A HYBRID APPROACH IN GENETIC ALGORITHM: COEVOLUTION OF THREE VECTOR SOLUTION ENCODING. A CASE-STUDY

Dmitriy BORODIN, Victor GORELIK, Wim DE BRUYN and Bert VAN VRECKEM
University College Ghent, Ghent, Belgium
Dorodnicyn Computing Center or Russian Academy of Sciences, Moscow, Russia
e-mail: dmitriy.borodin@hogent.be
vgor16@mail.ru
wim.debruyn@hogent.be
bert.vanvreckem@hogent.be

Genetic algorithms (GAs) are known for providing good quality solutions to combinatorial optimization problems relatively fast. Many scientific papers and works research different aspects of GAs such as different solution encodings, various operators for solution change, evolution strategy. This paper gives a case-study of applying a GA to a particular batch scheduling problem. The solution is encoded in three independents vectors; the fitness function takes all three vectors as variables. The objective is to minimize the fitness function by coevolving the three solution vectors. Different coevolution strategies are proposed and compared. The choices of genetic operators as well as their influence on the convergence speed and robustness of the algorithms are discussed.

Introduction

Genetic algorithms, among other heuristic and meta-heuristic optimization techniques, were successfully used for solving scheduling problems, and chemical batch production problems in particular [2], [7], [9]. The contribution of this research is in applying the genetic algorithm to the batch scheduling problem involving three independent vectors and one fitness function in the solution encoding. Moreover, these vectors have different structure: one vector encodes a permutation, two other vectors encode assignments. The results of the proposed algorithm are compared with the exact solutions of the same problem instances as well as with a Kernighan-Lin (KL) heuristic. The GA outperforms the KL algorithm and the solution quality is acceptable – the deviation from the exact solutions is 3-10% while for some instances GA demonstrated the ability to find the exact solution.

Problem background and statement

We describe the problem background and statement shortly as the technical details are important for the GA efficiency. For the detailed description and mathematical formulation the reader may see [5] (the work was presented at the 24th European conference on Operations Research in
Lisbon in 2010) and the yet unpublished paper [6]. Reference [6] also contains the optimal solutions for the problem instances (calculated using the commercial solver cplex) as well as the Kernighan-Lin heuristic results.

The problem is in finding the optimal schedule of an enzyme production according to the total-weighted tardiness optimality criteria. A mixed-integer programming model was created to solve the problem but the model turned out to be difficult to solve [5, 6]. The idea of fixing integer variables of the model lead to splitting up the problem into two parts in order to get a good solution within a reasonable speed. Indeed, if all integer variables are fixed, then only a contentious linear program (LP) is left to solve. The LP was named evaluation function \( (EF) \) and provided the schedule and the objective function value (the fitness value) by the given values of integer variables.

Following this idea, the problem can be reduced to the problem involving three vectors of integer variables which represented different stages of the production process. Once there is a need to calculate the fitness function \( (FF) \) value, the \( EF \) takes these three vectors as input, solves the LP and provides the \( FF \) value. From the technical point of view, the \( EF \) could be interpreted as an objective function in the common sense, and then the problem is in unconditional optimization.

The problem statement:
Find the values of the vectors
- \( R_{\text{sequence}}: (1,2,3,\ldots,n) \)
- \( MF_{\text{assignment}}: (1,5,5,2,\ldots,1) \)
- \( BT_{\text{assignment}}: (2,5,5,2,\ldots,1) \)

bringing the minimum to the \( EF \).

\( R_{\text{sequence}} \) is a sequence of unique integer numbers from 1 to \( n \) (\( n \) is the number of batches to be scheduled), which represents the order of how the batches are processed on a stage of production; each component of \( MF_{\text{assignment}} \) and \( BT_{\text{assignment}} \) represents the number of the resource the respective batch is assigned to, the values of the components are limited by the number of the resources (machines) and are from 1 to 5 for \( MF_{\text{assignment}} \) and from 1 to 4 for \( BT_{\text{assignment}} \). The dimension of all three vectors is \( n \).

**Genetic algorithm steps, operators and implementation**
The general steps of the GA for the problem under study:
1) Generate an initial set of solutions (a population) randomly, i.e. generate the acceptable values of the three vectors and calculate the \( EF \) for each one. Normally, the amount of the population varies from 30 to 1000 solutions.
2) Perform the selection of pairs of the solutions from the population in order to crossover them (exchange the minor data of two solutions
thus that the offspring inherit the best characteristics) and add their offspring into the new population (next generation).

3) Mutate (minor random change of a solution) some solutions of the population selected randomly. The mutation probability may vary from 0 to 50%, but the standard “patient” value is 10%. Mutation is necessary to escape from the local optima by trying different points of the search space.

4) If the stopping criteria is satisfied (the number of iterations is reached, the solution is not improving for a certain number of iterations, time limit is reached), then return the best found solution. Otherwise, continue to step 2.

As vector $R_sequence$ is a permutation, the crossover and mutation operators are similar to the sequencing problems as well as for the travelling salesman problem. We implemented partially-matched (PMX), cyclic (CX) and order (OX) crossovers for this vector. We refer to [1,3,4,8] for the details about these crossovers. For two other vectors (as they can contain duplicates and missing values) the one- and two-point and uniform crossovers were employed. Reference [8] describes these operators and analyzes theoretical difference between them.

Mutation was performed by the insert, swap or reverse operators.

For the selection strategy we tried tournament, roulette-wheel, Genitor-Whitley, two-random, all with best and cascade.

As the value of the $EF$ is computed only by the values of all three vectors the challenging point is to determine the best possible order of applying genetic operators to them. We considered the following choices: 1) evolve all three vectors simultaneously; 2) evolve them consequently (one by one) by fixing the values of two others; 3) evolve two vectors and fix another one.

**Numerical results analysis**

The algorithm was programmed in the C# language using the Microsoft Visual Studio .NET environment. Such choice enabled to use the full performance of the computer processor as well as to test the multicore processor features.

Besides the coevolution strategy, the choice of genetic operators – crossover, mutation and selection – is important as it influences the solution quality, convergence speed and the algorithm robustness.

Our numerical results showed that different operators and coevolution strategies provided different results for different data instances. But in each case either the robustness was not good (the algorithm could find the best possible solution only in 50-60% of executions) or the algorithm didn’t converge to the best possible solution, or the convergence speed was not
balanced (too fast or too slow). The question was: how to find the best combination of the operators?

One of the answers is in applying the hybrid approach. In this case, on each iteration different operators are applied, the random choice provided good results for the described problem.

Table 1 shows the time and solution quality comparison of the GA with the exact solutions and the Kernighan-Lin heuristics.

Table 1

<table>
<thead>
<tr>
<th>Instance</th>
<th>CPLEX best</th>
<th>KL best</th>
<th>KL time</th>
<th>GA best</th>
<th>GA time</th>
<th>SA</th>
<th>SR</th>
</tr>
</thead>
<tbody>
<tr>
<td>N10_1</td>
<td>90</td>
<td>91</td>
<td>7</td>
<td>93</td>
<td>5</td>
<td>S,C</td>
<td>1.2</td>
</tr>
<tr>
<td>N10_2</td>
<td>40</td>
<td>35</td>
<td>16</td>
<td>30</td>
<td>17</td>
<td>S,C</td>
<td>1.2</td>
</tr>
<tr>
<td>N10_3</td>
<td>42</td>
<td>56</td>
<td>10</td>
<td>44</td>
<td>6</td>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>N10_4</td>
<td>49</td>
<td>52</td>
<td>14</td>
<td>50</td>
<td>5</td>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>N10_5</td>
<td>43</td>
<td>48</td>
<td>6</td>
<td>45</td>
<td>12</td>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>N15_1</td>
<td>73</td>
<td>77</td>
<td>80</td>
<td>76</td>
<td>25</td>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>N15_2</td>
<td>43</td>
<td>45</td>
<td>112</td>
<td>45</td>
<td>34</td>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>N15_3</td>
<td>57</td>
<td>70</td>
<td>39</td>
<td>66</td>
<td>75</td>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>N20_1</td>
<td>52</td>
<td>54</td>
<td>490</td>
<td>54</td>
<td>180</td>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>N20_2</td>
<td>58</td>
<td>66</td>
<td>260</td>
<td>64</td>
<td>194</td>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>N30_1</td>
<td>-</td>
<td>180</td>
<td>1304</td>
<td>186</td>
<td>1560</td>
<td>C</td>
<td>2</td>
</tr>
</tbody>
</table>

Fig. 1. Convergence speed graph, consequent (C1) and hybrid (C2) coevolution approaches, instance N30_1

Conclusions

The stochastic character of GAs is important not only for the solution encoding and genetic operators, but also for the choice of these operators and, in the case of the described problem, the coevolution strategy. Deterministic approach (genetic operators are fixed for all iterations) provides good results as well but is less stable and can’t guarantee the solution quality for each instance or iteration.

The hybrid approach enhances the exploration and exploitation abilities of a GA thus providing an efficient pseudo optimal solution search method on a large solution space.
References