European Journal of Operational Research 64 (1993) 278–285
North-Holland

Benchmarks for basic scheduling problems

E. Taillard
École Polytechnique Fédérale de Lausanne, Département de Mathématiques, CH-1015 Lausanne, Switzerland

Received December 1989; revised March 1992

Abstract: In this paper, we propose 260 randomly generated scheduling problems whose size is greater than that of the rare examples published. Such sizes correspond to real dimensions of industrial problems. The types of problems that we propose are: the permutation flow shop, the job shop and the open shop scheduling problems. We restrict ourselves to basic problems: the processing times are fixed, there are neither set-up times nor due dates nor release dates, etc. Then, the objective is the minimization of the makespan.

Keywords: Combinatorial optimization; Scheduling; Benchmarks

Introduction

The types of problems discussed in this paper (permutation flow shop, job shop and open shop scheduling problems) have been widely studied in the literature using exact or heuristic methods, but a common comparison base is missing. We hope that this paper will fill a gap in this domain.

The three-field nomenclature described in Lawler et al. [9] names these problems $F \| C_{max}$, $J \| C_{max}$ and $O \| C_{max}$ respectively. They certainly belong to the most studied ones among scheduling problems. Let us describe them briefly.

There are n jobs that have to be performed on m unrelated machines; in our case, every job consists of m nonpreemptible operations. Every operation of a job uses a different machine during a given time and may wait before being processed.

For the permutation flow shop problem, the operations of every job must be processed on machines $1, \ldots, m$ in this order. Moreover, the processing order of the jobs on the machines is the same for every machine. The problem consists in finding a permutation of the n jobs that minimizes the makespan.

In the case of the job shop problem, any processing order of the jobs on the machines is allowed. For every job, the operations must be processed in a given order on the machines, but this order may differ according to the jobs.

For the open shop problem, every operation is assigned to a given machine but the order of the operations of every job is totally free.

The aim of this paper is to present unsolved problems whose size corresponds to the one of industrial problems. These problems must be easy to generate.

Generating interesting problems

As we do not know any method to solve exactly the problems we want to propose, we have used heuristic methods to get hopefully good solutions of these problems. These heuristic methods are based on taboo search techniques. Taboo search is described very generally in Glover [7] and one can find some of its practical applications to the flow shop sequencing problem in Taillard [13] and Widmer and Hertz [14], and to the job shop scheduling problem in Taillard [12].

Correspondence to: E. Taillard, École Polytechnique Fédérale de Lausanne, Département de Mathématiques, CH-1015 Lausanne, Switzerland email: TAILLARD@MAS65.EPFL.CH.

Taboo search is very easy to implement and can provides results that are better than those obtained by any other heuristic method described in [9] or in Applegate and Cook [1] if its parameters are well tuned.

In order to propose problems that are as difficult as possible (the most interesting ones) we have generated many instances of problems that we have 'solved' in a summary way with taboo search. Then, we have chosen the 10 problems that seemed to be the hardest ones and we have solved them once more, allowing our heuristic method to perform a higher number of iteration.

Obviously, the choice of the hardest problems is very subjective. We decided that a problem was interesting if the best makespan we found was far from a lower bound of the makespans and if many attempts to solve the problem (starting from various initial solutions) did not provide the same solution. Such a method enabled us to detect the simplest problems but we may not propose problems that have a local optimum with a large attraction basin.

The problems

The problems we propose are randomly generated with a good random number generator proposed in Bratley [4]. We recall its implementation so that this paper is self contained.

A problem will be entirely defined by the initial value of the seed of the random generator and by the way of generating it. For every type of problem, we give a simple manner of computing a lower bound of the makespan; in particular, this permits to verify the generation of the problem. For every size of problem, we give the total number of instances we have generated (summary resolution), the maximum number of iteration of taboo search that were done (long resolution) and the proportion of problems that were solved up to the lower bound, that is to say optimally. For every type and every size of problem, we give 10 instances. For each instance, we give the initial value of the random generator seed, the best value of the makespan we have found (i.e. an upper bound of the optimal makespan) and a lower bound of all the makespans.

The random number generator

Let us recall the implementation of the linear congruential generator we have used which is based on the recursive formula

$$X_{i+1} = (16\,807 X_i) \bmod (2^{31} - 1).$$

This implementation uses only 32-bit integers and provides a uniformly distribution sequence of numbers between 0 and 1 (not contained):

Step 0. Initial seed and constants:
X_0 $(0 < X_0 < 2^{31} - 1)$;
$a = 16\,807$, $b = 127\,773$, $c = 2\,836$, $m = 2^{31} - 1$.

Step 1. Modification of the seed:

$$k := \lfloor X_j / b \rfloor,$$

$$X_{i+1} := a(X_i \bmod b) - kc,$$

If $X_{i+1} < 0$, then let

$$X_{i+1} := X_{i+1} + m.$$

Step 2. New value of the seed: X_{i+1}.
Current value of the generator: X_{i+1}/m.

Below, we shall denote by $U(0, 1)$ the pseudo-random number that this generator provides. We have $0 < U(0, 1) < 1$ for every generated number.

We shall denote by $U[a, b]$ (with $a < b$, a and b integers) the integer number

$$\lfloor a + U(0, 1) \cdot (b - a + 1) \rfloor.$$

For every random integer generated, we have

$$a \le U[a, b] \le b$$

and every integer between a and b has the 'same' probability of being chosen. In order to implement the integer random procedure only with 32-bit integers, the problems have been chosen in such a way that one never has to deal with a seed X such that

$$\left\lfloor a + \frac{X \cdot (b - a + 1)}{m} \right\rfloor \neq \left\lfloor a + \frac{X}{\lfloor m/(b - a + 1) \rfloor} \right\rfloor.$$

Flow shop problems

There are in the literature some problems of this type; let us quote for example eight small and simple problems proposed in Carlier [5] and solved exactly in this reference.

The flow shop problems are characterized by the processing times d_{ij} of job j on machine i ($1 \le i \le m$, $1 \le j \le n$). We have generated the values of d_{ij} by the following way:
For $i = 1$ to m
  for $j = 1$ to n
    $d_{ij} := U[1, 99]$.

We propose problems with 5, 10 and 20 machines and from 20 to 500 jobs. We compute the lower bound of the makespan as presented below.

Let b_i be the minimum amount of time before machine i starts to work and a_i be the minimum amount of time that it remains inactive after its work up to the end of the operations, and let T_i be its total processing time. We have:

$$b_i = \min_j \left(\sum_{k=1}^{i-1} d_{kj} \right),$$

$$a_i = \min_j \left(\sum_{k=i+1}^{m} d_{kj} \right),$$

$$T_i = \sum_{j=1}^{n} d_{ij}.$$

Clearly, the optimal makespan $C^*_{\max}$ is greater than or equal to the maximum between the mini-

Time seed	UB	LB
20 jobs	5 machines	
873654221	1278	1232
379008056	1359	1290
1866992158	1081	1073
216771124	1293	1268
495070989	1235	1198
402959317	1195	1180
1369363414	1239	1226
2021925980	1206	1170
573109518	1230	1206
88325120	1108	1082
20 jobs	10 machines	
587595453	1582	1448
1401007982	1659	1479
873136276	1496	1407
268827376	1377	1308
1634173168	1419	1325
691823909	1397	1290
73807235	1484	1388
1273398721	1538	1363
2065119309	1593	1472
1672900551	1591	1356
20 jobs	20 machines	
479340445	2297	1911
268827376	2099	1711
1958948863	2326	1844
918272953	2223	1810
555010963	2291	1899
2010851491	2226	1875
1519833303	2273	1875
1748670931	2200	1880
1923497586	2237	1840
1829909967	2178	1900
50 jobs	5 machines	
1328042058	2724	2712
200382020	2836	2808
496319842	2621	2596
1203030903	2751	2740
1730708564	2863	2837
450926852	2829	2793
1303135678	2725	2689
1273398721	2683	2667
587288402	2554	2527
248421594	2782	2776

Time seed	UB	LB
50 jobs	10 machines	
1958948863	3037	2907
575633267	2911	2821
655816003	2871	2801
1977864101	3067	2968
93805469	3011	2908
1803345551	3021	2941
49612559	3124	3062
1899802599	3048	2959
2013025619	2910	2795
578962478	3100	3046
50 jobs	20 machines	
1539989115	3886	3480
691823909	3733	3424
655816003	3673	3351
1315102446	3755	3336
1949668355	3648	3313
1923497586	3719	3460
1805594913	3730	3427
1861070898	3737	3383
715643788	3772	3457
464843328	3791	3438
100 jobs	5 machines	
896678084	5493	5437
1179439976	5274	5208
1122278347	5175	5130
416756875	5018	4963
267829958	5250	5195
1835213917	5135	5063
1328833962	5247	5198
1418570761	5094	5038
161033112	5448	5385
304212574	5328	5272
100 jobs	10 machines	
1539989115	5776	5759
655816003	5362	5345
960914243	5679	5623
1915696806	5820	5732
2013025619	5491	5431
1168140026	5308	5246
1923497586	5600	5523
167698528	5640	5556
1528387973	5891	5779
993794175	5856	5830

Time seed	UB	LB
100 jobs	20 machines	
450926852	6330	5851
1462772409	6320	6099
1021685265	6364	6099
83696007	6331	6072
508154254	6405	6009
1861070898	6487	6144
26482542	6379	5991
444956424	6514	6084
2115448041	6386	5979
118254244	6534	6298
200 jobs	10 machines	
471503978	10872	10816
1215892992	10500	10422
135346136	10956	10886
1602504050	10893	10794
160037322	10537	10437
551454346	10347	10255
519485142	10882	10761
383947510	10754	10663
1968171878	10465	10348
540872513	10727	10616
200 jobs	20 machines	
2013025619	11393	10979
475051709	11445	10947
914834335	11522	11150
810642687	11461	11127
1019331795	11427	11132
2056065863	11368	11085
1342855162	11536	11194
1325809384	11544	11126
1988803007	11424	10965
765656702	11548	11122
500 jobs	20 machines	
1368624604	26316	25922
450181436	26807	26353
1927888393	26626	26320
1759567256	26642	26424
606425239	26509	26181
19268348	26654	26401
1298201670	26575	26300
2041736264	26794	26429
379756761	26241	25891
28837162	26662	26315

Figure 1. Instances of flowshop problems

mum amount of time required by the machines and the minimum of time required for each job:

$$LB = \max\left\{ \max_i (b_i + T_i + a_i),\ \max_j \left(\sum_{i=1}^{m} d_{ij} \right) \right\} \le C^*_{\max}.$$

This lower bound is easy to compute and we conjecture that for fixed m

$$\lim_{n\to\infty} \text{Prob}(C^*_{\max} = \text{LB}) = 1.$$

For every size of problem we give the following information (Table 1):

Nb jobs: The number of jobs.
Nb machines: The number of machines.
Nb instances: The total number of problems generated.
LB reached: The proportion of problems for which we found a solution for which the makespan was equal to the lower bound (or equal to the lower bound augmented by 2% for the 500-job 20-machine problems).
Nb iterations: The maximum number of iterations performed by taboo search (long resolution).
Nb resolutions: The number of attempts to solve the problem from various initial solutions (long resolution).

Then we give ten instances for every size of problem with the following information (Figure 1):

Time seed: The initial value of the random generator's seed.
UB: An upper bound of the optimal makespan (the best value we got).
LB: A lower bound of the makespans.

As the aim is to give an upper bound as good as possible but not a fast solving method, the computation time does not have much importance. However, let us mention that an iteration of taboo search needs about $4 \cdot 10^{-6} n^2 m$ seconds on a 'Silicon Graphics' personal workstation (10 Mips).

Table 1
Flow shop problems

Nb jobs	Nb machines	Nb instances	LB reached (%)	Nb iterations	Nb resolutions
20	5	100	35	10^4	3
20	10	100	1	10^4	3
20	20	100	0	$2 \cdot 10^4$	3
50	5	70	41	$5 \cdot 10^3$	3
50	10	70	3	10^4	3
50	20	70	0	$5 \cdot 10^4$	3
100	5	10000	54	$2 \cdot 10^3$	4
100	10	50	6	$2 \cdot 10^4$	3
100	20	50	0	10^4	3
200	10	300	28	$4 \cdot 10^3$	3
200	20	25	0	$4 \cdot 10^3$	10
500	20	100	14 [a]	$4 \cdot 10^3$	5

[a] The value reached for this size was less than or equal to 1.02 times the lower bound.

Job shop problems

In the literature, we may find instances of small problems in Lawrence [10] and Muth and Thompson [11]; most of the optimal values of these problems are given in [1] or Carlier and Pinson [6]. Some very good solution values of Lawrence's problems are also given in [12]. We can consider that problems up to ten machines may be solved satisfactorily with existing methods. This is why we propose problems with 15 and 20 machines and from 15 to 100 jobs.

The processing time d_{ij} of the i-th operation of job j ($1 \le i \le m$, $1 \le j \le n$) is obtained as follows:

For $j = 1$ to n
  For $i = 1$ to m
    $d_{ij} := U[1, 99]$.

The machine M_{ij} on which the i-th operation of job j has to be performed is given by the following procedure:

Step 0.
  $M_{ij} := i$ ($1 \le i \le m$, $1 \le j \le n$).

Table 2
Job shop problems

Nb jobs	Nb machines	Nb instances	Lb reached (%)	Nb iterations	Nb resolutions
15	15	50	0	$7 \cdot 10^5$	4
20	15	50	4	10^6	3
20	20	50	0	10^7	4
30	15	50	26	$3 \cdot 10^6$	4
30	20	50	0	$2 \cdot 10^6$	4
50	15	100	78	$3 \cdot 10^6$	4
50	20	26	27	$5 \cdot 10^5$	4
100	20	100	97	$5 \cdot 10^5$	3

Step 1.
For $j = 1$ to n
For $i = 1$ to m
Swap M_{ij} and $M_{U[i,m]j}$.

Let us note the use of another initial seed for the choice of the machines: machine seed.

An instance of a small open shop problem, obtained with the same procedures, is given extensively in Table 4.

The lower bound is computed as previously but b_i, a_i and T_i are defined as follows:

$$b_i = \min_j \left(\sum_{k=1}^{k'-1} d_{kj} \right),$$

with k' such that $M_{k'j} = i$;

$$a_i = \min_j \left(\sum_{k=k'+1}^{m} d_{kj} \right),$$

with k' such that $M_{k'j} = i$;

$$T_i = \sum_{j=1}^{n} d_{k'j},$$

with k' such that $M_{k'j} = i$.

We conjecture again that this bound is tight if $n/m \to \infty$, because we have always found an optimal schedule if $n/m \geq 6$, considering more than 2000 problems whose size was varying from 20

Time seed	Machine seed	UB	LB
15 jobs	15 machines		
840612802	398197754	1231*	1005
1314640371	386720536	1252	953
1227221349	316176388	1223	1036
342269428	1806358582	1181	973
1603221416	1501949241	1234	940
1357584978	1734077082	1243	1134
44531661	1374316395	1228	1103
302545136	2092186050	1221	980
1153780144	1393392374	1289	1020
73896786	1544979948	1261	940
20 jobs	15 machines		
533484900	317419073	1376	1254
1894307698	1474268163	1381	1267
874340513	509669280	1367	1243
1124986343	1209573668	1355	1329
1463788335	529048107	1366	1163
1056908795	25321885	1371	1211
195672285	1717580117	1480	1306
961965583	1353003786	1432	1315
1610169733	1734469503	1361	1202
532794656	998486810	1373	1213
20 jobs	20 machines		
1035939303	773961798	1663	1217
5997802	1872541150	1626	1314
1357503601	722225039	1574	1248
806159563	1166962073	1660	1284
1902815253	1879990068	1598	1256
1503184031	1850351876	1679	1245
1032645967	99711329	1704	1403
229894219	1158117804	1626	1387
823349822	108033225	1635	1352
1297900341	489486403	1614	1277
30 jobs	15 machines		
98640593	1981283465	1770	1764
1839268120	248890888	1853	1774
573875290	2081512253	1855	1733
1670898570	788294565	1851	1828
1118914567	1074349202	2007	1754
178750207	294279708	1844	1777
1549372605	596993084	1822	1771
798174738	151685779	1714	1673
553410952	1329272528	1824	1764
1661531649	1173386294	1723	1608

Time seed	Machine seed	UB	LB
30 jobs	20 machines		
1841414609	1357882888	2064	1850
2116959593	1546338557	1983	1761
796392706	1230864158	1896	1710
532496463	254174057	2031	1820
2020525633	978943053	2032	1785
524444252	185526083	2057	1940
1569394691	487269855	1947	1751
1460267840	1631446539	2005	1770
198324822	1937476577	2013	1758
38071822	1541985579	1973	1678
50 jobs	15 machines		
17271	718939	2760*	2760
660481279	449650254	2756*	2756
352229765	949737911	2717*	2717
1197518780	166840558	2839*	2813
1376020303	483922052	2689	2679
2106639239	955932362	2781*	2781
1765352082	1209982549	2943*	2943
1105092880	1349003108	2885*	2885
907248070	919544535	2655*	2655
2011630757	1845447001	2723*	2723
50 jobs	20 machines		
8493988	2738939	2921	2868
1991925010	709517751	3002	2848
342093237	786960785	2835	2755
1634043183	973178279	2775	2697
341706507	286513148	2800	2725
320167954	1411193018	2914	2845
1089696753	298068750	2895	2812
433032965	1589656152	2835	2764
615974477	331205412	3097	3071
236150141	592292984	3075	2995
100 jobs	20 machines		
302034063	1203569070	5464*	5464
1437643198	1692025209	5181*	5181
1792475497	1039908559	5568*	5552
1647273132	1012841433	5339*	5339
696480901	1689682358	5392*	5392
1785569423	1092647459	5342*	5342
117806902	739059626	5436*	5436
1639154709	1319962509	5394*	5394
2007423389	749368241	5358*	5358
682761130	262763021	5213	5183

Figure 2. Instances of job shop problems

Time seed	Machine seed	UB	LB
4 jobs	4 machines		
1166510396	164000672	193*	186
1624514147	1076870026	236*	229
1116611914	1729673136	271*	262
410579806	1453014524	250*	245
1036100146	375655500	295*	287
597897640	322140729	189*	185
1268670769	556009645	201*	197
307928077	421384574	217*	212
667545295	485515899	261*	258
35780816	492238933	217*	213
5 jobs	5 machines		
527556884	1343124817	300*	295
1046824493	1973406531	262*	255
1165033492	86711717	323*[a]	321
476292817	24463110	310*	306
1181363416	606981348	326*[a]	321
897739730	513119113	312*	307
577107303	2046387124	303*[a]	298
1714191910	1928475945	300*	292
1813128617	2091141708	353*	349
808919936	183753764	326*	321
7 jobs	7 machines		
1840686215	1827454623	438	435
1026771938	1312166461	449	443
609471574	670843185	479	468
1022295947	398226875	467	463
1513073047	1250759651	419	416
1612211197	95606345	460	451
435024109	1118234860	435	422
1760865440	1099909092	426	424
122574075	10979313	460	458
248031774	1685251301	400	398

Time seed	Machine seed	UB	LB
10 jobs	10 machines		
1344106948	1868311537	645[b]	637
425990073	1111853152	588*[b]	588
666128954	1750328066	611[b]	598
442723456	1369177184	577*[b]	577
2033800800	1344077538	641[b]	640
964467313	1735817385	538*[b]	538
1004528509	967002400	623	616
1667495107	818777384	596[b]	595
1806968543	1561913259	595*[b]	595
938376228	344628625	602[b]	596
15 jobs	15 machines		
1561423441	1787167667	937*[b]	937
204120997	213027331	918*[b]	918
801158374	1812110433	871*[b]	871
1502847623	1527847153	934*[b]	934
282791231	1855451778	950[b]	946
1130361878	849417380	933*[b]	933
379464508	944419714	891*[b]	891
1760142791	1955448160	893*[b]	893
1993140927	179408412	908[b]	899
1678386613	1567160817	902*[b]	902
20 jobs	20 machines		
957638	9237185	1155*[b]	1155
162587311	1489531109	1244[b]	1241
965299017	1054695706	1257*[b]	1257
1158457671	1499999517	1248*[b]	1248
1191143707	1530757746	1256*[b]	1256
1826671743	901609771	1209[b]	1204
1591533998	1146547719	1294*[b]	1294
937297777	92726463	1173[b]	1169
687896268	1731298717	1289*[b]	1289
687034842	684013066	1241*[b]	1241

Figure 3. Instances of open shop problems

jobs, 2 machines to 150 jobs, 15 machines, passing by 500 jobs, 4 machines.

The time needed to perform one iteration of taboo search is about $20 \cdot 10^{-6} nm$ seconds on 'Silicon Graphics' workstation. Table 2 and Figure 2 are analogous to Table 1 and Figure 1, but, for job shop problems, we give in addition *machine seed* in Figure 2. Stars in Figure 2 indicate optimum values.

Open shop problems

We do not know instances of such problems in the literature. This is why we give problems of

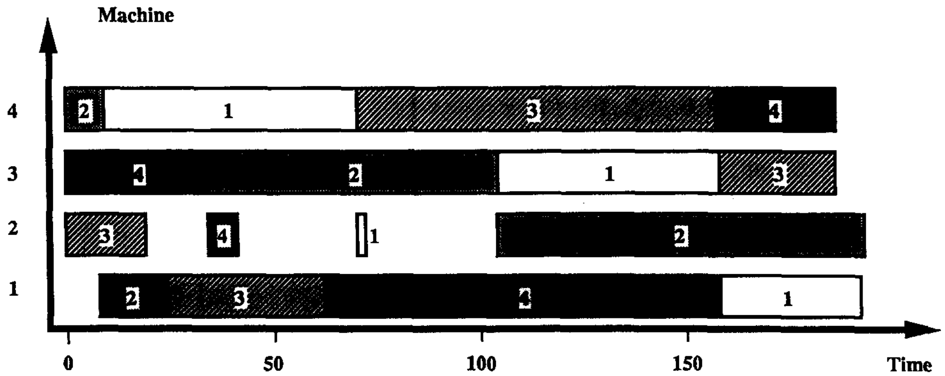

Figure 4. An optimum schedule for the first problem of Table 4

small size. These problems are obtained using exactly the same procedures as those used for the job shop problems, and the lower bound of the makespans corresponds to the maximum amount of time that a job or a machine requires, i.e.

$$\text{LB} = \max\left\{ \max_i \left(\sum_j d_{ij} \right), \max_j \left(\sum_{i,k \mid M_{ik}=j} d_{ik} \right) \right\}.$$

Because one has to choose the order of the operations of a job, one can find very often an optimal schedule, except for the problems in which the number of jobs is about the number of machines. In this case, either an optimal solution is easily reached, or the problem is harder than job shop problems of the same size.

For problems with $n \gg m$, we have observed empirically that the mean complexity of taboo search applied to open shop problems, $O(n^{2.37} \cdot m^{3.69})$, is lower than the complexity of taboo search applied to job shop problems, $O(n^{2.50} \cdot m^{3.81})$.

In Table 4, we describe extensively the first 4-job 4-machine problem we propose, i.e. the

```
function unif(var seed: integer; low, high: integer): integer;
const
   m =2147483647; a =16807;
   b =127773;       c =2836;
var
   k:            integer;
   value_0_1: double; (* floating point coded on 64 bits *)
begin
   k := seed div b;
   seed := a * (seed mod b) - k * c;
   if seed < 0 then seed := seed + m;
   value_0_1 := seed/m;
   unif := low + trunc(value_0_1 * (high - low + 1))
end;
procedure generate_flow_shop( var time_seed:            integer;
                                  nb_jobs, nb_machines: integer;
                                  var d:                matrix);
(* type matrix =array[1..20, 1..500] of integer; must be declared above *)
var i, j: integer;
begin
   for i := 1 to nb_machines do
      for j := 1 to nb_jobs do
         d[i, j] := unif(time_seed, 1, 99)
end;
procedure generate_job_and_open_shop( var time_seed, machine_seed: integer;
                                          nb_jobs, nb_machines:    integer;
                                          var d, M:                matrix);
var i, j: integer;
   procedure swap(var a, b: integer);
   var temp: integer;
   begin
      temp := a; a := b; b := temp
   end;
begin
   for j := 1 to nb_jobs do
      for i := 1 to nb_machines do
         d[i, j] := unif(time_seed, 1, 99);
   for j := 1 to nb_jobs do
      for i := 1 to nb_machines do
         M[i, j] := i;
   for j := 1 to nb_jobs do
      for i := 1 to nb_machines do
         swap(M[i, j], M[unif(machine_seed, i, nb_machines), j])
end;
```

Figure 5. Pascal code for the generation of scheduling problems

Table 3
Open shop problems

Nb jobs	Nb machines	Nb instances	LB reached (%)	Nb iterations	Nb resolutions
4	4	50000	98.5	10^5	5
5	5	45000	99.7	$5 \cdot 10^5$	4
7	7	1000	94	10^6	5
10	10	300	89	$2 \cdot 10^6$	5
15	15	40	52	$3 \cdot 10^5$	3
20	20	25	24	$3 \cdot 10^6$	3

processing times d_{ij} of the operation i of job j and its associated machine M_{ij}. We give an optimum schedule of this problem in the Gantt chart of Figure 4.

The time needed by taboo search to perform one iteration is about $23 \cdot 10^{-6} nm$ seconds. Tables 3 and Figure 3 are analogous to Table 2 and Figure 2. Stars in Figure 3 indicate optimum values; a) indicates solution values issued from Kleinau [8] who proves optimality for 4×4 and 5×5 open shop problems; b) indicates solution values issued from Bräsel et al. [3].

Finally, we give in Figure 5 the procedures, written in Pascal language, for generating the scheduling problems described in this paper.

Concluding remarks

We hope that the problems that we propose will constitute a comparison base for future resolution methods.

Everyone may send us his own results about these problems, specifying whether his solutions are proved optimal or not, in order to update the best solutions known. The data included in Figures 1 to 3 are available via e-mail in the OR-Library created by Beasley [2].

Table 4
The first instance of the 4-job 4-machine open shop problem

Operation i	Job j [a]				Job j [b]			
	1	2	3	4	1	2	3	4
1	54	9	38	95	3	4	1	1
2	34	15	19	34	1	1	2	3
3	61	89	28	7	4	2	3	2
4	2	70	87	29	2	3	4	4

[a] Processing times d_{ij}.
[b] Machines M_{ij}.

References

[1] Applegate, D., and Cook, W., "A computational study of the job-shop scheduling problem", *ORSA Journal on Computing* 3 (1991) 149–156.

[2] Beasley, J.E., "OR-Library: Distributing test problems by electronic mail", *Journal of the Operational Research Society* 41 (1990) 1069–1072.

[3] Bräsel, H., Tautenhahn, T., and Werner, F., "Constructive heuristic algorithms for the open shop problem", Math 21/91, Institut für Mathematische Optimierung, Technische Universität 'Otto von Guericke', Magdeburg, 1991.

[4] Bratley, P., Fox, B.L., and Schrage, L.E., "A guide to Simulation", Springer-Verlag, New York, 1983.

[5] Carlier, J., "Problèmes d'ordonnancement à contraintes de ressources: Algorithmes et complexité", Méthodologie et architecture des systèmes informatiques, Institut de programmation, Université P.&M. Curie, Paris, 1984.

[6] Carlier, J., and Pinson, E., "An algorithm for solving the job-shop problem", *Management Science* 35/2 (1989) 164–176.

[7] Glover, F., "Tabu Search – Part I", *ORSA Journal on Computing* 1 (1989) 190–206.

[8] Kleinau, U., "On some new methods for solving machine scheduling problems", Math 16/91, Institut für Mathematische Optimierung, Technische Universität 'Otto von Guericke', Magdeburg, 1991.

[9] Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G., and Shmoys, D.B., "Sequencing and scheduling", Report BS-R89xx, Centre for Mathematics and Computer Science, Amsterdam, Netherlands, 1989.

[10] Lawrence, S., "Resource constrained project scheduling: An experimental investigation of heuristic scheduling techniques (supplement)", Graduate School of Industrial Administration, Carnegie Mellon University, 1982.

[11] Muth, J.F., and Thompson, G.L., "Industrial scheduling", Prentice-Hall, Englewood Cliffs, NJ 1963, 225–251.

[12] Taillard, E., "Parallel taboo search technique for the job shop scheduling problem", ORWP 89/11, DMA, École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, 1989.

[13] Taillard, E., "Some efficient heuristic methods for the flowshop sequencing problem", *European Journal of Operational Research* 47 (1990) 65–74.

[14] Widmer, M., and Hertz, A., "A new heuristic for the flowshop sequencing problem", *European Journal of Operational Research* 41 (1989) 186–193.