Andrzej Bożek and Marian Wysocki
Rzeszów University of Technology,
Department of Computer and Control Engineering,
al. Powstańców Warszawy 12, 35-959 Rzeszów

April 10, 2014

Supplemental material - Appendix B

This document describes benchmark instances dedicated for the CF-FJS problem. Benchmarks parameters and generation rules as well as the file format used for instances storing are presented.

1. Parameters and generation rules of benchmark instances

We propose a set of ten benchmarks prepared to verify algorithms for solving the CF-FJS problem. Names **cff_*N*** are used for instances of these benchmarks, where **cff** stands for *continuous flow flexible* (*job shop*) and ***N*** is a two digits benchmark number. For each benchmark instance the number of jobs n and machines r is defined arbitrarily, and other parameters are given by range with a lower and upper limit (Table 1). There are the following parameters: the number of operations in a job $\mathrm{js}(k)$, the number of alternative machines for an operation $\mathrm{ms}(k,i)$, the nominal processing time of an operation $p_{k,i,p}$, the material buffer between operations in percentage $b_{k,i,p}$, as well as the machine efficiency pattern e_p defined by three parameters, namely the efficiency value in percentage e_p^{EV} and length of time periods when machine is on/off: $e_p^{\mathrm{ON}}/e_p^{\mathrm{OFF}}$.

Table 1. Benchmarks parameters.

Inst.	n	r	$\mathrm{js}(k)$		$\mathrm{ms}(k,i)$		$p_{k,i,p}$		$b_{k,i,p}$		e_p^{EV}		e_p^{ON}		e_p^{OFF}	
			min	max	min	max	min	max	min	max	min	max	min	max	min	max
cff_01	30	20	8	12	2	5	6	40	40	90	50	120	5	20	4	10
cff_02	40	25	5	12	2	4	20	30	20	90	30	150	5	17	5	9
cff_03	40	30	6	15	1	5	8	35	40	70	40	130	4	15	2	6
cff_04	60	50	5	12	1	4	6	40	10	80	20	140	4	20	5	12
cff_05	60	60	8	15	2	6	12	35	30	60	30	120	3	15	5	10
cff_06	80	50	10	15	1	5	10	30	30	70	50	120	2	16	3	8
cff_07	80	80	12	20	2	7	20	40	10	80	40	130	4	15	4	7
cff_08	100	80	10	15	2	5	10	40	30	80	60	140	5	12	2	5
cff_09	120	100	10	20	1	6	6	35	20	90	50	140	4	10	3	6
cff_10	150	120	15	25	1	5	8	40	20	100	50	150	3	8	2	7

The benchmark instances have been generated randomly on the basis of the given parameters. The number of operations for each job, the alternative machine sets and buffer values for each operation as well as intervals of machine speed calendars have been drawn independently with the use of integer uniform distribution. Any interval of machine speed has been drawn with the same probability equal to $\frac{1}{2}$ as machine stopping (duration from the range e_p^{OFF} and speed 0) or as machine working (duration from the range e_p^{ON} and speed from the range e_p^{EV}). For each machine 20 consecutive intervals have been drawn. Adjacent intervals of machine stopping have been aggregated into single intervals, thus the effective number of intervals is typically lower than 20. The speed calendars are considered as unrestricted in the sense that given intervals pattern repeats infinitely (see Fig. 1).

The benchmark instances are saved in the attached files: `bench_cf01.cff - bench_cf10.cff`. Format of these files is described below.

2. Benchmark instance file format

The files that represent benchmark instances of the CF-FJS scheduling problem have the following syntax

```
CF_FJS_benchmark:
    header
    jobs
    machines

header:
    [r] <n> <rm>

jobs:
    |n|*{job}

job:
    [r] <js> first_operation operations

first_operation:
    <ms> fprocs

fprocs:
    |ms|*{fproc}

fproc:
    <mi> <pt>

operations:
    |js-1|*{operation}

operation:
    <ms> procs

proc:
    <mi> <pt> <bf>

machines:
    |rm|*{machine}

machine:
    [r] <mp> efficiency_pattern

efficiency_pattern:
    |mp|*{efficiency_interval}

efficiency_interval:
    <tm> <ef>
```

where

- `CF_FJS_benchmark` is the root symbol which represents whole content of the file,
- `[r]` is the terminal control symbol which represents start of a new row,

- <x> is the terminal symbol that has the form of decimal encoded integer,
- |y|*{x} is the expression denoting that a symbol of type x is repeated y times,
- all other symbols are non-terminal.

The values used in the syntax definition have the following meaning

- n the number of jobs,
- rm the number of machines,
- js the number of operations in a job,
- ms the number of machines available for an operation,
- mi a machine identifier,
- pt a nominal processing time,
- bf a minimal buffer (given in percentage),
- mp the number of efficiency pattern intervals,
- tm the time at the end of an efficiency pattern interval,
- ef a value of efficiency (given in percentage).

For clarification, an exemplary content of such file (a very small problem instance) is presented.

```
(Row 1)  2 3
(Row 2)   3   2 1 8 3 11 1 2 16 83 2 1 9 47 2 17 78
(Row 3)   4   1 3 6 2 1 12 73 2 8 90 1 3 22 73 1 2 15 81
(Row 4)   4   4 120 7 110 12 0 14 90
(Row 5)   3   3 110 7 0 11 110
(Row 6)   4   1 130 5 120 9 80 11 120
```

The first row specifies that the problem has 2 jobs and 3 machines. The jobs are described in the rows 2-3 and the machine efficiency patterns are given in the rows 4-6. The first job (row 2) consists of 3 operations. The first operation can be processed by two machines with the parameters (machine identifier, nominal processing time) equal to (1, 8) and (3, 11), respectively. For successive operations of the job three parameters are defined (machine identifier, nominal processing time, minimal buffer [%]) for each available machine: (2, 16, 83) for the second operation and (1, 9, 47), (2, 17, 78) for the third one. Similarly, the row 3 describes the second job which consists of 4 operations. The row 4 specifies efficiency pattern for the first machine. The pattern (Fig. 1) should be considered as infinite (periodically repeated): $e_1 = ((4, 120), (7, 100), (12, 0), (14, 90), \dots)$.

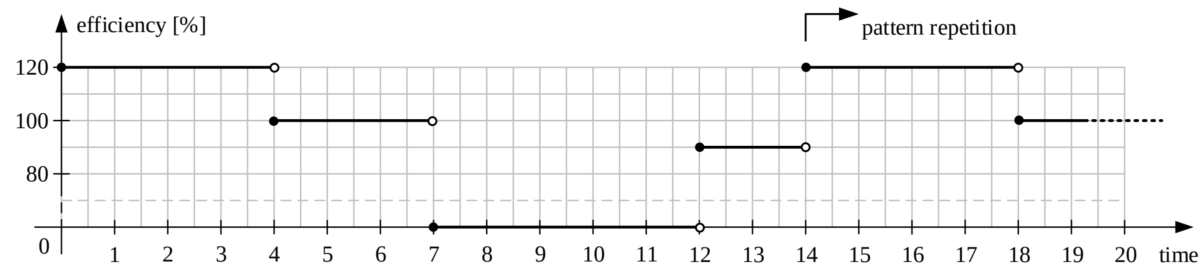

Figure 1. Efficiency pattern for the machine 1.