Minimizing the Tardiness in a Single Machine Batch Processing

Fahad Al-Ghamdi, Meshaal Al-Khaldi, Amaar Khoukhi and Muhammad Al-Slamah
Systems Engineering Department
King Fahd University of Petroleum and Minerals
Dhahran, 31261, Saudi Arabia

Abstract

This paper is about minimizing the tardiness of a single machine patch processing. It is a scheduling problem, which consist of minimizing the tardiness of jobs assigned in batches to be processed by machines that have size and time limitations. This problem is common in manufacturing operations such as heat treatment and wafer burn-in operations. It is well studied in the literature and many problems can be created from the basic batch processing machine problem. This research aims to solve the problem by formulating and compare the heuristic simulated annealing and CPLEX in term of accuracy and time.

Keywords
Minimize tardiness, batch, machine processing, SA, CEPLEX, GAMS.

1. Introduction
In a batch-processing machine, jobs are grouped to form batches and all the jobs in a batch are processed simultaneously. Batch-processing machines are an integral part of many practical applications such as heat-treating ovens, chemical processes performed in tanks or kilns, testing process of electrical circuits, wafer fabrication process, etc. In a batch-processing machine, all jobs are contained in a batch and processed simultaneously and then released together from the machine. The machine has a size restriction. So, it can process a batch only when the sum of all the job sizes in the batch does not exceed its capacity. The processing time of a batch is equal to the longest processing time of all the jobs in that batch. The processing time and the size of each job are known. The main objective is to minimize the tardiness (the completion time for each job corresponding to their due dates).

2. Related Work
As described in the introduction, the purpose of this research is to minimize the number of tardy jobs in single machine batch processing. There are several publications dealt with problems that are related to this topic. Ref. [1] and [2] did a research aiming to improve single batch-processing machine through scheduling a capacitated batch-processing machine to minimize make span. They used simulated annealing in performing the results and compared the results with CPLEX. Cheng-Shuo and Reha [3] studied the problem of minimizing the maximum lateness of a machine batch processing. They used dynamic programming algorithm to show that the due-date feasible batching exists to minimize the maximum lateness. Genetic algorithms are used to solve the problem. Sung, and Min [4] used two machines flow shops with batch processing machine(s) for earliness/tardiness measure under a common due date. Ali , Behrooz, and Masoud [5] studied a scheduling parallel batch processing machines with arbitrary job sizes. They used a hybrid genetic heuristic for solving the problem and compare the results with simulated annealing. Imelda, John, and Carlyle [6] studied the problem of minimizing the total weighted tardiness on a single batch process machine with incompatible job families and they developed several heuristics and tested their performance. Chang and Young [7] studied a scheduling problem in a two-machine flow shop of two batch processing machines. They analyzed the problem with respect to the tardiness, to minimize the number of tardy jobs and total tardiness. They used simulated annealing to solve the problem. Bertrand and Edwin [8] considered scheduling in the no-wait two-machine flowshop, where a set of jobs are processed simultaneously to minimize the make span. The literature reviewed indicates that the

problems that deal with batch-processing machines are typically addressed through a heuristic, simulated annealing, or CEPLEX. Therefore, in this paper both of them will be used their abilities to work as the size of the problem increase. The literature reviewed indicates that the problems that deal with batch-processing machines are typically addressed through a heuristic, simulated annealing, or CEPLEX. However, no one before compared the ability of both CPLEX and Simulated Annealing to minimize the number of tardy jobs as the size of the problem increase. Thus, this paper will do this as it is needed for many companies and its concept is needed for many areas .The key of comparison should be the time needed to solve the problem and the number of tardy jobs for both of them in different scenario.

3. Problem Statement

The word tardy has many meaning like late, slow and sluggish [9]. The tardiness in a machine(s) means that the completion time for job(s) is achieved after the due date assign for each job. Many companies need to increase the efficiency of the process to fully utilize their facilities. In this paper, we are going to minimize the number of tardiness in a single machine batch processing by assigning each job to specific batch in a manner that minimize the number of tardy jobs. Furthermore, we are going to compare the efficiency of two different approaches to solve this kind of problem which are Simulated Annealing and CPLEX.

The formulaion of the tardiness problem is summarized as the following.

$$\text{Min} \sum_{j=1}^{n} U_j \qquad (1)$$

Subject to.

$$\sum_{k=1}^{m} x_j^{[k]} = 1 \qquad (2)$$

$$\sum_{j=1}^{n} x_j^{[k]} \leq C \qquad (3)$$

$$R^{[k]} + P_j x_j^{[k]} \leq C^{[k]} \qquad (4)$$

$$r_j x_j^{[k]} \leq R^{[k]} \qquad (5)$$

$$c_j \geq C^{[k]} - M\left(1 - x_j^{[k]}\right) \qquad (6)$$

$$c_j \leq d_j + MU_j \qquad (7)$$

For each $k = 1,2, \ldots . m$

And for each $j = 1,2, \ldots . n$

Where U_j indicates whether job j is tardy or not, $x_j^{[k]}$ indicates whether job j is assigned to a batch k or not, $R^{[k]}$ is the ready time for batch k, P_j is the process time for job j, $C^{[k]}$ is completion time for batch k, r_j is ready time for job j, c_j is completion time for job j and d_j is ready time for job j.

Equation (1) is the objective function to be minimized. Equation (2) is used to insure that each job assign to one batch only. Equation (3) is used to insure that each batch does not exceed its capacity. Equation (4) is used to define the completion time of each batch. Equation (5) is used to define ready time for each batch. Equation (6) is used to define completion time for each job. Equation (7) is used to define U_j and links it to other constraints.

4. Approach

The previous formulated problem is considered as linear objective function with both linear and nonlinear constraints. This problem can be implemented using both simulated annealing and CPLEX to decide which of them are better for this case and similar ones. However, these terminologies are needed to be described before the implementation. **CPLEX** is an optimization software package. It was named for the simplex

method as implemented in the C programming language, although today it provides additional methods for mathematical programming and offers interfaces other than just C. CPLEX Optimizer solves integer programming problems, very large linear programming problems using either primal or dual variants of the simplex method or the barrier interior point method, convex and non-convex quadratic programming problems, and convex quadratically constrained problems (solved via Second-order cone programming, or SOCP). The CPLEX Optimizer is accessible through independent modeling systems such as AIMMS, AMPL, GAMS, MPL, OpenOpt, OptimJ and TOMLAB [10]. **Simulated Annealing (SA)** is a generic probabilistic meta-algorithm for the global optimization problem, namely locating a good approximation to the global optimum of a given function in a large search space. For certain problems, simulated annealing may be more effective than exhaustive enumeration. each step of the SA algorithm replaces the current solution by a random "nearby" solution, chosen with a probability that depends on the difference between the corresponding function values and on a global parameter T (called the temperature), that is gradually decreased during the process. The dependency is such that the current solution changes almost randomly when T is large, but increasingly "downhill" as T goes to zero. The allowance for "uphill" moves saves the method from becoming stuck at local minima—which are the bane of greedier methods [11]. The system to be used is the General Algebraic Modeling System **(GAMS)** which is specifically designed for modeling linear, nonlinear and mixed integer optimization problems. The system is especially useful with large, complex problems. GAM is available for use on personal computers, workstations, mainframes and supercomputers.

5. Implementation

A. Testing the codes:

As shown in the formulation there are five variables to be evaluated by the solver to give the best solution that are U_j ,$x_j^{[k]}$, c_j (completion time for job j), $R^{[k]}$ (the ready time for batch k) and $C^{[k]}$ (completion time for batch k). Also, there are six more values that must be known which are r_j (ready time for job j), d_j (ready time for job j), P_j (the process time for job j), number of jobs, number of batches and capacity of the machine. These parameters are generated randomly, taking into account that $r_j$must be greater than d_j. The formulated problem is coded in GAMS as CPLEX and SA. After the codes were made, ten experiments were done by fixing the seed (random number generator) to generate the same random number for r_j, d_j and P_j for both CPLEX and SA with the same condition (i.e. same number of jobs, batches and capacities) to compare the results. Ten experiments were done with 10 jobs, 4 batches and capacity is 5. Table 1 compares the results of both CPLEX and SA in term of the minimum number of tardy jobs that they could produce. Figure 1 present data provided in Table1 to make the comparison easier.

Table 1: The number of tardy jobs in SA and CPLEX

Seed	1	2	3	4	5	6	7	8	9	10
CPLEX	7	9	8	8	9	7	8	7	6	9
SA	6	9	7	7	8	6	8	8	6	7

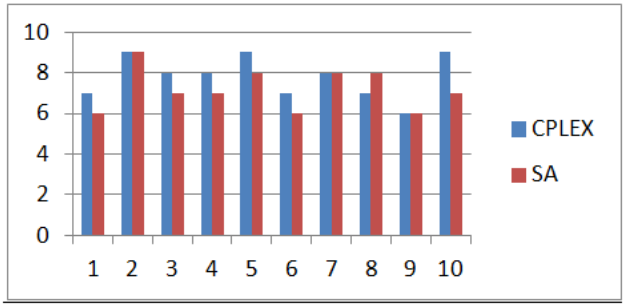

Figure 1: Comparing the results in SA and CPLEX

As shown in Figure 1, the codes seem to be work as expected. Then, more scenarios were generated to observe the performance of each approach as the size of the problem increase.

B. Generating scenarios.

To test the performance of CPLEX and SA, again the seed was fixed to one to generate the same random number for r_j, d_j and P_j for both CPLEX and SA and four more scenarios were generated such that.

Table 2: Scenarios to be tested

Scenario	1	2	3	4	5
# of jobs	20	40	60	80	100
# of batches	5	7	11	12	20
Capacity	5	6	6	7	5

C. Solving the generated scenarios:

Then, these scenarios were implemented as SA and CPLEX using GAMS. Table 3 shows the results of the experiments were done. It shows how many tardy jobs were found in each experiment. Figure 2 present the data provided in Table3 to make the comparison easier. Data in Table 4 shows the time needed to solve each scenario in seconds for both CPLEX and SA in GAMS. Figure 3 present the data provided in Table 4 to make the comparison easier.

Table 3: Number of tardiness in SA and CPLEX

Scenario	1	2	3	4	5
SA	7	19	23	37	40
CPLEX	3	15	24	40	45

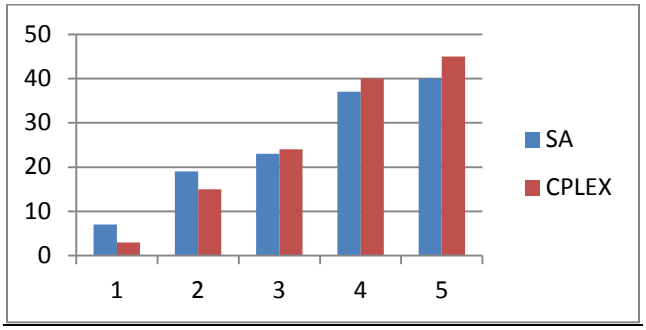

Figure 2: Number of tardy jobs for both SA and CPLEX

Table 4: Time needed to solve the problem

Scenario	1	2	3	4	5
SA	850	1467	3243	3989	6000
CPLEX	30	54	83	115	843

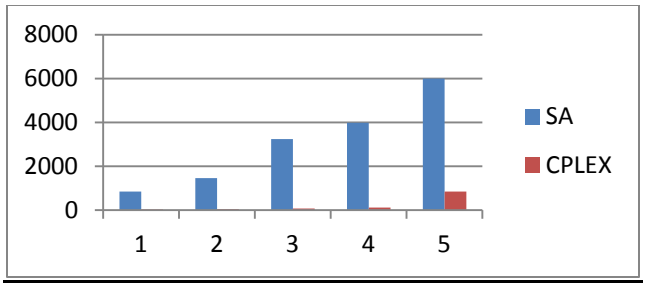

Figure 3: Time needed to solve both SA and CPLEX

D. Results discussion

In 10 seeds experiment the number of tardy jobs in simulated annealing is less than or equal to CPLEX and the time for all that experiments was almost the same. This indicates that in such a small number case simulated annealing is better than CPLEX. However, as the problem's parameters increase (i.e. the number of jobs, batches and capacity) the time needed to solve simulated annealing as a result will increase. As shown in the previous experiments, the results for both CPLEX and simulated annealing in term of number of tardy jobs are approximately in the same range. From the tardy jobs table, one can notice that 60% of simulated annealing scheduled the problem with less tardy jobs than CPLEX. However, the time needed to solve the problem in SA case is extremely higher than CPLEX. These rustles can be changed also if the range of the random numbers generator was changed even though the seed was fixed for all of them. For example, the random numbers range to be generated in 1^{st} scenario was as follow, for process time of a job P(j) = uniform (1,15) for ready time of a job = uniform (5,20) for the due date of each job d(j) = r(j)+ uniform(1,10). If that ranges were changed, both CELPEX and SA will generate another results which may allow SA to generate better output than CELPEX. Furthermore, to avoid that randomness, a reasonable numbers were generated with 10 numbers of jobs scenario to test each method and the result was that the CELPEX scheduled the problem with 3 tardy jobs while SA scheduled it with 4 tardy jobs. It is recommended in the future to study the same problem with multiple machines rather than one single machine such that it can solve this problem as a special case (i.e. the number of machines =1).

6. Conclusion

This paper considers the minimizing of tardiness of a single machine batch processing. This research formulated the problem to minimize the number of tardy jobs in single machine batch processing. Two types of codes were used to solve the problem CPLEX and simulated annealing. From the experiment, simulated annealing gave better results but it took too long time to show the results. It recommended for those who concern about the time to use CPLEX and for those who is looking for more accurate results to use SA.

Acknowledgement

We would like to thank Dr Fouad Al-Sunni the chairman of System Engineering for offering this course. And we would thank Dr. Amar Khoukhi for his effort in teaching this interesting course from which we learn many things about the research and how it is done. Also, we would thank our faculty supervisor Dr. Muhammad Al-Salamah for the very useful theoretical and technical knowledge that we learnt from him that helped us in our research.

References

[1] Melouk,Sharif, Damodaran Purushothaman, and Chang Ping-Yu, "Minimizing makespan for single machine batch processing with non-identical job sizes using simulated annealing." *International Journal of Production Economics*, vol. 87, pp. 141-147, 2004.

[2] Purushothaman, Damodaran, Srihari Krishnaswami, and Lam Sarah S. "Scheduling a capacitated batch-processing machine to minimize makespan." *Robotics and Computer-Integrated Manufacturing*, Vol. 23, pp. 208-216, 2007.
[3] Cheng-Shuo, Wang, and Uzsoy Reha. "A genetic algorithm to minimize maximum lateness on a batch processing machine." *Computers and Operations Research*, vol. 29 pp. 1621-1640, 2002.
[4] C. S., Sung, and Min J. I., "Scheduling in a two-machine flowshop with batch processing machine(s) for earliness/tardiness measure under a common due date." *European Journal of Operational Research*, vol. 131, pp. 95-106, 2001.
[5] Ali Husseinzadeh, Kashan, Karimi Behrooz, and Jenabi Masoud. "A hybrid genetic heuristic for scheduling parallel batch processing machines with arbitrary job sizes." *Computers & Operations Research*, vol. 35, pp. 1084-1098, 2008.
[6] Imelda C., Perez, Fowler John W., and Carlyle W. Matthew. "Minimizing total weighted tardiness on a single batch process machine with incompatible job families." *Computers and Operations Research*, vol. 32, pp. 327-341, 2005.
[7] Chang, Sup Sung, and Young Hwan Kim. "Minimizing due date related performance measures on two batch processing machines." *European Journal of Operational Research*, vol. 147, pp. 644-656, 2003.
[8] Bertrand M.T., Lin, and Cheng T.C. Edwin. "Batch scheduling in the no-wait two-machine flowshop to minimize the makespan." *Computers and Operations Research*, vol. 27, pp. 613-624, 2001.
[9] http://en.wikipedia.org/wiki/Tardy
[10] http://en.wikipedia.org/wiki/CPLEX
[11]http://en.wikipedia.org/wiki/Simulated_annealing

Appendix I: Code for CPLEX

```
option seed=1;
option MIP=CPLEX
set j/1*10/
   k/1*4/;
binary variable U(j),x(j,k);
variable tardy,R_batch(k),C_batch(k),c_job(j);
equations obj,assign_job(j),batch_size(k),CT_batch(j,k),CT_job(j,k),tardy_job(j),RT_batch(j,k);
scalar c/3/;
scalar M/1000000/;
parameter P(j);
parameter r_job(j);
parameter d(j);
p(j) = uniform (1,15);
r_job(j) = uniform (5,30);
d(j) = r_job(j)+ uniform(1,20);
obj..tardy=e=sum(j,U(j));
assign_job(j)..sum(k,x(j,k))=e=1;
batch_size(k)..sum(j,x(j,k))=l=c;
CT_batch(j,k)..R_batch(k)+P(j)*x(j,k)=l=C_batch(k);
RT_batch(j,k)..r_job(j)*x(j,k)=l=R_batch(k);
CT_job(j,k)..c_job(j)=g=C_batch(k)-M*(1-x(j,k));
tardy_job(j)..c_job(j)=l=d(j)+M*U(j);
model smbp/all/;
solve smbp using MIP min tardy;
display U.l;
```

Appendix II: code for simulated annealing

```
option seed=1;
set j/1*10/
   k/1*4/;
parameters x(j,k),R_cap(k),C_cap(k),c(j),u(j),cost;
parameters p(j),r(j),d(j),rp,cap,Temp,rbatch;

scalar cap/3/;
parameter p(j);
parameter r(j);
parameter d(j);
p(j) = uniform (1,15);
r(j) = uniform (5,30);
d(j) = r(j)+ uniform(1,20);

rp=0.0001;
x(j,k)=0;
loop(j,
       rbatch=uniformint(1,card(k));
       loop(k$(ord(k)=rbatch), x(j,k)=1;););

loop(k,R_cap(k)=smax(j,r(j)*x(j,k)););
```

```
loop(k,C_cap(k)=R_cap(k)+smax(j,p(j)*x(j,k)););
loop(j,c(j)=sum(k,C_cap(k)*x(j,k)););

loop(j,if(c(j)>d(j),u(j)=1;
      else u(j)=0;););

cost= sum(j,u(j))+rp*sum(k,sqr(max(0,sum(j,x(j,k))-cap)));

parameters x_new(j,k),u_new(j),cost_new,i_p,i_SA,x_glob(j,k),u_glob(j),cost_glob,rjob;

x_glob(j,k)=x(j,k);u_glob(j)=u(j);cost_glob=cost;

for(i_p=1 to 100,
Temp=50000;
x(j,k)=x_glob(j,k);u(j)=u_glob(j);cost=cost_glob;
for(i_SA=1 to 10000,
   x_new(j,k)=x(j,k);
   u_new(j)=u(j);
      rbatch=uniformint(1,card(k));
      rjob=uniformint(1,card(j));

loop(j$(ord(j)=rjob),loop(k,if(x_new(j,k),x_new(j,k)=-0;);));
loop((j,k)$(ord(j)=rjob and ord(k)=rbatch),x_new(j,k)=1;);

loop(k,R_cap(k)=smax(j,r(j)*x_new(j,k)););
loop(k,C_cap(k)=R_cap(k)+smax(j,p(j)*x_new(j,k)););
loop(j,c(j)=sum(k,C_cap(k)*x_new(j,k)););

loop(j,if(c(j)>d(j),u_new(j)=1;
      else u_new(j)=0;););

cost_new= sum(j,u_new(j))+rp*sum(k,sqr(max(0,sum(j,x_new(j,k))-cap)));

if(exp((cost-cost_new)/Temp)>uniform(0,1),
      x(j,k)=x_new(j,k);
      cost=cost_new;
      u(j)=u_new(j);
       if(cost_glob>cost,cost_glob=cost;x_glob(j,k)=x(j,k);u_glob(j)=u(j);););
Temp=Temp*0.99;
);
rp=rp*1.0001;);

display cost_glob;
```