International Journal of Management Science
Vol 11, No 2, November 2005

Bi-Criteria Process Routing Based on COMSOAL Approach

Sung-Youl Lee*

Professor, Division of Computers, Kwandong University,
San 7, Yangyangeup, Kangwondo 215-802, Korea

(Received Sep. 2004 ; Revised Jan 2005/Aug. 2005 ; Accepted Aug. 2005)

ABSTRACT

This paper investigates the application of the computer method COMSOAL (Computer Method of Sequencing Operations for Assembly Lines) to the process routing (PR) problem with multiple objectives. In any computer aided process planning (CAPP) system, one of the most critical activities for manufacturing a part could be to generate the sequence that optimizes production time, production cost, machine utilization or with multiple these criteria. The COMSOAL has been adopted to find the optimum sequence of operations that optimizes two major conflicting criteria; production cost and production quality. The COMSOAL is here slightly modified to simultaneously generate and evaluate a set of possible solutions (called as population) instead of processing a solution stepwise in each iteration. The significant features of the COMSOAL include: no parameters settings needed, and a guarantee of feasible solutions. Experimental results show that COMSOAL is a simple but powerful method to quickly generate multiple feasible solutions which are as good as the ones obtained from several other well-known process routing algorithms.

Keywords: Multiple Objectives, COMSOAL, Process Routing, Heuristics

* Corresponding author, Email: sylee@kd.ac.kr

1. INTRODUCTION

The Process Routing (PR) refers to activities to determine the optimum production sequence which converts a raw material into a completed part through multi-stage process given a certain criteria such as minimum cost, minimum time, maximum quality, maximum machine utilization, or with multiple these criteria. The implicit enumeration of all these alternatives can be formulated using network flow. Network flow formulation has been widely used to find the best process routing under single objective criterion where the problem is equivalent to solving the Shortest Path Problem (SPP) with precedence constraints. However, since the network flow that has been implemented in this study contains two conflicting objectives, COMSOAL heuristic approach has been adopted as a solution method.

Recently, several GA based approaches for the PR problem have been reported [2, 6, 8]. However, a major drawback of the GA based approach is a significant time consuming to find a proper set of genetic parameters such as population size, crossover rate and mutation rate. This is mainly because the performance of the GAs highly relies on the combination of the genetic parameters.

The COMSOAL, originally a solution approach for the assembly line balancing problem, is a computer heuristic that can be used to generate a feasible solution to the process routing problem at each iteration of the heuristic. A solution methodology of repeatedly running COMSOAL will result in many feasible solutions from which the best is chosen. The significant features of the COMSOAL include that the algorithm does not need any sensitive parameters settings and always generates feasible solutions. Therefore, the COMSOAL provides a relatively simple but powerful solution methodology to be applied in PR field. This solution approach now becomes viable given the increased speed of inexpensive computers.

The COMSOAL is here slightly modified to generate and evaluate simultaneously a set of solution population instead of a solution at each iteration. Therefore, major issue of this study is to adopt the COMSOAL in the PR problem and to report the findings and corresponding modifications.

2. PR PROBLEM DESCRIPTION

The PR usually consists of a series of machining operations, such as turning,

drilling, grinding and so on, to transform a raw material into its final shape. The whole process can be divided into several stages. At each stage, there can be a set of manufacturing alternatives. The PR problem is to find the optimal process routing among all possible alternatives given a certain criteria such as minimum cost, minimum time, maximum quality, maximum machine utilization, or under multiple of these criteria which are defined on the operations to be chosen. In this study, two conflicting objectives are considered to be achieved; *i.e.*, minimizing production cost and maximizing production quality.

The bi-criteria PR problem can be defined as follows:

$$\begin{aligned} &\min \mathrm{C}(x_1,x_2,\cdots,x_n)=\textstyle\sum_{k=1}^{n} u_k(s_k,x_k) \\ &\max \mathrm{Q}(x_1,x_2,\cdots,x_n)=\textstyle\sum_{k=1}^{n} v_k(s_k,x_k) \\ &s.t.\quad x_k \in D_k(s_k) \end{aligned} \tag{1}$$

where s_k is the some state at stage k, $D_k(s_k)$ is the set of possible states to be chosen at stage k, $k=1,2,\cdots,n$, and let x_k be the decision variable to determine which state to choose at stage k, obviously $x_k \in D_k(s_k)$, $k=1,2,\cdots,n$. $u_k(s_k,x_k)$ and $v_k(s_k,x_k)$ represent the criterion to determine x_k under state s_k at stage k, usually defined as real number such as cost, time or quality, and so on.

3. A MODIFIED COMSOAL APPROACH

COMSOAL was first developed by Arcus [1] as a computer method to solve the assembly line balancing problem. While most references to COMSOAL are in the assembly line balancing area, a few papers discuss the applications of COMSOAL to the other problems such as resource assignment, Depuy and Whitehouse [3] and vehicle routing, Lee [4]. Whitehouse [7] discusses the application of COMSOAL to resource allocation and finds the COMSOAL results at each iteration to be fairly stable even though only random sampling is used to choose the next activity to be scheduled. As mentioned previously, since fast, inexpensive computers have become readily available, COMSOAL is worthwhile to be reexamined as a possible solution methodology for the PR problem.

As a solution method, COMSOAL quickly generates multiple feasible solutions and uses the best solution as its final reported result. To schedule processes

for the PR problem, COMSOAL generates an availability list. This list contains those processes in the manufacturing alternatives that could be assigned next. To appear on this list, the processes must have met all precedence. The next process to be scheduled is randomly chosen from this availability list and then a new availability list is formed correspondingly to reflect the chosen process. COMSOAL continues in this fashion until all required processes have been scheduled.

The original COMSOAL has been slightly modified to accelerate the computing time in this study. The modification is basically done in two aspects; generating and evaluating solutions. A set of PRs can be randomly formed in parallel using state permutation encoding without forming availability list.

A set of possible solutions (so called 'population') is evaluated in parallel instead of a solution at a time at each iteration in original COMSOAL.

Finally, the two conflicting objective values from the current iteration are compared to the ones obtained from the previous iteration and the non-dominant solutions are being kept. This process continues until the maximum number of iterations is reached.

3.1 Generating Possible Solutions

In the network flow representation shown in Figure 1, the PR can be naturally identified by indicating which node or state is chosen for a particular operation at each processing stage. Here, a stage stands for a specific process required in every process sequence. A state stands for the one of the alternative machine tools which can be used in the corresponding stage. The alternative states at each stage can be expressed by a series of integers to indicate the node or state. If a state for an operation is chosen at some stage for the PR, then its corresponding integer for that node or state can be assigned whereas the integer is within the number of possible states at that stage. Therefore, the PR solution can be concisely encoded in a state permutation format by concatenating all the set states of the stages [8].

As there is always only one state to be chosen at the last stage, it need not be indicated in the encoding. This state permutation encoding is one-to-one mapping for the PR problem. As to the population for an n-stage PR problem, each individual is a permutation with n-1 integers whereas the each integer is generated randomly within the number of all possible states at the corresponding stage. In Figure 1, the dotted arrow line indicates a possible PR path which is represented by (1 3 2) in state permutation format.

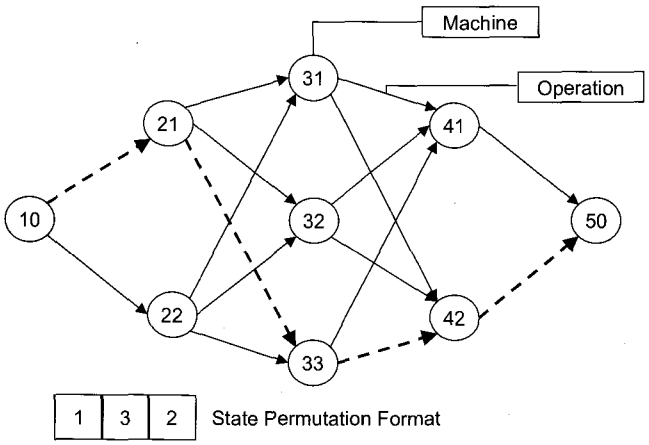

Figure 1. Example Network Flow

3.2 Pareto Solutions

Since the two objectives (minimum production cost and maximum production quality) usually conflict with each other in practice, we can only calculate each objective value of the problem but can not simply evaluate both objective values at the same time. In addition, these two factors are non-commensurate because they can not be measured on the same scale or unit. In other words, we can not obtain the absolute optimal solution, but we can only get the Pareto optimal solutions.

Figure 2 shows the example of Pareto optimal solutions. In Figure 2, the solutions A, B, and C are Pareto optimal (or nondominated) solutions since solution E is dominated by the B and D is dominated by the C.

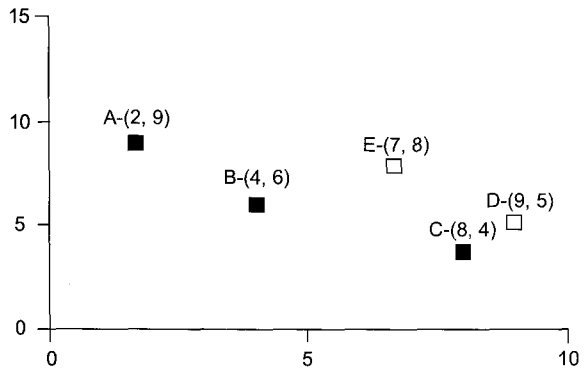

Figure 2. Example Pareto Optimal Solutions

Given a set of feasible solutions X for the problem, solution $x^* \in X$ is denoted as the Pareto optimal solution (or nondominated solution) for the problem if and only if there is no any other solution $x \in X$, satisfying the following conditions:

$$\begin{aligned} V_q(x) &< V_q(x') \quad for \quad some \quad q \in \{1,2,\cdots,p\} \\ V_k(x) &\le V_k(x') \quad for \quad all \quad k \neq q \end{aligned} \tag{2}$$

Assuming minimization problem: Min $V_q(x)$, k=1,2, ..., p, there are respectively p optimal solution x^j, j=1, 2, ... , p, and corresponding objective value:

$$V^j{}_k = V_k(x^j),\ k = 1, 2, \cdots,\ p, \quad j = 1, 2, \cdots, p$$

Pareto stratum (or Pareto frontier) are then identified for a population. Among the solutions generated in a population, nondominated solutions which form a Pareto stratum are filtered and kept. This process is repeated through all the iterations. Only individuals which are not dominated by any other individuals can be kept up to the end. The more the number of iterations, the more the number of Pareto solutions likely maintains.

3.3 Modified COMSOAL Procedure

Figure 3 shows a Flowchart for the modified COMSOAL algorithm and a summary of the modified COMSOAL procedure follows:

[Step 1] Randomly generate a population of possible solutions using the state permutation encoding within the number of all possible states at the corresponding stage.

[Step 2] Calculate the two objective values respectively for the population.

[Step 3] Find the Pareto stratum in a current population based on the two objective values. If the Pareto stratum is the first one, save the stratum and then go to [Step 1]. Otherwise, go to [Step 4].

[Step 4] Test the dominance between the current Pareto stratum and the previous one. Then update and save the resulting Pareto stratum as a current one to reflect the test result.

[Step 5] If the number of current iteration is equal to the number of maximum iteration, print the current Pareto stratum and stop. Otherwise, increase the current iteration number by 1 and go to [Step 1].

The number of iterations used is a user defined variable. Clearly the number of iterations from which the better solution is chosen is a variable of interest.

More iterations will give COMSOAL a chance to find a better solution; however, the run time of the procedure will also increase.

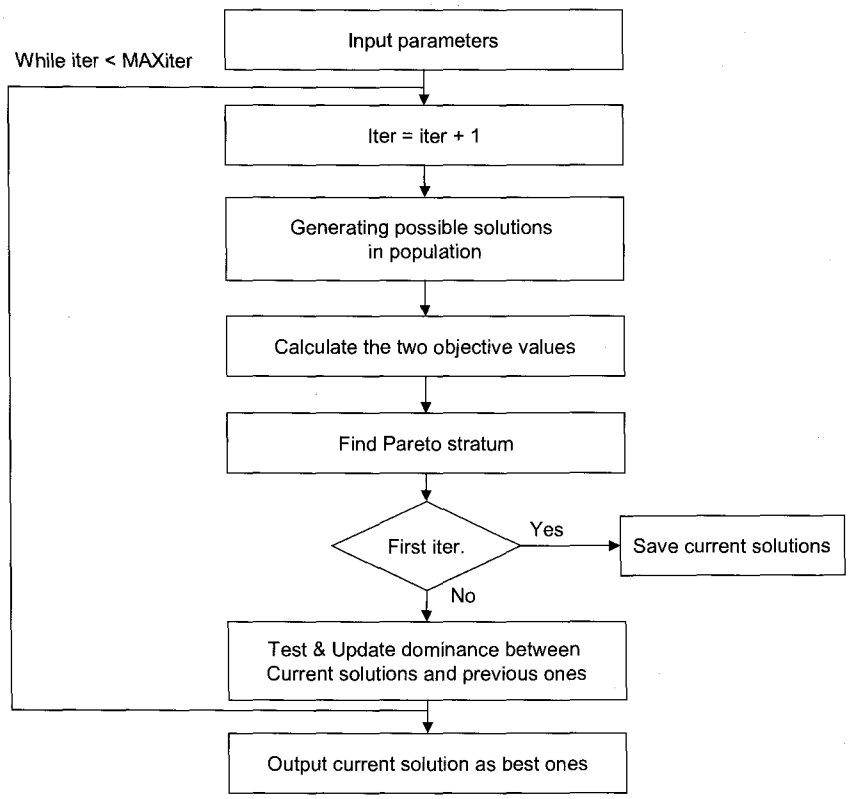

Figure 3. Flowchart of Modified COMSOAL Algorithm

4. NUMERICAL ANALYSIS

The three different levels of example size have been used in this study: a small size example (SSE), medium size example (MSE), and large size example (LSE). The data set of SSE has been taken from the known literature to be compared with GA's results in terms of solution's quality as well as computing speed [8]. The data sets of MSE and LSE have been randomly extended from the SSE's to test the COMSOAL's performance in terms of computing speed. All of the three experiments have been repeated 20 times each to obtain average performance values.

4.1 A Small Size Example

The numerical example with single objective reported by Awadh *et al.* [2] has been adopted in this study. Some random values have been added to the example in order to serve as a second objective. The problem consists of 7 stages, 24 nodes, 80 arcs, and thus the total number of 1,440 possible process routings.

Figure 4 shows the example network and the corresponding two attribute values. In the network, optimum path for each objective which has been obtained using a SPP package is marked only for comparison purpose.

The dotted line stands for the least cost routing path ($1 \rightarrow 1 \rightarrow 1 \rightarrow 3 \rightarrow 2 \rightarrow 1$). The thick line stands for the maximum quality routing path ($3 \rightarrow 3 \rightarrow 3 \rightarrow 2 \rightarrow 2 \rightarrow 2$). The first objective, F1 was to find the least cost and the second, F2 to find the maximum quality process routing among all possible paths in the network. Here, the F1 is modified to be maximization problem as F1 = U – f1. Where the U is the some big number greater or equal to the maximum cost value (in this specific example, 150) and f1 is the total cost for a specific routing path. Thus, the two objectives F1 and F2 now become both maximization problems.

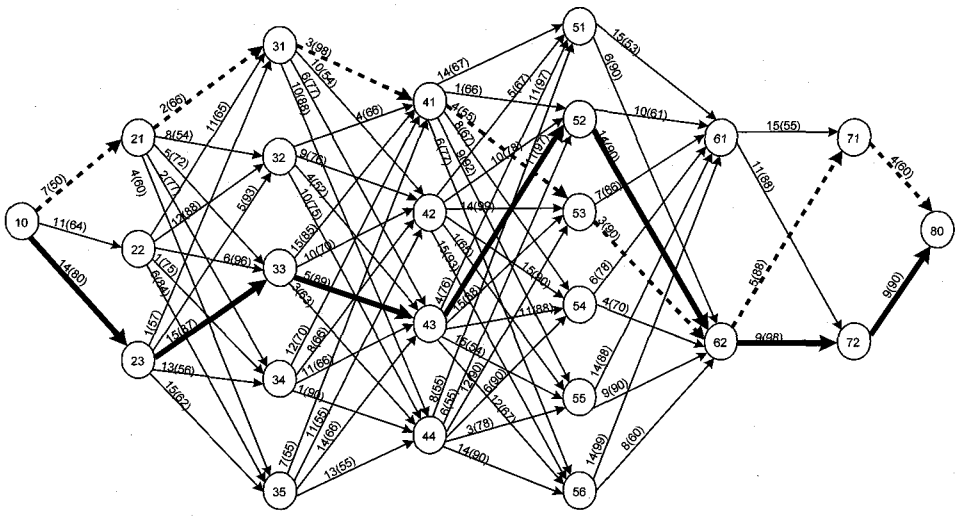

Figure 4. The Example Network Flow with Two Attribute Values: Cost and Quality

The experiment is performed in two ways: i.e., (i) number of iterations from 6, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100,110, 120, and 200 for a fixed population size 50 and (ii) number of iterations from 3, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, and 100 for a fixed population size 100. The experiment is repeated 20 times each

for the various parameters set. There was no difference in terms of the number of Pareto solutions found and their quality between the corresponding two ways except a slightly shorter running time at the first case. When the number of iterations reached at 20 and a population size 50, the solution quality becomes reasonable but the number of Pareto solutions found in between 12~16 were still not enough. When the total number of solutions tested (= population size x number of iterations) reached at 4,000 (50×80 or 100×40), the number of Pareto solutions found and the solution quality both became reasonable and stable: *i.e.*, mostly 19 solutions found and all the same and consistent routing resulted.

Table 1 shows average number of solutions obtained at each iteration for the experiment (i) case. Correspondingly, Figure 5 shows a plot for the average number of Pareto solutions obtained at given iterations.

In Table 1, the first column from the left represents the number of iterations applied, second column for the computing time in seconds, and the last for the average number of Pareto solutions obtained from the 20 trials.

Figure 6 shows a sample plot of the final results obtained at the end of 80^{th} iterations for the given population size, 50. The x-axis gives the total cost from the PR found, while on the y-axis the PR quality is given.

When the total number of solutions tested is greater than 5000, the results converge and are always the same as given in Figure 6.

Table 1. Average Number of Solutions Obtained at Each Iteration (POP_SIZE = 50)

Number of iterations	Time(sec)	Average number of solutions
6	.28	9.2
10	.54	9.1
20	1.21	12.4
30	1.76	14.5
40	2.47	15.0
50	3.14	16.3
60	3.84	17.2
70	4.34	16.2
80	4.95	18.3
90	5.50	18.6
100	6.71	18.6
110	7.40	18.8
120	8.52	18.8
:	:	:
200	14.89	18.8

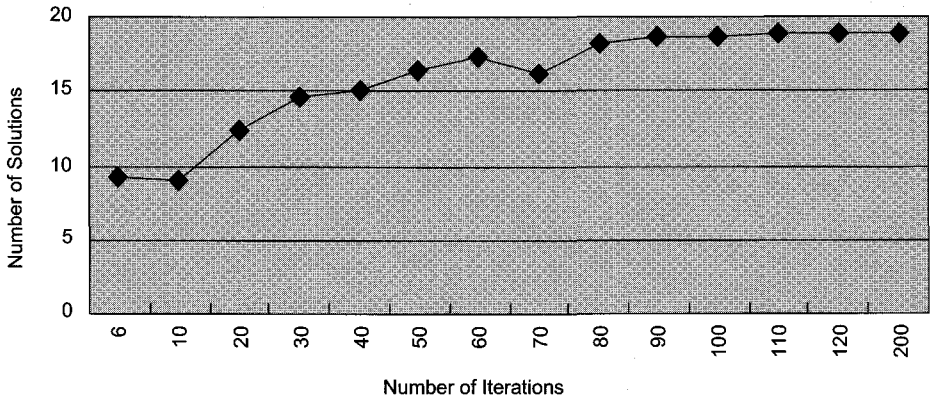

Figure 5. A Plot for the Average Number of Solutions Obtained at Given Iterations

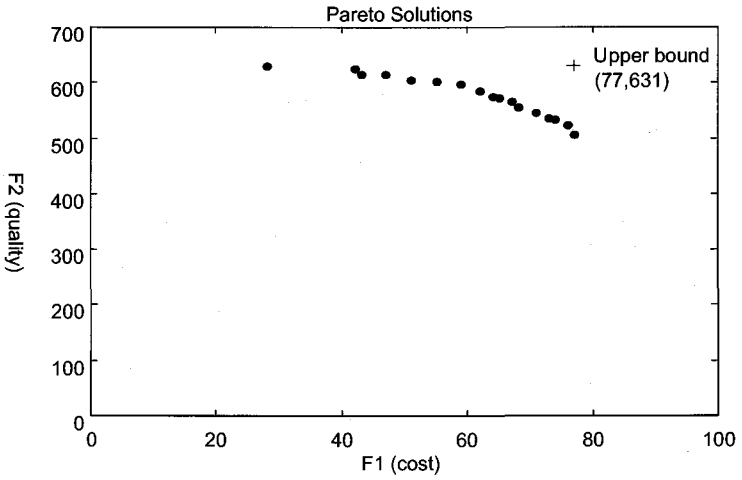

Figure 6. A Sample Plot of the Pareto Solutions Obtained at (MAX_ITER=80 AND POP_SIZE=50)

Thus we consider the results closely reached at the optimum. The upper bound (77, 631) was calculated using a SPP package for each of the two objectives individually. The figure clearly indicates that the Pareto frontier converges near to the upper bounds as close as possible.

Table 2 shows the corresponding 19 different Pareto solutions in Figure 4 including two different routings with the same pair of objective values. The two extreme points are (28, 631) and (77, 507), which equal the upper bound value in one of two objectives. Now, the user can choose one among the 19 solutions based on the user's specific company environment.

These results are exactly the same one that is obtained at the previous research, Lee [5] using multi-objective Genetic Algorithm. The average run time took 4~5 seconds for this specific example. This is much shorter time than the one of the GA which took about 14 seconds [5]. As a result, COMSOAL provided the same quality solutions with around one third computing times comparing to GA.

Table 2. Pareto Quasi-Optimal Solutions Obtained at the Given Parameters Set

F1(cost)	F2(quality)	Pareto solutions
28	631	3 3 3 2 2 2
42	626	3 2 2 3 2 2
43	616	3 2 4 3 2 2
47	615	2 3 3 3 2 2
51	605	3 1 1 5 2 2
55	603	2 3 3 1 2 2
59	597	2 4 4 3 2 2
62	585	1 4 4 3 2 2, 2 4 4 5 2 2
64	577	2 4 4 4 2 2
65	573	1 4 4 5 2 2
67	565	1 4 4 4 2 2
68	557	2 4 4 3 2 1
71	545	2 4 4 5 2 1, 1 4 4 3 2 1
73	537	2 4 4 4 2 1
74	533	1 4 4 5 2 1
76	525	1 4 4 4 2 1
77	507	1 1 1 3 2 1
Parameters	Maxiter=80, Population=50(Run time=4.9 sec)	

Furthermore, COMSOAL always guarantees not only feasible solutions, but also no sensitive parameter selections needed, while GA usually needs lots of time consumption to find a proper combination of parameters.

The algorithm was programmed using Matlab software and operated at the Pentium IV and 128 MB RAM PC.

4.2 A Medium Size Example

Here, we used a set of 7 stages example which is basically modified from the SSE. Each stage was extended randomly so that the 7 stages consisted of 4, 6, 5, 7, 4, and 5 states in order except the last one. In other words, this example problem consists of 7 stages, 33 nodes, and thus the total number of 16,800 possible process routings. Figure 7 shows the data set of cost and quality in each stage.

```
stage1_c=[5,7,11,14];          % cost data for each state in each stage
stage2_c=[4,3,12,15,7,8; 3,2,8,5,2,4; 8,11,12,6,1,16; 10,1,5,15,13,15];
stage3_c=[8,11,14,10,6; 8,3,10,6,10; 11,4,9,4,10; 12,15,10,5,3; 9,12,8,11,1; 8,7,11,14,13];
stage4_c=[12,10,7,5,11,14,4; 8,14,1,4,8,9,6; 7,9,10,14,15,1,15; 10,4,11,15,11,15,12; 4,8,
          6,12,6,3,14];
stage5_c=[12,10,7,8; 4,15,6,8; 8,10,14,9; 9,7,3,8; 10,6,4,10; 12,14,9,8; 11,14,8,6];
stage6_c=[14,10,8,9,12; 10,15,11,12,14; 10,5,9,8,8; 4,6,10,7,9];
stage7_c=[7,4,9,10,13];
stage1_q=[90,50,64,80];      % quality data for each state in each stage
stage2_q=[73,66,70,88,69,58; 70,66,54,72,77,60; 72,65,88,96,75,84; 68,57,93,87,56,62];
stage3_q=[58,72,66,80,90; 60,98,54,77,88; 81,66,76,52,75; 72,85,70,89,63; 80,70,66,66,
          90; 58,55,55,66,55];
stage4_q=[87,65,70,58,72,68,56; 58,67,66,55,67,92,77; 66,67,78,99,90,65,93; 80,76,97,88,
          88,54,67; 90,55,55,90,90,78,90];
stage5_q=[66,87,58,61; 55,53,90,81; 66,61,90,60; 65,66,90,65; 72,78,70,71; 66,88,90,66;
          83,99,60,55];
stage6_q=[65,83,66,70,77; 82,55,88,70,75; 73,88,98,80,82; 88,99,89,77,76];
stage7_q=[70,60,90,82,78];
```

Figure 7. Cost and Quality data set used in each state in each stage

When the total number of solutions tested reached 25,000, the results converged. The number of Pareto solutions found and the solution quality both became reasonable and stable: *i.e.*, mostly 11 solutions found. The Figure 8 shows a sample plot of the final results obtained at the end of 25,000 iterations (50x500). The Table 3 shows a summary of the results. The calculated upper bound is (123, 642). The average run time took 20 seconds with Pentium III 930MHz, 256 RAM PC.

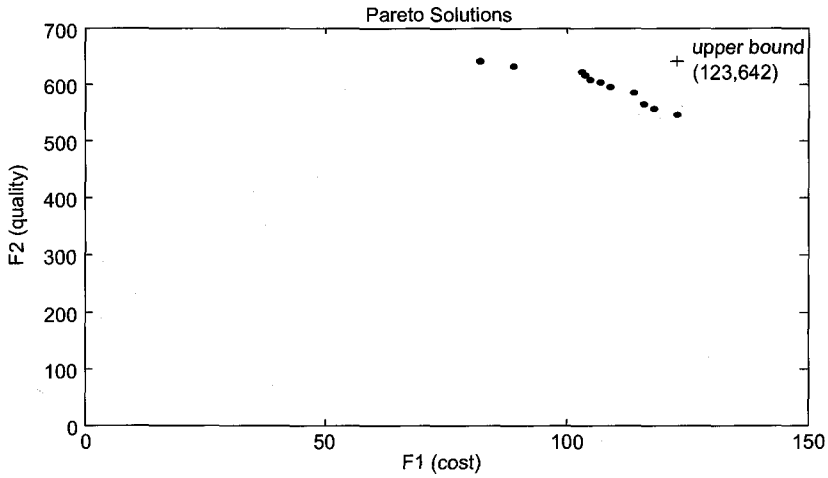

Figure 8. A Sample Plot of the Pareto Solutions Obtained at 25,000 loops

Table 3. Results Summary of Medium Size Example (10 Runs)

Number of possible solutions	33 nodes, total number of possible solutions = 16,800 (no. of states in 7 stages = 4, 6, 5, 7, 4, 5, 1)
Parameters used	population size =50, max no. of iterations =500
Average number of solutions	11
Average computing times	20 seconds

4.3 A Large Size Example

Here, we used the 8 stages example where only the last stage has been added as 10 nodes from the MSE. Consequently, this example problem consists of 8 stages, 43 nodes, and thus the total number of 168,000 possible process routings. Figure 9 shows the data set of cost and quality in the last two stages.

```
% cost data for each state in each stage
stage7_c=[6,10,14,11,8,11,4,9,4,10;10,6,10,3,8,12,15,10,5,4; 8,7,11,14,13,1,11,8,12,9;
          9,6,12,5,8,10,7,7,10,6; 13,5,8,8,10,9,7,7,11,9];
stage8_c=[10,15,4,3,12,8,5,11,9,7];

% quality data for each state in each stage
stage7_q=[90,80,66,72,58,75,52,76,66,81; 88,77,54,98,60,72,85,70,89,63; 55,66,55,72,58,
          80,70,65,93,86; 57,99,60,70,80,66,79,90,56,82; 80,66,58,88,90,65,76,87,59,66];
stage8_q=[84,57,67,62,98,73,50,86,90,88];
```

Figure 9. Cost and Quality data set used in each state in last two stages

When the total number of solutions tested reached 200,000, the results converged. The number of Pareto solutions found and the solution quality both became reasonable and stable: *i.e.*, mostly 16 solutions found. The Figure 10 shows a sample plot of the final results obtained at the end of 200,000 loops (50x4000). The Table 4 shows a summary of the results. The calculated upper bound is (121, 735). The average run time took 191 seconds with Pentium III 930MHz, 256 RAM PC.

It is expected that the bigger sized example is applied, the performance in terms of average run time could be improved since the algorithm structure is simple compare to other methods such as GA. Depuy and Whitehouse [3] proves this in his previous research by reporting for the COMSOAL to give better results for large sample sizes than other, more traditional, resource allocation heuristics.

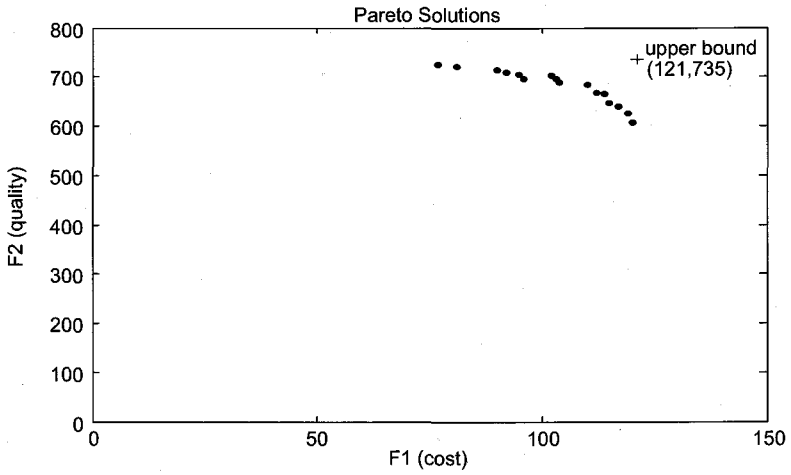

Figure 10. A Sample Plot of the Pareto Solutions Obtained at 200,000 loops

Table 4. Results Summary of Large Size Example (10 Runs)

Number of possible solutions	43 nodes, total number of possible solutions = 168,000 (no. of states in 8 stages = 1, 4, 6, 5, 7, 4, 5, 10, 1)
Parameters used	population size =50, no. of iterations =4,000
Average number. of solutions	16
Average computing times	191 seconds

4.4 Summary of the Examples

Data set for the small size example has been taken from the literature for comparison with GA's results. However, the MSE and LSE have been randomly extended from the data set of the SSE by approximately 10 times. That is to say 1,440 possible solutions for SSE, 16,800 for MSE, and 168,000 for LSE. The Table 5 shows the comparison among the results from the three levels of example size.

In case of SSE, as shown in Table 5, when the total number of solutions tested reached around 2.7 times bigger than the number of possible ones, the COMSOAL generated 19 of quasi-optimal solutions in 4 seconds.

For this specific example, Pentium IV PC was used to compare with the results of GA's which were generated using same type of PC.

In MSE, when the ratio between total number of solutions tested to number of possible solutions was 1.5 times, the algorithm generated 11 solutions in 20 seconds. In LSE, when the ratio was 1.2 times, the algorithm generated 16 solutions in 191 seconds. Based on these ratios and the last row of the Table 5, we can

easily deduce that the algorithm tends to improve its performance in terms of computing speed when the bigger sample size is applied.

Table 5. Results Comparison Among Three Different Example Size

	Small Size Example	Medium Size Example	Large Size Example
Number of possible solutions	7 Stages 24 nodes 1,440 solutions	7 Stages 33 nodes 16,800 solutions	8 Stages 43 nodes 168,000 solutions
Average computing times	4 seconds (Pentium IV 128 MB RAM)	20 seconds (Pentium III 930 MHz 256 MB RAM)	191 seconds (Pentium III 930 MHz 256 MB RAM)
Total number of solutions tested	4,000	25,000	200,000
Average number of quasi-optimal solutions obtained	19	11	16
Performance (number of possible solutions/ second)	1440/4=360	16,800/20=840	168,000/191=880

These results could not be directly compared with the other meta heuristics such as GA and Simulated Annealing because the data sets used in these examples have been created randomly. However, based on the ratio of the number of possible solutions to the computing times as shown in the last row of the Table 5, it is derived that the algorithm could generate more than 800 quasi-optimal solutions per second. Based on the third and fourth rows of the Table 5, we can easily forecast around 5 hours computing times when the total number of solutions tested becomes 20,000,000 which is large enough to be realistic problem sizes and is 100 times bigger than LSE example size.

Therefore, if we consider that process routing is usually done by non-real time base, it is fairly fast enough to satisfy shop floor's requirement.

5. CONCLUSIONS

In this study, a simple but an efficient COMSOAL algorithm was adopted to obtain diverse optimum process routings with conflicting two objectives; *i.e.*, minimizing production cost and maximizing production quality. The COMSOAL which seeks a set of diverse non-dominated solutions has been proposed to solve the PR

problem. The COMSOAL, which does not need any sensitive parameter settings and always guarantees feasible solutions, is relatively simple yet powerful solution approach to be applied in PR.

The three different sizes of numerical examples show that the proposed system could generate good amounts of and various feasible solutions, which are quasi-optimal or optimal, in a short time. The experimental results also show that the performance in terms of average run time could be improved when the bigger sized example is applied.

REFERENCES

[1] Arcus, A. L., "COMSOAL: a computer method of sequencing operations for assembly lines," *International Journal of Production Research* 4 (1966), 259-277.

[2] Awadh, B., N. Sepehri, and O. Hawaleshka, "A Computer-Aided Process Planning Model Based On Genetic Algorithms," *Computers & Operations Research* 22 (1995), 841-856.

[3] Depuy, G. W. and G. E. Whitehouse, "Applying the COMSOAL computer heuristic to the constrained resource allocation problem," *Computers & Industrial Engineering* 38 (2000), 413-422.

[4] Lee, Sung-Youl, "Optimal Vehicle Routing Selection Using COMSOAL," *Proceedings of the KIIE/KORMS Spring Joint Conference*, 2002.

[5] Lee, Sung-Youl, "Process Routing Using Multi-Objective Genetic Algorithm," *Proceedings of the APIEMS conference*, Hong Kong, 2000.

[6] Moon, Chiung, and Gen, Mitsuo, "Evolutionary Algorithm for Process Plan Selection with Multiple Objectives," *Proceedings of the APIEMS'99 conference*, 1999, 323-326.

[7] Whitehouse, G. E., *Practical Partners: Microcomputers and the Industrial Engineer*, Norcross, GA: Industrial Engineering and Management Press, 1985.

[8] Zhou, Gengui and Gen, Mitsuo, "Evolutionary Computation on Multicriteria Production Process Planning," *IEEE (*1997*)*, 419-424.