A Single-Machine Deteriorating Job Scheduling Problem of Minimizing the Makespan with Release Times

Wen-Chiung Lee, Chin-Chia Wu, and Yu-Hsiang Chung

Abstract

In this paper, we study a single-machine deteriorating job scheduling problem with job release times where the objective is to minimize the makespan. The problem is known to be strongly NP-complete. Therefore, we first develop a branch-and-bound algorithm incorporating with several dominance properties and a lower bound to derive the optimal solution. Then, we propose two heuristic algorithms for the near-optimal solutions. Finally, we conduct some computational experiments to evaluate the performance of the proposed algorithms.

***Keywords*:** single machine; deteriorating jobs; makespan; release time

1. Introduction

For many years, job processing times are assumed to be known and fixed from the first job to be processed until the last job to be completed. However, there are many situations in which a job that is processed later consumes more time than the same job when processed earlier. Examples can be found in the fire fighting, emergency surgery, machine maintenance, and cleaning assignment where any delay in processing a job is penalized and often implies additional time for accomplishing the job. Scheduling in this setting is known as deteriorating job scheduling problems and has received increasing attention in recent years.

Gupta and Gupta [1] and Browne and Yechiali [2] were the pioneers that introduced the deteriorating job scheduling problems. They constructed models where the processing time of a job is a function of its starting time. Since then, many researchers have devoted to this vivid area. For instance, Mosheiov [3] considered the simple linear deterioration model. He showed that the problems of minimizing the makespan, total flow time, sum of weighted completion times, total lateness, number of tardy jobs, maximum lateness, and maximum tardiness are polynomially solvable. Cheng and Ding [4] examined the single machine makespan scheduling problems involving starting time dependent tasks with release times and linearly increasing/decreasing processing rates. They solved a series of models in the literature, and further gave a sharp boundary delineating the complexity of the problems. Bachman and Janiak [5] proved that the maximum lateness problem is NP-hard if the actual processing time is a linear function of its starting time. Under the same model, Bachman *et al*. [6] proved that minimizing the total weighted completion time is NP-hard. Under the decreasing linear deterioration model, Wang and Xia [7] provided optimal solutions for single-machine problems of minimizing the makespan, maximum lateness, maximum cost and number of tardy jobs. Wu *et al*. [8] studied the total weighted completion time problem on a single machine. In addition, Janiak and Kovalyov [9] studied an important problem of scheduling jobs executed by human resources in a contaminated area. The specificity of this problem is that the dynamics of the harmful factor should be taken into account as well as the norms of organism recovery in rest periods. They also constructed two polynomial time algorithms for the both cases-- with and without precedence constraints. Lee *et al*. [10] studied the two-machine makespan problem while Wu and Lee [11] considered the two-machine total completion time problem.

On the other hand, there are some situations in which the job will require extra time for successful accomplishment if certain maintenance procedures fail to be completed prior to a pre-specified deadline. Kubiak and van de Velde [12] considered a single machine problem of scheduling independent jobs to minimize the makespan, and they showed the problem is NP-complete in the ordinary sense. Cheng and Ding [13] assumed that the actual processing time p_i is a step function of its starting time and showed the makespan problem is NP-complete in the ordinary sense, and the total completion time problem is NP-complete. They further identified a variety of solvable cases for some related problems. Researchers have formulated the deterioration phenomenon into different models and solved different problems for various criteria. Alidaee and Womer [14] surveyed the rapidly growing literature, while Cheng *et al*. [15] gave a detailed review of scheduling problems with deteriorating jobs.

However, most of the works assume that the jobs are available at all times. To the best of our knowledge, Cheng and Ding [4] were the only authors that considered deteriorating job scheduling problems with release times. They considered a family of scheduling problems for a set of starting time dependent tasks with release times and linearly increasing or decreasing processing rates on a single

Manuscript received November 20, 2007. This paper is support by NSC under grant number NSC 96-2628-E-035-002.

Wen-Chiung Lee is with Feng Chia University, Taichung, Taiwan, ROC. (phone: 886-4-2456-7250x4016; e-mail: wclee@fcu.edu.tw).

Chin-Chia Wu is with Feng Chia University, Taichung, Taiwan, ROC (e-mail: cchwu@fcu.edu.tw).

Yu- Hsiang Chung is with Feng Chia University, Taichung, Taiwan, ROC.

machine to minimize the makespan. In particular, they showed that the makespan problem with identical basic processing time, arbitrary release times and increasing processing rates is strongly NP-complete. However, no reports on developing the exact solution or providing the approximately solutions are found. In this paper, we will investigate the model where each job *j* has a common basic processing time *a*, a deterioration rate b_j, and a release time r_j. We will develop a branch-and-bound and two heuristic algorithms to obtain the optimal and near-optimal solutions, respectively. The remainder of this paper is organized as follows. The problem formulation is given in the next section. The elimination rules and a lower bound for the problem are presented in Section 3. In Section 4 we develop two heuristic algorithms to solve the proposed problem. In Section 5, we provide the results of the computational experiment. The conclusion is given in the last section.

2. Problem formulation

The problem description is given as follows. Let $N = \{1,...,n\}$ be the set of jobs to be scheduled. Each job *j* has a basic processing time *a*, a deterioration rate b_j, and a release time r_j. The actual processing time of J_j is $p_j = a + b_j t$, where *t* is the starting time for J_j. Moreover, let $C_j(S)$ denote the completion time for J_j under schedule S. The objective of this paper is to find an optimal schedule S^* such that

$$\max\{C_1(S^*), C_2(S^*), \ldots, C_n(S^*)\} \le \max\{C_1(S), C_2(S), \ldots, C_n(S)\}$$

for any schedule *S*.

3. Elimination rules and lower bound

Cheng and Ding [4] showed that the problem under consideration is strongly NP-complete. Therefore, the branch-and-bound method is a good way to derive the optimal solution. To facilitate the search process, some elimination rules are needed. Suppose that *S* and S' are two job schedules where the difference between *S* and S' is a pairwise interchange of two adjacent jobs *i* and *j*. That is, $S = (\pi\ i\ j\ \pi')$ and $S' = (\pi\ j\ i\ \pi')$, where π and π' denote the partial sequences. Furthermore, let *A* denote the completion time of the last job in π. To show that *S* dominates S', it suffices to show that $C_j(S) \le C_i(S')$.

Property 1. If $r_i \le A$ and $(1+b_i)A + a \le r_j$, then S dominates S'.

Proof: From $A \ge r_i$, it implies that

$C_i(S) = (1+b_i)A + a$.

Since $(1+b_i)A + a \le r_j$, we have

$C_j(S) = (1+b_j)r_j + a$.

From $r_i \le A$ and $(1+b_i)A + a \le r_j$, it implies that $r_i \le r_j$. Thus,

$C_j(S') = (1+b_j)r_j + a$,

and

$C_i(S') = (1+b_i)(1+b_j)r_j + (1+b_i)a + a$.

Since $b_i > 0$, we have

$C_j(S) < C_i(S')$.

Therefore, S dominates S'.

The proofs of Properties 2 to 6 are omitted since they are similar to that of Property 1.

Property 2. If $r_i \le A \le r_j$, $(1+b_i)A + a \ge r_j$, and $b_j < b_i$, then S dominates S'.

Property 3. If $A \ge \max\{r_i, r_j\}$ and $b_i > b_j$, then S dominates S'.

Property 4. If $r_j \le A \le r_i$, $(1+b_j)A + a \ge r_i$, and $a(b_j - b_i) < (1+b_i)(1+b_j)(A - r_i)$, then S dominates S'.

Property 5. If $A \le r_i \le r_j$, $b_i > b_j$, and $(1+b_i)r_i + a > r_j$, then S dominates S'.

Property 6. If $A \le r_i$ and $(1+b_i)r_i + a \le r_j$, then S dominates S'.

To further facilitate the searching process, a theorem is used in the branch-and-bound algorithm to determine the order of unscheduled jobs. Let (π, π^c) be a sequence of jobs where π is the scheduled part consisted of *k* jobs and π^c is the unscheduled part. In addition, let (π, π^*) be a sequence in which the unscheduled jobs in π^* are arranged in non-increasing order of deterioration rates b_i. For simplicity, we use the symbol [ ] to signify the order of jobs in a sequence.

Theorem 1. If $C_{[k]} > \max_{j \in N}\{r_j\}$, then sequence (π, π^*) dominates sequences of any other types of (π, π^c).

3.1 Lower bound

In this subsection, we establish a lower bound to curtail the branching tree. Suppose that *PS* is a partial schedule in which the order of the first *k* jobs is determined and *S* is a complete schedule obtained from *PS*. By definition, the completion time for the (*k*+1)th job is

$$C_{[k+1]} = (1+b_{[k+1]})\max\{C_{[k]}, r_{[k+1]}\} + a \ge (1+b_{[k+1]})C_{[k]} + a.$$

Similarly, the completion time for the (*k*+r)th position job is

$$C_{[k+r]} = (1+b_{[k+r]})\max\{C_{[k+r-1]}, r_{[k+r]}\} + a \ge \prod_{i=1}^{r}(1+b_{[k+i]})C_{[k]} + \sum_{j=1}^{r-1}\prod_{i=r-j+1}^{r}(1+b_{[k+i]})a + a, \quad (1)$$

where $k + r = n$. In equation (1), the first term is constant, and the second term is an increasing function of *j*. Thus, it is minimized by arranging the deterioration rates in a non-increasing order for the unscheduled jobs. Therefore, the lower bound can be obtained as in the following way.

$$LB = \prod_{i=1}^{r}(1+b_{[k+i]})C_{[k]} + \sum_{j=1}^{r-1}\prod_{i=1}^{j}(1+b_{(k+i)})a + a, \quad (2)$$

where $b_{(k+1)} \le b_{(k+2)} \le \ldots \le b_{(k+r)}$ denotes the deterioration rates in the unscheduled part.

4. The heuristic algorithms

An alternative approach to solve an NP-hard problem is to provide the heuristic algorithm. In this section, two heuristic algorithms are presented. The main idea of the first algorithm is to choose the job with the smallest release time. However, the schedule might be affected by the deterioration rates, which in turn yields a larger value of makespan. To overcome this problem, the second heuristic algorithm is proposed. Instead of only choosing the job with the smallest release time among all the unscheduled jobs, we choose the smallest value of release time and deterioration rates from the jobs already released. The steps of the proposed algorithms are described as follows.

Heuristic Algorithm 1 (HA_1)

***Step* 1**: Let $N=\{1,\ldots,n\}$ and $k=1$.

***Step* 2**: Choose job j with the smallest release time from N and place the job on the kth position. Compute the completion time for the kth job, say $C_{[k]}$. Delete job j from N.

***Step* 3:** If $C_{[k]} > \max_{j\in N}\{r_j\}$, then go Step 5, otherwise, go to Step 4.

***Step* 4**: Let $k=k+1$. If $k \le n$, go to Step 2.

***Step* 5:** Arrange the unscheduled jobs in a non-increasing order of the deterioration rates and go to Step 6.

***Step 6*:** Output the final solution.

Heuristic Algorithm 2 (HA_2)

***Step* 1**: Let $N=\{1,\ldots,n\}$, $C_{[0]}=0$, and $k=1$.

***Step* 2**: Choose job j which has the smallest value of $(1+b_j)\max\{C_{[k-1]}, r_j\}+a$ in N and place it on the kth position. Compute the completion time for the kth job, say $C_{[k]}$. Delete job j from N.

***Step* 3**: Let $k=k+1$. If $k \le n$, then go to Step 2. Otherwise, go to Step 4.

***Step* 4:** Output the final solution.

To further refine the quality of the proposed heuristics, some neighborhood search movements are necessary. In this paper, we use the pairwise interchange, extraction and backward-shifted, extraction and forward-shifted movement in our neighborhood search since they improve the quality of the solution significantly.

5. Computational experiment

A computational experiment is conducted to evaluate the performance of the branch-and-bound and the heuristic algorithms. The programs are coded in Fortran 90 and run on a Pentium 4 personal computer. The basic processing times take the values of 5 and 10, and the release times are generated from a uniform distribution between 0 to $50.5n\lambda$, where n is the job number and λ is a factor of the range of the release times.

The computational experiment consists of two parts. In the first part, the branch-and-bound and the heuristic algorithms are conducted with three different job numbers (n= 20, 30 and 40) associated with ten different ranges of λ (λ=0.2, 0.4, 0.6, 0.8, 1.0, 1.25, 1.5, 1.75, 2.0, and 3.0). The deterioration rates are generated from a uniform distribution between 0 and 0.25. These tests are designed to study the effects of the dominance rules and the lower bound, and the accuracy of the heuristic algorithms. The average and maximum number of nodes and the average and maximum execution time (in seconds) are reported for the branch-and-bound algorithms. For the heuristic algorithms, the average and maximum error percentages are recorded. The error percentage of the solution produced by the heuristic algorithm is calculated as

$$(V - V^*)/V^* \times 100\%,$$

where V is the value of the makespan generated from the heuristic method and V^* is the optimal value of the makespan obtained from the branch-and-bound method. In addition, the performance of the heuristic $BH = \min\{HA_1, HA_2\}$ is also reported. As a consequence, 60 experimental conditions are examined, and 100 replications are randomly generated for each condition. The results are summarized in Table 1. It is observed that the number of nodes and the execution time grow exponentially as the job size increases. The most time-consuming case takes about 20 minutes and the maximal number of nodes is over 1.3×10^6 when n= 40. As λ increases, the execution time and the number of nodes increase for a fixed number of jobs. In addition, the problems are easier to solve as the values of the basic processing times increase. This is due to the fact that it is easier to surpass the maximal release time, and Theorem 1 is more powerful in that case. Both heuristic algorithms perform well in terms of the average error percentages. Overall, the performance of HA_1 heuristic is better than that of HA_2, especially for small values of λ. Moreover, the average error percentages of both heuristics tend to decrease as λ increases when the job number is fixed. In addition, the performance of BH heuristic is much superior to those of HA_1 and HA_2, which means there is no clear dominance relation between the performances of HA_1 and HA_2. Thus, BH is recommended for small job-sized problems.

Table 1. The Performance of Branch-and-Bound and Heuristic Algorithms for $b \sim U(0, 0.25)$.

			Number of Nodes		CPU Times		Error Percentage					
							$\overline{HA_1}$		HA_2		$\overline{BH}$	
n	a	λ	mean	max	mean	max	mean	max	mean	max	mean	max
20	5	0.20	134.8	4780	0.017	0.453	0.058	2.548	1.117	21.669	0.037	2.548
		0.40	621.9	24982	0.055	2.125	0.010	0.930	0.607	23.052	0.009	0.930
		0.60	332.3	7019	0.029	0.469	0.007	0.669	1.054	38.622	0.000	0.000
		0.80	1728.5	108969	0.132	7.641	0.000	0.000	0.252	11.447	0.000	0.000
		1.00	1972.1	104518	0.102	5.188	0.004	0.159	0.386	9.194	0.001	0.082
		1.25	800.1	34002	0.038	0.906	0.001	0.126	0.233	9.663	0.000	0.000
		1.50	1558.7	45693	0.089	2.094	0.002	0.190	0.079	4.325	0.000	0.000
		1.75	911.8	33924	0.052	1.750	0.000	0.000	0.142	6.641	0.000	0.000
		2.00	600.5	12346	0.040	0.766	0.001	0.033	0.030	1.455	0.000	0.016
		3.00	6292.1	454641	0.226	12.516	0.000	0.004	0.068	6.591	0.000	0.000
	10	0.20	37.1	1026	0.006	0.109	0.263	3.633	0.646	10.882	0.104	3.396
		0.40	130.0	3010	0.014	0.266	0.089	3.055	1.179	29.493	0.024	0.826
		0.60	293.1	8886	0.027	0.594	0.048	3.204	0.648	13.060	0.017	1.167
		0.80	793.6	25975	0.068	2.109	0.014	0.884	0.606	16.839	0.000	0.000
		1.00	2498.0	104878	0.182	7.906	0.007	0.350	0.708	24.051	0.004	0.256
		1.25	527.8	11842	0.035	0.719	0.003	0.309	0.366	19.134	0.000	0.000
		1.50	945.4	51929	0.053	2.359	0.002	0.118	0.426	14.160	0.000	0.000
		1.75	457.9	17008	0.031	1.125	0.000	0.000	0.587	24.200	0.000	0.000
		2.00	1366.0	101970	0.068	4.219	0.003	0.164	0.437	14.198	0.000	0.035
		3.00	449.8	10169	0.028	0.344	0.000	0.000	0.151	6.979	0.000	0.000
30	5	0.20	178.7	3092	0.049	0.688	0.155	3.468	2.590	45.559	0.074	3.428
		0.40	495.1	9503	0.116	1.766	0.054	1.462	1.655	30.962	0.040	1.462
		0.60	637.4	9728	0.152	2.109	0.005	0.288	0.798	13.685	0.005	0.288
		0.80	891.0	37351	0.237	9.609	0.003	0.154	0.721	27.342	0.001	0.080
		1.00	6747.1	305936	1.629	71.391	0.001	0.087	0.188	7.574	0.001	0.087
		1.25	1398.2	84733	0.482	25.094	0.000	0.013	0.430	12.694	0.000	0.013
		1.50	3038.3	142875	1.105	46.484	0.001	0.076	0.403	16.337	0.000	0.034
		1.75	2079.6	122634	0.599	26.281	0.002	0.125	0.566	42.661	0.001	0.125
		2.00	342819.8	30272444	103.294	9420.109	0.000	0.000	0.021	0.656	0.000	0.000
		3.00	2646.1	160279	0.856	43.719	0.001	0.102	0.096	6.214	0.001	0.102
	10	0.20	50.8	649	0.023	0.312	0.349	5.967	1.534	17.016	0.223	5.967
		0.40	160.5	1486	0.048	0.453	0.116	3.968	1.967	18.838	0.065	3.968
		0.60	843.3	66905	0.197	14.578	0.104	2.817	1.849	22.836	0.054	2.817
		0.80	681.9	20254	0.174	4.234	0.034	2.013	1.122	21.661	0.022	2.013
		1.00	573.1	14095	0.184	4.219	0.027	1.851	0.591	9.925	0.022	1.851
		1.25	629.6	9322	0.196	3.219	0.012	0.680	1.645	21.999	0.007	0.680
		1.50	547.1	16616	0.177	4.109	0.004	0.345	0.886	26.625	0.000	0.041
		1.75	769.1	27646	0.271	9.016	0.002	0.172	0.526	12.836	0.000	0.034
		2.00	704.2	21981	0.251	6.281	0.000	0.019	0.406	17.684	0.000	0.019
		3.00	589.4	15683	0.219	6.438	0.006	0.344	0.154	6.729	0.000	0.000
40	5	0.20	316.3	12089	0.214	7.328	0.100	2.033	1.776	57.402	0.061	2.003
		0.40	739.8	16622	0.445	8.359	0.022	0.937	1.969	54.610	0.018	0.937
		0.60	3798.3	255461	2.350	154.141	0.021	0.613	1.162	29.136	0.011	0.613
		0.80	731.9	26998	0.761	26.422	0.006	0.366	1.162	21.457	0.004	0.366
		1.00	1326.0	57543	1.238	46.062	0.005	0.381	0.234	7.515	0.001	0.093
		1.25	2777.3	77313	3.049	89.469	0.003	0.185	0.206	12.445	0.001	0.107
		1.50	15111.1	1314899	14.390	1239.703	0.002	0.137	0.788	35.551	0.001	0.090
		1.75	15719.2	892501	14.845	825.766	0.002	0.197	0.089	4.638	0.000	0.034
		2.00	11684.7	972043	10.434	844.422	0.003	0.190	0.142	6.327	0.002	0.190
		3.00	4098.8	240433	3.475	186.281	0.001	0.099	0.449	29.110	0.000	0.019
	10	0.20	67.9	909	0.061	0.891	0.315	4.886	1.922	34.178	0.191	4.886
		0.40	220.7	3179	0.155	2.312	0.152	2.692	2.193	25.540	0.088	2.669
		0.60	586.6	31335	0.428	21.484	0.079	1.910	2.171	28.679	0.056	1.910
		0.80	1547.1	85794	1.601	85.906	0.006	0.526	1.486	19.860	0.006	0.526
		1.00	1111.4	51365	1.159	46.281	0.019	0.556	1.569	42.724	0.016	0.556
		1.25	597.0	19990	0.726	24.094	0.027	1.323	1.417	32.551	0.014	0.958
		1.50	469.4	10843	0.547	10.203	0.004	0.147	1.020	26.874	0.001	0.146
		1.75	2635.6	179920	2.742	179.625	0.004	0.371	1.057	34.116	0.004	0.371
		2.00	1635.6	43663	1.704	35.953	0.006	0.311	1.542	33.460	0.003	0.311
		3.00	1077.1	23355	1.162	22.641	0.002	0.148	0.871	63.449	0.000	0.000

Table 2. The Performance of Branch-and-Bound and Heuristic Algorithms for $b \sim U(0, 0.50)$

			Number of Nodes		CPU Times		Error Percentage					
							HA_1		HA_2		BH	
n	a	λ	mean	max	mean	max	mean	max	mean	max	mean	max
60	5	0.20	35.5	761	0.100	1.984	0.124	3.949	1.050	22.796	0.042	2.002
		0.40	42.8	240	0.122	0.672	0.060	2.647	1.508	53.171	0.038	2.647
		0.60	60.3	889	0.187	2.562	0.008	0.370	0.714	33.753	0.003	0.183
		0.80	42.3	1250	0.169	4.297	0.004	0.235	0.133	5.977	0.000	0.000
		1.00	33.4	241	0.199	1.453	0.003	0.225	1.107	51.215	0.000	0.000
		1.25	59.1	2086	0.373	11.625	0.007	0.455	0.021	0.689	0.003	0.141
		1.50	43.9	664	0.255	3.047	0.000	0.000	0.200	18.566	0.000	0.000
		1.75	49.7	1272	0.253	5.859	0.000	0.011	0.030	1.616	0.000	0.000
		2.00	89.2	2665	0.472	12.453	0.003	0.301	0.367	16.352	0.000	0.000
		3.00	32.8	204	0.193	1.062	0.000	0.000	0.046	2.337	0.000	0.000
	10	0.20	30.2	308	0.074	1.094	0.488	7.726	1.396	24.497	0.151	3.667
		0.40	34.1	412	0.093	1.109	0.105	2.828	1.293	18.474	0.084	2.828
		0.60	34.9	325	0.098	0.875	0.063	2.703	0.969	31.068	0.029	2.703
		0.80	34.1	283	0.099	0.828	0.069	1.969	0.958	26.599	0.002	0.143
		1.00	45.2	1308	0.153	4.094	0.005	0.439	1.039	46.719	0.005	0.439
		1.25	41.5	549	0.148	1.953	0.020	0.727	0.454	19.942	0.009	0.654
		1.50	35.7	396	0.139	1.453	0.010	0.419	0.130	5.778	0.000	0.000
		1.75	39.0	629	0.155	2.266	0.020	0.970	0.837	33.597	0.000	0.000
		2.00	36.3	177	0.150	0.625	0.033	2.251	0.883	40.291	0.004	0.225
		3.00	44.6	538	0.197	2.250	0.004	0.367	0.018	0.479	0.000	0.036
80	5	0.20	36.8	248	0.369	2.938	0.236	5.417	1.964	37.576	0.030	1.004
		0.40	44.8	459	0.520	4.969	0.028	1.023	0.857	26.262	0.009	0.702
		0.60	30.4	230	0.389	2.703	0.001	0.133	0.532	18.300	0.001	0.133
		0.80	49.9	424	0.607	4.562	0.006	0.400	1.321	61.562	0.002	0.239
		1.00	40.8	417	0.515	5.016	0.006	0.436	0.084	4.829	0.000	0.000
		1.25	53.5	659	0.650	7.375	0.006	0.378	0.154	5.800	0.006	0.378
		1.50	41.3	647	0.533	7.969	0.001	0.097	0.098	3.535	0.001	0.097
		1.75	39.3	328	0.492	4.000	0.002	0.084	0.032	0.604	0.001	0.076
		2.00	47.3	419	0.598	5.203	0.000	0.000	0.536	50.976	0.000	0.000
		3.00	64.4	1275	0.790	14.188	0.000	0.000	0.080	6.503	0.000	0.000
	10	0.20	27.9	182	0.178	0.766	0.276	4.381	1.308	19.308	0.106	4.089
		0.40	37.8	204	0.282	1.828	0.173	3.511	1.718	37.617	0.099	3.511
		0.60	32.5	189	0.316	1.875	0.024	0.692	1.916	30.255	0.016	0.625
		0.80	51.1	831	0.525	8.453	0.056	2.535	1.170	22.897	0.022	1.540
		1.00	54.2	1500	0.628	17.016	0.014	0.746	0.486	13.200	0.001	0.052
		1.25	59.0	1395	0.733	16.781	0.000	0.000	0.252	8.039	0.000	0.000
		1.50	38.8	456	0.486	5.469	0.006	0.340	0.284	20.115	0.003	0.286
		1.75	43.2	293	0.565	3.594	0.001	0.030	0.226	8.047	0.000	0.024
		2.00	45.1	627	0.581	7.656	0.001	0.116	0.072	1.699	0.001	0.116
		3.00	42.1	507	0.546	6.047	0.001	0.097	0.537	24.088	0.000	0.000
100	5	0.20	36.6	259	0.456	3.594	0.069	1.920	1.316	28.413	0.039	1.568
		0.40	73.6	3076	0.937	39.016	0.034	1.370	1.122	24.346	0.017	1.370
		0.60	41.6	496	0.736	9.484	0.015	1.118	1.369	42.292	0.011	1.118
		0.80	53.8	537	1.243	11.766	0.009	0.597	0.476	32.700	0.009	0.597
		1.00	63.8	1168	1.627	28.109	0.002	0.177	1.357	38.680	0.000	0.000
		1.25	76.1	2170	1.913	53.578	0.001	0.097	0.095	3.275	0.000	0.025
		1.50	642.9	58559	15.408	1396.719	0.008	0.530	0.499	43.582	0.005	0.530
		1.75	49.1	602	1.275	15.688	0.003	0.205	0.473	24.259	0.001	0.093
		2.00	70.1	865	1.765	20.219	0.001	0.067	0.041	0.705	0.001	0.067
		3.00	34.2	241	0.950	6.484	0.000	0.000	0.091	3.800	0.000	0.000
	10	0.20	66.3	2943	0.702	39.625	0.563	15.577	2.619	31.125	0.298	5.467
		0.40	43.4	223	0.446	3.094	0.310	6.996	1.534	34.170	0.109	3.585
		0.60	43.0	386	0.546	5.422	0.079	3.911	2.556	75.087	0.051	3.911
		0.80	44.3	288	0.761	4.172	0.027	0.888	1.267	56.015	0.019	0.888
		1.00	146.8	11348	3.159	253.219	0.086	3.860	1.052	43.904	0.040	1.392
		1.25	57.4	877	1.467	23.219	0.012	0.567	1.263	53.018	0.004	0.220
		1.50	37.5	324	0.991	8.750	0.005	0.275	0.522	30.403	0.003	0.275
		1.75	49.3	495	1.180	12.375	0.020	0.788	1.008	32.713	0.017	0.788
		2.00	40.8	203	1.064	5.141	0.012	1.226	1.335	87.073	0.012	1.226
		3.00	68.1	1928	1.726	46.797	0.001	0.110	0.359	31.012	0.000	0.017

In order to study the performance of HA_1 and HA_2 algorithms for large job-sized problems, three different job numbers (n = 60, 80, and 100) combined with ten different values of λ are tested in the second part of the experiment. The deterioration rates are generated from a uniform distribution between 0 and 0.5. As a consequence, 60 experimental conditions are examined, and 100 replications are randomly generated for each condition. The results are summarized in Table 2. The same phenomenon is also observed from Table 2. Both heuristic algorithms perform well in terms of the average error percentages. Overall, the performance of HA_1 heuristic is better than that of HA_2, especially for small values of λ. Moreover, the average error percentages of both heuristics tend to decrease as λ increases when the job number is fixed.

6. Conclusions

This paper considers a single-machine makespan problem where each job has a basic processing time, a deterioration rate, and a release time. This problem is known to be NP-complete. Thus, a branch-and-bound algorithm with several dominance properties and a lower bound is proposed to derive the optimal solution. In addition, two effective heuristic algorithms are also provided.

The computational results show that the branch-and-bound algorithm performs well in terms of the number of nodes and the execution time. Moreover, the computational experiments also show the proposed heuristic algorithms perform well. The extension of arbitrary basic processing times might be an interesting issue for future research.

Acknowledgements. The authors are grateful to the editor and the referees, whose constructive comments have led to a substantial improvement in the presentation of the paper.

References

[1] J.N.D. Gupta, and S.K. Gupta, "Single facility scheduling with nonlinear processing times." *Computers Industrial Engineering*, 14, 1988, pp. 387-393.

[2] S. Browne, and U. Yechiali, "Scheduling deteriorating jobs on a single processor." *Operations Research*, 38, 1990, pp. 495-498.

[3] G. Mosheiov, "Scheduling jobs under simple linear deterioration." *Computers and Operations Research*, 21(6), 1994, pp. 653-659.

[4] T.C.E. Cheng, and Q. Ding, "The complexity of scheduling starting time dependent tasks with release times." *Information Processing Letters*, 65(2), 1988, pp. 75-79.

[5] A. Bachman, and A. Janiak, "Minimizing maximum lateness under linear deterioration." *European Journal of Operational Research*, 126, 2000, pp. 557-566.

[6] A. Bachman, T.C.E. Cheng, A. Janiak, and C.T. Ng, "Scheduling start time dependent jobs to minimize the total weighted completion time." *Journal of the Operational Research Society*, 53(6), 2002, pp. 688-693.

[7] J.B. Wang, and Z.Q. Xia, "Scheduling jobs under decreasing linear deterioration." *Information Processing Letters*, 94, 2005, pp. 63-69.

[8] C.C. Wu, W.C. Lee, and Y.R. Shiau, "Minimizing the total weighted completion time on a single machine under linear deterioration." *International Journal of Advanced Manufacturing Technology*, 33, 2007, pp. 1237-1243.

[9] A. Janiak, and M.Y. Kovalyov, "Scheduling in a contaminated area: A model and polynomial algorithms." *European Journal of Operational Research*, 173, 2006, pp. 125-132.

[10] W.C. Lee, C.C. Wu, C.C. Wen, and Y.H. Chung "A two-machine flowshop makespan scheduling problem with deteriorating jobs." *Computers & Industrial Engineering*, 2007. (DOI 10.1016/j.cie.2007.10.010)

[11] C.C. Wu and W.C. Lee "Two-machine flowshop scheduling to minimize mean flow time under linear deterioration." *International Journal of Production Economics*, 103, 2006, pp. 572-584.

[12] W. Kubiak, and S. van de Velde, "Scheduling deteriorating jobs to minimize makespan." *Naval Research Logistics*, 45, 1998, pp. 511-523.

[13] T.C.E. Cheng, and Q. Ding, "Single-machine scheduling with step-deteriorating processing times." *European Journal of Operational Research*, 134, 2001, pp. 623-630.

[14] B. Alidaee, and N.K. Womer, "Scheduling with time dependent processing times: Review and extensions." *Journal of the Operational Research Society*, 50, 1999, pp. 711-720.

[15] T.C.E. Cheng, Q. Ding, and B.M.T. Lin, "A concise survey of scheduling with time-dependent processing times." *European Journal of Operational Research*, 152, 2004, pp. 1-13.