Hybrid Evolutionary Approaches to Terminal Assignment in Communications Networks

X. Yao[1], F. Wang[2], K. Padmanabhan[1] and S. Salcedo-Sanz[1]

[1] The Centre of Excellence for Research in Computational Intelligence and Applications (CERCIA), School of Computer Science, The University of Birmingham Edgbaston, Birmingham B15 2TT, UK. Email: x.yao@cs.bham.ac.uk

[2] Intelligent Systems Lab, BTExact Orion 1/12, Adastral Park, Ipswich, IP5 3RE, UK. Email: fang.wang@bt.com

Summary

Terminal assignment is an NP-hard problem in communications networks. It involves assigning a set of terminals to a set of concentrators with a cost for each assignment. The objective is to minimize the total cost of the assignment and the number of concentrators used. A number of heuristic algorithms, including genetic algorithms, have been proposed for solving this problem. This chapter studies several evolutionary and hybrid approaches to terminal assignment. Firstly, a novel chromosome representation scheme based on concentrators is proposed. This representation compares favourably against the existing terminal-based representation, which scales poorly for large problems. Extensive experiments have been carried out. The results show that our evolutionary algorithms using the concentrator-based representation outperform significantly existing genetic algorithms using the terminal-based representation. Secondly, a number of new search operators used in our algorithms are also investigated empirically in order to evaluate their effectiveness for the terminal assignment problem. Finally, different combinations of evolutionary algorithms and local search are studied in this chapter. Both Lamarckian evolution and Baldwin effect have been examined in combining an evolutionary algorithm and local search. Our results show that hybrid algorithms perform better than either evolutionary algorithms or local search. However, there is no significant difference between Lamarckian-evolution-style combination and Baldwin-effect-style combination.

1 Introduction

Evolutionary algorithms (EAs) and their hybridisation with local search have been widely studied and applied to solve many real world problems. Communications network design is a typical combinatorial optimization problem for which no efficient algorithm exists unless P=NP. A good design of communications networks requires certain constraints to be met and at the same time, one or more objectives to be optimized. The algorithms for designing communications networks must have good scalability and be able to deal with large-scale applications with a large number of network nodes.

The optimal design of communications networks considering both cost and capacity has been investigated in the literature using different heuristic algorithms, such as tabu search, simulated annealing and greedy search. Recently, EAs have been shown to perform well in communications network design, especially for the terminal assignment problem, which has been shown to be NP hard [11]. However, the performance of such EAs is still unsatisfactory for large problems.

This chapter studies novel hybrid EAs. Unlike previous EAs which used a terminal-based chromosome representation, a concentrator-based evolutionary approach for solving the terminal assignment problem is proposed in this chapter. This evolutionary approach uses a novel concentrator-based representation and associated search operators. It is hybridised with local search methods to form hybrid EAs. The concentrator-based representation is proposed to overcome the difficulties encountered by the terminal-based representation previously used by other evolutionary approaches. Attempts are also made to design appropriate search operators that work well with the concentrator-based representation. In addition to minimising costs, we also consider reducing the number of concentrators used. The objective of minimising the total cost is explicitly dealt with by the fitness function during the evolution. Minimising the total number of concentrators used is considered as an implicit constraint for the cost objective, or a second objective encoded in the fitness function. Hence, there are two problem formulations for the terminal assignment problem, i.e., single-objective and multi-objective optimisation. In this chapter, both formulations will be studied using the hybrid EAs and the concentrator-based representation.

In our hybrid EAs, two methods are considered for hybridization with local search, i.e., Lamarckian evolution and Baldwin effect. Lamarckian evolution forces the genotype to reflect the result of local improvement. The improved individual is placed back into the population and allowed to compete for reproductive opportunities [10]. The Baldwin Effect allows an individual's fitness (phenotype) to be determined after local search. Similar to natural evolution (Darwinian evolution), the result of the improvement is not reflected in the genetic structure (genotype) of the individual. It only changes the individual's chance of survival [10]. Baldwin effect as used in EAs may introduce undesirable offspring after crossover. When crossing two individuals, which after

local search converge to the same local basin, it is likely that the offspring may be similar to the parents and will converge to the same basin. To avoid this problem, the use of memory is considered for both Lamarckian evolution and Baldwin effect in the work presented here.

The concentrator-based evolutionary approach and its hybrid evolution are fully tested and examined by a series of computational experiments designed for the terminal assignment problem. The results have shown that the EA's performance was better with concentrator-based representation than with the terminal-based representation. The generation of a feasible initial population is simpler and more scalable in the concentrator-based representation even for a large number of terminals. The concentrator-based hybrid EAs outperformed EAs without local search. However, there is no significant difference between two different approaches to hybridise EAs with local search, i.e., Lamarckian evolution and Baldwin effect.

The remainder of this chapter is organised as follows. The next section introduces the terminal assignment problem and the previous work in solving this problem. Section 3 presents our concentrator-based representation and the search operators designed for it. A set of experiments are carried out to test the performance of the representation and operators and to compare them with the traditional terminal-based EAs. Concentrator-based hybrid EAs that integrate Lamarckian evolution or Baldwin effect are studied in Section 4. Lamarckian-style and Baldwin-style evolution with and without memory are investigated using the terminal assignment problems with single-objective or multi-objectives. Section 5 concludes this chapter with a brief summary of our work and some future work.

2 The Terminal Assignment Problem

2.1 Problem Representation

In this chapter, we will focus on the two-terminal network (also called source-link network) design. The work, however, can also contribute to the design of other kinds of networks, i.e., all-terminal networks. In the two-terminal network, a set of pre-specified source nodes communicate with the pre-specified sink nodes through non-specified paths. This can be simplified as a terminal assignment problem that concerns the assignment of certain terminals to some concentrators. This assignment should keep the total cost minimum. The cost may include material cost of cabling, installation cost and connection or communication cost between the concentrators and terminals. The cost may be fixed or varied per connection depending on the real situation. In general, it can be summarized as a weight that is used as the complete cost for each connection [2], [3], and [14].

In addition to minimizing the total cost, the terminal assignment problem should take the concentrators' capacity limit into account by satisfying two constraints:

1. Every terminal is assigned to one and only one concentrator.
2. The sum of weights of connections between terminals and a concentrator should not exceed the capacity of that concentrator.

Single-objective Optimisation Formulation

Given

K: number of concentrators,
T: number of terminals,
C_i: Capacity of concentrator $i = 1, 2, \ldots, K$,
d_ij: weight of the connection between concentrator i and terminal j, where $i = 1, 2, \ldots, K$, $j = 1, 2, \ldots T$,

the single objective optimisation problem of terminal assignment is to minimise the total cost,

$$\sum_{i=1}^{K} \sum_{j \in J} d_{ij}$$

subject to

$$\sum_{j \in J_i} d_{ij} \leq C_i,$$

where $j \in J_i$ is the terminal j assigned to concentrator i, and is the set of all terminals connected to concentrator i.

Multi-objective Optimisation Formulation

The most common objective of the terminal assignment problem is to minimise the total cost of the network. However, in many situations, it makes sense to also minimise the number of concentrators used so that the whole network can work with a less cost. The minimisation of the number of concentrators can be treated as an implicit constraint to be considered in the above single objective optimisation, or as another objective to optimise. In the latter case, the problem becomes a true multi-objective optimisation problem that minimises both the cost and the number of concentrators used at the same time. A weighted sum approach for this problem is described below.

Given

f_i: objective i, $i = 1, 2, \ldots, n$, and

w_i: weight of objective,

then the purpose of the multi-objective optimisation is to minimise

$$G = \sum_{i=1}^{n} w_i f_i.$$

In the problem presented here, n is 2 (for two objectives), f_1 is the total cost of all the connections between concentrators and terminals, which is F as described in the single objective optimisation; and f_2 is the total number of concentrators used.

2.2 Previous Work on Terminal Assignment in Communications Networks

Various approaches have been applied to the optimisation of communications networks. Previous work in [3] utilised simulated annealing to find the optimal design of small-scale networks (less than five nodes). Simulated annealing was also adopted in [15] to find solutions for packet switched networks with considerations of delay and capacity. Tabu search was used in [7] and in [13] to find an appropriate design of communications networks by considering cost and capacity together.

Using greedy algorithms and genetic algorithms (GAs) to assign terminal nodes to concentrators was investigated by [1]. The greedy algorithm assigns terminals to nearby (but maybe not the nearest) concentrators, if this assignment can help other terminals to be assigned to nearby concentrators. This kind of assignment can lead to infeasible solutions even if a feasible solution exists. This means that sometimes there are unassigned terminals that cannot be allocated to any concentrator.

The GA used in [1] had two possible chromosome representations for the terminal assignment problem, LC1 and LC2. Both representations are composed of an integer string. Each integer indicates the concentrator to which a terminal is assigned. The integers are arranged in the sequence of terminals, so the length of the string is the same as the number of terminals. In LC1, the first n1 terminals are assigned to n2 different concentrators, one terminal per concentrator. The remaining terminals are assigned in a greedy fashion considering the different costs of the concentrators. [1], used a seeding strategy to initialise the population in order to reduce the number of infeasible individuals in the initial population. Unfortunately, this kind of representation sometimes may cause inappropriate assignments with a great waste of concentrator capacities after the first n1 terminals are allocated to n2 concentrators. In case of large-scale problems with large numbers of terminals and concentrators, the computation time may increase considerably due to the continuous evaluation of the lowest costs for the assignment of the remaining terminals.

The second representation LC2 do not adopt the strategy of assigning the first n1 terminals to n2 concentrators. All the terminals are assigned in a greedy fashion. Therefore, unlike LC1, the infeasibility in the initial population of LC2 is likely to be high. In case of large-scale problems, the computation of the cost can be very high as well. The results in [1] showed that GAs outperformed the greedy algorithm.

[12] compared greedy algorithms, GAs and grouping GAs (GGAs) for solving the terminal assignment problem. A terminal is assigned to the nearest concentrator which has sufficient capacity to take this terminal, and if not, the next closest concentrator is chosen for assignment. In this algorithm, there are many chances that some terminals may not be allocated and hence make the solutions infeasible. If the number of terminals or concentrators is large, it may take a long time to search for concentrators with the least cost for assigning all terminals.

The GAs in [12] used both binary and non-binary representations indicating the concentrators with which the terminals are connected. If the terminal size is large, e.g., 1000, the chromosome length will also be large, e.g., 1000. Therefore, generating a feasible initial population and evolving such long chromosomes can be a challenge to GAs. [12] have incorporated a penalty term in the fitness function to deal with infeasibility. Infeasible solutions are not discarded but included in the population with the penalty incorporated in it. The penalty term clearly distinguishes infeasible solutions from feasible ones. A higher penalty imposes more selective pressure on infeasible solutions.

[12] used a GGA as a third approach to solve the terminal assignment problem. The representation in GGAs consists of two parts. The first part is the same as the representation used in GAs, but there is an additional part which groups the terminals and their connected concentrators together. The first part of the representation is only used for selection and fitness evaluation. A special crossover operator is designed for the group part in the group representation, which selects an entire group from one parent and inserts it into the other parent at the crossover point. After crossover there is a high possibility that the individuals may become infeasible. So the infeasible chromosomes have to be repaired. The repair process needs to remove the duplicate concentrators and re-assign the associated terminals to other concentrators. [12] demonstrated that GAs with the non-binary representation outperformed greedy algorithms in most cases, but GGAs did not perform very well comparatively.

It can be seen from previous work that using evolutionary approaches (especially GAs) in communications network design has potentials. These approaches showed better performance than other search algorithms such as greedy algorithms. However, there is a crucial limitation in the previous evolutionary approaches concerning their encoding methods, which is usually a list of all possible connections to concentrators, arranged in the sequence of terminals. Such encoding methods usually cannot work well with large-scale problems, and in particular, they have extreme difficulties in generating a feasible initial population within a reasonable time. Because those encoding schemes are all based on terminals and cannot reflect well the relationship between terminals and concentrators, good couplings between terminals and concentrators discovered in evolution may not be maintained after search operations such as crossover and mutation. This makes the evolution more difficult to find and keep optimal solutions. Though [12] introduced the concept

of group based representation, it was used together with the terminal based representation and only for crossover. The performance of GGAs was not satisfactory. In order to overcome the difficulties presented in the previous work, a new chromosome representation is proposed for the terminal assignment problem in the next section.

3 Concentrator-based Evolutionary Approach

In this chapter, a novel concentrator-based evolutionary approach is proposed to make use of the group structure in the terminal assignment problem. This approach is especially used to overcome the incapability of previous evolutionary approaches in handling large-scale networks. The concentrator-based evolutionary approach differs from classic EAs in two aspects. First, a special encoding scheme is designed to introduce the structure of groups into the genes of chromosomes. Second, given the distinctive encoding, special genetic operators are designed to evolve the concentrator-based chromosomes for solving the terminal assignment problem.

In the remaining of this section, we will introduce the encoding method and the corresponding search operators. The concentrator-based EA is then examined and compared with other EAs by a series of experiments with different experimental settings.

3.1 Concentrator-based Representation

The concentrator-based representation is composed of a set of trees in one level, in each of which the concentrator is the root node and the terminals associated with the concentrator are the leaves. Each tree therefore indicates a concentrator together with its terminals. An example of the representation is shown in Figure 1,

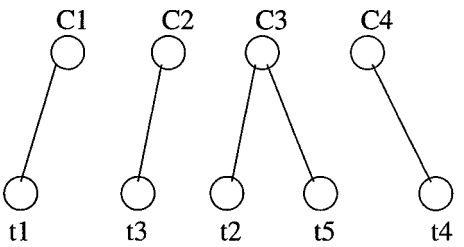

Fig. 1. An example of the concentrator-based representation.

In this example, there are 4 concentrators (c1 to c4) and 5 terminals (t1 to t5). Terminal t1 is assigned to concentrator c1, terminal t3 to concentrator c2, and so forth. The representation of this example can be written as:

$$c1(t1)c2(t3)c3(t2,t5)c4(t4)$$

When composing the concentrator-based representation, both constraints of the terminal assignment problem must be met. Any infeasible representations should be either repaired or eliminated in evolution.

The initial population of the concentrator-based EAs is generated in a way similar to that of the terminal-based EA (e.g., the non-binary representation used in [12]). Every concentrator has equal probability to serve terminals. A terminal is first assigned to a randomly selected concentrator. If the concentrator has not enough capacity to serve the terminal, then another concentrator is randomly chosen. An individual is included in the population only if it is feasible.

The concentrator-based representation, i.e. the tree-based representation, allows for variable length genotypes, so the chromosomes are not restrained by terminal or concentrator numbers. It is both efficient and flexible. Because there is no need to search and evaluate the least cost concentrators when generating individuals, the concentrator-based representation works well even with a large number of terminals or concentrators. The generation of the initial population is simpler than the terminal-based representation. The terminals that are to be assigned to a concentrator are taken from a pool where terminals are stored, eliminating any duplicates. By generating populations in this way the constraints of assigning a terminal to only one concentrator is implicitly satisfied.

3.2 Search operators

A series of search operators including selection, crossover, and mutation have been designed to work with the new concentrator-based representation, as neither the standard nor the ordering genetic operators are suitable for grouping problems [4]. These operators are introduced below.

Selection

Selection is the operation by which individuals are selected from a population for mating. There are many different models of selection such as ranking, roulette wheel selection and tournament selection. Because these models select chromosomes according to their ranks or fitness values, they can be easily applied to the concentrator-based evolution without major changes. In the following experiments, tournament selection is used due to its good performance in selecting optimum or nearly optimum solutions.

Crossover

The purpose of crossover is to pass on the genetic material from the current generation to the next one. A typical crossover recombines two individual parents to produce two offspring. Several crossover operators can be used on the concentrator-based representation.

One Point Crossover

This is one of the most common crossover methods used in EAs. A crossover point is randomly chosen and children are obtained by swapping the tails of the parents' chromosomes. Figure 2 is an example of how one point crossover works on the concentrator-based representation. If the crossover point divides the parent in equal halves then equal information is inherited. Sometimes repair has to be done to make the children feasible. The process of repair is explained later. In this type of crossover, the order of concentrators in a chromosome is not very important.

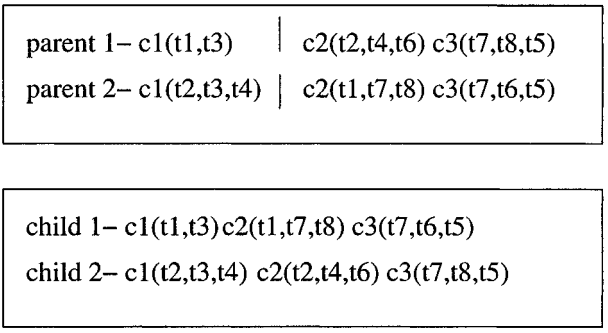

Fig. 2. One point crossover.

Two Point Crossover

In two point crossover, two crossover points are randomly chosen and the chromosome parts in between are exchanged between the parents, as shown in Figure 3. The information that is inherited depends on the crossover points. If the crossover points are far apart, more information is then inherited. Similar to one point crossover, this type of crossover is also commonly used in EAs.

Modified Uniform Crossover

A typical example of the uniform crossover is shown in Figure 4. The order of concentrators remains the same in all the chromosomes before and after crossover, but only some the terminals associated with each concentrator are inherited.

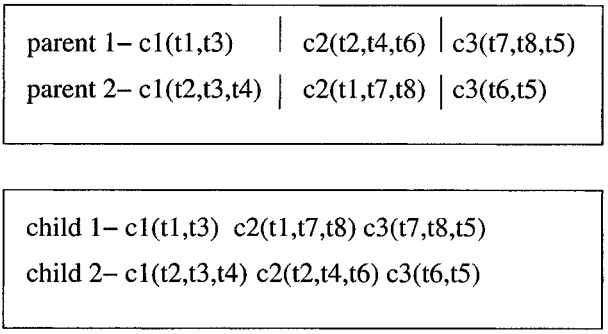

Fig. 3. Two point crossover.

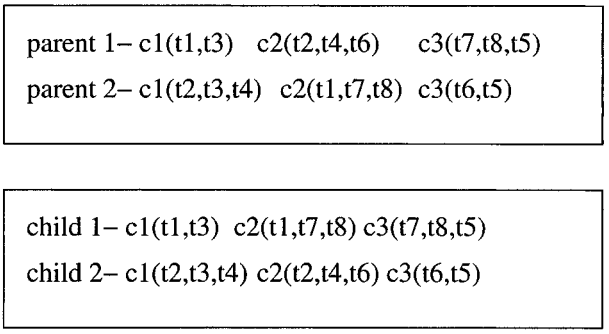

Fig. 4. Uniform crossover.

In Figure 4, for child1, the terminal set of c1 is inherited from parent1, for c2 it is inherited from parent2 and for c3 it is again inherited from parent1. In this example, the probability of inheriting a gene from a parent is set as 0.5. In the following experiments, the probability is calculated based on the available capacities of the concentrators. For example, if the available capacity of c1 is 60% in parent1 and is 30% in parent2, then the probability of selecting c1 from parent1 will be greater than from parent2. Such a uniform crossover is different from the classical one, and thus called modified uniform crossover. In this type of crossover it is possible that both terminal assignment constraints may be violated. The crossover may result in infeasible solutions. For example, terminals may be assigned to more than one concentrator, such as terminal t4 of child2 shown in Figure 4, which is assigned to both c1 and c2. Also, there may be some terminals that are not assigned to any concentrator, such as terminals t1, t7 and t8 in child2. In order to resolve the violation of constraints, repair should be done.

One node Crossover

The crossover operators introduced above are similar to classical crossover methods on the terminal-based representation. To exploit our representation better, two specific crossover methods based on concentrators are also designed. Figure 5 illustrates one of the methods, which is called one node crossover. Each concentrator is deemed as a node in this method. A random node point is chosen (such as c2 in the example shown in Figure 5) and the nodes together with their associated terminals in two parents are swapped to produce offspring. Repair is used to make the offspring feasible whenever necessary.

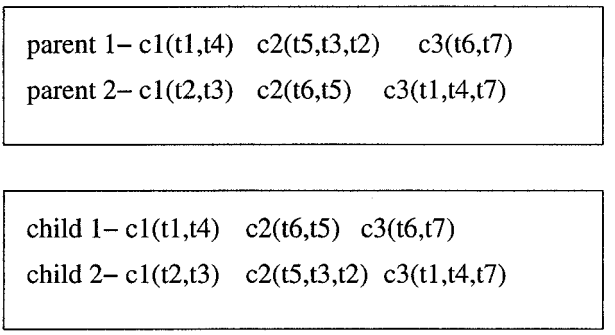

Fig. 5. One node crossover.

Best Node Crossover

In addition to one node crossover, another node-based crossover is proposed to exploit the best concentrator in the chromosomes. The best concentrator is chosen from each parent and then passed to the offspring. In Figure 6, concentrator c3 from parent1 and concentrator c1 from parent2 are transferred to both children. The offspring replace their parents only if the cost is less than or equivalent to that of the parents. This type of crossover can reduce the number of concentrators used.

One Group Crossover

This operator is inspired by the crossover used in GGAs as described in Section 2.2. Two random crossover points are generated separately and independently for two parents as shown in Figure 7. The crossover points may be different for two parents. Repair will be needed for infeasible children.

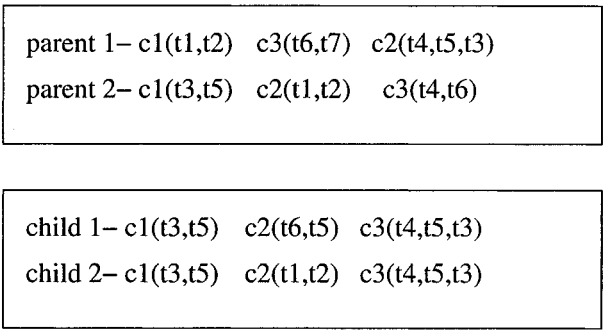

Fig. 6. Best node crossover.

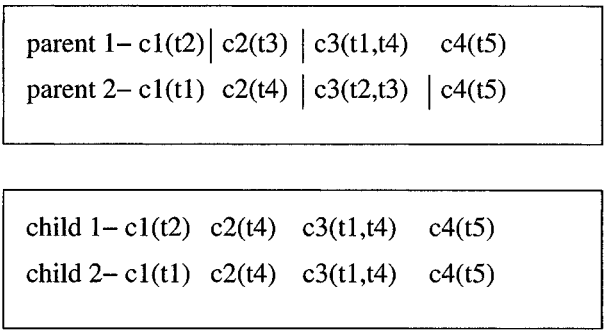

Fig. 7. One group crossover.

Best Group Crossover

This crossover is similar to the above one group crossover, but the best concentrator in the group will be retained in the offspring instead of being replaced and lost through crossover. Figure 8 illustrates this crossover. The concentrator c2 in parent1 is best utilized and hence is retained in child1. The concentrator c3 in parent1 is the best and hence is inserted into the chromosome of child2.

Repair

After crossover some terminals may be either presented in duplicates or completely missing and hence cause infeasible individuals. Stochastic repair is then used to make the individuals feasible. The repair process can be described by two steps:

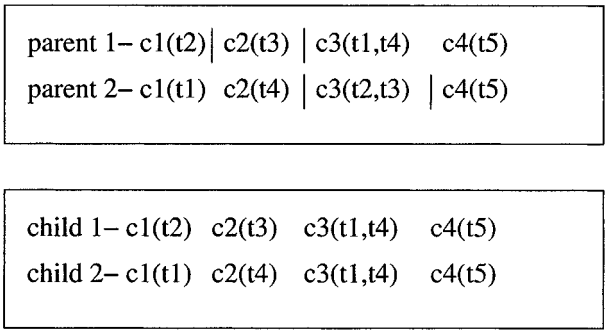

Fig. 8. Best group crossover.

1. Deletion of duplicate terminals - Each terminal is examined for duplicates and if there is any, a duplicate terminal in a less loaded concentrator is deleted;

2. Stochastic assignment of missing terminals - Missing terminals are assigned to less loaded concentrators which are randomly chosen.

Mutation

Mutation makes (usually small) alterations to one or more genes in a chromosome. It is considered as a method to recover lost genetic material during evolution. Here several mutation methods are used for the concentrator-based evolution.

Point Concentrator Swap

Two concentrators c1 and c2 are chosen stochastically and all the terminals associated are swapped between them. Because concentrators may have different capacities, swapping their terminals may reduce the cost but not the number of concentrators. This is shown in Figure 9.

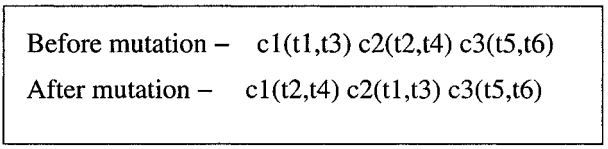

Fig. 9. Two point concentrator swap.

Two Point Terminal Swap

Here two concentrators are chosen at random and then two terminals are randomly chosen from the two concentrators, respectively. The selected terminals are then interchanged. An example of this mutation is shown in Figure 10.

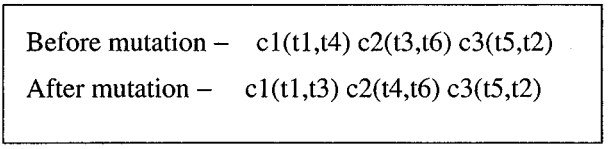

Fig. 10. Two point terminal swap.

In Figure 10, concentrators c1 and c2 are selected and in them terminals t3 and t4 are selected and then interchanged. The mutated individual is included in the population only if it is feasible and fitter than its parent.

Delete-Insert One Mutation

This mutation is designed to alter the concentrator of a terminal. A concentrator c1 is first chosen at random. Then a random terminal t1 is deleted from it and then inserted into the terminal set of another randomly chosen concentrator, c3 in this case. The mutated individual joins the population only if it is fitter than its parent and is feasible. This type of mutation is designed to reduce the number of concentrators used. For example, c1 in Figure 11 is no longer needed.

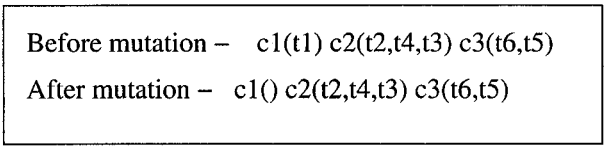

Fig. 11. Delete-insert one mutation.

In this mutation, a concentrator c1 is chosen at random and all the terminals in c1 are removed and inserted into another concentrator c2, which is also chosen at random, as shown in Figure 12. This type of mutation is designed to reduce the number of concentrators by shifting all of the terminals of a concentrator to other concentrators. In Figure 12, all the terminals of c1 are shifted to concentrator c2, however they may be reassigned to more than

one concentrator if the currently selected concentrator does not have sufficient capacity. After this mutation the individual joins the population only if it is fitter than its parent and is feasible.

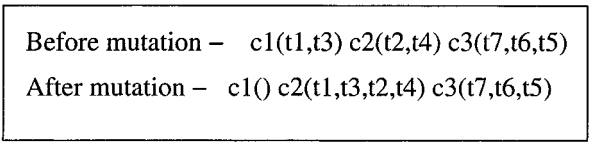

Fig. 12. Delete-insert all mutation.

Self Crossover Mutation

Two concentrators c1 and c2 are chosen at random. The terminal set of each concentrator is regarded as a small "individual" and the two sets are crossed using one point crossover.

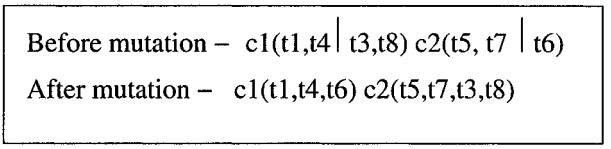

Fig. 13. Self crossover mutation.

One Group Mutation

A less loaded concentrator is chosen at random from a parent and all the terminals associated with the concentrator are deleted. The deleted terminals are then reassigned to other concentrators chosen randomly. In Figure 14 concentrator c2 is chosen at random and terminal t3 is deleted. The missing terminal t3 is added to the terminal list of concentrator c1. This type of mutation may result in fewer concentrators.

Multi-Group Mutation

This mutation is similar to the one group mutation except that more than one concentrator is involved. Several less loaded concentrators are chosen at random. In Figure 15 concentrators c2 and c4 are randomly chosen and terminals t4 and t2 are removed from their terminal lists. The removed terminals t4 and t2 are reassigned to a random concentrator c3. This type of mutation operator can also reduce the number of concentrators used.

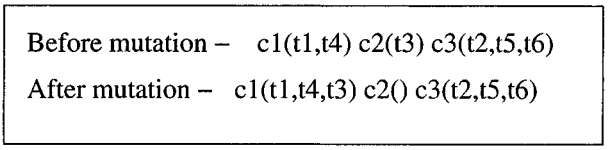

Fig. 14. One group mutation.

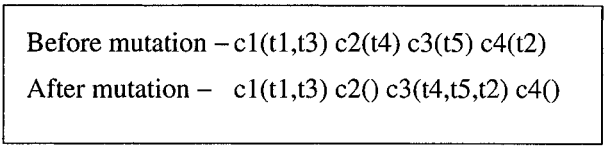

Fig. 15. Two group mutation.

3.3 Experimental Studies

In the previous sections, we introduced the concentrator-based representation and a number of search operators that can be used on the representation. In order to evaluate the proposed concentrator-based EA, a number of experiments were run with different experimental settings. In this section, both the concentrator-based representation and the corresponding operators will be tested for their performance.

Performance Test of the Concentrator-based Representation

The first experiment is used to examine the performance of the proposed concentrator-based representation. For the purpose of comparison, the terminal-based representation was also tested in the experiments. The initial populations in the experiments were generated in the way as described in Section 3.1. Tournament selection with uniform crossover and two-point interchange mutation was used for both representations in this comparison test. The EAs are terminated after the fitness value remains unchanged for 25 generations. Both concentrator-based and terminal-based representations are tested on different problems in which the number of terminals ranges from 100 to 1000. However, problems with more than 500 terminals were not considered in the terminal-based representation because the generation of their initial populations took too long. Table 1 gives a list of the experimental parameters used in the experiment. The experiment was run for 30 times and the results are shown in Table 2.

The results in Table 2 show that the concentrator-based representation generally found solutions better and much faster (with fewer generations) than the terminal-based representation, especially when the problem was large.

Table 1. Experimental setting

Population size:	100
Chromosome:	terminal based/concentrator based representation
Selection:	tournament selection
Crossover:	uniform crossover
Mutation:	two-point interchange
Termination Criterion:	the fitness value presents no change for 25 generations
Elitism:	yes
Number of runs:	30
Ratio of number of terminals to terminal number of concentrators:	2:1
Number of terminals:	100 to 500/100 to 1000
Number of concentrators:	50 to 250/50 to 500
Weight of terminals to: concentrators	1 to 6
Capacity of concentrators:	15 to 25

Table 2. Comparison between concentrator-based and terminal-based representations, where Size indicates the number of terminals, s.d. indicates standard deviations and N indicates the number of generations.

	Concentrator-based					Terminal-based				
Size	cost				N	cost				N
	best	mean	s.d.	worst		best	mean	s.d.	worst	
100	1254	1479	250	1720	250	932	1062	72	1212	435
200	1607	1806	116	2029	587	1565	1782	122	2051	1005
300	2018	2259	150	2568	1008	2038	2305	171	2756	1618
400	2375	2711	190	3152	1306	2676	3119	241	3843	2004
500	2962	3508	193	3843	1709	3194	3753	320	4665	2665
600	3274	3618	213	3959	2160					
700	3959	4399	246	4922	2306					
800	3920	4409	258	5029	2761					
900	4337	5179	342	6055	3224					
1000	4395	5295	378	5936	3624					

During the experiments, we found that the difference in cost between two representations was higher in the first generation as compared with the final generation. Though the concentrator-based representation produced a relatively uncompetitive population at the beginning, it obtained superior final results through evolution, except when the problem is very small, e.g., for terminal sizes 100 and 200. The concentrator-based representation achieved the results in fewer generations for all terminal sizes. The genetic operators worked more effectively on the concentrator-based representation than its counterpart.

The concentrator-based representation also showed good scalability. It easily generated and evolved populations for terminals up to 1000. This is in contrast to the terminal-based representation which became incapable of solving the problem when the number of terminals involved was more than 500. This incapability inevitably restricts the application of the terminal-based representation in real world communications networks, which usually involve a great number of network nodes. The number of generations required by the concentrator-based representation was approximately linearly increased with the increased number of terminals, as shown in Figure 16. The well-presented scalability of the concentrator-based representation shows that it is suitable for large-scale network applications.

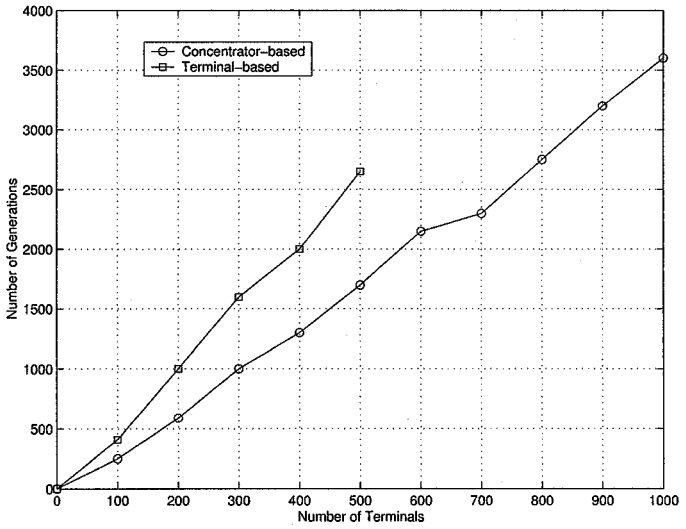

Fig. 16. Scalability of the concentrator-based representation. The generations required by the concentrator-based EA was linearly increased with the problem size. On the contrary, the terminal-based EA became incapable of solving the problem when terminals were more than 500.

Performance Tests of Search Operators

Various crossover and mutation operators designed for the concentrator-based representation were tested for their performance here. The EA guided by the operators should achieve the objective of minimizing the total cost between terminals and concentrators, and at the same time, the operators should keep the number of concentrators used at a minimum. In our experiments, the

number of terminals was set as 100 and the number of concentrators was 50. The other settings of the experiments were the same as those used in the previous tests (see Table 1). The experimental results over 30 runs are presented in Table 3.

Table 3. Performance tests of various search operators, where s.d. indicates standard deviation and N indicates the number of generations.

	Search operators	Number of conc.	Cost best	mean	worst	s.d.	N
1	One point crossover	20	4160	4567	4797	25622	50
2	Two point crossover	44	4380	4553	4728	8855	28
3	Uniform crossover	19	4498	4751	5052	23093	50
4	One node crossover	23	4145	4485	4779	23481	50
5	Best node crossover	43	4222	4503	4726	13887	50
6	One group crossover	44	4361	4563	4737	9863	25
7	Best group crossover	44	4251	4549	4720	16256	25
8	Delete-insert one mutation	39	681	702	744	259	296
9	Delete-insert all mutation	23	2877	3120	3408	13759	115
10	Two point concentrator swap	45	2479	2711	2984	19294	110
11	Two point terminal swap	50	1172	1420	1737	20588	254
12	Self crossover mutation	44	1357	1653	2171	31268	258
13	One group mutation	44	4380	4553	4728	8855	28
14	Multi-group mutation	44	4380	4553	4728	8855	28

¿From Table 3 we can see that, when the cost alone is considered, delete-insert one mutation was the best among all search operators. Two point terminal swap mutation, self crossover mutation, two point concentrator swap mutation and delete-insert all mutation followed delete-insert one mutation, but all of them required more generations. All the other operators performed similarly. When the number of concentrators used alone is considered, uniform crossover is the best, followed by one point crossover, one node crossover, delete-insert all mutation and delete-insert one mutation. The remaining operators had similar performance.

Generally speaking, crossover operators showed better performance in reducing the number of concentrators because most crossover operators are designed for this purpose. Moreover, the stochastic repair mechanism is also very effective in reducing the number of concentrators used. While the exchange of genes is more frequent (and hence more repair is required) in one point crossover and uniform crossover, only 40% of the total concentrators were finally utilized in the solutions found by these two crossover methods. Unlike one point crossover, two point crossover may not have frequent gene exchange and the repair mechanism only works on limited numbers of concentrators. The reduction of the concentrators used is therefore not so significant in two point crossover. Best node crossover is similar to one node crossover except

that the best utilized concentrator is chosen for crossover in both parents. Although repair may help reducing the concentrators used in both cases, it is unclear why the former is ineffective in the concentrator usage while the later is relatively more effective. This issue will be our future work. Compared with other crossover operators, the performance of group based crossover methods (e.g., one group crossover and best group crossover) was unsatisfactory. In most cases, group based crossover involves less exchange of genes than others. This may be the main reason why more concentrator used in group based crossover. Group based crossover also required much longer time to meet the termination criterion. All crossover operators showed insufficient effect on reducing the total cost.

In contrast to crossover, most mutation operators effectively reduced the total cost because the assignment of terminals to concentrators was continuously altered and only fitter individuals after mutation were allowed to join the population. Among these operators, delete-insert one and delete-insert all mutation performed best by maintaining better utilized concentrators and mutating less loaded concentrators. Two point concentrator swap, two point terminal swap and self crossover mutation were also good at reducing cost, but less effective than delete-insert one and delete-insert all mutation. This is because these operators do not take the concentrator load into account when swapping genes. Similar to group based crossover, one group mutation and multi-group mutation demonstrated unsatisfactory performance in both cost minimization and reduction of the number of concentrators used.

It is worth emphasizing that our study of genetic operators was carried out for each operator independently. We did not run EAs with two or more operators in the above experiments (Table 3). We expect EA's performance will improve further if we use two or more appropriate operators together.

4 Concentrator-based Hybrid Evolutionary Approaches

Hybrid EAs have been shown to be quite effective in solving a wide range of real world problems. How EAs and local search are combined is an extremely important issue that influences the final solution quality and the computational efficiency of the algorithm [10]. Hybridization of EA with local search gives rise to the concepts of Lamarckian evolution and Baldwin effect [10], which are the most often studied techniques in hybrid EAs.

In this section, both Lamarckian evolution and Baldwin effect are incorporated with the concentrator-based EA to form hybrid EAs for communications network design. Lamarckian evolution or Baldwin effect is applied to all the individuals in every generation. The two different forms of hybrid EAs are fully investigated on the terminal assignment problem, for both single-objective and multi-objective optimisation as introduced in Section 2.1. A series of experiments are designed to examine the performance of the hybrid EAs.

4.1 Lamarckian Evolution and Baldwin Effect

In Lamarckian evolution individuals improve during their lifetime through local search and the improvement is passed to the next generation. The individuals are selected based on improved fitness and are transferred to the next generation with the improvement incorporated in the genotype.

The Baldwin effect utilized in EAs was first investigated by Hinton and Nolan in [8]. Unlike Lamarckian evolution, the improvement does not change the genetic structure (genotype) of the individual that is transferred to the next generation. The individual is kept the same as before local search, but the selection is based on the improved fitness after local search. Baldwin effect follows natural evolution (Darwinian), i.e., learning improves the fitness and selection is based on fitness. The improvement is passed indirectly to the next generation through fitness in Baldwin effect.

While Lamarckian learning may disrupt the schema processing of a GA, Baldwin learning certainly aggravates the mapping problem of multiple genotypes to one phenotype. In a comparison of Baldwin and Lamarckian learning, [16] showed that utilizing either form of learning would be more effective than the classical GA without any local improvement procedure. They argued that, while Lamarckian learning is faster, it may be susceptible to premature convergence to a local optimum as compared to Baldwin learning [10].

4.2 Use of Memory

In Baldwin effect, if two individuals are different but map to the same local basin, the evolutionary approach will try to exploit both individuals. If these two individuals are crossed over and produce offspring in the same basin, computational effort will then be wasted on applying the local search to search the same basin again [10].

In Lamarckian evolution, these individuals are possibly identical and will reproduce clones of themselves if crossed over. The local improvement is therefore unnecessary as the children are the same as the parents. Slight mutation change may be useless since it may leave the individual in the same basin or in a later generation the EA may generate an individual that falls in a basin already explored. Therefore, the local improvement procedure may be reapplied to search the same basin while valuable computational cycles could be used to explore other regions in the search space. To solve this problem, random linkage, a search algorithm taken from global optimization, was designed [10] to prevent repeated searches by using an accept/reject function that determines whether a local search is appropriate.

In the work presented in this chapter, it is assumed that the offspring will converge to the same local basin after local search (though in practice it may not be the case), so these individuals are forbidden from crossing over with themselves. Consequently, the computational effort can be used to explore

other basins. Because this technique will check the fitness values of the offspring before crossover, it is similar to the use of memory. The memory is used with both Lamarckian evolution and Baldwin effect, and their performances are compared with those without memory in the following experiments. Tables 4 and 5 list the algorithmic descriptions of Lamarckian evolution and Baldwin effect with and without memory, respectively.

Table 4. Algorithmic descriptions of Lamarckian evolution with and without memory

Lamarckian evolution without memory

```
BEGIN
   Generate initial population P(0) randomly,
   i ← 0;
   REPEAT
      Select the parents from P(i) based on their fitness in P(i);
      Apply crossover to the parents and repair if necessary to make it feasible.
      Replace the parents only if the offspring is better;
      Apply mutation to the individuals and replace the population
            if the mutated individual is better and feasible;
      For each solution s0 from the population:
            REPEAT
               Perform local search to get a new solution s;
               If (f(s) < f(s0)) replace s0 by s;
            UNTIL terminal size
   UNTIL the population converges
END
```

Lamarckian evolution with memory

```
BEGIN
   Generate initial population P(0) randomly,
   i ← 0;
   REPEAT
      Select the parents from P(i) based on their fitness in P(i);
      Apply crossover to the parents only if their fitness are different and repair
            if necessary to make it feasible
            and replace the parents only if the offspring is better;
      For each solution s0 from the population;
        REPEAT
           Perform local search to get new a solution s
           If (f(s) < f(s0)) replace s0 by s;
        UNTIL terminal size
   UNTIL the population converges
END
```

Table 5. Algorithmic descriptions of Baldwin effect with and without memory

Baldwin effect without memory

```
BEGIN
   Generate initial population P(0) randomly,
   i ← 0;
   REPEAT
      Select the parents from P(i) based on their fitness(Baldwin) in P(i);
      Apply crossover to the parents and repair if necessary to make it feasible.
      Replace the parents only if the offspring is better;
      Apply mutation to the individuals and replace the population
            if the mutated individual is better and feasible;
      For each solution s0 from the population:
        REPEAT
           Perform local search to get a new solution s;
           Replace s0 by s;
        UNTIL terminal size
   UNTIL the population converges;
END
```

Baldwin effect with memory

```
BEGIN
   Generate initial population P(0) randomly,
   i ← 0;
   REPEAT
      Select the parents from P(i) based on their fitness(Baldwin) in P(i);
      Apply crossover to the parents only if their fitness are different and repair
            if necessary to make it feasible and
            replace the parents only if the offspring is better;
      For each solution s0 from the population;
        REPEAT
           Perform local search to get new a solution s
           Replace s0 by s;
        UNTIL terminal size
   UNTIL the population converges;
END
```

4.3 Experimental Studies

Concentrator-based hybrid EAs using Lamarckian evolution or Baldwin effect are evaluated and compared. Lamarckian evolution and Baldwin Effect are first combined with the various search operators introduced in Section 3.2 to solve the terminal assignment problem with a single-objective, then that with multi-objectives. The experimental setup is the same as the one used previously, as introduced by Table 1 in Section 3.3. Delete-insert one mutation

is used in the local search due to its effectiveness in reducing the cost as well as the number of concentrators.

Single Objective Optimization

Lamarckian evolution and Baldwin effect with and without memory are tested on the single-objective terminal assignment problem. Table 6 shows the total cost obtained by Lamarckian evolution and Baldwin Effect without memory and Table 7 shows the cost obtained with memory. All tests eventually used the same number of concentrators, which is 40, regardless of the use of memory. Because the use of memory influences only crossover, no mutation operators were used in the tests of Lamarckian and Baldwin learning with memory.

The experimental results listed in Tables 6 and 7 show that there is no significant difference in performance between Lamarckian and Baldwin evolution. When the local search is used without memory, the best results obtained are all around 680 for different combinations of search operators. The two-tail t-test on the mean cost also indicates that for a=0.5, none of the local search is significantly different from others. The hybrid EAs found the same basin for different combinations of search operators.

In the case of memory, the two tail t-test on the mean cost again shows no significant difference between Lamarckian and Baldwin evolution. However, the best cost obtained with memory can be lower than 680 when one point crossover, uniform crossover, one node crossover or best node crossover is used (the lowest is 461). This suggests that the use of memory aids the crossover operators to explore other basins and hence the computational effort can be saved from repeated exploration of the same basin. However, the standard deviation is quite high for those operators. It is worth noting that the performance of all the hybrid EAs outperformed the concentrator-based EA without local search as introduced in Section 3.

Multi-objective Optimisation

In real world communications networks, minimising cost and number of concentrators are both important and should be considered at the same time. It is therefore more sensible to deal with them as two independent objectives like in a multi-objective optimisation problem. To enable this, concentrator-based hybrid EAs with multiple objectives are studied.

In multi-objective optimisation, more than one objective should be optimised and these objectives are often in conflict with each other. Obtaining a global optimal solution for all the objectives is therefore not easy. Usually only a set of solutions that are non-dominated (known as Pareto optimal solutions) can be obtained. There are three main approaches to evolutionary multi-objective optimisation: the weighted sum approach, population-based non-Pareto approach and Pareto-based approach [6].

Table 6. Cost comparison between Lamarckian and Baldwin effect without memory in single-objective optimisation

Crossover	Mutation	Lamarckian			Baldwin			t-test
		best	mean	s.d.	best	mean	s.d.	
1	Del-ins one mut.	680	685	8	680	685	11	0.00
One point	Del-ins all mut.	680	685	8	680	685	15	0.00
	Two point conc.	680	685	18	680	684	9	0.27
	Two point term.	680	684	11	680	684	6	0.00
	Self crossover	680	684	10	680	684	5	0.00
2	Del-ins one mut.	680	686	18	680	684	16	0.45
Two point	Del-ins all mut.	680	685	17	680	683	8	0.58
	Two point conc.	680	684	9	680	684	12	0.00
	Two point term.	680	684	14	680	683	8	0.34
	Self crossover	680	684	8	680	683	10	0.42
3	Del-ins one mut.	680	686	18	680	684	7	0.56
Uniform	Del-ins all mut.	680	685	17	680	685	11	0.00
	Two point conc.	680	684	9	680	683	10	-0.40
	Two point term.	680	684	14	680	683	6	0.36
	Self crossover	680	684	8	680	685	11	-0.40
4	Del-ins one mut.	681	685	11	680	685	5	0.00
One node	Del-ins all mut.	680	684	7	680	685	7	-0.55
	Two point conc.	680	685	8	680	685	6	0.00
	Two point term.	680	685	7	680	685	10	0.00
	Self crossover	680	685	9	680	685	7	0.00
5	Del-ins one mut.	680	686	15	680	684	9	0.62
Best node	Del-ins all mut.	680	686	20	680	683	4	0.80
	Two point conc.	680	684	9	680	684	6	0.00
	Two point term.	680	685	16	680	683	4	0.66
	Self crossover	680	684	11	680	684	9	0.00
6	Del-ins one mut.	680	684	7	680	682	5	1.26
One group	Del-ins all mut.	680	684	10	680	684	8	0.00
	Two point conc.	680	684	11	680	684	8	0.00
	Two point term.	680	684	10	680	683	4	0.50
	Self crossover	680	683	15	680	683	4	1.04
7	Del-ins one mut.	680	686	19	680	683	6	0.81
Best group	Del-ins all mut.	680	685	12	680	683	4	0.86
	Two point conc.	680	685	15	680	683	6	0.67
	Two point term.	680	685	13	680	684	9	0.34
	Self crossover	680	686	22	680	684	8	0.46

When hybrid EAs are used for the multi-objective terminal assignment problem, the weighted sum approach is used. It is similar to the single-objective optimisation except that the fitness function explicitly deals with two objectives: one is to minimise the total cost and the other is to minimise the number of concentrators used. The mathematical formulation for

Table 7. Comparison between Lamarckian evolution and Baldwin effect with memory in single-objective optimisation

	Lamarckian			Baldwin			t-test
Type	best	mean	s.d.	best	mean	s.d.	
a) One point crossover	542	741	19021	461	715	15418	0.01
b) Two point crossover	680	684	11	461	683	6	0.43
c) Uniform crossover	583	705	4643	551	726	6577	-0.02
d) One node crossover	661	687	284	662	683	101	0.07
e) Best node crossover	668	684	35	665	685	38	-0.10
f) One group crossover	680	685	17	680	683	4	0.62
g) Best group crossover	680	686	17	680	684	7	0.59

this problem was shown in Section 2.1. However, there are some weaknesses in the weighted sum approach [9]:

1. It can provide only one Pareto solution from one run;
2. It has been shown that the weighted sum approach is unable to deal with a multi-objective optimisation problem with a concave Pareto front [5].

If the weights for different objectives are changing during optimisation, the optimiser may go through all points on the Pareto front. If the searched non-dominated solutions are archived, the whole Pareto front can be achieved. This has been shown to be working well for both convex and concave Pareto fronts. Whether the weighted sum approach is able to converge to a Pareto-optimal solution depends on the stability of the Pareto solution corresponding to the given weight combination. Without considering the time consumption, the whole Pareto front can be obtained by running the optimiser as long as possible [9].

Investigation of Varied Weights

To examine the weight effect on optimisation, varying weights between 0.1 and 0.9 are set for both objectives of the terminal assignment problem. In the experiments, EAs with one point crossover and delete-insert one mutation are used, and all the other experimental settings are the same as those used for the performance tests of various search operators, as introduced by Table 1 in Section 3.3. Table 8 summarises the experimental results, including the number of concentrators used, the total cost obtained and the number of generations required by each EA. Figure 17 shows the relationship between the values obtained for both objectives, i.e., cost vs the number of concentrators.

In Table 8, Weight One indicates the weight assigned to the first objective (cost) and Weight Two is that assigned to the number of concentrators. As Weight One increases, the cost decreases as expected. The decrease in cost became less obvious while weight one is higher than 0.4. The EA reaches a relatively reasonable performance for both objectives when their weights are

around 0.5. In the following experiments of hybrid EAs for multi-objective optimisation, we choose 0.5 as the weights for both objectives.

Table 8. Results obtained by varied weights in the weighted sum approach to multi-objective optimisation

	Weight one	Weight two	Number of concentrators	Cost best	mean	worst	s.d.	Generations
1	0.10	0.90	20	1623	2101	2528	50012	62
2	0.20	0.80	21	1029	1256	1466	10207	50
3	0.30	0.70	23	888	1008	1126	3538	52
4	0.40	0.60	24	795	846	918	1110	53
5	0.50	0.50	27	740	770	813	236	59
6	0.60	0.40	29	713	733	760	170	50
7	0.70	0.30	30	697	714	745	144	51
8	0.80	0.20	33	689	696	712	27	52
9	0.90	0.10	35	684	690	697	10	50

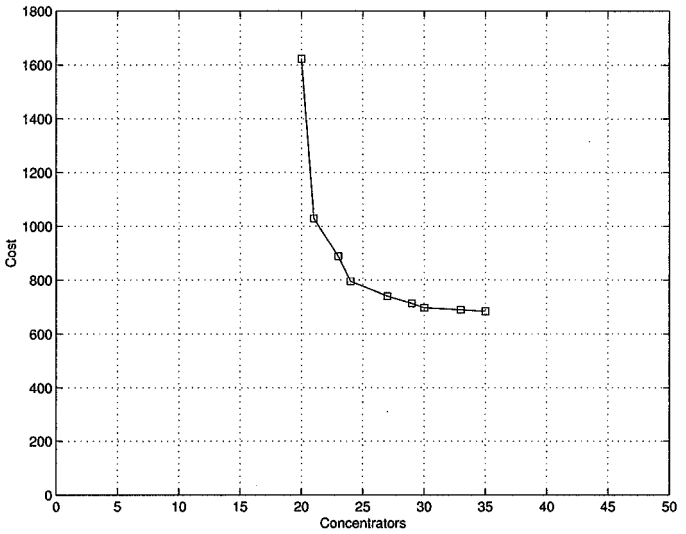

Fig. 17. Relationship between obtained cost and number of concentrators.

Comparison Between Lamarckian and Baldwin Effect in Multi-objective Optimization

In multi-objective optimization, both Lamarckian evolution and Baldwin effect are tested with various combinations of search operators. For simplicity, the use of memory is not considered in these tests. The weights for the two objectives are set as 0.5. Table 9 shows the comparison results.

In the Lamarckian-style hybrid EA, the best cost obtained was 719 when one point crossover was used with delete-insert all mutation and two point crossover was used with two point concentrator swap. The corresponding number of concentrators used was 26 in both cases. In Baldwin effect, the best cost obtain was 720 and the corresponding number of concentrators used was also 26, when two point crossover and two point concentrator swap mutation are used together. The experimental results again demonstrate that there is no significant difference between these two local search methods.

When comparing the results obtained for single-objective and multi-objective optimization, hybrid EA in single-objective optimization sometimes obtained a lower cost than in multi-objective optimization, such as the Lamarckian evolution with one point crossover plus two point terminal swap mutation, and Baldwin effect with uniform crossover plus self crossover mutation and with best node crossover plus delete-insert all mutation. The cost achieved for the single objective case is around 685, compared with the cost around 760 achieved for the multi-objective case. The number of concentrators used, however, is much lower in the multi-objective case, which is around 26, compared with 40 obtained in the single-objective case. If taking both objectives into consideration, the multi-objective optimization performed better in satisfying two objectives simultaneously than the single objective optimization. It is worth noting that in either case, hybrid EAs with local search outperformed EAs without local search as given in Section 3.

5 Conclusions and Future Work

Communications network design is essential to the development and implementation of widely used packet switch networks and fiber optical networks. Optimal communications network design is challenging since it needs to satisfy multiple constraints and to minimize one or more objectives at the same time. EAs have been shown to perform well for the terminal assignment problem. Their performance can be further enhanced by a new concentrator-based chromosome representation and by hybridization with local search.

This chapter proposes a novel concentrator-based representation that utilizes the group character of terminals and concentrators to overcome the limitations of the traditional terminal-based representation. A series of new search operators including crossover and mutation are designed for the concentrator-based representation. The concentrator-based EAs have been shown to outperform other terminal-based EAs. Our computational study also demonstrates

Table 9. Comparison between Lamarckian evolution and Baldwin effect in multi-objective optimisation

Xover	Mutation	Lamarckian				Baldwin				t-test
		conc.	best	mean	s.d.	conc.	best	mean	s.d.	
One point	Del-ins one mut.	27	740	770	236	27	732	765	418	0.06
	Del-ins all mut.	26	719	757	431	26	723	756	495	0.01
	Two point conc.	26	731	757	394	26	721	756	323	0.01
	Two point term.	26	740	767	211	26	737	764	424	0.04
	Self crossover	26	724	753	10	26	725	759	325	-0.08
Two point	Del-ins one mut.	26	745	780	592	26	734	770	543	0.07
	Del-ins all mut.	26	727	761	342	26	721	752	349	0.10
	Two point conc.	26	719	753	229	26	720	756	575	-0.03
	Two point term.	26	739	776	453	26	729	768	824	0.05
	Self crossover	26	728	753	242	26	727	753	788	0.00
Unif.	Del-ins one mut.	27	733	768	513	26	736	774	771	-0.04
	Del-ins all mut.	26	724	753	260	26	726	757	461	-0.04
	Two point conc.	26	733	760	312	26	729	751	246	0.12
	Two point term.	26	731	764	473	26	736	766	281	-0.02
	Self crossover	26	723	754	511	26	725	750	162	0.04
One node	Del-ins one mut.	27	734	767	617	26	728	761	374	0.08
	Del-ins. all mut.	26	723	754	517	26	723	750	232	0.04
	Two point conc.	26	730	756	429	26	725	747	281	0.10
	Two point term.	27	732	761	353	27	727	257	261	0.05
	Self crossover	26	724	750	262	27	725	758	555	-0.07
Best node	Del-ins one mut.	27	738	767	617	27	724	754	221	0.11
	Del-ins all mut.	26	724	763	399	26	726	751	113	0.16
	Two point conc.	26	725	757	499	26	722	750	233	0.07
	Two point term.	26	739	770	413	26	721	757	311	0.14
	Self crossover	26	723	749	259	26	731	751	229	-0.03
One group	Del-ins one mut.	26	740	772	384	26	741	767	281	0.06
	Del-ins all mut.	26	726	762	308	25	731	762	267	0.00
	Two point conc.	26	725	755	251	26	727	753	282	0.03
	Two point term.	26	733	771	494	26	735	761	221	0.10
	Self crossover	26	727	752	293	26	722	751	274	0.02
Best group	Del-ins one mut.	26	744	776	381	26	740	764	204	0.15
	Del-ins all mut.	25	721	768	616	26	730	755	198	0.11
	Two point conc.	26	729	754	436	26	732	755	264	-0.01
	Two point term.	26	766	685	236	27	720	328	766	0.00
	Self crossover	26	733	762	212	26	727	750	146	0.25

the good scalability of the concentrator-based EAs, which can still work well with the number of terminals up to 1000.

Hybrid EAs integrating Lamarckian evolution or Baldwin effect with or without memory have been designed to tackle both the single-objective and multi-objective formulations of the terminal assignment problem. Our experimental results reveal that Lamarckian evolution and Baldwin effect performed

similarly in most cases for the terminal assignment problem. However, the hybrid EAs obviously outperformed the EAs without local search.

It is worth noting that the proposed concentrator-based hybrid EAs are not limited to the terminal assignment problem. They can also be applied to other real world applications, such as bin packing and cutting stock problems. Further study of the concentrator-based hybrid EAs in these applications will be carried out. Although the work presented here includes a comprehensive investigation of the performance of various search operators for the concentrator-based representation, the most proper combination of these operators for the concentrator-based hybrid EA still needs further study. In particular, we are interested in analysing those group based crossover and mutation, which showed unsatisfactory performance in the experiments. Another work we want to investigate is the use of memory in hybrid EAs. Our experiments show that there is no significant difference between the EAs with and without memory. This is somewhat anti-intuitive and needs to be investigated further.

References

1. Abuali, F., Schoenefeld, D. and Wainwright, R. (1994). Terminal assignment in a Communications Network Using Genetic Algorithms. Proceedings of the 22nd Annual ACM Computer Science Conference, pages 74-81. ACM Press.
2. Aggarwal, K.K., Chopra, Y.C. and Bajwa, J.S. (1982). Reliability evaluation by network decomposition. IEEE Transactions on Reliability, R-31:355-358.
3. Atiqullah, M.M. and Rao, S.S. (1993). Reliability optimization of communication networks using simulated annealing. Microelectronics and Reliability, 33:1303-1319.
4. Falkenauer, E. (1996). A hybrid grouping genetic algorithm for bin packing. Journal of Heuristics, 2(1):5-30.
5. Fleming, P.J. (1985). Computer aided control systems using a multi-objective optimization approach. Proceedings of IEE Control'85 Conference, pages 174-179.
6. Fonseca, C.M. and Flemming, P.J. (2000). Multiobjective optimisation. In Back, T., Fogel, D.B. and Michalewicz, Z., editors, Evolutionary computation, volume 2, pages 25-27. Institute of Physics Publishing, Bristol.
7. Glover, F., Lee, M. and Ryan, J. (1991). Least-cost network topology design for a new service: and application of a tabu search. Annals of Operations Research, 33:351-362.
8. Hinton, G.E. and Nolan, S.J. (1987). How learning can guide evolution. Complex Systems, 1:495-502.
9. Jin, Y., Olhofer, M. and Sendhoff, B. (2001). Dynamic Weighted Aggregation for Evolutionary Multi-Objective Optimization: Why Does It Work and How? Proceedings of the Genetic and Evolutionary Computation Conference GECCO, pages 1042-1049. Morgan Kaufmann.
10. Joines, A.J. and Kay, M.G. (2002). Utilizing Hybrid Genetic Algorithms. In Sarker, R., Mohammadian, M and Yao, X., editors, Evolutionary Optimisation. Kluwer Academic Publishers. pp.199-228.
11. Kershenbhaum, A. (1993). Telecommunications Network Design Algorithms. McGraw-Hill.
12. Khuri, S. and Chiu, T. (1997). Heuristic Algorithms for the Terminal Assignment Problem. Proceedings of the 1997 ACM Symposium on Applied Computing, pages 247-251. ACM Press.
13. Koh, S.J. and Lee, C.Y. (1995). A tabu search for the survivable fiber optic communication network design. Computers and Industrial Engineering, 28:689-700.
14. Kumar, A., Pathak, R.M., Gupta, Y.P. and Parsaei, H.R. (1995). A genetic algorithm for distributed system topology design. Computers and Industrial Engineering, 28:659-670.
15. Pierre, S., Hyppolite, M.A., Bourjolly, J.M. and Dioume, O. (1995). Topological design of computer communication networks using simulated annealing. Engineering Applications of Artificial Intelligence, 8:61-69.
16. Whitley, D., Gordon, V.S. and Mathias, K. (1994). Lamarckian evolution, the Baldwin effect and function optimization. In Davidor, Y., Schwefel, H.-P. and Mnner R., editors, Lecture Notes in Computer Science, 866:6–15, Springer-Verlag.