0.020	1042.6

Table 8.6: Results for n33 with OWA for different population size, pheromone deposit rate, and evaporation rate. Goodness = average overall goodness with standard deviation.

Number of Ants	DEP = 0.2		DEP = 0.3		DEP = 0.4		DEP = 0.5	
	OG	Time	OG	Time	OG	Time	OG	Time
10	0.249 ±0.024	430.4	0.333 ±0.027	426.7	0.328 ±0.028	426.0	0.317 ±0.023	426.0
15	0.236 ±0.013	445.7	0.331 ±0.020	439.3	0.340 ±0.028	438.7	0.335 ±0.023	438.3
20	0.254 ±0.023	458.2	0.342 ±0.030	452.5	0.333 ±0.020	451.0	0.339 ±0.022	451.2
25	0.256 ±0.021	474.8	0.343 ±0.024	466.3	0.341 ±0.026	464.3	0.341 ±0.020	464.9
30	0.267 ±0.022	489.5	0.348 ±0.024	478.5	0.343 ±0.022	478.7	<b>0.362</b> ±0.022	553.3

Table 8.7: Results for n25 with OWA for different population size, pheromone deposit rate, and evaporation rate. Goodness = average overall goodness with standard deviation.

Number of Ants	DEP = 0.2		DEP = 0.3		DEP = 0.4		DEP = 0.5	
	OG	Time	OG	Time	OG	Time	OG	Time
10	0.272 ±0.018	214.9	0.389 ±0.015	215.0	0.403 ±0.027	213.3	0.404 ±0.024	211.6
15	0.281 ±0.018	221.8	0.410 ±0.028	221.0	0.417 ±0.030	217.1	0.421 ±0.022	217.1
20	0.277 ±0.021	230.7	0.414 ±0.022	225.5	0.421 ±0.022	222.9	0.415 ±0.020	222.9
25	0.293 ±0.021	237.0	0.419 ±0.024	229.2	0.423 ±0.030	228.3	0.425 ±0.030	228.4
30	0.294 ±0.016	238.8	0.426 ±0.023	233.8	<b>0.430</b> ±0.023	234.0	0.423 ±0.023	234.1

Table 8.8: Results for n15 with OWA for different population size, pheromone deposit rate, and evaporation rate. Goodness = average overall goodness with standard deviation.

Number of Ants	DEP = 0.2		DEP = 0.3		DEP = 0.4		DEP = 0.5	
	OG	Time	OG	Time	OG	Time	OG	Time
10	0.276 ±0.029	12.9	0.251 ±0.026	13.0	0.244 ±0.044	13.0	0.237 ±0.029	13.2
15	0.290 ±0.018	19.0	0.257 ±0.032	19.0	0.250 ±0.033	19.1	0.254 ±0.041	19.2
20	0.294 ±0.033	39.3	0.249 ±0.023	40.3	0.253 ±0.039	27.5	0.244 ±0.024	25.4
25	0.302 ±0.027	47.9	0.279 ±0.034	47.5	0.256 ±0.027	31.7	0.248 ±0.024	44.6
30	<b>0.313</b> ±0.032	50.3	0.277 ±0.044	31.0	0.259 ±0.027	31.2	0.256 ±0.033	31.2

Table 8.9: Results for n50 with UAO for different population size, pheromone deposit rate, and evaporation rate. OG = average overall goodness with standard deviation.

Number of Ants	DEP = 0.2		DEP = 0.3		DEP = 0.4		DEP = 0.5	
	OG	Time	OG	Time	OG	Time	OG	Time
10	0.258 ±0.007	6073.4	0.331 ±0.004	2729.9	0.333 ±0.004	2237.0	0.333 ±0.004	2008.9
15	0.262 ±0.008	9452.9	0.333 ±0.002	4029.3	0.333 ±0.002	3418.6	0.334 ±0.004	2980.7
20	0.264 ±0.006	12461.4	0.334 ±0.002	5498.3	0.334 ±0.002	4239.4	0.334 ±0.004	3980.6
25	0.266 ±0.005	15940.8	0.334 ±0.003	6681.8	0.335 ±0.003	5478.4	0.336 ±0.004	5110.3
30	0.265 ±0.009	16546.5	0.335 ±0.003	8132.1	<b>0.336</b> ±0.004	6478.8	0.334 ±0.002	5891.5

Table 8.10: Results for n40 with UAO for different population size, pheromone deposit rate, and evaporation rate. OG = average overall goodness with standard deviation.

Number of Ants	DEP = 0.2		DEP = 0.3		DEP = 0.4		DEP = 0.5	
	OG	Time	OG	Time	OG	Time	OG	Time
10	0.288 ±0.003	1032.3	0.344 ±0.008	741.9	0.344 ±0.007	727.8	0.345 ±0.008	716.0
15	0.289 ±0.003	1416.5	0.346 ±0.006	947.6	0.345 ±0.005	931.3	0.343 ±0.005	919.6
20	0.290 ±0.003	1741.2	0.346 ±0.006	1178.5	0.346 ±0.006	1143.0	0.345 ±0.005	1142.5
25	0.290 ±0.003	2178.4	0.348 ±0.008	1403.9	0.349 ±0.005	1358.5	0.350 ±0.007	1341.3
30	0.290 ±0.002	2602.5	0.349 ±0.006	1614.2	<b>0.352</b> ±0.006	1561.9	0.348 ±0.007	1534.3

Table 8.11: Results for n33 with UAO for different population size, pheromone deposit rate, and evaporation rate. Goodness = average overall goodness with standard deviation.

Number of Ants	DEP = 0.2		DEP = 0.3		DEP = 0.4		DEP = 0.5	
	OG	Time	OG	Time	OG	Time	OG	Time
10	0.265 ±0.007	459.8	0.343 ±0.005	451.8	0.345 ±0.007	451.4	0.345 ±0.005	451.2
15	0.264 ±0.005	487.5	0.345 ±0.004	477.8	0.346 ±0.006	477.1	0.345 ±0.005	476.6
20	0.268 ±0.007	516.3	0.347 ±0.005	502.9	0.347 ±0.005	502.5	0.347 ±0.004	501.9
25	0.271 ±0.006	551.5	0.348 ±0.005	528.4	0.348 ±0.006	528.1	<b>0.349</b> ±0.006	528.1
30	0.270 ±0.006	177.5	0.348 ±0.005	153.9	0.348 ±0.004	152.7	0.348 ±0.004	152.7

Table 8.12: Results for n25 with UAO for different population size, pheromone deposit rate, and evaporation rate. Goodness = average overall goodness with standard deviation.

Number of Ants	DEP = 0.2		DEP = 0.3		DEP = 0.4		DEP = 0.5	
	OG	Time	OG	Time	OG	Time	OG	Time
10	0.284 ±0.004	227.4	0.355 ±0.007	227.0	0.357 ±0.010	224.4	0.358 ±0.008	223.4
15	0.288 ±0.007	233.9	0.355 ±0.009	238.7	0.358 ±0.013	237.8	0.357 ±0.007	237.9
20	0.289 ±0.004	245.4	0.360 ±0.009	246.0	0.358 ±0.008	246.1	0.359 ±0.008	246.1
25	0.288 ±0.006	266.8	0.359 ±0.009	257.1	0.361 ±0.008	257.2	0.361 ±0.006	257.2
30	0.289 ±0.004	278.7	<b>0.363</b> ±0.008	268.5	0.362 ±0.008	268.7	0.359 ±0.006	268.6

Table 8.13: Results for n15 with UAO for different population size, pheromone deposit rate, and evaporation rate. Goodness = average overall goodness with standard deviation.

Number of Ants	DEP = 0.2		DEP = 0.3		DEP = 0.4		DEP = 0.5	
	OG	Time	OG	Time	OG	Time	OG	Time
10	0.310 ±0.011	13.0	0.333 ±0.000	13.2	0.333 ±0.000	13.2	0.333 ±0.000	13.2
15	0.312 ±0.011	19.2	0.333 ±0.000	19.2	0.333 ±0.000	19.3	0.334 ±0.002	19.3
20	0.312 ±0.009	25.3	0.333 ±0.001	25.5	0.333 ±0.000	25.5	0.333 ±0.000	25.4
25	0.319 ±0.008	48.1	0.334 ±0.002	47.8	0.334 ±0.002	40.6	0.334 ±0.002	34.8
30	0.315 ±0.007	53.1	0.333 ±0.001	31.5	0.333 ±0.000	31.4	<b>0.334</b> ±0.002	31.3

Table 8.14: Improvement with respect to increase in number of ants for different DEP rates using OWA. Statistically significant improvements are in italics.

Case	% Improvement			
	DEP = 0.2	DEP = 0.3	DEP = 0.4	DEP = 0.5
n15	<i>12.03</i>	<i>9.46</i>	5.66	<i>7.66</i>
n25	<i>7.35</i>	<i>8.77</i>	<i>6.21</i>	<i>4.50</i>
n33	<i>6.91</i>	<i>4.34</i>	<i>4.54</i>	<i>12.43</i>
n40	<i>5.43</i>	<i>6.93</i>	<i>5.65</i>	1.30
n50	<i>4.94</i>	<i>7.12</i>	3.91	<i>11.21</i>

Table 8.15: Improvement with respect to increase in number of ants for different DEP rates using UAO. Statistically significant improvements are in italics.

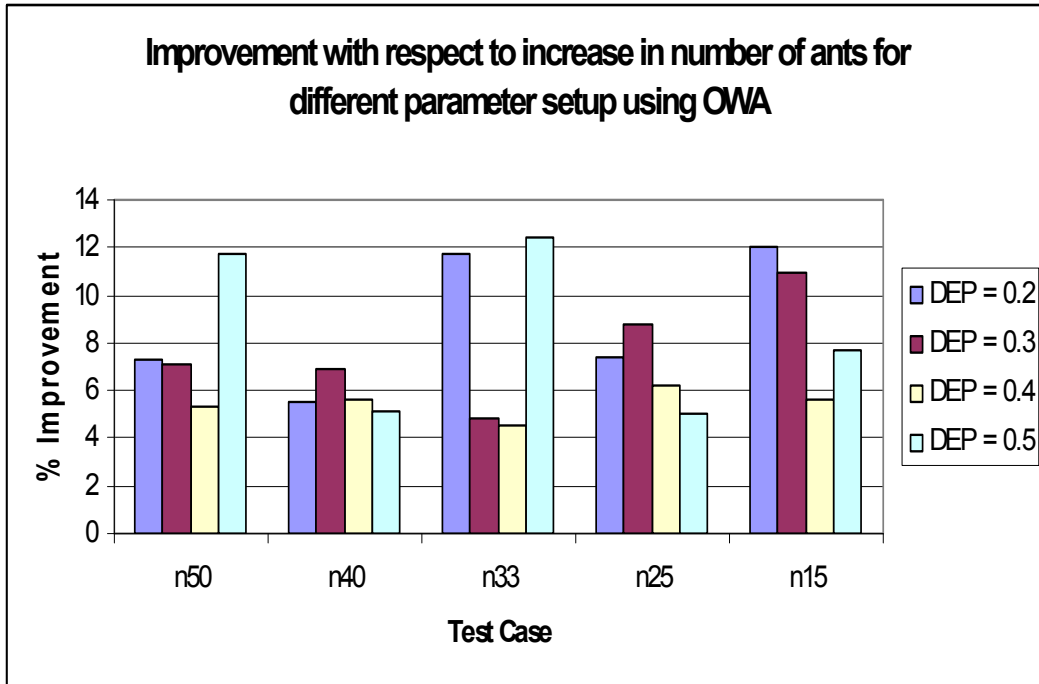
Case	% Improvement			
	DEP = 0.2	DEP = 0.3	DEP = 0.4	DEP = 0.5
n15	<i>1.73</i>	0.07	0.07	0.20
n25	<i>1.82</i>	<i>2.22</i>	<i>1.58</i>	0.26
n33	<i>2.09</i>	<i>1.39</i>	0.83	<i>1.09</i>
n40	<i>0.50</i>	<i>1.53</i>	<i>2.29</i>	0.91
n50	<i>2.62</i>	<i>1.26</i>	<i>0.69</i>	0.31

the solutions. For the case with no evaporation, the bad links continue generation by generation, thus preventing the algorithm from converging to a good quality solution for the same amount of time as that of the other deposit and evaporation setups.

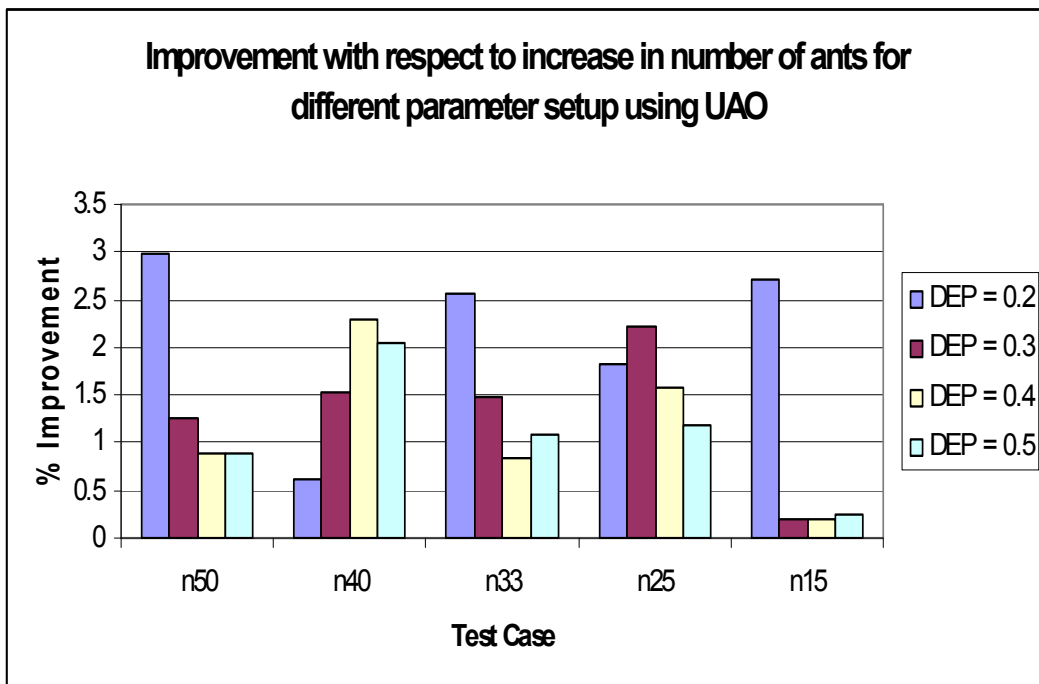
### 8.2.3 Comparison of OWA and UAO

Table 8.16 compares the OWA and UAO operators. This comparison is done using linear regression analysis. The data for the analysis consisted of the best results with respect to different numbers of ants. That is, the data consisted of columns where the best results appear (in boldface) in tables 8.4 to 8.13. Each of these columns





(a)



(b)

Figure 8.2: Percentage improvement with increase in number of ants for different parameter setup using (a) OWA (b) UAO

contains five values (one for each different number of ants), where each value is an average of 30 runs. Thus, each regression coefficient in Table 8.16 was obtained using 150 values (5 values  $\times$  30 runs). The regression coefficients of OWA and UAO are provided in columns 2 and 3 of Table 8.16 for each test case. Column 4 enlists the ratio of UAO and OWA. Based on this ratio, the sensitivity of OWA and UAO with respect to an increasing number of ants was assessed. A comment related to this is provided in column 5.

It is observed in Table 8.16 that UAO was more sensitive with respect to an increasing number of ants. For cases  $n50$ ,  $n25$ , and  $n15$ , both OWA and UAO increased the overall goodness almost at the same rate when the number of ants increased. However, UAO performed better than OWA for  $n33$  with a rate of 1.7. In other words, if the number of ants is increased, then UAO will improve the goodness 1.7 faster than OWA. For the case  $n40$ , UAO showed a rate of more than 6, which is significantly higher than the rate of OWA.

### 8.3 Conclusions

This chapter proposed and investigated a fuzzy multi-objective ant colony optimization algorithm. A fuzzy heuristic value was proposed. Furthermore, since pheromone deposit and evaporation, and the population of ants, are important parameters in ACO, empirical analysis was done to study the effect of these parameters. The analysis revealed that better results were obtained when the difference in pheromone deposit and evaporation rates was high. This was a general trend for both OWA and UAO. As for the number of ants, better results were obtained when the number of

Table 8.16: Comparison of OWA and UAO for ACO.

Case	Regression coefficients		Ratio = $\frac{UAO}{OWA}$	Comment
	OWA	UAO		
n50	0.343	0.314	0.92	UAO increases goodness at almost the rate as that of OWA with increase in number of ants
n40	0.072	0.456	6.33	UAO increases goodness 6.3 times faster than OWA with increase in number of ants
n33	0.242	0.418	1.73	UAO increases goodness 1.7 times faster than OWA with increase in number of ants
n25	0.305	0.327	1.07	UAO increases goodness at almost the rate as that of OWA with increase in number of ants
n15	0.158	0.163	1.03	UAO increases goodness at almost the rate as that of OWA with increase in number of ants

ants was high. Moreover, mutual comparison of OWA and UAO suggested that UAO produced better results in the context of its usage in the ant colony optimization algorithm.

The next chapter investigates another swarm intelligence technique, the particle swarm optimization algorithm, with its application to the DLAN topology design problem.