Application of Genetic Algorithms in Access Network Planning

Alexander Tsenov[1]

Abstract – **In recent work a possible presentation of location-allocation problems in access network planning was introduced [1] – the set-based presentation. The new presentation was tested over many topological problems and has shown acceptable results as well.**

However, one important problem has arisen - the application of standard genetic operators, such a crossover and mutation has leaded to generation of many invalid individuals (unusable decisions of the topological problem).

Therefore, in this paper new genetic operators are introduced – set-based crossover, based on Random Transmitting Recombination, and set-based Mutation. Both are oriented to the set-based representation of location - allocation problems. The paper presents the functionality of these types of operators and some practical results as well.

Keywords – **Network design, Optimization, Genetic Algorithms, Crossover, Mutation**

I. Introduction

The problem of optimally designing a network in order to meet a given set of specifications (such as prescribed traffic requirements, achieving a desired level of reliability, respecting a given maximum transit time), while minimizing total cost, arises in a wide variety of contexts: computer networks, telecommunications networks, transportation networks, distribution systems.

Network design algorithms draw an increasing amount of attention nowadays. Considering the complexity, high cost factor and fast deployment times of today's communications systems (such as IP and ATM backbones, optical networks, numerous types of access structures etc.), network operators can benefit a lot from the use of network design tools. These tools can help speeding up and 'automating' the design process, ensuring superior quality (i.e. lower cost and/or better Quality of Service) and more justifiable solutions. Network design tools typically incorporate a wide range of functionality, such a geographical database handling, traffic estimation, link dimensioning, cost calculation, equipment configuration databases etc. The real benefit of using these tools, however, comes from the possibility of using the algorithmic network optimization approaches. In this way, there arises a possibility for finding solutions of better quality in much shorter time, as compared to the manual network design.

This paper introduces new genetic operators that are appropriate to new type of presentation of location - allocation problems for access network planning and applies them to some simple network optimization problems.

The first section describes a new set-based crossover operator based on Random Transmitting Recombination [2] and the second section – the set-based mutation operator, both used for application in genetic set-based representations in the context of a general location-allocation problem.

II. Set-Based Crossover

The set-based representation described in [1] is such that traditional crossover and mutation operators will not perform well. The representation is not orthogonal as for example

$$\xi_{a\,b} \cap \xi_{bc} \cap \xi_{\overline{ac}} = \emptyset$$

The non-orthogonality displays itself as dependencies between forma ("forma" is every simple of the representation): that means that a traditional operator would generate many invalid individuals. This is illustrated using an example:

Let consider a simple case where there are four customers to be connected. If the following two parents are selected for crossover witch represent the sets $\{\{a\},\{b,c,d\}\}$ and $\{\{a,b\},\{c,d\}\}$, Fig. 1 shows how an illegal child can be generated when uniform crossover is used. In fact, of the eight possible children under uniform crossover, five would be infeasible. Infeasible means that a customer is specified as sharing a set with a customer *and* not to sharing. E.g. a shares with b, b shares with c and a does not share with c.

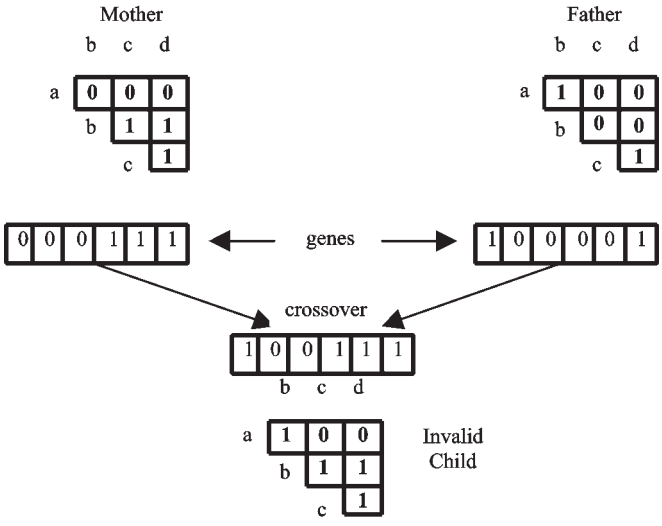


Fig. 1. Example uniform crossover, generating illegal children with set-based representation

[1]Alexander Tsenov is with Telecom Department at Technical University of Sofia, "Kliment Ohridsky" Blvd 8, 1756 Sofia, Bulgaria, E-mail: akz@vmei.acad.bg

As a consequence of this non-orthogonality and the fact the forma are separable (as assortment and respect are compatible) an operator based on Random Transmitting Recombination (RTR) was developed. The operator is a version of uniform crossover adapted for problems where the alleles are non orthogonal.

The operator functions in a number of stages:

1. Step 1: Gene values that are common to both parents are transmitted to the children;
2. Step 2: For each remaining uninitialized gene in the child: randomly select a parent and take the allele at the same location, set the child's gene to this allele. Update all dependant values in the array;
3. Repeat Step 2 until all values are specified.

Figs. 2-4 show a working example of the described operator. "1" indicates that two customers are sharing a set, (e.g. physically they are connected to the same node of the network) and "0" indicates not sharing.

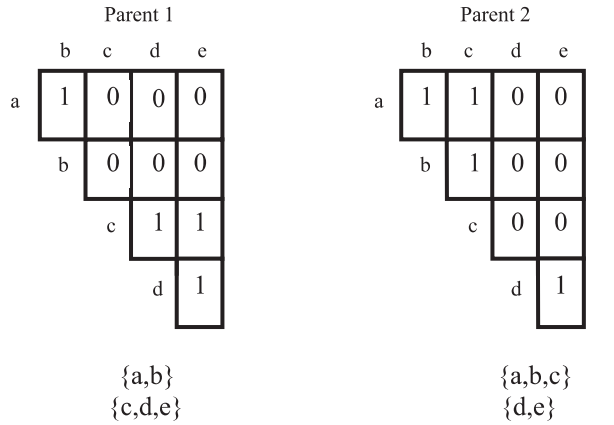


Fig. 2. Two example parents and the sets that they represent

So, in the first stage of crossover a child is built that contains all the allele values that are common to both parents. These values are shown on Figure 3, where for example the gene for customer a and b have an allele value of "1" in both parents. This applies to all common allele values.

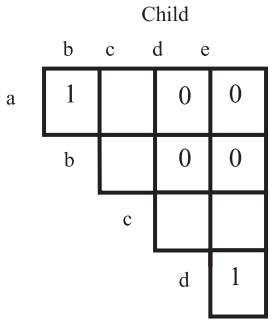


Fig. 3. The new child after the first stage of crossover

In the next stage of crossover a random uninitialized gene is selected and the corresponding value from a random parent is chosen to place there. After this value has been set the array must be updated to show any consequential changes. Figure 4 illustrates all the possible children that can be generated by this process given the two parents shown in Figure 2.

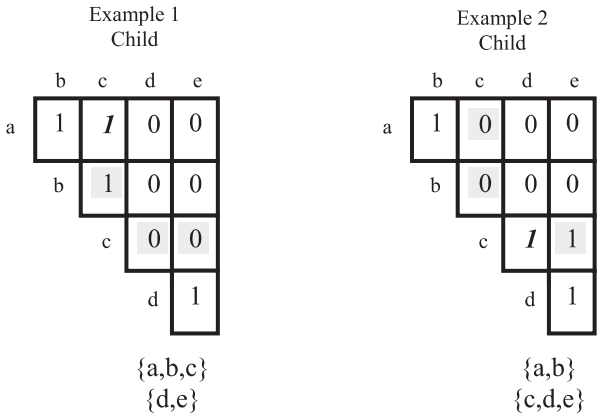
In example 1 in the figure for customers a and c is chosen and randomly assigned the value "1" from parent 2 (shown with italic in the Figure). As a consequence of this the following happens (shown with gray background in the Figure):

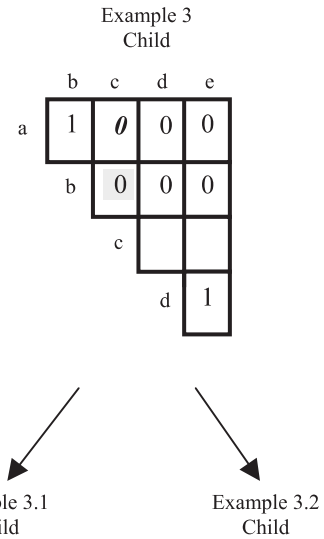
- because $(a, c) = 1$ and $(a, b) = 1$; that means a shares with c and a shares with b, therefore, b must share with c. So, $(b, c) = 1$.
- because $(b, c) = 1$ and $(b, d) = 0$; that means b shares with c and b doesn't share with d, therefore c cannot share with d. So $(c, d) = 0$.
- because $(b, c) = 1$ and $(b, e) = 0$; that means b shares with c and b doesn't share with e, therefore c cannot share with e. So $(c, e) = 0$.

In this case, there are no longer any uninitialized genes, therefore crossover has finished. Same reasonings may be made in example 2.

In examples 3 and 4 this is not the case and Step 2 is repeated to initialize all the remaining genes.

The two parents in the example are very similar, and there are not many customers to allocate to sets, therefore the children are similar, in fact there are children same as the parents.





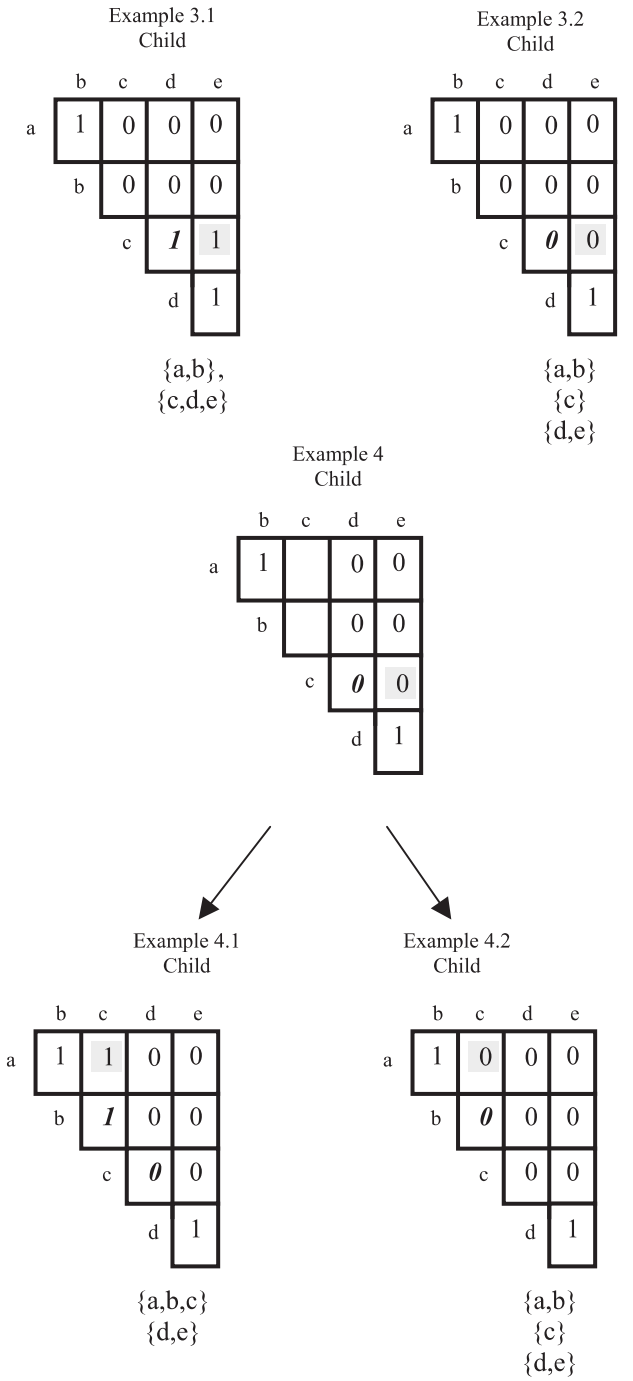


Fig. 4. All the possible children that can be generated for the parents shown in Figure 1. Those values shown in *italic* indicate the randomly chosen allele for a randomly chosen gene. Those values shown with gray background are consequential changes

Clearly, for larger examples much more variety is present in the children.

III. Set-Based Mutation

The aim of a mutation operator is to help the algorithm explore new areas of the search space by generating new genetic material. Given the representation described in [1] it is necessary to devise a compatible mutation operator. Typically genetic algorithms mutate an individual by randomly flipping the value of a gene in the individual; this corresponds to changing the value of a bit at a single point in the array. The nature of the representation is such that a change to the value of a single point in the array will lead to a cascade of changes throughout the array.

The effect of a change in the value of a bit can be understood in terms of the effect that it will have in the problem domain. If a value is changed from zero to one, this means that a customer that previously did not belong so a set of customers now does. That is, the customer changes its connection from one site to another – Fig. 5.

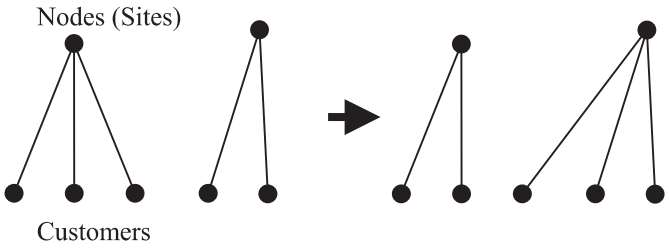


Fig. 5. The effect of mutation – changing an array value from “0” to “1” causes a customer to swap sites

Changing the value from one to zero means that s customer is removed from a set but not added to another – Fig. 6 The effect of this is to create a new set of customers with a single member. This means a new site must be initialized too.

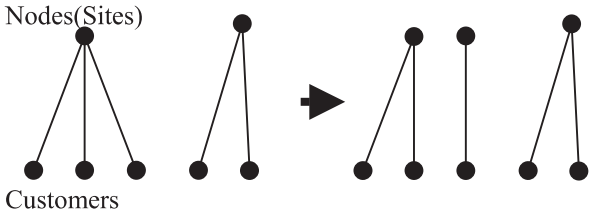


Fig. 6. The effect of mutation – changing an array value from “1” to “0” causes a customer to be removed from a set

The next section provides a worked example of the operation of the simple set-based mutation operator. The operator is implemented so that it has one of two effects: either the customer is moved from one set to another, or a new set is formed with a single customer.

Fig. 7 demonstrates the latter case where a gene is chosen randomly – in this case gene (a, b). This gene has the value “1”, which is mutated to “0”. This change leads no consequential changes, and customers a and b are split into two separate sets. If more then two customers where in the initial set, e.g. $\{a, b, c\}$, and the same gene had been chosen, then there would be two possible children: $\{\{a, b\}, \{c\}\}$ and $\{\{a\}, \{b, c\}\}$. The mutation operator chooses randomly between the two possible children.

Figure 8 demonstrates the mutation operator when the gene chosen has a value of “0” which is mutated to “1”. This is implemented so that one of the customers associated with the gene is moved to a new set. So, in the figure gene (a, c) is chosen and changing its value leads to two possible children

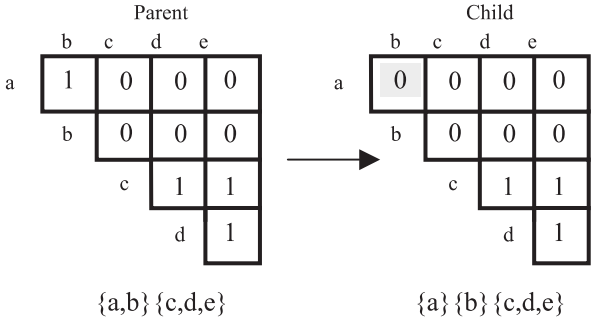


Fig. 7. The effect of mutation – changing an array value from "1" to "0" causes a customer to be removed from a set and new set is build.

one where customer a is moved to another set and the other when customer *c* is moved to another set. The mutation operator chooses randomly between the two possible children. It can be seen from the diagram that there are many consequential changes (shown in *italic*) produced by this change.

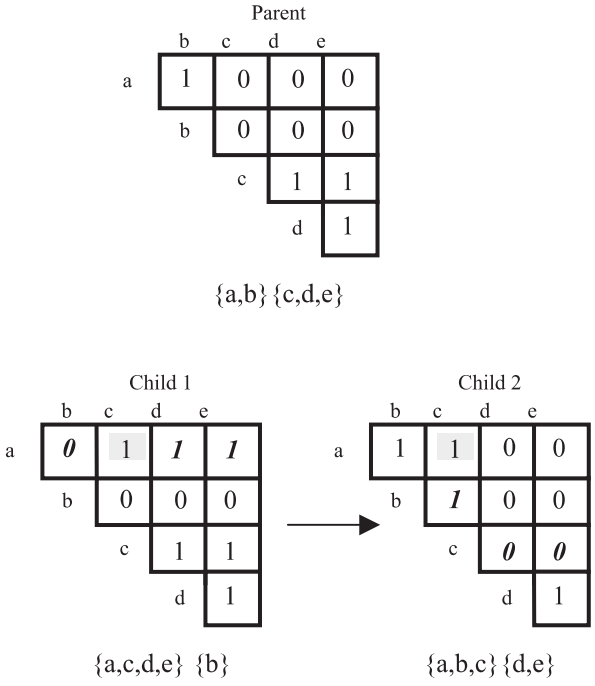


Fig. 8. The effect of mutation – changing an array value from "0" to "1" causes consequential changes (shown with *italic*)

IV. Conclusion

In this work two new genetic operators where introduced – the set-based crossover and the set-based mutation. Both are oriented to a new type of presentation of location-allocation problems in access network planing. It was proved that the implementation of these operators overcomes the disadvantages of implementing the standard crossover and mutation operators by using the set-based representation. This work is a part of the dissertation work of the author, where the implementation of both new operators was developed further and has led to good results.

References

[1] Tsenov A., "Presentation of location problems", *ICEST 2002*, pp. 299-302, Nish, Yugoslavia, 2-4 October 2002

[2] Radcliffe N.J, The Algebra of Genetic Algorithms, *Annals of Mathematics and Artificial Intelligence*, 10, pp. 339-384, 1994

[3] Routen T., Genetic Algorithms and Neural Network Approaches to Local Access Network Design, *Proceedings of the 2nd International Workshop on Modeling, Analysis and Simulation of Computer and Telecommunications Systems*, 1994

[4] Kratica J., V. Filipovitch, V. Ljeljum, D. Toljijg, Solving of the Uncapacitated Warehouse Location Problem Using a Simple Genetic Algorithm, *Proceedings of the XIV International Conference on Material handling and warehousing*, 3.33-3.37, Belgrade, 1996

[5] Kratica J., Improvement of Simple Genetic Algorithm for Solving the Uncapacitated Warehouse Location Problem, *Proceedings of the 3rd On-line World Conference on Soft Computing (WSC3)*, 1998

[6] P. Chardaire, A. Kapsalis, J.W. Mann, V.J. Rayward-Smith and G.D. Smith, Applications of Genetic Algorithms in Telecommunications. In J. Alspector, R. Goodman, T.X. Brown (Eds.), *Proceedings of the 2nd International Workshop on Applications of Neural Networks to Telecommunications*, pp. 290-299, 1995

[7] N.J. Radcliffe The Algebra of Genetic Algorithms, *Annals of Mathematics and Artificial Intelligence*, 10, pp. 339-384, 1994.

[8] Hoang, H.H.: A Computational Approach to the selection of an Optimal Network, *Management Science*, vol. 19, pp. 488-498,1973