Genetic Algorithm

Another category of local search methods is formed by genetic algorithms. In contrast to neighbourhood search methods, genetic algorithms operate with a *population* of solutions. A new population is created by allowing *parent solutions* in one generation to produce *offspring,* which are included in the next generation. A '*survival of the fittest*' principle is applied to ensure that the overall quality of solutions increases as the algorithm progresses from one generation to the next.

A general framework and a possible implementation of a genetic algorithm for a permutation scheduling problem is given below.

- **Initialisation**. Choose initial population P containing q solutions to be the current population (randomly generate q permutations also called *strings*).
- **Evaluation**. Compute a fitness value for each solution of P (compute the value $F(S)$ for each solution S).
- **Reproduction**. Use fitness values to select solutions from P to form a mating pool (select $q/2$ best permutations).
- **Regeneration**. Apply *crossover*, *mutation* and any other selected operations to solutions of the mating pool to form a new population ($q/2$ new permutations are obtained and replace $q/2$ worst permutations in the population).
- **Termination test**. Test whether the algorithm should terminate. If it terminates, output the best solution generated; otherwise, return to the evaluation step.

In the initialisation step, we create a population by generating q random permutations. A non-negative fitness function $F(S)$ is used in the evaluation step.

In the reproduction step, a mating pool of size $q/2$ is created. To apply the crossover operation in the regeneration step, solutions in the mating pool are randomly partitioned into pairs. With probability p_{cross}, each pair undergoes a crossover; otherwise, the pair is unchanged. Under a crossover operation, the two solutions, which we refer to as *parents*, combine to produce two *offspring*, each containing some characteristics of each parent. The hope is that one of the offspring will inherit the desirable features of each parent to produce a good quality solution. A mutation operation is applied to solutions before placing them into the new population, each element of each string (each jobs in the permutation) is selected with probability p_{mut} to be perturbed. E.g., if a job of a string is selected for mutation, then it is swapped with another randomly selected job in the same string (which yields a neighbour in the swap neighbourhood).

As a termination test, a time limit is set and the algorithm terminates when this limit is exceeded.

Consider one type of crossover, which is described below. Our description refers to a two-point crossover in which two randomly selected crossover points are used.

Reorder Crossover. Select two positions at random as crossover points and then reorder the sub-sequences between these positions to match the order of the elements in the other sequence.

For example, suppose that we start with two sequences

$\pi_1 = (1\ \ 2\ \ 3 \mid 4\ \ 5\ \ 6\ \ 7 \mid 8\ \ 9\ \ 10)$

$\pi_2 = (3\ \ 8\ \ 7 \mid 1\ \ 9\ \ 4\ \ 2 \mid 10\ \ 6\ \ 5)$

and choose two crossover points, as indicated.

Then crossover would produce offspring

$\pi_1' = (1\ \ 2\ \ 3 \mid 7\ \ 4\ \ 6\ \ 5 \mid 8\ \ 9\ \ 10)$

$\pi_2' = (3\ \ 8\ \ 7 \mid 1\ \ 2\ \ 4\ \ 9 \mid 10\ \ 6\ \ 5)$.

We illustrate this approach using an instance of problem $F2 \mid \mid \sum w_j C_j$:

j	a_j	b_j	w_j
1	3	5	7
2	1	4	2
3	6	2	4
4	2	6	1

We set p_{cross}=0.6 and p_{mut}=0.05. The original population consists of the following 8 schedulues:

S	$F(S)$	Rank
(1,2,3,4)	156	2
(1,3,2,4)	151	1
(1,3,4,2)	159	3
(2,3,1,4)	172	4
(4,1,3,2)	197	6
(3,1,2,4)	190	5
(4,1,2,3)	209	8
(2,4,1,3)	205	7

We select schedules ranked 1-4 for mating. Suppose two random pairs of parents are 1, 4, and another pair is 2, 3.

Take the first pair: (1,3,2,4) and (2,3,1,4), and apply Reorder Crossover with positions 2 and 3. We have that

$\pi_1 =(1, | 3,2 |, 4)$ → $\pi_1' = (1,2,3,4)$

$\pi_2 =(2, | 3,1 |, 4)$ → $\pi_2' = (2,1,3,4)$

Suppose the first child is subject to mutation for job 4, and that job is swapped with 1, so that the final child is (4,2,3,1).

Take the second pair: (1,2,3,4) and (1,3,4,2), and apply Reorder Crossover with positions 1 and 3. We have that

$\pi_1 =(|1,2,3 |, 4)$ → $\pi_1' = (1,3,2,4)$

$\pi_2 =(|1,3,4 |, 2)$ → $\pi_2' = (1,3,4,2)$

Suppose the second child is subject to mutation for job 4, and that job is swapped with 1, so that the final child is (4,3,1,2).

The next population:

S	*F*(*S*)	Rank
(1,2,3,4)	156	4
(1,3,2,4)	151	2-3
(1,3,4,2)	159	5
(2,3,1,4)	172	6
(4,2,3,1)	221	8
(2,1,3,4)	146	1
(1,3,2,4)	151	2-3
(4,3,1,2)	200	7

We select the four top-ranked schedules for mating etc.

Conclusions

Advantages:

- Implementation of the genetic algorithm usually does not require much knowledge about the structural properties of the problem.
- The algorithm can be easily coded.
- Often genetic algorithms produce fairly good solutions.

Disadvantages:

- May be less efficient (in terms of the running time and the accuracy of the solution) than problem-specific approaches