

Paper:

Optimization of Genetic Operators for Scheduling Problems

António Ferrolho* and Manuel Crisóstomo**

*Dept. of Electrotechnical Engineering, Superior School of Technology, Polytechnic Institute of Viseu
3504-510 Viseu, Portugal

E-mail: antferrolho@elect.estv.ipv.pt

**Institute of Systems and Robotics, Dept. of Electrical and Computer Science Engineering, University of Coimbra
Polo II, 3030-290 Coimbra, Portugal

E-mail: mcris@isr.uc.pt

[Received February 15, 2007; accepted June 14, 2007]

Genetic algorithms (GA) can provide good solutions for scheduling problems. But, when a GA is applied to scheduling problems various crossovers and mutations operators can be applicable. This paper presents and examines a new concept of genetic operators for scheduling problems. A software tool called hybrid and flexible genetic algorithm (HybFlexGA) was developed to examine the performance of various crossover and mutation operators by computing simulations of job scheduling problems.

Keywords: scheduling, genetic algorithms, crossover operators, mutation operators

1. Introduction

The strong performance of the genetic algorithm (GA) depends on the choice of good genetic operators. Then, the selection of appropriate genetic operators is very important for constructing a high performance GA. Various crossover and mutation operators have been examined for sequencing problems in the literature (for example, see [3–5, 8–11, 13]).

This paper presents and examines a new concept of genetic operators for scheduling problems. Each one of these genetic operators was evaluated with the objective of selecting the best performance crossover and mutation operators. When the performance of a crossover operator is evaluated, a GA without mutation is employed and the evaluation of a mutation operator is carried out by a GA without crossover. In the literature, various crossover operators and mutation operators were examined in this manner (for example, see [8, 10]).

The organization of this paper is as follows: section 2 proposes a new concept of genetic operators for scheduling problems; section 3 presents the developed software tool, called HybFlexGA. This software is used to examine the performance of various crossover and mutation operators by computing simulations of scheduling problems. Section 4 presents the computer simulations and section 5 concludes this paper.

2. Genetic Operators

This section presents a new concept of genetic operators for scheduling problems.

2.1. Crossover Operators

Crossover is an operation to generate a new sequence (child chromosome) from two sequences (parent chromosomes). **Fig. 1(a)** presents a one-point crossover: 1 child (OPC1C). In OPC1C we randomly chose one parent, and then one point is also randomly selected in this parent. The jobs on one side are inherited from the parent to the child, and the other jobs are placed in the order of they appeared in the other parent. **Fig. 1(b)** presents a two-point crossover: 1 child (Version I) - TPC1CV1. In TPC1CV1 we chose one parent, and then two points are also randomly selected in this parent. The jobs outside the two selected points are always passed down from one parent to the child, and the other jobs are placed in the order of they appeared in the other parent. **Fig. 1(c)** presents a two-point crossover: 1 child (Version II) - TPC1CV2. In TPC1CV2 we chose one parent, and after two points are also randomly selected in this parent. The jobs between the two selected points are always passed down from one parent to the child, and the other jobs are placed in the order of they appeared in the other parent.

Figure 1(d) presents a one-point crossover: 2 children (OPC2C). OPC2C is similar to OPC1C but in this crossover operator we always obtain two children. **Fig. 1(e)** presents a two-point crossover: 2 children (Version I) - TPC2CV1. TPC2CV1 is similar to TPC1CV1 but in this crossover operator we always obtain two children. **Fig. 1(f)** presents a two-point crossover: 2 children (Version II) - TPC2CV2. TPC2CV2 is similar to TPC1CV2 but in this crossover operator we always obtain two children.

Crossover operators with 3 and 4 children were also developed. The crossover operators with 3 children are called two-point crossover: 3 children (Version I) - TPC3CV1 and two-point crossover: 3 children (Version II) - TPC3CV2. TPC3CV1 is a mix of TPC1CV1 plus TPC2CV1 and TPC3CV2 is a mix of TPC1CV2 plus TPC2CV2. The crossover operator with 4 children is

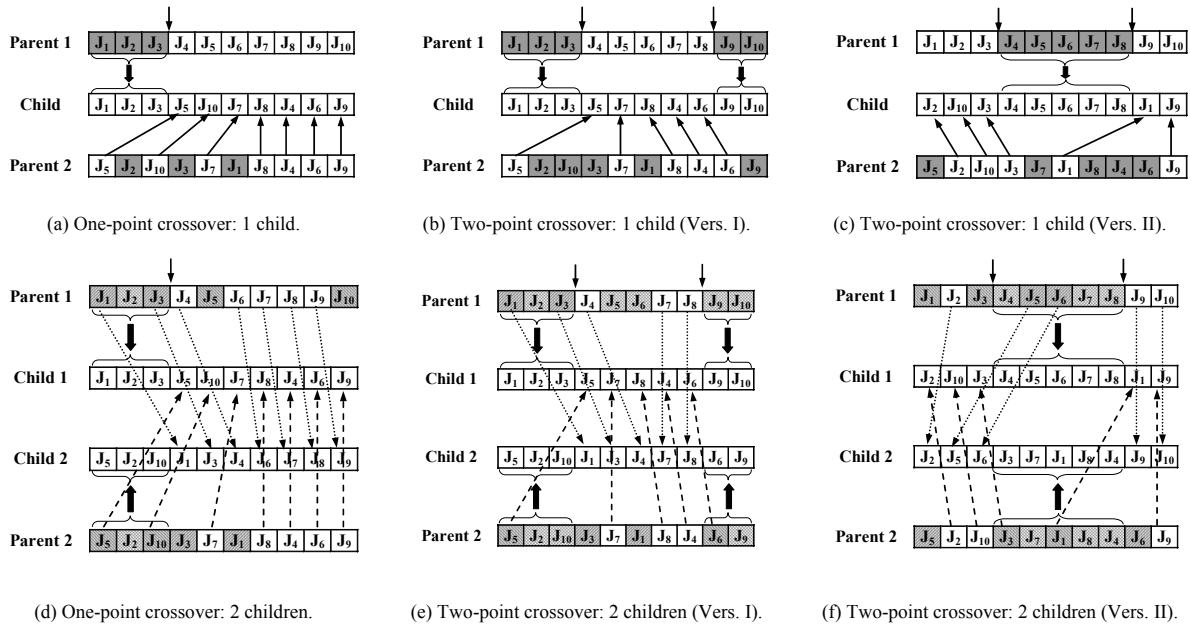


Fig. 1. Illustration of crossover operators.

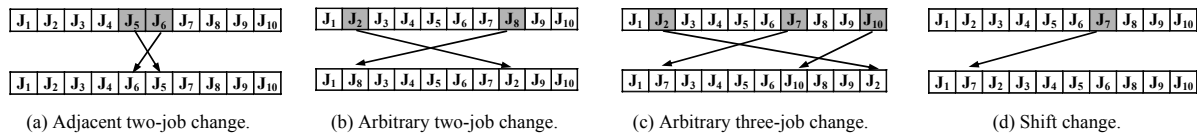


Fig. 2. Illustration of mutation operators.

called two-point crossover: 4 children (TPC4C). This operator is a mix of TPC2CV1 plus TPC2CV2.

In our computational tests in section 4 we also used the following crossover operators: order crossover (OX) in Goldberg [5], cycle crossover (CX) in Oliver [11], and position based crossover (PBX) in Syswerda [13].

2.2. Mutation Operators

Mutation is an operation to change the order of n jobs in the generated child. The mutation operators are used to prevent the loss of genetic diversity. We examined the following four mutations used by Murata in [9, 10]: adjacent two-job change (Adj2JC), arbitrary two-job change (Arb2JC), arbitrary three-job change (Arb3JC) and shift change (SC).

As we can see in Fig. 2(a), in Adj2JC the two adjacent jobs to be changed are randomly selected. In Arb2JC (see Fig. 2(b)) the two jobs to be changed are arbitrarily and randomly selected. In Arb3JC (see Fig. 2(c)) the three jobs to be changed are arbitrarily and randomly selected. In SC a job at one position is removed and placed at another position, as shown in the Fig. 2(d).

A new mutation operator called the arbitrary 20%-jobchange (Arb20%JC) was developed, as we can see in Fig. 3. This mutation selects 20% of the jobs in the child chromosome. The 20% of the jobs to be changed are arbitrarily and randomly selected, and the order of the selected jobs after the mutation is randomly specified. The

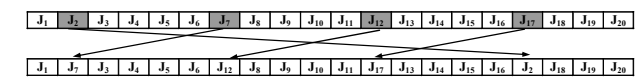


Fig. 3. Arbitrary 20%-job change.

percentage in this mutation operator gives the operator some flexibility, i.e., the number of jobs to be changed depends on the size of the chromosome. For example, if we have a chromosome with 40 jobs and other with 100 jobs, the number of the jobs to be changed is 8 and 20 respectively.

3. Developed Software

We developed a software tool, called Hybrid and Flexible Genetic Algorithm (HybFlexGA), to examine the performance of various crossover and mutation operators by computing simulations on scheduling problems. The HybFlexGA was coded in C++ language, and Fig. 4 shows its architecture. Its architecture is composed of three modules: interface, preprocessing, and scheduling module.

The interface module with the user is very important for the scheduling system's success. Thus, this interface should be user friend and dynamic so as to allow easy manipulation of the scheduling plan, jobs, and so forth.