International Journal of Technology and Engineering Studies
volume 5 issue 4 pp. 106-112 doi: https://dx.doi.org/10.20469/ijtes.5.10001-4

An Instance Generator for Scheduling Problems Featuring Options for Unequal Stages and Unequal Parallel Machines

Jakkrit Latthawanichphan
King Mongkuts University of Technology North Bangkok,
Bangkok, Thailand

Watchara Songserm [*]
Rajamangala University of Technology Phra Nakhon,
Bangkok, Thailand

Teeradej Wuttipornpun
King Mongkuts University of Technology North Bangkok,
Bangkok, Thailand

Abstract: This research aims to develop a new Instance Generator for Scheduling Problems (IGSP). There are three main features differentiating IGSP generator from extant Instance Generators (IGs). Firstly, the IGSP can generate datasets for various types of production shop (single machine, parallel machines, flow shop, and job shop), while the extant IGs can generate datasets for only a specific production shop. Secondly, for the multi-stage problem, in the case of a flow shop and job shop, a number of stages of each job can be unequal. Finally, a number of parallel machines in each stage of the multi-stage problem can be unequal. The final two features make datasets obtained by the IGSP more realistic than the extant IGs. In addition, IGSP features a comprehensive graphical user interface and well-organised output text files. Researchers can use it to evaluate the performances of their scheduling algorithms and determine benchmark problem instances.

Keywords: *IG, scheduling problems, unequal stages, unequal machines, Sequence-Dependent Setup Time (SDST)*

Received: 21 May 2019; ***Accepted:*** 17 July 2019; ***Published:*** 26 August 2019

I. INTRODUCTION

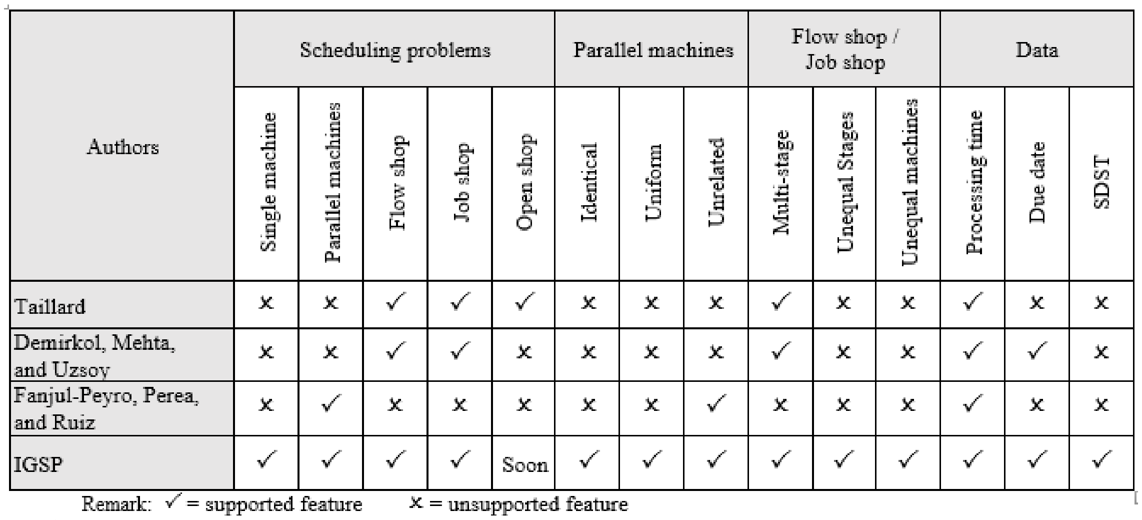
Mathematical modelling is an important tool for operations research. Datasets are normally required to evaluate performances of the developed math models. Generally, one of two approaches is used to obtain these datasets. The first approach is collecting data from real industries, reflecting the actual conditions. Unfortunately, this approach is very time and cost consuming and involves managing human errors. Complications are compounded when many replicate datasets are required. In the second approach, an IG is used to generate datasets based on random distributions (hypothetical data). Most researchers prefer the second approach because it saves time and cost, and avoids having to manage human errors [1, 2, 3, 4, 5]. Furthermore, the datasets obtained by the second approach also reflect the actual conditions, assuming the appropriate random distributions are employed [6, 7].

A survey of the extant IGs proposed in the literature shows that there have not been many IGs developed in the past three decades. The most famous IG appears to have been developed by Taillard [8], as many researchers employ this IG to evaluate their algorithms [9, 10, 11, 12]. Taillards IG generates datasets using a Linear Congruen-

[*]Correspondence concerning this article should be addressed to Watchara Songserm, Rajamangala University of Technology Phra Nakhon, Bangkok, Thailand. E-mail: watchara.s@rmutp.ac.th

tial (LCG) equation. It can generate datasets for three production shops: flow shop, job shop, and open shop. Unfortunately, Taillard focuses on the makespan. Therefore, only job processing time is provided. Other data are ignored, including due date and setup time, which are essential for determining penalty costs such as those for tardiness, earliness, and total setup time. In addition, the single machine and parallel machine production shops are not presented in this IG [8, 13, 14]. Given the due date is directly related to penalty costs, it was later included in an IG developed by Demirkol et al. [13]. This IG provides both due dates and processing times so that penalty costs can be determined. However, it generates datasets for scheduling problems in only the flow shop and the job shop, without considering setup time. Recently, an IG with additional resources was developed for scheduling problems [14]. However, this IG provides only processing times for jobs on unrelated parallel machines. Note that for all of the IGs in the literature reviewed, every job is defined for operation in the same number of stages, ignoring some industries requiring unequal stage.

The extant IGs described were all developed according to specific interests and specific problems. This research presents a new IGSP that addresses the oversights of extant IGs. The IGSP provides three important datadue date, processing time, and SDST for both single-stage production shops (single machine and parallel machines) and multi-stage production shops (flow shop and job shop). Using IGSP, in multi-stage production shops, a number of stages of each job and a number of parallel machines of each stage are allowed to be unequal. These two features are considered the novel features of IGSP. A comparison between IGSP and the extant IGs is summarised in following figure.



Authors	Scheduling problems					Parallel machines			Flow shop / Job shop			Data		
	Single machine	Parallel machines	Flow shop	Job shop	Open shop	Identical	Uniform	Unrelated	Multi-stage	Unequal Stages	Unequal machines	Processing time	Due date	SDST
Taillard	✗	✗	✓	✓	✓	✗	✗	✗	✓	✗	✗	✓	✗	✗
Demirkol, Mehta, and Uzsoy	✗	✗	✓	✓	✗	✗	✗	✗	✓	✗	✗	✓	✓	✗
Fanjul-Peyro, Perea, and Ruiz	✗	✓	✗	✗	✗	✗	✗	✓	✗	✗	✗	✓	✗	✗
IGSP	✓	✓	✓	✓	Soon	✓	✓	✓	✓	✓	✓	✓	✓	✓

Remark: ✓ = supported feature ✗ = unsupported feature

Fig. 1. Comparison between IGSP and extant IGs

The rest of this paper is organised as follows. The next section explains how IGSP calculates processing time (p), due date (d), and SDST. Section 3 presents an IGSP User Interface (UI) and explains the output file format. Finally, Section 4 concludes this research and makes recommendations for further study.

II. CALCULATION OF *P*, *D*, SDST

This section elaborates on the calculation of *p*, *d*, and SDST. All of the variables and formulas used to calculate these data are shown in Figures 13. Fig. 2 demonstrates the use of Taillards concept to calculate the processing time of job *i* stage *j* on machine *k* (p_i,*j*,*k*). This calculation chooses a random number in a range between Umin and Umax, which are lower and upper limits of a uniform distribution specified by the user. Taillard calculates processing time by applying a LCG along with four important parameters: *a, b, c,* and *m*. These parameters are configured at certain numbers, as shown in Fig. 2, so that processing time is perfectly randomised [15]. Due date is a function of p_i, *j, k*, based on the concept proposed by Hasija and Rajendran [16]. It is randomly generated using the equations in steps 2 and 3 of Fig. 3. Similarly to Ruiz and Stutzles IG, the IGSP calculates SDST as a proportion of maximum processing time (Umax), as shown in Fig. 4 [17]. Two types of setup times constitute SDST data: Loading (LD) and Unloading (UD). The time required to set up a machine before it starts production is LD, while UD refers to the time required to remove all essential devices used in production. For the IGSP, the SDST of job*i, j* is randomly generated, but it must be less than or equal to UDi +LDj. Since the SDST data represents a sequence of jobs, they are provided in an easily understood matrix form, as shown in Fig. 8.

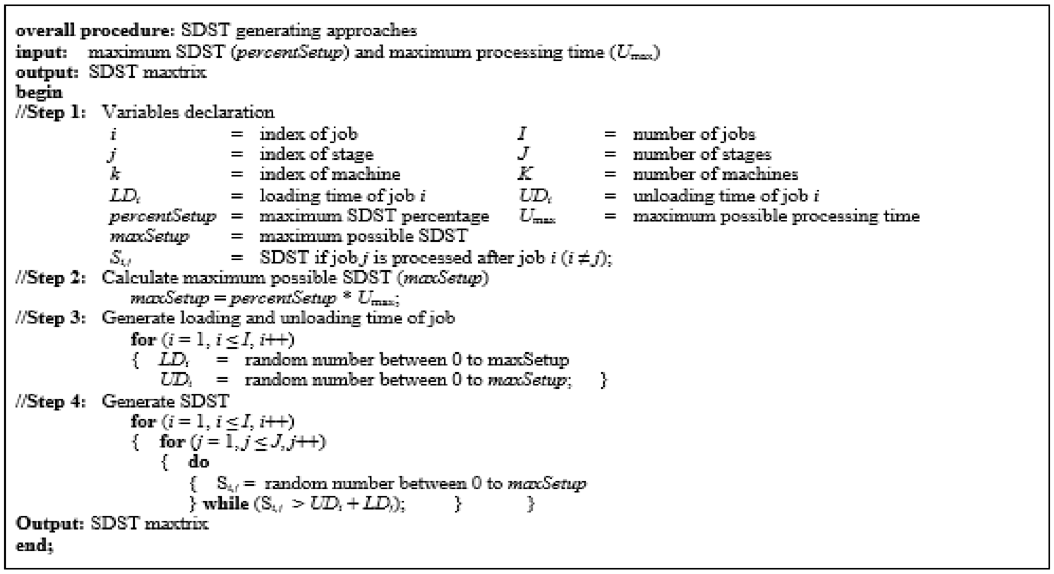
```
overall procedure: processing time generating approaches
input:    initial seed ($X_r$; $r$=0)                    // $0 < X_0 < 2^{31}$-1
          minimum possible processing time ($U_{min}$) and maximum possible processing time ($U_{max}$)
          number of jobs, number of stages, number of machines
output:   processing time based on uniform distribution
begin
//Step 1:   Variables declaration
            $i$   =  index of job          $I$  =  number of jobs
            $j$   =  index of stage        $J$  =  number of stages
            $k$   =  index of machine      $K$  =  number of machines
            $X_r$ =  initial seed          $r$  =  index of seed
            Construct parameters referring to Taillard
            $a$ = 16,807     $b$ = 127,773     $c$ = 2,836        $m = 2^{31}$-1
//Step 2:   Find a modification of seed ($X_{r+1}$)
            for ($i$ = 1, $i \leq I$, $i$++)
            {for ($j$ = 1, $j \leq J$, $j$++)
                {for ($k$ = 1, $k \leq K$, $k$++)
                    {for ($r$ = 0, $r$ < ($I$*$J$*$K$) - 1, $r$++)
                        {
                            $X_{r+1} = a(X_r \bmod b) - \frac{X_r}{b}c$
                        if ($X_{r+1} < 0$)
                        {
                            $X_{r+1} = X_{r+1} + m$
                        }
//Step 3:   Find a random number between 0 and 1 ($Rand_{i,j,k}$)
                            $Rand_{i,j,k} = \frac{X_{r+1}}{m}$
//Step 4:   Calculate a processing time based on uniform distribution
                            $processing\ time_{i,j,k} = U_{min} + \mathrm{trunc}(Rand_{i,j,k} \cdot (U_{max} - U_{min} + 1))$
                        }
                    }
                }
            }
Output: processing time based on uniform distribution
end;
```

Fig. 2. Processing time calculation

```
overall procedure: due date generating approaches
input:    processing time for each machine
output:   due date of job
begin
//Step 1:   Variables declaration
            $i$         =  index of job          $I$  =  number of jobs
            $j$         =  index of stage        $J$  =  number of stages
            $k$         =  index of machine      $K$  =  number of machines
            $d_i$       =  due date of job $i$
            $P_i$       =  processing time of job $i$
            $p_{i,j,k}$ =  processing time of job $i$, stage $j$, machine $k$
            $random$    =  random number uniformly distributed in [0,1);

//Step 2:   Find processing time of job $i$ ($P_i$)
                for ($i$ = 1, $i \leq I$, $i$++)
                {  $P_i$      =      $\sum_{j=1}^{J}\sum_{k=1}^{K} \frac{p_{i,j,k}}{K}$ ;
//Step 3:   Find due date of job $i$ ($d_i$)
                    $d_i$  =  $P_i \times (1 + random \cdot 3)$;   }
Output: due date of job $i$
end;
```

Fig. 3. Due date calculation



```
overall procedure: SDST generating approaches
input:   maximum SDST (percentSetup) and maximum processing time ($U_{max}$)
output: SDST maxtrix
begin
//Step 1:  Variables declaration
           $i$             =  index of job                  $I$         =  number of jobs
           $j$             =  index of stage                $J$         =  number of stages
           $k$             =  index of machine              $K$         =  number of machines
           $LD_i$          =  loading time of job $i$       $UD_i$      =  unloading time of job $i$
           percentSetup  =  maximum SDST percentage       $U_{max}$   =  maximum possible processing time
           maxSetup      =  maximum possible SDST
           $S_{i,j}$       =  SDST if job $j$ is processed after job $i$ ($i \neq j$);
//Step 2:  Calculate maximum possible SDST (maxSetup)
              maxSetup = percentSetup * $U_{max}$;
//Step 3:  Generate loading and unloading time of job
              for ($i = 1, i \leq I, i$++)
              {  $LD_i$  =  random number between 0 to maxSetup
                 $UD_i$  =  random number between 0 to maxSetup;    }
//Step 4:  Generate SDST
              for ($i = 1, i \leq I, i$++)
              {  for ($j = 1, j \leq J, j$++)
                 {  do
                    {  $S_{i,j}$ = random number between 0 to maxSetup
                    } while ($S_{i,j} > UD_i + LD_j$);     }     }
Output: SDST maxtrix
end;
```

Fig. 4. SDST calculation

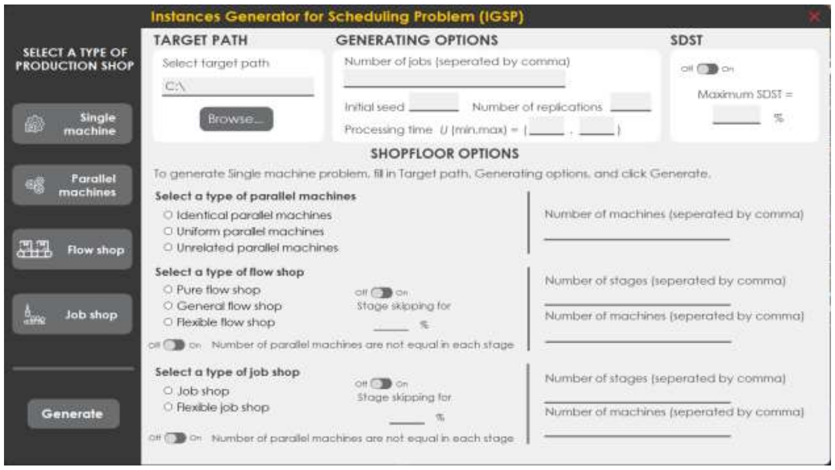
III. USER INTERFACE AND OUTPUT FILE FORMAT OF IGSP

This section details the user interface, output file folders and file names, and output text files. These elements are represented in Fig. 5-8.

A. User Interface

Fig. 5 shows the IGSP user interface. It consists of five main panels:

- Panel 1 is for selecting or creating the target path where the generated datasets will be sacved.
- Panel 2 is for entering the number of jobs, the number of replications, initial seed value, and minimum and maximum processing-time limits (Umin and Umax).
- Panel 3 is for entering a percentage of Umax, which is the SDST upper limit. This option can be switched to On or Off.
- Panel 4 is for selecting a production shop. The Generate button is included in this panel but only enabled when the essential parameters have been entered. (Note that the open shop will be added in the next phase.)
- Panel 5 consists of four categories, which are used separately by specific production shops. Category 5-1 provides instruction to single machine operators. Categories 5-2, 5-3 and 5-4 allow essential parameters to be entered for parallel machines, flow shop and job shop environments.



Instances Generator for Scheduling Problem (IGSP)

SELECT A TYPE OF PRODUCTION SHOP

TARGET PATH

Select target path: C:\

GENERATING OPTIONS

Number of jobs (seperated by comma)

Initial seed ____ Number of replications ____

Processing time U (min,max) = (____ , ____)

SDST

Off On

Maximum SDST = ____ %

SHOPFLOOR OPTIONS

To generate Single machine problem, fill in Target path, Generating options, and click Generate.

Select a type of parallel machines
- Identical parallel machines
- Uniform parallel machines
- Unrelated parallel machines

Number of machines (seperated by comma)

Select a type of flow shop
- Pure flow shop
- General flow shop
- Flexible flow shop

Off On Stage skipping for ____ %

Off On Number of parallel machines are not equal in each stage

Number of stages (seperated by comma)

Number of machines (seperated by comma)

Select a type of job shop
- Job shop
- Flexible job shop

Off On Stage skipping for ____ %

Off On Number of parallel machines are not equal in each stage

Number of stages (seperated by comma)

Number of machines (seperated by comma)

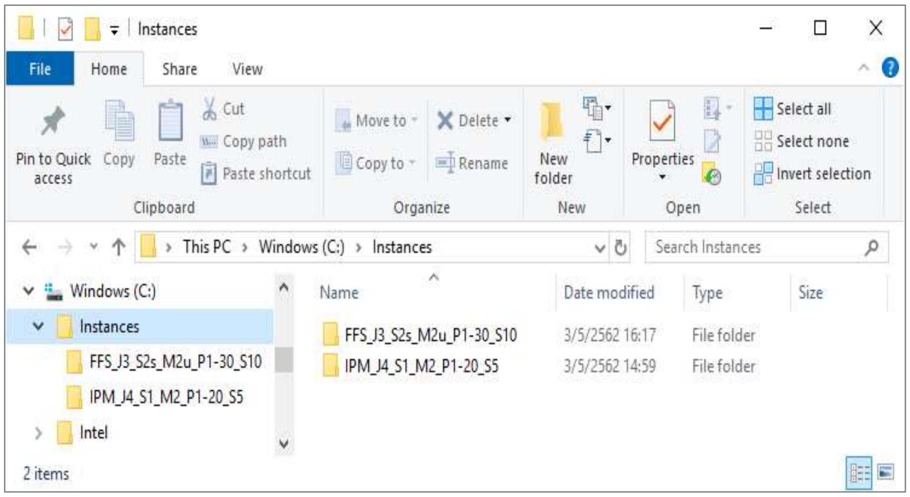
Fig. 5. User interface of IGSP

B. Output File Folder and File Name

This section explains output file folders and file names. Six components of the file folder provide information about the type of production shops, number of jobs, number of production stages, number of parallel machines in each stage, lower and upper limits of processing time, and SDST percentage. Table 1 provides notations and definitions of each component. Using the table, it is known that the file folder FFS_J3_S2s_M2u_P1-30_S10 (presented in Fig. 6) describes a job with the following characteristics: a flexible flow shop, 3 jobs, two-stage with unequal stages, two machines with unequal machines at each stage, a 130 processing time range, and maximum SDST set to 10% of the maximum processing time.

TABLE 1
NOMENCLATURE OF FILE FOLDERS

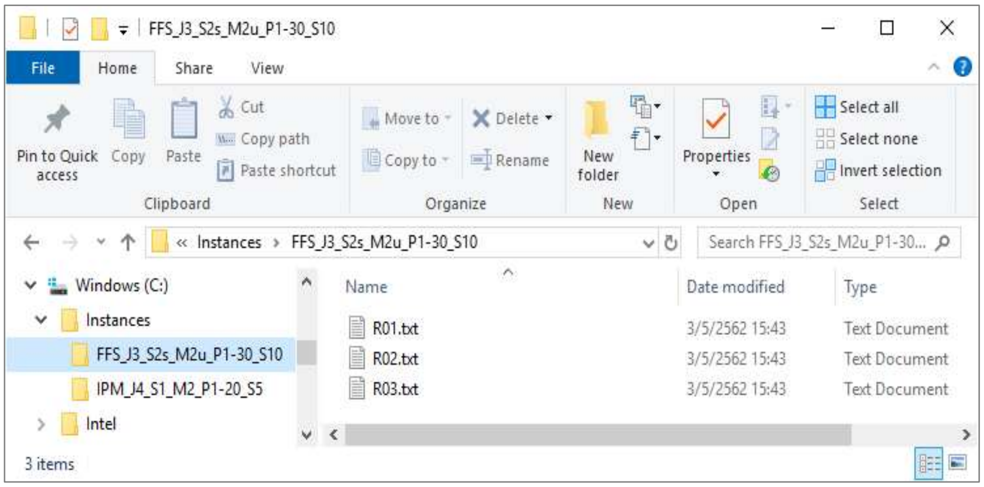
Component	Description	Notation / Example
1	Type of production shop	SM : Single Machine
		IPM : Identical Parallel Machines
		UFP : Uniform Parallel Machines
		URP : Unrelated Parallel Machines
		PFS : Pure Flow Shop
		GFS : General Flow Shop
		FFS : Flexible Flow Shop
		JS : Job Shop
		FJS : Flexible Job Shop
2	No. of jobs	J3 : These are 3 jobs.
3	No. of stages	S2 : There are 2 stages for each job.
		S2s : The maximum stages for each job is 2 (skipping in some stages is allowed).
4	No. of parallel machines	M2 : There are 2 parallel machines for each stage.
		M2u : The maximum parallel machines for each stage is 2 (number of machines for each stage are unequal).
5	Range of processing time	P1-30 : The processing times is in the range of 1 to 30.
6	SDST percentage	S10 : The maximum SDST value is 10% of max processing time.



Name	Date modified	Type	Size
FFS_J3_S2s_M2u_P1-30_S10	3/5/2562 16:17	File folder	
IPM_J4_S1_M2_P1-20_S5	3/5/2562 14:59	File folder	

Fig. 6. Example of a file folder

The text files for generated instances are kept in each file folder. Each file name indicates a replicate number. For example, the files R01.txt, R02.txt, and R03.txt in Fig. 7 refer to replicate numbers 13.



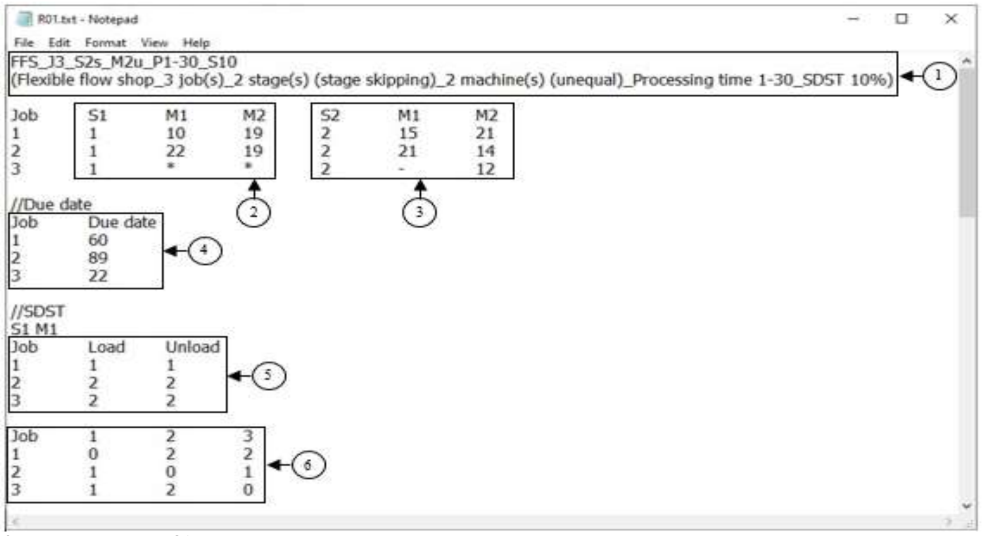
Name	Date modified	Type
R01.txt	3/5/2562 15:43	Text Document
R02.txt	3/5/2562 15:43	Text Document
R03.txt	3/5/2562 15:43	Text Document

Fig. 7. Example of a file name

C. Output Text File

IGSP provides a more comprehensive output text file than all of the IGs reviewed. Fig. 8 shows an example. The six parts of output text file are:

- Part 1 provides details of the generated data set, similarly to the file folder name.
- Part 2 provides processing times for each job (3 jobs) in Stage 1 for Machine 1, and 2. ‘*’ indicates the unequal stages option. In this example, it means that Stage 1 is not necessary for Job 3. Please note that each job must be processed during at least 1 stage.
- Part 3 provides processing times for each job (3 jobs) of Stage 2 for Machine 1 and 2. “-” indicates the unequal machines option. In this example, it means that there is only one machine (Machine 2) for Job 3, Stage 2.
- Part 4 provides the due date for each job. In this example, Job 3 is due in 22 units of time.
- Part 5 provides the LD and UD for each job. The LD is only used for the first job in the sequence, while the UD is only used for the last job in the sequence. For example, if the sequence on Machine 1 is Job 1 Job 3 Job 2, then the LD of Job 1, Stage 1 is 1 unit of time.
- Part 6 provides SDST. It is randomly generated, as explained in Section II. For example, the SDST is 2 units of time if Job 3 is processed after Job 1.



```
FFS_J3_S2s_M2u_P1-30_S10
(Flexible flow shop_3 job(s)_2 stage(s) (stage skipping)_2 machine(s) (unequal)_Processing time 1-30_SDST 10%)

Job   S1   M1   M2     S2   M1   M2
1     1    10   19     2    15   21
2     1    22   19     2    21   14
3     1    *    *      2    -    12

//Due date
Job   Due date
1     60
2     89
3     22

//SDST
S1 M1
Job   Load   Unload
1     1      1
2     2      2
3     2      2

Job   1   2   3
1     0   2   2
2     1   0   1
3     1   2   0
```

Fig. 8. Example of an output text file

IV. CONCLUSION AND RECOMMENDATIONS FOR FURTHER STUDY

This paper developed a new instance generator for scheduling problems named IGSP. It can generate hypothetical data set for various production shops, including new options reflecting real production scheduling environments (unequal stages of each job and unequal parallel machines of each stage). The user interface and output file obtained from IGSP are more comprehensive than the extant IGs. Researchers can use it to evaluate the performances of their scheduling algorithms and determine benchmark problem instances. At this point, IGSP can generate datasets for a single machine, parallel machines, flow shop, and job shop environments. Other production shops, such as an open shop, cellular shop, and assembly shop, will be developed for further study to increase the capacity of the IGSP. The IGSP can be freely downloaded from http://www.ie.kmutnb.ac.th/upload/IGSP/IGSP.zip. All user feedback is welcomed in the service of further developing its utility.

REFERENCES

[1] E. Vallada, R. Ruiz, and J. M. Framinan, "New hard benchmark for flowshop scheduling problems minimising makespan," *European Journal of Operational Research*, vol. 240, no. 3, pp. 666–677, 2015. doi: https://doi.org/10.1016/j.ejor.2014.07.033

[2] Q.-K. Pan, R. Ruiz, and P. Alfaro-Fernández, "Iterated search methods for earliness and tardiness minimization in hybrid flowshops with due windows," *Computers & Operations Research*, vol. 80, pp. 50–60, 2017. doi: https://doi.org/10.1016/j.cor.2016.11.022

[3] T. Yamada and R. Nakano, "A genetic algorithm applicable to large-scale job-shop problems," in *Parallel Problem Solving from Nature 2 (PPSN-II),* Brussels, Belgium, 1992.

[4] G. Minella, R. Ruiz, and M. Ciavotta, "A review and evaluation of multiobjective algorithms for the flowshop scheduling problem," *INFORMS Journal on Computing*, vol. 20, no. 3, pp. 451–471, 2008. doi: https://doi.org/10.1287/ijoc.1070.0258

[5] L. Fanjul-Peyro and R. Ruiz, "Scheduling unrelated parallel machines with optional machines and jobs selection," *Computers & Operations Research*, vol. 39, no. 7, pp. 1745–1753, 2012. doi: https://doi.org/10.1016/j.cor.2011.10.012

[6] A. Drexl, R. Nissen, J. H. Patterson, and F. Salewski, "ProGen - an instance generator for resource-constrained project scheduling problems with partially renewable resources and further extensions," *European Journal of Operational Research*, vol. 125, no. 1, pp. 59–72, 2000. doi: https://doi.org/10.1016/s0377-2217(99)00205-2

[7] R. Kolisch, C. Schwindt, and A. Sprecher, "Benchmark instances for project scheduling problems," in *Project Scheduling*. New York, Ny: Springer, 1999, pp. 197–212.

[8] E. Taillard, "Benchmarks for basic scheduling problems," *European Journal of Operational Research*, vol. 64, no. 2, pp. 278–285, 1993. doi: https://doi.org/10.1016/0377-2217(93)90182-m

[9] B. Wang, K. Huang, and T. Li, "Permutation flowshop scheduling with time lag constraints and makespan criterion," *Computers & Industrial Engineering*, vol. 120, pp. 1–14, 2018. doi: https://doi.org/10.1016/j.cie.2018.04.021

[10] H. Ye, W. Li, and A. Abedini, "An improved heuristic for no-wait flow shop to minimize makespan," *Journal of Manufacturing Systems*, vol. 44, pp. 273–279, 2017. doi: https://doi.org/10.1016/j.jmsy.2017.04.007

[11] M. Ciavotta, G. Minella, and R. Ruiz, "Multi-objective sequence dependent setup times permutation flowshop: A new algorithm and a comprehensive study," *European Journal of Operational Research*, vol. 227, no. 2, pp. 301–313, 2013. doi: https://doi.org/10.1016/j.ejor.2012.12.031

[12] M. Rauf, Z. Guan, S. Sarfraz, J. Mumtaz, E. Shehab, M. Jahanzaib, and M. Hanif, "A smart algorithm for multi-criteria optimization of model sequencing problem in assembly lines," *Robotics and Computer-Integrated Manufacturing*, vol. 61, no. 10, pp. 18–44, 2020. doi: https://doi.org/10.1016/j.rcim.2019.101844

[13] E. Demirkol, S. Mehta, and R. Uzsoy, "Benchmarks for shop scheduling problems," *European Journal of Operational Research*, vol. 109, no. 1, pp. 137–141, 1998. doi: https://doi.org/10.1016/s0377-2217(97)00019-2

[14] L. Fanjul-Peyro, F. Perea, and R. Ruiz, "Models and matheuristics for the unrelated parallel machine scheduling problem with additional resources," *European Journal of Operational Research*, vol. 260, no. 2, pp. 482–493, 2017. doi: https://doi.org/10.1016/j.ejor.2017.01.002

[15] P. Bratley, B. L. Fox, and L. E. Schrage, *A Guide to Simulation*. New York, NY: Springer-Verlag, 1983.

[16] S. Hasija and C. Rajendran, "Scheduling in flowshops to minimize total tardiness of jobs," *International Journal of Production Research*, vol. 42, no. 11, pp. 2289–2301, 2004. doi: https://doi.org/10.1080/00207540310001657595

[17] R. Ruiz and T. Stutzle, "An iterated greedy heuristic for the sequence dependent setup times flowshop problem with makespan and weighted tardiness objectives," *European Journal of Operational Research,*, vol. 187, pp. 1143–1159, 2008.