A Method for Chromosome Handling of r-Permutations of n-Element Set in Genetic Algorithms

Göktürk Üçoluk

I. Introduction

COMBINATORIAL optimization problems are in the domain of Genetic Algorithms (GA) interest. Unfortunately ordinary crossover and mutation operators cause problems for chromosome representations of permutations and some types of combinations. This is so because offsprings generated by means of the ordinary operators are of a great possibility no more valid chromosomes. A variety of methods and new operators that handle that sort of obscenities are introduced throughout the literature.

Solutions are observed to fall into one of the following three categories:

• *Disqualification:* The idea is to allow the generation of those invalid chromosomes but assign such a low fitness values that they got eliminated in the forthcoming selection process. This simple method has its disadvantage of being extensively time consuming. The genetic engine spends most of its time generating invalid chromosomes and then eliminating them.

• *Repairing:* In this approach invalid chromosomes are generated but then fed into a intermediate process where they are transformed into valid ones. Here the key idea is to do the least modification such that the merits of crossover is preserved.

• *Inventing Specialized Operators:* Instead of creating invalid chromosomes the GA operators are modified to generate only valid chromosomes.

Falling into the third category and concerning permutation-respecting crossover operators, the following operators are worth to mention:

- Partially mapped crossover (PMX). [1]
- Order crossover (OX). [2]
- Edge recombination crossover (ERX). [3]

All of the above mentioned works are solutions to the genetic handling of n-permutations of n objects problem. The proposed technique is an alternative to these and is a solution to a more general problem. In the following section a technique for GA to deal with r-permutations of n-element sets, where the representation is crossover and mutation robust, is introduced. This means that the off-springs generated by crossover and mutation are still valid chromosomes and no special definitions for these operators are needed: the conventional bit-string crossover and mutation operators suffices.

G. Üçoluk is with the Department of Computer Engineering, Middle East Technical University, 06531 Ankara, Turkey. E-mail:ucoluk@ceng.metu.edu.tr

II. Proposed Method

In the proposed method the problem of representing r-permutations of n-elements in a chromosome is viewed as a problem with two stages. For each of these stages a different genetic encoding is proposed. So, a chromosome that will correspond to a specific r-combination (without repetition) of some n-elements will have two components:

1. The encoding of which r-combination of the n-element set it is.
2. The encoding of which permutation of that specific r-combination set it is.

With these encodings the conventional crossover and mutation operators will cause no problem and will generate valid offsprings.

So for example consider the case where we are going to represent the below given 7-permutation of the 12-element set $\{1,2,3,4,5,6,7,8,9,10,11,12\}$:

$$10,2,3,8,1,11,9$$

Our approach will be to

1. encode the 7-combination $(1,2,3,8,9,10,11)$,
2. then encode the way the places of that 7-combination has to be permuted to obtain the given $10,2,3,8,1,11,9$. For this example this is the permutation $(6,2,3,4,1,7,5)$.

A. The Encoding of "Which r-combination"

The proposed method is to enumerate all the r-combinations in a lexicographical order and to use the enumeration number for the GA coding. Let us assume that the n-element set is represented by the set

$$\{m \mid 1 \le m \le n\}.$$

An r-combination $\boldsymbol{\alpha}$ is of the form $(\alpha_1, \ldots, \alpha_r)$ where

$$1 \le \alpha_i \le n \quad \wedge \quad \alpha_i < \alpha_{i+1} \qquad \text{for } i = 1, \ldots, r$$

We define $\mathcal{S}_n : N^r \mapsto N$ to be a bijective function that provides the lexicographical order number of any r-combination of a n-element set defined above. $\mathcal{S}_n$ can be expressed [5] in mathematical terms as:

$$\mathcal{S}_n(\boldsymbol{\alpha}) = \binom{n}{r} - \sum_{i=1}^{r} \binom{n-\alpha_i}{r-i+1}$$

The inverse of $\mathcal{S}_n$ exists and is defined through:

$$\begin{aligned} \alpha_i &= \left[\mathcal{S}_n^{-1}(N)\right]_i \\ &= n - \max\{x \mid \binom{x}{r-i+1} \le \Omega(n,r,N,i)\} \end{aligned}$$

where $\Omega(n,r,N,i)$ is defined as:

$$\Omega(n,r,N,i) \triangleq \binom{n}{r} - N - \sum_{j=1}^{i-1} \binom{n-\alpha_j}{r-j+1}$$

Though this definition of $\mathcal{S}_n^{-1}(N)$ seems somewhat complex, it is not so actually: The elements of the permutation, namely the α_i s are obtained in the $i = 1, 2, \ldots, r$ order. Here in each step the previously obtained α_i values are made use of. The point in the calculation where one has to find the maximal integer x that satisfy

$$\binom{x}{r-i+1} \leq \text{(Some calculated value)}$$

is from the computational point of view not problematic. Since x and $r-i-1$ are bound to be in the ranges $[1,n]$ and $[1,r]$ respectively, constructing a precomputed table

$$T_{m,k} \triangleq \binom{m}{k} \quad \text{for} \quad m = 1, 2, \ldots, n \quad k = 1, 2, \ldots, r$$

at the initialization phase (of the GA engine) and performing a binary search in there is sufficient to locate the x value.

B. The Encoding of "Which permutation"

The proposed method for this phase is to describe a permutation by means of its inversions [4]. For a permutation $i_1, i_2, \ldots, i_r$ of the set $\{1, 2, \ldots, r\}$ we let a_j denote the number of integers in the permutation which precede j but are greater than j. So, a_j is a measure of how much out of order j is. The sequence of numbers $a_1, a_2, \ldots, a_r$ is called the *inversion sequence* of the permutation $i_1, i_2, \ldots, i_r$. For example the inversion sequence of the permutation $4, 8, 6, 2, 5, 1, 3, 7$ is $5, 3, 4, 0, 2, 1, 1, 0$. Here, for example, the 4 (which is the 3rd element in the inversion sequence) is saying that there are exactly 4 elements in the permutation which are to the left of 3 end are greater than 3 (Yes this is true, they are: $4, 8, 6, 5$).

The inversion sequence $a_1, a_2, \ldots, a_r$ satisfies the conditions

$$0 \leq a_i \leq r - i \qquad \text{for} \qquad i = 1, 2, \ldots, r$$

As seen there is no restriction on the elements which says $a_i = a_j$ is forbidden for $i \neq j$. This is of course very convenient for the crossover and mutation operations in GA.

Below two iterative algorithms are given. The first generates the inversion sequence of a given permutation and the second does the inverse (generates the corresponding permutation of a given inversion sequence).

Input perm : array holding the permutation
Output inv : array holding the inversion sequence

```
for i ← 1..r do
    { inv_i ← 0
      m ← 1
      while perm_m ≠ i do
            { if perm_m > i then inv_i ← inv_i + 1
              m ← m + 1 } }
```

Input inv : array holding the inversion sequence
Output perm : array holding the permutation
Uses pos : dummy array for intermediate result[1]

```
for i ← r..1 do
    { for m ← i + 1..r do
          if pos_m ≥ inv_i + 1 then pos_m ← pos_m + 1
      pos_i ← inv_i + 1 }
for i ← 1..r do perm_{pos_i} = i
```

C. Using the Method in GA

By this method a chromosome or a subsection of it, which has to keep an r-permutation of n-elements, will consist of

1. A gene which has alleles that are natural numbers and are limited above by $\binom{n}{r}$.
2. A sequence of r genes[2] where the allele of each element is a natural number. The maximal allele value allowed decreases by one at each element from the first position of the sequence to the last one.

In GA applications natural number valued genes are usually represented by bit strings, which are the binary representation of that number. The limitation is very easily controlled by the choosing a restricted bit length and/or a modulo operation. Except this limitation of the maximal values, which always is the case in GA applications with numerical alleles, there is no extra restriction or order that has to be preserved throughout the GA operations. Whatever crossover or mutation will produce will inheritly correspond to a valid r-permutation. Now there is a question to be answered:

- What characteristics of the parents will be inherited by the offspring?

Assuming that an ordinary bit-string one-point crossover is performed on both components of the chromosome, we can state that the offsprings in:

the combination part will have a close similarity to their parents. One of the two offsprings will receive its high-most bits from one of the parents and the other offspring will receive them from the other. Since the allele is a number that corresponds to a place in a lexicographical order the donor of the high-most bit segment will determine the rough position in the order. The low-most bits coming from

[1] The use of this array can be avoided by a more elaborated algorithm but this will not reduce the time complexity.

[2] Actually $(r-1)$ suffices, since inv_r is always zero.

the other parent will somewhat provide a minor alternation to this position.

the permutation part will inherit characteristics from both parents. For one of the offspring, one of the parents, p_1, will provide the *displacement* information of some of the permutation elements (lets call them $\mathcal{E}'$) and the other parent, p_2 will provide a similar information for the remaining permutation elements ($\mathcal{E}''$). Of course the other offspring will receive the displacement information for $\mathcal{E}'$ and $\mathcal{E}''$ from p_2 and p_1, respectively.

Similar properties can be stated for mutation. It has to be pointed out though, a mutation that hits the high-most bits in the combination part is more severe than a hit of the low-most bits. This is not so for the permutation part, here any valid alternation has the same effective severeness.

D. An Example

Lets assume we are representing 7-permutations out of a 12 element set in a chromosome. We will display a one-point crossover of two such permutations. Consider the following two permutations[3].

Perm$_1$: $10, 2, 3, 8, 1, 11, 9$
Perm$_2$: $7, 3, 6, 1, 4, 12, 8$

In the proposed method each of these two permutations will be represented by a chromosome that holds two pieces of information:

1. "Which 7-combination of the 12 elements"
2. "Which permutation of that 7-combination"

The S_n function introduced in section II-A provides the answer for the first question:

Perm$_1$ is a permutation of the 7-combination $(1, 2, 3, 8, 9, 10, 11)$. Feeding it into S_n we obtain the sequence number for this particular 7-combination: 122.

Perm$_2$ is a permutation of the 7-combination $(1, 3, 4, 6, 7, 8, 12)$. S_n applied on this combination yields the sequence number: 291.

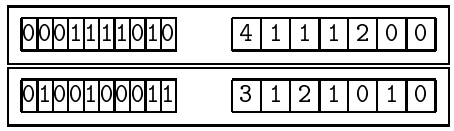
To obtain the answer to the second question we will first consider the 7-combinations stated above the base form of the permutation. For a moment let us deal with $Perm_1$ only. If we consider $(1, 2, 3, 8, 9, 10, 11)$ as the base permutation and label the positions with $1, 2, 3, 4, 5, 6, 7$ respectively, $Perm_1$ is labelwise the $(6, 2, 3, 4, 1, 7, 5)$ permutation. Similarly $Perm_2$ is labelwise the $(5, 2, 4, 1, 3, 7, 6)$ permutation when $(1, 3, 4, 6, 7, 8, 12)$ is considered as the base and labeled with $1, 2, 3, 4, 5, 6, 7$ respectively. The 'inverted sequence' method introduced in section II-B will provide the answer of the second question:

Perm$_1$ is labelwise represented by $(6, 2, 3, 4, 1, 7, 5)$ with respect to its 7-combination (when it is labeled as $1, 2, 3, 4, 5, 6, 7$). For $(6, 2, 3, 4, 1, 7, 5)$ The inversion algorithm yields the inversion sequence:(4,1,1,1,2,0,0).

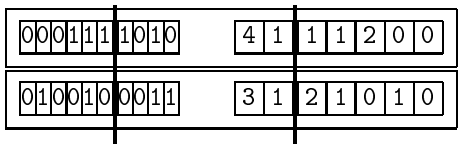
Perm$_2$, similarly, is labelwise represented by the permutation $(5, 2, 4, 1, 3, 7, 6)$ (w.r.t. its own 7-combination) and this has an inversion sequence of:(3,1,2,1,0,1,0).

This is what the chromosomes look like:

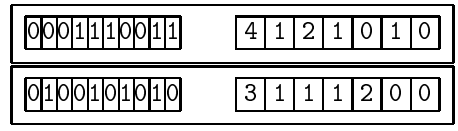
[3]these permutations were generated randomly.



Here the sequence numbers 122 and 291 are displayed in their binary form since crossover will take place at the bit boundaries of the binary representation. On the other hand, the inversion sequence is not represented so, since a crossover will not take place within the binary representation of the sequence elements, but will merely be carried out by swapping the sequence elements themselves. Now assume the crossover process has chosen the following one-point-crossover boundaries (for each component of the chromosomes).



The two offsprings are:



The crossover process has produced the new sequence numbers 115 and 298 for the Offspring$_1$ and Offspring$_2$, respectively (those are what the new binary numbers say). Using the S_n^{-1} operation introduced in section II-A one obtains that 115 corresponds to the 7-combination $(1, 2, 3, 7, 8, 10, 11)$ and 298 corresponds to $(1, 3, 4, 6, 8, 9, 10)$. Furthermore doing the inverted sequence $\Rightarrow$ permutation conversion by means of the algorithm described in section II-B we observe that the bred labelwise permutations are $(5, 2, 4, 3, 1, 7, 6)$ for Offspring$_1$ and $(6, 2, 3, 1, 4, 7, 5)$ for Offspring$_2$. Rewriting the (new) 7-combinations in these (new) label orders we obtain:

Offspring$_1$: $8, 2, 7, 3, 1, 11, 10$
Offspring$_2$: $9, 3, 4, 1, 6, 10, 8$

III. Conclusion

A new method for representing r-permutations of n-elements as GA chromosomes has been introduced. In contrast to the conventional ones this proposed representation is not handicaped under crossover and mutation. The proposed method is used in various scheduling and timetabling GA applications problems and is observed to perform exteremly well.

References

[1] D.E. Goldberg and R. Lingle, Alleles, Loci, and the Travelling Salesman Problem, in:J.J. Grefenstette (ed), *Proceedings of the First International Conference on Genetic Algorithms and Their Application*, Lawrence Erlbaum Associates, Hillsdale, NJ, 1985, pp. 154-159

[2] L. Davis, Applying Adaptive Algorithms to Epistatic Domains, in: *Proceedings of the International Joint Conference on Artificial Intelligence*, 1985, pp. 162-164
[3] M. Oliver, D.J. Smith and J.R.C. Holland, A Study of Permutation Crossover Operators on the Travelling Salesman Problem, in:J.J. Grefenstette (ed.), *Proceedings of the Second International Conference on Genetic Algorithms*, Lawrence Erlbaum Associates, Hillsdale, NJ, 1987, pp. 224-230
[4] M. Hall, *American Mathematical Society, Proceedings of the Symposium on Pure Mathematics*, 1963, **6** pp. 203
[5] S. Even, *Algorithmic Combinatorics*, The Macmillan Company, NY, 1973