# A Comparison of Optimization Techniques for Integrated Manufacturing Planning and Scheduling[1].CSRP 408.

M. McIlhagga[2], P. Husbands, R. Ives.

COGS, University of Sussex, Brighton, BN1 9QH, UK.

**Abstract.** We describe a comparison between Simulated Annealing (SA), Dispatch Rules (DR), and a Coevolutionary Distributed Genetic Algorithm (DGA) solving a random sample of integrated planning and scheduling (IPS) problems. We found that for a wide range of optimization criteria the DGA consistently outperformed SA and DR. The DGA finds 8-9 unique high quality solutions per run, whereas the other techniques find one. On average, each DGA solution is 10-15% better than SA solutions and 30-35% better than DR solutions.

## 1. Introduction

This paper describes a comparison of SA, DR, and a Coevolutionary DGA applied to a highly generalized class of job-shop scheduling problems. These problems involve the simultaneous optimization of a number of flexible manufacturing plans. The application of Coevolutionary GAs to this class of problems has been investigated in [Husbands P and Mill F, 1991, Husbands P, 1993]. Prior to that Khoshnevis and Chen used DR to solve problems from a restricted subset of the class [Khoshnevis B and Chen Q, 1990]. Recently Palmer applied SA to a range of industrial problems of this sort [Palmer G, 1994]. To date the relative performance of all three approaches has not been measured. In his Ph.D. thesis Palmer [Palmer G, 1994] detailed algorithms for generating random IPS problems. He used these to compare his SA method with Khoshnevis and Chen's approach on sets of 100 industrially realistic problems.

A comparison of the results obtained with a coevolutionary DGA with those in Palmer' s thesis is reported here. His problem generation algorithms have been reimplemented, as have his evaluation criteria: makespan, mean flow time, total tardiness and proportion of tardy jobs. The coevolutionary approach was found to significantly outperform the two other techniques on all these measures. This work involved adapting an earlier 'ecosystems' model of integrated production for use with a new set of problems and cost functions. This turned out to be relatively straightforward, supporting the claim that the coevolutionary model is very general [Husbands P, 1993].

[1] To appear in Parallel Problem Solving From Nature IV.
[2] malcolm@cogs.susx.ac.uk, http://www.cogs.susx.ac.uk/users/malcolm

Section 2 explains IPS more fully, followed by an overview of each technique used in this study. Problem, cost function and implementation details are then given before the results of the comparison are presented.

## 2. Integrated Manufacturing Planning and Scheduling

The traditional academic view of job-shop scheduling (JSS) is shown in Figure 1 [French S, 1982; Zweben M and Fox M, 1994]. A number of *fixed* plans, one for each component to be manufactured, are interleaved by a scheduler so as to minimize some criteria such as total schedule length.

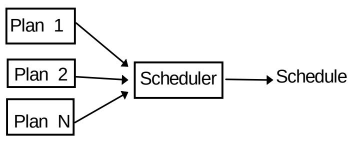

**Fig 1: Traditional Scheduling**

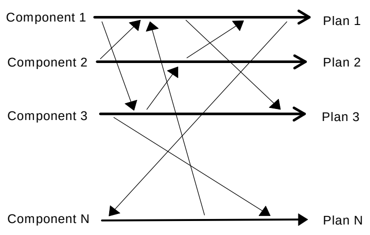

**Fig 2: Emergent Scheduling**

A problem that would often be more useful to solve is that illustrated in Figure 2. Here the intention is to optimize the individual manufacturing plans *in parallel*, taking into account the numerous interactions between them resulting from the shared use of resources. This is a much harder and far more general problem than the traditional JSS problem.

In many manufacturing environments there are a vast number of legal component plans. These vary in the number of manufacturing operations, the ordering of the operations, the machines used, the tool used for an operation and orientation of the work-piece (setup) given the machine and tool choices. All these choices are subject to constraints on the ordering of operations, and technological dependencies between operations. Optimizing a single process plan is an NP-hard problem [Husbands P and Mill F, 1991]. Optimizing several in parallel requires a powerful search technique. It is this class IPS with which we are concerned.

## 3. Approaches to Integrated Planning and Scheduling.

Section 3 reviews the three approaches to IPS investigated in this study.

### 3.1 Simulated Annealing

SA is a stochastic search technique fully described in the literature [Aarts E and Korst J, 1989; Kirkpatrick S, Gelatt C D, Vecchi M P, 1983]. By sometimes allowing temporary jumps to worse solutions using the Boltzman distribution the technique tends to avoid local minima.

In order to apply SA to a problem it is necessary to have a solution representation and a set of operators to move from the current solution to new candidate solutions. Palmer chose to represent solutions to the IPS problem as digraphs [Palmer G, 1994]. Such a graph is shown in Fig 3 and the schedule it represents is shown in Figure 4. The solid arrows represent ordering constraints between operations and the shaded arrows represent particular linearisations of the process plans which are combined in parallel to form the overall schedule.

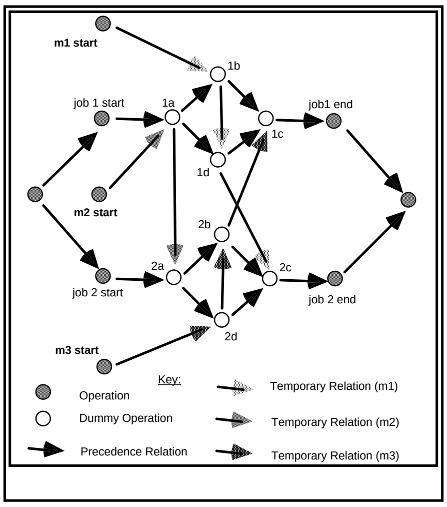

**Fig 3: Digraph of Schedule**

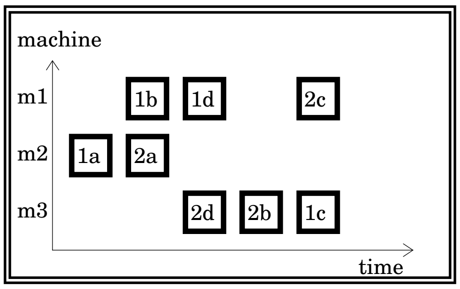

**Figure 4: Gantt Chart of Schedule**

He implemented three plan change operations. Each of these respected problem ordering and operation-machine combination constraints: reverse the order of two sequential operations on a machine; reverse the order of two sequential operations within a job; change the machine performing an operation. Each move was generated by one randomly chosen operator from the set above. The annealing schedule reduced the temperature by 10% every 10N moves, or after N moves without any improvement since the last drop in temperature, whichever came first. N is the number of problem variables.

## 3.2 Khoshnevis and Chen's Dispatching Rules Approach

Dispatching (priority) rules are a popular heuristic used in constructing schedules in classical JSS problems [French S, 1982]. Typically used within simple constructive search algorithms to choose the next operation to process, the most common are: **SPT,** Shortest Processing Time; **FCFS,** First Come First Served; **MWKR**, Most Work Remaining; **LWKR**, Least Work Remaining; **MOPNR,** Most Operations Remaining; **RDM,** Random.

[Khoshnevis B and Chen Q, 1990] use a dispatching rule based on slack time (the difference between time remaining to due date and anticipated total process time). Whenever a machine becomes available, the job chosen to be processed next is the one with least slack time (LST). Their approach allows process plan flexibility in the order of operations within a job and the machine chosen for each operation. In order to decide between candidate operation-machine combinations in the next job to be processed, they use the SI (Shortest Imminent processing time) rule. Thus the IPS problem is tackled using a LST rule to choose between jobs, and the SI rule to choose an operation-machine combination within a job.

## 3.3 Distributed GAs & the Coevolutionary Ecosystem Model

The DGA model is fully described in [Husbands P, 1992; McIlhagga M, Husbands P, Ives R, 1996]. In the basic DGA [Collins R and Jefferson D, 1991] a population of chromosomes is kept in a non-ordinal data structure similar to that of a traditional GA [Goldberg D E, 1989]. However a landscape grid (a 2D torus) is maintained, allowing geographically local selection and replacement strategies. That is, members of the population mate with other members nearby on the 2D grid and their offspring are placed in the same 'neighbourhood'. The advantages of a DGA are: low variance in best solutions found over multiple runs; high variance in good solutions represented in the population at the end of a run; better quality and quantity of solutions found and faster decent to very good solutions [McIlhagga M, Husbands P, Ives R, 1996; Collins R and Jefferson D, 1991].

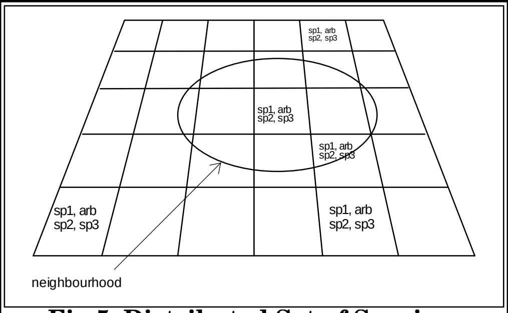

**Fig 5: Distributed Set of Species**

In the ecosystems model for handling the IPS problem, a number of different species coevolve on the 2D grid. Each cell on the grid contains one member of each species. This is illustrated in Fig 5. The genotype of each species represents a plan for a component to be manufactured in the machine shop. Separate populations evolve under the pressure of selection to find near-optimal plans for each component. However, their costs account for shared resources in their common world (a machine shop model). This means that without the need for an explicit scheduling stage, a low cost schedule will emerge as the plans are being optimized. Manufacturing data is used to randomly construct populations of plan structures, one for each component. Arbitrator chromosomes, who resolve conflicts between members of the other populations, are an important part of this model—their fitness depends on how well they achieve this. Each population, including Arbitrators, evolve under the influence of selection, crossover and mutation [Husbands P, 1993].

Selection works by using a ranking scheme within a 12 chromosome neighbourhood: the fittest individual is twice as likely to be selected as the median. Offspring replace individuals from their parents' neighbourhood.

Replacement is probabilistic using the inverse scheme to selection. Genetic material remains spatially local and a robust and coherent coevolution (particularly between Arbitrators and process plans) is allowed to unfold.

The cost, hence selection, functions for plan organisms involve two stages: i) population specific criteria (machining costs), and ii) takes into account interactions between populations. Arbitrators are only evaluated at the second stage, their fitness depending on how well they reduce conflicts between plans. The second stage of the plan cost function involves simulating the simultaneous execution of several plans, one for each component manufactured. These plans (plus an Arbitrator) are taken from the same cell on the grid—an important factor in allowing coherent coevolution. Any waiting time due to interactions between plans (resource conflicts) are added to manufacturing costs to give the total plan cost.

## 4. Description of the Problems

The test problems were generated from data collected from David Brown Vehicles Ltd. [Palmer G, 1994]. The statistics shown later are mean figures taken from 100 sample problems. A problem is a number of jobs (1-14), each of which requires a plan scheduled for a specific shop-floor. A job represents the manufacture of one or more identical parts which (usually) remain together as they move through the shop. Each part can have 1-14 processes. A process plan may be either fixed or flexible, it describes the processes that are carried out (including possible ordering constraints) for a specific set of features to appear on a work piece. The shop-floor does not alter between problems. It comprises 25 machines which vary in the number and diversity of processes that they can carry out.

Each plan is generated from a representation of a part (descriptions of its features and operation order constraints) and the possible processes that can generate those features on the work-piece; in this case there are 1-2 processes per feature. Most can be carried out on a large selection of machines, greatly increasing the search space for this IPS problem.

### 4.1 Problem Generation

The problems used were generated according to plan templates as detailed in [Palmer G, 1994]. A template forms the basis of a job, giving possible operations, machine options and ordering constraints. The number of operations for an instance of a job is chosen at random within the limits defined in the template. The ordering constraints and machine options in the generated (flexible) plans form the basis for the IPS search space. The major problem parameters were as follows:

| | |
|---|---|
| 1-14 | operations per job (generated at random from a plan template) |
| 5-10 | jobs per problem (generated at random) |
| 1-2 | applicable methods per operation (generated from a plan template) |

There were 24 available operation methods. The earliest availability date for each machine was randomly generated from an appropriate range. Release and due dates, set-up and machine times, were generated in accordance with lookup tables and random functions [Palmer G, 1994]. Operation times were calculated using the company's estimation program.

## 4.2 The Cost Function

Each job j has the following data associated with it:

- release date $r_j$
- due date $d_j$
- completion time $C_j$
- flowtime $F_j = C_j - r_j$
- lateness $L_j = C_j - d_j$
- tardiness $T_j = \max(0, L_j)$
- processing time of job j on machine i, $P_{ij}$

The following cost functions can then be calculated:

- makespan: $C_{\max}$
- mean flowtime:

$$\overline{F} = \frac{1}{N} . \sum_{j=1}^{N} Fj$$

- total tardiness: $\sum_{j=1}^{N} T_j$
- proportion of tardy jobs.

In addition, machine utilization, U, for each machine can be calculated:

$$U_i = \frac{1}{C_{\max} - a_i} . \sum_{j=1}^{N} P_{ij}$$

Where $a_i$ is the initial availabe date of machine $i$.

All of Palmer's results reproduced here were found using the compound cost function 'mean flowtime plus twice the total tardiness' (MFTT2). The more distributed coevolutionary GA approach uses slightly different cost functions. These were adapted from Palmer's to fit the coevolutionary architecture. It should be pointed out that results from the two methods were compared over exactly the same set of statistical evaluation criteria.

Below we show the results for two cost functions used with the coevolutionary DGA: **Grp**. and **Cont**. The difference between these two costing criteria is as follows. **Cont**. stands for *contribution*. Here the plan chromosomes have a cost that is in part proportional to it's own efficiency and in part proportional to the efficiency of the group that is belongs to. Each cell on the DGA grid contains a single group comprising one unique plan for each part being planned plus an arbitrator. The evaluation of an individual may be tardiness or plan cost (includes machine set-up and machining costs). **Grp**. stands for *group*, here each chromosome in the group is given the same cost: a weighted sum of the efficiency of *all* of the chromosomes in a cell. Below, $P_G$ and $P_C$ are the Grp. and Cont. process plan chromosomes cost functions respectively. $A_G$ and $A_C$ are the Grp. and Cont. arbitrator cost functions respectively.

PG = MFTT2, PC = (flowtime + 2*tardiness)*N (N is No of plans),
AG = MFTT2, AC = total wait time + 2*total tardiness.

## 5. The Coevolutionary DGA Implementation

Section 5 describes problem specific details of the DGA implementation.

### 5.1 Plan Encoding

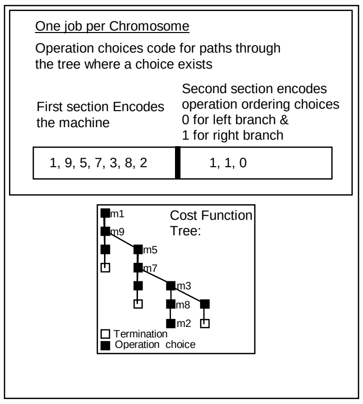
One job per Chromosome
Operation choices code for paths through the tree where a choice exists
First section Encodes the machine
Second section encodes operation ordering choices 0 for left branch & 1 for right branch
1, 9, 5, 7, 3, 8, 2
1, 1, 0
Cost Function Tree:
m1
m9
m5
m7
m3
m8
m2
Termination
Operation choice

**Fig 6: Chromosome**

- Each job is encoded in a separate fixed length of (varies between jobs) chromosome.
- Chromosomes have two sections, the first deals with method (machine) choices, the second with sequence (ordering) choices.
- Currently, all methods that do have a choice have two options and are therefore represented in binary. Lookup tables translate these values into a machine choice. Method choices are held on the genome in an order which maps to a set of known operations (1-N).
- For each job, the cost function maintains a tree containing the space of legal sequences of operations. Sequence choices on the chromosome are interpreted as routes down the tree for that job. The default sequence is always legal, so in cases where the problem description constrains the genome to only one legal sequence, the sequencing information is implicit.
- The evaluation function is a set of routines that traverse a given tree, following a given route, returning with a necessarily valid operation sequence.

The sequence choices may be much larger than the method choices, depending simply on the branch rate of the tree. Thus, a separate data structure is maintained to hold the maximum possible value of each gene. Legal crossover is made trivial with this representation, provided it only occurs at gene boundaries. Translocation is not possible with this scheme.

### 5.2 Arbitrator Encoding

An arbitrator is a series of lookup tables, flattened into a bit string, used to resolves resource (machine) conflicts between different plans. The Nth table encodes preference relationships for the Nth manufacturing operations. Relationships for every possible component pairing are represented. See [Husbands P, 1994] for details. The event-processing in the cost function responds to a request for the use of a machine as follows:

1. If machine is unoccupied, provisionally put the operation on that machine.
2. If the machine is already provisionally occupied by another operation, an arbitrator will decide whether the new operation is placed on the machine instead, and the first operation placed in a waiting state, or whether the second operation should wait until the machine is released.

The provisional status of an operation placement is lost when the **time** in the event-processing routines reaches the operation completion time. As operations cannot be performed **in part**, an operation taken off a machine is said to have been waiting all along. No consideration is made for the time an operation has provisionally occupied a machine when conflict arises. The binary table encodes which of the two plans wins machine-use.

## 6. Results

The results for the SA and dispatch rule algorithms presented here are reprinted from [Palmer G, 1994]. Due to time constraints, we chose not to re-implement these algorithms. The results in [Palmer G, 1994] show the mean results over a set of 100 problems. Here we present our results along side those shown in his thesis. Statistics were derived from 100 problems. The number of objective function calls per run was 525,000.

| Algorithm | Make span | Proportion Tardy | Total Tardiness | Total Time Machining | Machine Utilization | Mean Flowtime |
|---|---|---|---|---|---|---|
| GPDGA cont | 81.22 | 0.14 | 5.84 | 171.75 | 0.18 | 34.86 |
| GPDGA grp | 80.40 | 0.15 | 5.47 | 172.23 | 0.18 | 34.63 |
| SA | 89.09 | 0.18 | 8.87 | 191.22 | 0.18 | 36.10 |
| DR | 95.96 | 0.31 | 30.28 | 218.13 | 0.19 | 41.37 |

**Table 1: Problem Set Comparison Over Assorted Cost Functions**

| Objective function: | CONTRIBUTI ON, GROUP |
|---|---|
| popsize: | 1500. |
| Evaluations per problem: | 525,000 (1.5 hours on an Sparc ipx). |

**Table 2: Parameters**

The DGA outperforms SA and DR for all optimisation criteria (Table 1). Mean improvements over SA, averaged over all optimization criteria (not machine utilization), were **16.58%** and **15.75%** for GPDGA cont. and grp. respectively. The mean improvements over DR, averaged over the optimization criteria, were **37.60%** and **39.00%** for GPDGA cont. and grp. respectively. See section 4.2 for an explanation of the two GPDGA cost functions. The DGA performs ≈1% better when the cost function reflect the efficiency of an individual and not just the group that it came from.

One advantage of using the coevolutionary DGA over SA is that in one run it is capable of generating more than one solution to the problem at hand. Table 3 shows the mean number of chromosomes within 5% of the best found. The figures are averages over 100 problems. The data is taken from the same set of experiments used to generate Table 1. This emphasizes the fact that the DGA is not only finding good solutions to an NP-Hard problem, but is finding multiple different near optimal solutions. It is impossible to make a detailed comparison of the solutions found by different techniques (other than mean costs over 100 problems) because Palmer does not include problem solutions in his thesis [Palmer G, 1994].

| Algorithm | Mean Flow time + Total Tardiness * 2 | Mean No. Competing Solutions |
|---|---|---|
| GPDGA Cont. | 46.54 | 8.76 |
| GPDGA Grp. | 45.56 | 9.74 |
| SA | 53.84 | N/A |
| DR | 101.93 | N/A |

**Table 3: MFTT2 Comparison**

Further analysis indicated the presence of a few aspects which significantly swung the results for the Total Tardiness criterion. These occur in cases where a problem includes a number of plans generated from the same templates-section. In such cases, the due-dates turn out to be similar for a number of jobs which largely demand the same machines (i.e. the method flexibility required to avoid waiting-times turns out to be particularly limited). Because there are 14 job-types, and 5-10 different jobs per problem, the probability of getting three or more jobs from the same job-type class is ≈0.071 (given 105 jobs, an average of ≈7.5 jobs would be the same as at least two others). This aspect adds to the difficulty of the problem by making the total tardiness hard to minimize. Table 4 shows the results for two identical DGA runs, differing only in the problems used: i.e. two different sets of 100 problems.

| Algorithm | Makespan | Proportion Tardy | Total Tardiness | Total time Machining | Machine Utilization | Mean Flowtime |
|---|---|---|---|---|---|---|
| GPDGA 1st | 62.75 | 0.18 | 16.67 | 112.86 | 0.16 | 23.93 |
| GPDGA 2nd | 63.50 | 0.16 | 12.89 | 117.22 | 0.16 | 23.99 |
| SA | 89.09 | 0.18 | 8.87 | 191.22 | 0.18 | 36.10 |
| K&C | 95.96 | 0.31 | 30.28 | 218.13 | 0.19 | 41.37 |

**Table 4: Comparison of Two sets of 100 Problems (GPDGA Cont. cost function)**

The results stay reasonably constant except for tardiness factors. Thus the effect of the MFTT2 in Table 5. This suggests that, although 100 problems would provide statistical significance for most of the optimization criteria, the sample may not be large enough to give a fair comparison of tardiness.

## 7. Conclusions

We found that for all the optimization criteria described in [Palmer G, 1994] the coevolutionary DGA consistently outperformed the SA algorithm and the DR algorithm. Results suggest that variance in costs due to sample size should be calculated in this sort of comparative study. It is clear that some optimisation criteria are more sensitive to this effect than others. A larger sample than that provided by Palmer would have been desirable. Both DGA cost function configurations investigated improved on the performance of the SA by a factor of over 15%, and on the dispatching rule by more than 35%. The results suggest that a cost function that includes factors that are related to both its individual performance as well as it's group performance will outperform,

| Algorithm | MFTT2 |
|---|---|
| GPDGA 1st | 57.27 |
| GPDGA 2nd | 49.77 |
| SA | 53.84 |
| K&C | 101.93 |

Table 5: MFTT2

by about 1%, one which only accounts for the group performance. Unlike the other techniques, the DGA produced a number of unique, high quality solutions, to the problem on each run (typically 8 or 9).

## 8. References

Aarts E and Korst J, 1989. Simulated Annealing and Boltzmann Machines. Wiley.

Collins R and Jefferson D, 1991. Selection in massively parallel genetic algorithms. In Belew R and Booker L B (Eds.), Proceeding on the fourth International Conference on Generic Algorithms, ICGA-91, pp. 249-256. Morgan Kaufmann, 1991.

French S, 1982. Sequencing and Scheduling. Ellis Horwood.

Garey M and Johnson D, 1979. Computers and Intractability. W H Freeman.

Goldberg D E, 1989. GAs in Search, Optimization and Machine Learning, Reading, Massachusetts: Addison Wesley.

Husbands P and Mill F, 1991. Simulated Co-evolution as the mechanism for Emergent Planning and Scheduling. In: R. K. Belew and L. B. Booker Eds. Proceedings of the Fourth International Conference on GAs, ICGA-91. Morgan Kaufmann Publishers.

Husbands P, 1992. GAs in Optimization and Adaptation. In L. Kronsjö and D. Shumsheruddin, Eds. 1992. Advances in Parallel Algorithms, pp. 227-276. Blackwell Scientific Publications.

Husbands P, 1993. An Ecosystems Model for Integrated Production Planning. International Journal of Computer Integrated Manufacturing, Vol. 6, nos. 1 & 2, pp. 74-86.

Husbands P, 1994. Distributed coevolutionary Genetic Algorithms for muti-criteria and multi-constraint optimisation. In: Fogarty T, Ed., *Evolutionary Computing, AISB Workshop selected papers,* pp. 150-165. Springer-Verlag, Lecture Notes in Computer Science Vol. 865, 1994.

Khoshnevis B and Chen Q, 1990. Integration of Process Planning and Scheduling Functions. Journal of Intelligent Manufacturing. pp. 165-176.

Kirkpatrick S, Gelatt C D, Vecchi M P, 1983. Optimisation by Simulated Annealing, Science 220, 671-680.

McIlhagga M, Husbands P, Ives R, 1996. A Comparison of Various Search Techniques on a Wing-Box Optimization Problem. In Parallel Problem Solving From Nature IV (available as a CSRP).

Palmer G, 1994. An Integrated approach to Manufacturing Planning (Ph.D. thesis, University of Huddersfield).

Rinnooy Kan A, 1976. Machine Scheduling Problems. Martinus Nijhoff, the Hauge.

Zweben M and Fox M, 1994. Intelligent Scheduling. Morgan Kaufmann.