A STATE-OF-THE-ART REVIEW OF JOB-SHOP SCHEDULING TECHNIQUES

Anant Singh Jain • Sheik Meeran[†]

Department of Applied Physics, Electronic and Mechanical Engineering
University of Dundee, Dundee, Scotland, UK, DD1 4HN
{A.Z.Jain, S.Meeran}@dundee.ac.uk

[†] To whom correspondence should be addressed

ABSTRACT

A great deal of research has been focused on solving the job-shop problem (Π_J), over the last forty years, resulting in a wide variety of approaches. Recently, much effort has been concentrated on hybrid methods to solve Π_J as a single technique cannot solve this stubborn problem. As a result much effort has recently been concentrated on techniques that combine myopic problem specific methods and a meta-strategy which guides the search out of local optima. These approaches currently provide the best results. Such hybrid techniques are known as iterated local search algorithms or meta-heuristics. In this paper we seek to assess the work done in the job-shop domain by providing a review of many of the techniques used. The impact of the major contributions is indicated by applying these techniques to a set of standard benchmark problems. It is established that methods such as Tabu Search, Genetic Algorithms, Simulated Annealing should be considered complementary rather than competitive. In addition this work suggests guide-lines on features that should be incorporated to create a good Π_J system. Finally the possible direction for future work is highlighted so that current barriers within Π_J maybe surmounted as we approach the 21st Century.

KEY WORDS: Job shop, scheduling, review, exact, approximation

1. INTRODUCTION

Research in scheduling theory has evolved over the past forty years and has been the subject of much significant literature with techniques ranging from unrefined dispatching rules to highly sophisticated parallel branch and bound algorithms and bottleneck based heuristics. Not surprisingly, approaches have been formulated from a diverse spectrum of researchers ranging from management scientists to production workers. However with the advent of new methodologies, such as neural networks and evolutionary computation, researchers from fields such as biology, genetics and neurophysiology have also become regular contributors to scheduling theory emphasising the multidisciplinary nature of this field.

One of the most popular models in scheduling theory is that of the job-shop, as it is considered to be a good representation of the general domain and has earned a reputation for being notoriously difficult to solve. It is probably the most studied and well developed model in deterministic scheduling theory, serving as a comparative test-bed for different solution techniques, old and new and as it is also strongly motivated by practical requirements it is clearly worth understanding.

Formally the deterministic job-shop scheduling problem, hereinafter referred to as Π_J, consists of a finite set $\mathcal{J}$ of n jobs $\{\mathcal{J}_i\}_{i=1}^{n}$ to be processed on a finite set $\mathcal{M}$ of m machines $\{\mathcal{M}_k\}_{k=1}^{m}$. Each job $\mathcal{J}_i$ must be processed on every machine and consists of a chain or complex of m_i operations O_{i1}, O_{i2}, …, O_{im_i}, which have to be scheduled in a predetermined given order, a requirement called a *precedence constraint*. There are N operations in total, $N = \sum_{i=1}^{n} m_i$. O_{ik} is the operation of job $\mathcal{J}_i$ which has to be processed on machine $\mathcal{M}_k$ for an uninterrupted processing time period τ_{ik} and no operation may be pre-empted, i.e., interrupted and then completed at a later time. Each job has its own individual flow pattern through the machines which is independent of the other jobs. Furthermore, the problem is attended by *capacity constraints* or *disjunctive constraints* which stipulate that each machine can process only one operation and each operation can be processed by only one machine at a time. If the completion time of $\mathcal{J}_i$ on $\mathcal{M}_k$ is C_{ik} then the duration in which all operations for all jobs are completed is referred to as the makespan C_{max}. In the optimisation variant[1] of Π_J, the objective of the scheduler is to determine starting times for each operation, $t_{ik} \geq 0$, in order to minimise the makespan while satisfying all the precedence and capacity constraints. That is, the goal maybe expressed as determining C^*_{max}, where

$$C^*_{max} = min\left(C_{max}\right) = \min_{feasible\ schedules}\left(max\left(t_{ik} + \tau_{ik}\right) : \forall\ \mathcal{J}_i \in \mathcal{J},\ \mathcal{M}_k \in \mathcal{M}\right).$$

Note the dimensionality of each Π_J instance is specified as $n \times m$ and N is often assumed to be nm provided that $m_i = m$ for each job $\mathcal{J}_i \in \mathcal{J}$ and that each job has to be processed exactly once on each machine. In a more general statement of the job-shop problem, machine repetitions (or machine

[1] The optimisation variant is used to denote the problem itself and is distinguished from the decision variant of the problem which considers the question: Does there exist a solution in the search space that does not exceed a given upper bound ?

absence) are allowed in the given order of the job $J_i \in \mathcal{J}$, and thus m_i may be greater (or smaller) than m. The main focus of this work is given to the case of $m_i = m$-non pre-emptive operations per job J_i, unless otherwise stated.

The paper is organised as follows: The various techniques applied to Π_J, spanning from mathematical approaches and branch and bound techniques to bottleneck based heuristics, artificial intelligence and local search methods, are reviewed and compared. The best methods are then analysed with respect to their required computing times and degree of divergence from optimality. The paper is concluded with suggestions for future work and possible features that should be incorporated within a Π_J scheduling system.

Figure 1 summarises the main techniques applied to solve the Π_J benchmark problems and the category each technique belongs to, i.e. whether it is an approximation or optimisation method and whether it is constructive, builds a solution from the problem data, or iterative, modifies a solution by continually reordering the sequence of operations. One can either apply an approximate method that delivers a good solution in acceptable time or an optimisation procedure that yields a globally optimal solution, but requires a very high computing time. The majority of these methods, be they optimisation or approximation, represent Π_J using the disjunctive graph model, $\mathcal{G} = \{\mathcal{N}, \mathcal{A}, \mathcal{E}\}$ of Roy and Sussmann (1964). Hence before analysing the various optimisation procedures the disjunctive graph model is described first.

In this node-weighted graph (figure 2) there is a vertex for each operation where $\mathcal{N}$ is the set of nodes representing operations to be processed on the set of machines $\mathcal{M}$. Included within $\mathcal{N}$ are two special (fictitious) nodes ⓪ and ☆ which correspond to the initial and final operations respectively and are known as the source and sink: $\mathcal{N} = \{⓪, 1, 2, \ldots, ☆\}$. The positive weight of each node j is equivalent to the processing time of the corresponding operation, where $\tau_{⓪} = \tau_{☆} = 0$ and the starting and completion times of these nodes represents the start and finishing times of Π_J respectively. ⓪ is connected to the initial operation of each job and similarly the final operation of each job is connected to ☆. In Π_J each operation, j, except ⓪ and ☆, has exactly two immediate predecessors and successors, its job and machine predecessor ($\mathcal{JP}(j)$ and $\mathcal{MP}(j)$) and its job and machine successor ($\mathcal{JS}(j)$ and $\mathcal{MS}(j)$).

$\mathcal{A}$ is the set of directed conjunctive arcs representing the precedence constraints for each job, such that $(i, j) \in \mathcal{A}$ indicates that operation i is an immediate predecessor of operation j ($i \prec j$) in the chain of operations of the job. Capacity constraints are represented by a uni-directed but orientable edge set, $\mathcal{E}$, where each member of $\mathcal{E}$ is associated by a pair of disjunctive arcs, requiring a common machine, such that $[i, j] = \{(i \prec j), (j \prec i)\}$ and $\{i, j \in O\}$. The disjunctive graph for the problem described in table 1 is illustrated in figure 2. In figure 2 complete lines represent arcs in $\mathcal{A}$ while the dotted and dashed lines represent edges in $\mathcal{E}$. A schedule is any feasible solution ($\mathcal{P}$) to the following

problem:

minimise $t_{\bigstar}$	$\bigstar \in \mathcal{N}$
subject to	
$t_i - t_j \geq \tau_j$	(conjunctive constraint) $\forall\ i, j \in \mathcal{N}, (i, j) \in \mathcal{A}$
$t_j \geq 0$	(earliest starting time constraint) $\forall\ i \in \mathcal{N}$
$t_i - t_j \geq \tau_j \vee t_j - t_i \geq \tau_i$	(disjunctive constraint)$\forall\ i, j \in \mathcal{N}, i \neq j, (i, j) \in \mathcal{E}_k, \forall\ k \in \mathcal{M}$
	$(\mathcal{P})$

Important parameters in the disjunctive graph formulation are heads and tails which are based on the operation ordering fixed in the current selection. The head or release date, r_j, is the length of the longest path, l, from the source to the start of O_j and is given by $r_j = l(⓪, j)$. Analogously, the length of the longest path from the completion of O_j to the sink is referred to as the tail or latency duration, q_j, and is calculated from $q_j = l(j, \bigstar) - \tau_j$. Thus if one of the longest paths in G goes through operations i and j and $i \prec j$ then $C_{max} = l(i, j) = r_i + \tau_i + \tau_j + q_j$.

2. OPTIMISATION PROCEDURES

In exact procedures the time requirement increases exponentially or as a high degree polynomial for a linear increase in problem size except for selected, restricted versions (special cases) of Π_J.

2.1 Efficient Methods

An efficient algorithm solves a given problem optimally with a requirement that increases polynomially with respect to the size of the input. These methods simply build an optimum solution from the problem data by following a simple set of rules which exactly determine the processing order.

The first example of an efficient method and probably the earliest work in scheduling theory is Johnson (1954) who develops an efficient algorithm for a simple two machine flow shop which minimises the maximum flow time. This problem is described notationally as $n/2/F/F_{max}$ (Conway *et al.* 1967)[2] where each job has to be processed by both machines. This early work has had a great influence on subsequent research because the criterion of minimising the makespan is attributed to Johnson (1954). In addition this algorithm can be easily extended to provide optimal solutions to the $n/2/G/F_{max}$ instance and to a special case of the $n/3/F/F_{max}$ problem. Other efficient methods developed for the job-shop include those by Akers (1956) for the $2 \times m$ problem and Jackson (1956) for the $n \times 2$ instance where there are no more than 2 operations per job. More recently Hefetz and Adiri (1982) have developed an efficient approach for the $n \times 2$ problem where all operations are of unit processing time, while Williamson *et al.* (1997) prove that determining the existence of a schedule with a makespan of three can be done in polynomial time as long as the total processing time required by all the operations on each machine is no more than three.

Even though the above special cases of Π_J have been solved optimally in the job-shop problem

[2] A three field representation scheme ($\alpha/\beta/\gamma$) is also provided by Graham *et al.* (1979) and interchanges between this three field notation and the four field notation of Conway *et al.* (1967) (A/B/C/D) shall be made depending on which scheme is considered by the authors to describe the problem most aptly.

there are $(n!)^m$ possible solutions. Thus a 20×10 problem has 7.2651×10^{183} possible solutions in principle, which exceeds the postulated age of the universe in microseconds. Even though many of these solutions will not be feasible due to precedence and disjunctive constraints, complete enumeration of all the feasible sequences to identify the optimal one is not practical. As a result of this intractability Π_J is considered to belong to the class of decision problems which are $\mathcal{NP}$ (Garey *et al*. 1976). $\mathcal{NP}$ stands for non deterministic polynomial, which means that it is not possible to solve an arbitrary instance in polynomial time unless $\mathcal{P} = \mathcal{NP}$ (Cook 1971, Karp 1972, Garey and Johnson 1979). $\mathcal{P}$ is a subclass of $\mathcal{NP}$ and consists of the set of problems which can be solved deterministically in polynomial time (e.g. the above mentioned special cases of Π_J all belong to the class $\mathcal{P}$). A decision problem is $\mathcal{NP}$-Complete if it belongs to the set $\mathcal{NP}$ and is as least as difficult as any other problem in $\mathcal{NP}$. The corresponding optimisation variant of such an instance is said to be $\mathcal{NP}$-Hard. Some $\mathcal{NP}$-Hard problems can be solved polynomially with respect to different representations of the input, e.g. the one machine total tardiness problem has a running time that is polynomial in the sum of the processing times. Such algorithms are known as pseudo-polynomials. However if a problem is described as strongly $\mathcal{NP}$-Hard or $\mathcal{NP}$-Hard in the strong sense, such as Π_J, then a pseudo-polynomial cannot be found for the problem unless $\mathcal{P} = \mathcal{NP}$.

Not surprisingly with only a very slight variation in problem definition the efficiently solvable problems quickly become $\mathcal{NP}$-Hard or strongly $\mathcal{NP}$-Hard. For example Lenstra *et al*. (1977) show that the 3×3 problem, the n×2 instance with no more than 3 operations per job and the n×3 problem with no more than 2 operations per job are all $\mathcal{NP}$-Hard. Lenstra and Rinnooy Kan (1979) prove the n×2 instance where operations last for no more than 2 units of processing time or the n×3 problem where all operations are of unit processing time both belong to the set of $\mathcal{NP}$ instances, even if pre-emption is allowed (Gonzalez and Sahni 1978). More recently, Sotskov (1991) prove the $\mathcal{NP}$ characteristic of the 3×m problem, while Williamson *et al*. (1997) indicate that determining the existence of a schedule with an upper bound of four is $\mathcal{NP}$-Complete even when all operations are of unit processing time. In addition, in randomly generated solutions precedence relations are not uniformly distributed (Mattfeld *et al*. 1998) and unlike other $\mathcal{NP}$-Hard problems local optimisation does not lead to a fixation of favourable solution characteristics. The intractability of Π_J is further emphasised by the fact that a 10×10 problem proposed by Fisher and Thompson (1963) could only be solved by Carlier and Pinson (1989) even though every possible algorithm had been tried on it.

If an algorithm for Π_J is to run in polynomial time then it can only guarantee a solution that is a fixed percentage, ρ, from the optimum. Such methods are referred to as ρ-approximation algorithms. For example Shmoys *et al*. (1994) propose several poly-logarithmic approximations of the optimal Π_J schedule which are evaluated in terms of their worst case relative error, while Fizzano *et al*. (1997) present a series of distributed approximation algorithms which configure the machines in a ring architecture. The most recent contribution in this area (Williamson *et al*. 1997) provides the first non

trivial theoretical evidence to indicate that scheduling problems are hard to solve even approximately. They prove that for any $\rho < {}^5/_4$ there does not exist a polynomial time ρ approximation algorithm for Π_J unless $\mathcal{P} = \mathcal{NP}$.

Despite the progress made by these recent works, efficient methods cannot be found for Π_J instances where $m \geq 3$ and $n \geq 3$ and French (1982) predicts that no efficient algorithms will ever be developed for the majority of scheduling problems. As a result the focus of optimisation research has turned to enumerative approaches. Enumerative methods generate schedules one by one using clever elimination procedures to verify if the non optimality of one schedule implies the non optimality of many others which are not yet generated, thereby preventing the need to search the complete space of feasible solutions.

2.2 Mathematical Formulations

It has been recognised by many researchers that scheduling problems can be solved optimally using mathematical programming techniques and one of the most common forms of mathematical formulation for Π_J is the mixed integer linear programming (MIP) format of Manne (1960) which is highlighted below. The MIP format is simply that of a linear program with a set of linear constraints and a single linear objective function, but with the additional restriction that some of the decision variables (y_{ipk}) are integers. Here the integer variables are binary and are used to implement the disjunctive constraints. $\mathcal{K}$ is a large number and Van Hulle (1991) indicates that in order for the feasible region to be properly defined $\mathcal{K}$ has to be greater than the sum of all but the smallest of the processing times.

Minimise C_{max} *subject to :*		
starting times	$t_{ik} \geq 0$	$\{i, p\} \in \mathcal{J}$ $\{k, h\} \in \mathcal{M}$
precedence constraint	$t_{ik} - t_{ih} \geq \tau_{ih}$	if O_{ih} precedes O_{ik}
disjunctive constraint	$t_{pk} - t_{ik} + \mathcal{K}(1-y_{ipk}) \geq \tau_{ik}$	$y_{ipk} = 1$, if O_{ik} precedes O_{pk},
	$t_{ik} - t_{pk} + \mathcal{K}(y_{ipk}) \geq \tau_{pk}$	$y_{ipk} = 0$, otherwise
		where $\mathcal{K} > \left(\sum_{i=1}^{n} \sum_{k=1}^{m} \tau_{ik} - min\left(\tau_{ik}\right) \right)$

Despite its conceptual elegance the number of integer variables scales exponentially (Bowman 1959) and even if better and more compact formulations are used they still require a large number of constraints (Manne 1960). Giffler and Thompson (1960) also mention that integer programs have not led to practical methods of solutions while French (1982) expresses the view that an integer programming formulation of scheduling problems is computationally infeasible. Nemhauser and Wolsey (1988) and Blazewicz *et al.* (1991) further emphasise such difficulties and indicate that mathematical programming models have not yet achieved the breakthrough for scheduling problems. As a result these techniques are only able to solve highly simplified "toy" instances, within a reasonable amount of time. This, not surprisingly, suggests that suitable techniques for Π_J lie in other domains.

Any success that has been achieved using mathematical formulations can be attributed to

Lagrangian relaxation (LR) approaches (Fisher 1973a, b, Van De Velde 1991, Della Croce *et al*. 1993, Hoitomt *et al*. 1993) and decomposition methods (Ashour 1967, Applegate and Cook 1991[3], Chu *et al*. 1992, Krüger *et al*. 1995). In LR methods precedence and capacity constraints are relaxed using non negative Lagrangian Multipliers, with penalty terms also incorporated into the objective function, while decomposition approaches partition the original problem into a series of smaller, more manageable subproblems which are then solved optimally.

Table 2 illustrates the solutions and times achieved by the Lagrangian Relaxation approach of Della Croce *et al.* (1993) and the decomposition approach of Krüger *et al*. (1995) on FT (06, 10, 20). Comparisons are made using the mean relative error (MRE) measure. The MRE is calculated from the best known lower bound (LB_{BEST}), and the makespan or upper bound achieved by the particular technique being analysed (UB_{SOL}), using the "relative deviation" formula $MRE = (UB_{SOL} - LB_{BEST}) / LB_{BEST}$. The results indicate that even these strategies suffer from excessive computational effort while the resulting solutions are usually of poor quality, resulting in a large deviation from optimum. Even when these mathematical formulations are combined with other techniques and applied in the calculation of the lower bound (Fisher *et al.* 1983, Applegate and Cook 1991) they have not performed well. Results indicate that the lower bounds they generate are not very good, can be difficult to calculate, and in some cases because of the excessive computing time required the search has to be terminated at the root node.

It is evident that mathematical approaches are inadequate for Π_J. Consequently, the main focus of enumerative approaches for the job-shop is branch and bound techniques.

2.3 Branch and Bound Techniques

Branch and Bound (BB) algorithms use a dynamically constructed tree structure as a means of representing the solution space of all feasible sequences. The search begins at the topmost (root) node and a complete selection is achieved once the lowest level (leaf) node has been evaluated (fathomed). Each node at a level p in the search tree represents a partial sequence of p operations. As implied by their name a branching as well as a bounding scheme is applied to perform the search. From an unselected (active) node the branching operation determines the next set of possible nodes from which the search could progress. The two most common branching strategies are Generating Active Schedules (GAS) and Settling Essential Conflicts (SEC) (Lageweg *et al*. 1977, Barker and McMahon 1985). GAS is derived from the work of Giffler and Thompson (1960). Here each node consists of a partial schedule and the branching mechanism fixes the set of operations to sequence next, while in SEC branching determines whether O_i should be sequenced before O_j or vice versa. Barker and McMahon (1985) indicate that SEC provides increased flexibility and in general is found to be superior to GAS.

The bounding procedure selects the operation which will continue the search and is based on an estimated LB and the currently best achieved UB. In most BB techniques, the first UB is normally

[3] Applegate and Cook (1991) have developed several algorithms, for example lower bound techniques derived from various mathematical models, a branch and bound strategy as well as a bottleneck based scheme. Each one shall be described in its appropriate section.

provided by means of a heuristic and is applied before the BB search actually begins. If at any node the estimated LB is found to be greater than the current best UB then there is no need to continue the search any further with this partial selection as it cannot improve the existing UB. Hence the partial selection and all its subsequent descendants are disregarded. Once either a leaf node or a node where the LB is greater than the current best UB is fathomed the search returns (backtracks) to the highest unfathomed node in the tree. Searching stops once all nodes have been implicitly or explicitly searched.

Tight bounds are therefore essential to BB techniques as they prevent the need to search large sections of the solution space. Many types of bounds are described in the literature. Although Akers (1956), Brucker (1988) and Brucker and Jurisch (1993) have generated LBs by reducing Π_J into subproblems of dimensionality $2\times m$, $2\times m$ and $3\times m$ respectively, the most popular formulation is to decompose the set of operations into m one machine instances ($1|r_j|L_{max}$). The single machine bound is obtained from the makespan of the bottleneck processor which is the strongest bound over all the machines. Despite the fact that this problem is $\mathcal{NP}$-Hard (Lenstra *et al*. 1977) good techniques have been developed (Potts 1980, Carlier 1982).

In addition to branching and bounding strategies inference rules or propositions, which attempt to fix the order of several operations, are also an integral part of many BB algorithms. By successfully combining all these three components large areas of the solution space can be removed from consideration at an early stage of the search. (See the survey of Pinson (1995) for more detail.)

The BB search technique was initially studied by Brooks and White (1965), Ignall and Schrage (1965) and Lomnicki (1965). Other early work of note includes that of Brown and Lomnicki (1966) and Greenberg (1968). Balas (1969) presents one of the first applications of a BB scheme to Π_J. This method applies the disjunctive graph model and only considers critical operations. Another early BB algorithm for Π_J is by Florian *et al*. (1971).

McMahon and Florian (1975) present one of the first successful applications of the popular one machine decomposition. Here branching commences by finding the critical job, i.e. the one that achieves the maximum lateness, and determining all the jobs with due dates greater than the critical job. Another algorithm based on very similar principles is that of Carlier (1982) where the Schrage algorithm (Schrage 1970) is also used to generate an initial schedule. A critical job, C, and a critical set, J of operations, are derived from this sequence and the dichotomy defined by the position of C relative to J serves as the basis for the branching rule.

Following on from this work Carlier and Pinson (1989) calculate the LB using Jackson's Pre-emptive Schedule (JPS) based on Jackson's Most Work Remaining Heuristic (Jackson 1955). The machine with the largest initial pre-emptive lower bound, which is determined using the approach of Carlier (1982), is scheduled first. The input (resp. output) set of operations on this machine are used in the branching strategy which fixes an unscheduled operation to be before (after) one of these sets. Improvements to this work are made by Carlier and Pinson (1990) who propose two further inference

rules. In addition a dichotomic search (Pinson 1988) is also applied to strengthen the initial LB. In their most recent Π_J work Carlier and Pinson (1994) propose four propositions and another lower bound scheme to further improve branching and the fixing of disjunctions.

Using many of the rules applied by Carlier and Pinson (1989), Applegate and Cook (1991) determine if an unsequenced operation *i* should be sequenced before or after a fixed set of operations, *g*. This strategy is referred to as Edge-finding as it determines at which edge of *g*, O_i will be scheduled and is applied in both the branching as well as the bounding strategy. Edge finder has also been applied by Lourenço (1994), combined with LBs derived by decomposing Π_J into one machine scheduling problems with lags.

Another example of a work derived from that of Carlier and Pinson (1989) is the BB approach of Brucker *et al*. (1994) where branching is based on the critical block scheme of Grabowski *et al*. (1986). Perregaard and Clausen (1995) modify these two techniques in order to allow the solution space to be searched more efficiently. The first method provides a parallel search strategy for the algorithm of Carlier and Pinson (1989) using a load balancing approach, while the second method is a parallel version of the algorithm of Brucker *et al*. (1994) and applies a master / slave arrangement. Another example of such a BB algorithm is by Boyd and Burlingame (1996) who construct a parallel version of edge finder. A depth first strategy is applied and the partial enumeration tree is represented by a collection of data structures held in shared memory.

While all of the methods described so far apply the disjunctive graph model, Martin (1996) adopts a time oriented representation to the decision variant of Π_J. The most powerful bounding procedure created is a technique called shaving. Each operation is allocated a time window in which it can be processed and based on various rules and selections each time a target time *T* can be met one or more time units is attempted to be removed (shaved) off *T*, by reducing the time windows of various operations. The aim is to make the time window of each operation as small as possible, while avoiding resource conflicts.

2.3.1 COMPARATIVE ANALYSIS

To highlight the improvement made by BB techniques table 3 compares the results of many of the works reviewed here for FT (06, 10, 20). Note the solutions and times reported for Perregaard and Clausen (1995) and Boyd and Burlingame (1996) are for parallel algorithms using 16 and 8 processors respectively. The results emphasise that the algorithm of McMahon and Florian (1975) is one of the best methods constructed till proof of optimality for FT 10 was achieved. As problems of this dimensionality are no longer a limitation table 4 provides a more complete comparative study of the most recent techniques on larger and harder benchmark instances. The results indicate that the best BB method is that of Martin (1996). However shaving demands phenomenal computing time and in general the performance of these BB techniques is quite sensitive to individual instances and initial upper bound values (Lawler *et al*. 1993).

Although the computational study indicates improvements have been achieved by BB methods, this is mainly attributed to the technology available rather than the techniques used. In general they cannot be applied to large problems and their execution necessitates the need for a very good understanding of the Π_J domain, as highly specialised inference rules and selection procedures are required in order to fathom nodes at high levels in the solution tree, without explicit searching. Consequently, as optimal procedures have generally appeared to be unsuitable for the job-shop many researchers have turned their attention to approximation methods.

3. APPROXIMATION METHODS

Although approximation methods do not guarantee achieving exact solutions, they are able to attain near optimal solutions, within moderate computing times and are therefore more suitable for larger problems. The importance of approximation methods is indicated by Glover and Greenberg (1989) who suggest that directed tree searching is wholly unsatisfactory for combinatorially difficult problems. They indicate that heuristics inspired by natural phenomena and intelligent problem solving are most suitable providing an appropriate bilateral linkage between operations research and artificial intelligence. In this analysis four main categories of approximation technique are considered: priority dispatch rules, bottleneck based heuristics, artificial intelligence and local search methods.

3.1 Priority Dispatch Rules (pdrs)

Approximation procedures applied to Π_J were first developed on the basis of priority dispatching rules and due to their ease of implementation and their substantially reduced computational requirement they are a very popular technique (Baker 1974, French 1982, Morton and Pentico 1993). At each successive step all the operations which are available to be scheduled are assigned a priority and the operation with the highest priority is chosen to be sequenced. Usually several runs of pdrs are made in order to achieve valid results.

The earliest work on pdrs was done by Jackson (1955, 1957), Smith (1956), Rowe and Jackson (1956), Giffler and Thompson (1960) and Gere (1966). Of particular note is the algorithm of Giffler and Thompson (1960). This is now considered as the common basis of all pdrs and its importance is derived from the fact that it generates active schedules. The procedure commences by choosing the unsequenced operation, O_j, with the earliest completion time, then it finds all the other operations that use the same machine ($\mathcal{M}_k$) and which start earlier than the completion time of O_j. These operations are placed in a conflict set. An operation is then selected from the conflict set and sequenced as early as possible. The procedure is repeated until all operations have been sequenced. Pdrs are characterised by the method applied to select operations from the conflict set where a random choice is the simplest priority assignment.

The most well known and comprehensive survey of scheduling heuristics is by Panwalker and Iskander (1977) where 113 pdrs are presented, reviewed and classified. Examples of some popular dispatch rules are provided in table 5. Blackstone *et al.* (1982), Haupt (1989) and Bhaskaran and

Pinedo (1991) provide an extended discussion and summary of these and many other pdrs. A common conclusion found in many studies, and originally stated by Jeremiah *et al.* (1964) is that for the makespan performance measure no single priority rule dominates. The most recent comparative study is by Chang *et al.* (1996) who evaluate the performance of 42 pdrs using a linear programming model. Their analysis indicates that the shortest processing time (SPT) related rules consistently perform well while the longest processing time (LPT) based rules consistently perform badly.

As individual rules perform so poorly, and provide no clear conclusion with respect to the criterion of makespan, more involved heuristics have been formulated. For example, Viviers algorithm (Viviers 1983) incorporates three levels of priority classes within the SPT heuristic. The most common method of improving solution performance is to have a probabilistic combination of individual pdrs. The earliest examples of such a strategy are by Crowston *et al.* (1963) and Fisher and Thompson (1963). Lawrence (1984) compares the performance of ten individual priority dispatch rules with a randomised combination of these rules and shows that the combined method provides far superior results but requires substantially more computing time. Other more sophisticated methods used to control the choice of which rule to apply include a genetic algorithm (Dorndorf and Pesch 1995) (c.f. §3.4.3) and fuzzy logic (Grabot and Geneste 1994).

It is evident that pdrs just choose one possible operation to add to the current partial sequence, while branch and bound techniques evaluate all possible operations, either implicitly or explicitly. The technique of beam search (Morton and Pentico 1993) provides a balance between these approaches by evaluating a number of best solutions at any given decision point. Such an approach is generalised by the sequential fan candidate list strategy (Glover and Laguna 1997). A filtered beam search strategy has been superimposed by Sabuncuoglu and Bayiz (1997) on selection decisions made by a set of pdrs. Best descendants are chosen based on LB calculations using these pdrs.

Another technique which incorporates beam search is the job insertion algorithm of Werner and Winkler (1995) which consists of two phases. The first phase applies an insertion scheme to construct a schedule and extends the algorithm of Nawaz *et al.* (1983) for the permutation flow shop. An operation is positioned in the partial schedule such that it minimises the length of the longest path passing through it. The second phase applies a reinsertion strategy to iteratively improve the initial solution where neighbours are chosen based on the critical block approach. In both phases beam search is applied to improve the search. (As the second phase applies the technique of local search it is analysed with other such methods c.f. table 12.)

3.1.1 COMPARATIVE ANALYSIS

Table 6 compares the results of Sabuncuoglu and Bayiz (1997) and ten different pdrs (table 5), which perform best with respect to the criterion of makespan minimisation as proved by Chang *et al.* (1996). Also included in the analysis is a rule that selects an operation from the conflict set at random (*random*), here all operations have an equal probability of being chosen, and a technique that randomly

combines all of these rules (*combo*). Due to the nature of these heuristics, each rule has been run ten times on each problem in order to get valid results. *Combo* has been applied twice. The first application employs *combo* in the same way as all the other rules, for ten runs ($combo_{10}$), while the second application employs *combo* as in Lawrence (1984) by stopping the algorithm if the best solution has not been improved in the last 200 runs ($combo_{200}$). All ties are broken arbitrarily.

As the CPU times for all the individual pdrs is the same, in table 6 only one time is specified for them. Although results of the individual pdrs are achieved extremely quickly they are of very poor quality (deviations from optimum can be as great as 74 %) and in general solution quality degrades as problem dimensionality increases. This is due to the highly myopic nature of these heuristics, as they only evaluate one possible operation at each decision point, thereby just considering the current state of the machine and its immediate surroundings. (For all practical purposes *mopr* and *fhalf* are the same.) As no single rule shows clear dominance it is more prudent to choose the best solution from several pdrs or apply a combination of several rules. However this requires more computing time as shown by $combo_{200}$ in comparison with the other simple pdrs.

The best results are clearly those of Sabuncuoglu and Bayiz (1997), with an average MRE of 8.33 %, indicating the suitability of beam search, as it is able to improve on the myopic selections normally made by pdrs. However the deviations from optimum are still high, especially compared with other approaches, and with respect to the individual pdrs the computing times are three orders of magnitude greater. The results suggest that pdrs are more suitable as an initial solution technique rather than being considered as a complete Π_J system and if beam search is to be truly exploited it should be applied in combination with other more powerful methods (see Jain 1998).

3.2 Bottleneck Based Heuristics

Although for many years the only viable approximation methods were priority dispatch rules recently the advent of more powerful computers as well as the emphasis on carefully designed, analysed and implemented techniques (Fisher and Rinnooy Kan 1988) has allowed more sophisticated approaches to be developed which can bridge the gap between pdrs and time consuming combinatorially exploding exact methods. One example of such an approach is the Shifting Bottleneck Procedure (SBP) of Adams *et al*. (1988).

SBP is characterised by the following tasks: Subproblem identification, bottleneck selection, subproblem solution, and schedule reoptimisation. The actual strategy involves relaxing the Π_J problem into m one machine problems and iteratively solving each subproblem $(1|r_j|L_{max})$ one at a time using the approach of Carlier (1982). Each one machine solution is compared with all the others and the machines are ranked on the basis of their solution. The unsequenced machine having the largest solution value is identified as the bottleneck machine. SBP sequences the bottleneck machine based on the machines already scheduled, with the remaining unsequenced machines ignored. Selection of the bottleneck machine is motivated by the conjecture that scheduling it at a later stage would deteriorate

the makespan further.

Every time the machine identified as the bottleneck is scheduled all the previously scheduled machines, susceptible to improvement, are locally reoptimised by solving the one machine problem again. The main contribution of this approach is the way the one machine relaxation is used to decide the order in which machines should be scheduled. Adams *et al*. (1988) refer to this technique as SBI. SBI has also been applied to the nodes of a partial enumeration tree (SBII) allowing different sequences of machines to be considered. A computational analysis of the individual components of SBP is provided by Holtsclaw and Uzsoy (1996) and Demirkol *et al*. (1997), who indicate solution quality and computing time are affected significantly by routing structure. A comprehensive review is provided in Alvehus (1997).

Based on SBII Applegate and Cook (1991) have constructed an initial solution procedure known as "Bottle-*k*" (*k* is chosen as 4, 5 and 6) where for the last *k* unscheduled machines the algorithm branches by selecting each remaining machine in turn. An algorithm called "Shuffle" has also been formulated with Edge-finder as its core. From an initial schedule constructed by Bottle-*k*, Shuffle fixes the processing order of one or a small number of heuristically selected machines with the remainder of machines optimally solved by Edge-finder.

Dauzère-Pérès and Lasserre (1993) and Balas *et al*. (1995) report several drawbacks to the strategies proposed by Adams *et al*. (1988), which are also applicable to the SBP variant proposed by Applegate and Cook (1991). When SBI produces the sequence on a machine it can create precedence constraints between pairs of jobs on an unsequenced machine. These constraints, known as delayed precedence constraints (DPCs), arise because sequencing a given machine may impose conditions on the sequence of some other machine, of the type that job *i* has to precede job *j* by at least a specified time lapse. Due to this job dependency when the instance is relaxed to a one machine scheduling problem it is less constrained than it should be, hence, the true bottleneck machine is not selected, the best sequence is not computed, reoptimisation does not guarantee a monotonic decrease of the makespan and the final SBI solution can be infeasible.

As a result Dauzère-Pérès and Lasserre (1993) propose a heuristic strategy while Dauzère-Pérès (1995) and Balas *et al*. (1995) utilise an exact scheme to deal with DPCs. The most recent work (Balas and Vazacopoulos 1998) embeds a variable depth guided local search procedure (GL) within SBP. GL applies an interchange scheme based on a local neighbour-hood structure that reverses more than one disjunction at a time.

3.2.1 COMPARATIVE ANALYSIS

Table 7 compares the results of the various SBP methods applied to Π_J. From table 7 it is clear that the strategy proposed by Balas and Vazacopoulos (1998) is superior, achieving the best solutions on all of the problems tested. This approach is also able to achieve the best known upper bound for the open problem ABZ 9. The primary weakness of this algorithm however is the high computing effort

required, in comparison with the other SBP methods, as many reoptimisations have been applied to achieve these results. In addition, best solutions are achieved from several different parameter settings. A general weakness of the SBP approaches is the level of programmer sophistication required and the whole procedure has to be completed before a solution is obtained. Also no method is available to decide the size of the subproblem or which machine(s) to fix. Even though Balas and Vazacopoulos (1998) suggest several elaborate reoptimisation schemes as yet there is no strategy to indicate how this is best performed and in order to solve problems where job routings are more structured SBP will require modification.

Nevertheless as SBP is a well designed, analysed and implemented procedure it has been incorporated in many other works (Caseau and Laburthe 1995, Yamada and Nakano 1996a, Vaessens 1996) which have improved the upper and lower bounds of several hard problems. SBP has also been applied to many generalisations of Π_J. For example Morton (1990) extends SBP to project scheduling. Ovacik and Uzsoy (1992, 1996) apply this technique to a semi conductor testing facility. Ramudhin and Marier (1996) adapt the procedure to assembly, dag and open shop applications, while Ivens and Lambrecht (1996) extend SBP to deal with a variety of additional constraints. The most recent generalisation is applied to the job shop problem with deadlines (Balas *et al*. 1998).

3.3 Artificial Intelligence

Artificial intelligence (AI) is the subfield of computer science concerned with integrating biological and computer intelligence. It has fundamental origins from biological understanding and uses principles in nature to find solutions. Using this natural understanding one of the primary objectives of AI is to make computers more useful in problem solving. Two main AI methodologies are analysed here: constraint satisfaction approaches and neural network methods. Many other AI techniques, such as expert systems, have been applied to Π_J however their effect has been limited.

3.3.1 CONSTRAINT SATISFACTION (CS)

Constraint Satisfaction techniques aim at reducing the effective size of the search space by applying constraints that restrict the order in which variables are selected and the sequence in which possible values are assigned to each variable. After a value is assigned to a variable any inconsistency arising is removed. The process of removing inconsistent values is called consistency checking, while the method of undoing previous assignments is referred to as backtracking. A backtrack search fixes an order on the variables and also determines a fixed ordering of the values of each domain. The Constraint Satisfaction Problem (CSP) is solved when a complete allocation of variables is specified that does not violate the constraints of the problem. Although considered within the domain of AI, many constraint based scheduling methods apply a systematic tree search and have close links with BB algorithms.

Examples of early constraint based scheduling systems include a train scheduling scheme by Fukumori (1980), a logic based system by Bullers *et al*. (1980) and a planning method called

"Deviser" by Vere (1983). More recently Fox (1987) designs and constructs an interactive constraint-directed heuristic search model called ISIS (Intelligent Scheduling and Information System), in which constraints are used to bound, guide and analyse the scheduling process. An improved version of this technique has also been created which incorporates parallel scheduling (Ow 1986).

Modifications to ISIS has led to the OPIS family of planning / scheduling systems (Smith *et al*. 1986, Ow and Smith 1988) and the CORTES scheduling system (Fox and Sycara 1990). Pesch and Tetzlaff (1996) note several other constraint directed scheduling systems: SOJA (Lepape 1985), OPAL (Bensana *et al*. 1988) and FURNEX (Slotnick *et al*. 1992). However Pesch and Tetzlaff (1996) indicate that many of these methods only provide high level guide-lines to human schedulers and are therefore of little practical use.

Apart from the construction of such industrial Π_J systems, much work has also been performed on applying constraint satisfaction procedures to solve the standard Π_J model. One of the earliest examples of such a work is by Erschler *et al*. (1976) which determines whether $[(i \prec j), (j \prec i)]$ given $\mathcal{M}_i = \mathcal{M}_j$. Each time a disjunction can be fixed all times are propagated so that further disjunctions can be fixed.

The majority of the more recent work on constraint satisfaction methods has been concentrated within a small of group of researchers.[4] In one of their earliest works Fox and Sadeh (1990) provide a series of constraint satisfaction approaches to increasingly harder scheduling problems while Sadeh (1989, 1991) constructs a short term Π_J scheduler based on release and due dates. Further improvements are made by Sadeh *et al*. (1995) who implement an approach that involves scheduling operations with predefined earliest / latest possible start time windows. Sadeh and Fox (1996) apply a framework that assigns a subjective probability, based on resource contention, to each unscheduled operation. The most recent work is by Cheng and Smith (1997) who construct a procedure called Precedence Constraint Posting (PCP). PCP has also been extended to Multi-PCP (MPCP) where it is applied several times in order to improve results.

Nuijten *et al*. (1993) is a deterministic constraint satisfaction strategy that uses the approach of Sadeh (1991) for initialisation. Consistency is achieved by using the logical considerations derived by Carlier and Pinson (1989). The study by Baptiste and Le Pape (1995) indicates that, the number of backtracks significantly decreases when more propagation is performed, the cost of the additional constraint propagation is more than balanced by the reduction in search effort and algorithms which use edge-finding strongly out perform algorithms which do not. These authors also present a series of constraint based optimisation and approximation algorithms (Baptiste *et al*. 1995). Nuijten and Aarts (1996) adapt many of these methods to the multiple capacitated job-shop scheduling problem ($MC\Pi_J$).

Other recent works out with the above are by Caseau and Laburthe (1994, 1995) Harvey and

[4] The majority of research on these methods has been performed by the groups of Mark Fox and Norman Sadeh and Wim Nuijten and Claude Le Pape.

Ginsberg (1995) and Pesch and Tetzlaff (1996). Caseau and Laburthe (1994) introduce the concept of task intervals. Given two operations i and j (which can be the same), where $\mathcal{M}_i = \mathcal{M}_j$, then a task interval (TI) consists of all the operations which can be scheduled between the earliest start time of i and the latest completion time of j. The edge finding propositions of Applegate and Cook (1991) are then applied to determine the operation order. Caseau and Laburthe (1995) formulate a two stage hybrid model. The first step permutes the disjunctions on the critical path, while the second step fixes a small portion of the solution and then applies the edge finder algorithm with task intervals to complete the rest of the schedule.

Harvey and Ginsberg (1995) present a method called Limited Discrepancy Search (LDS) which attempts to remove wrong decisions made early in the search, by systematically searching all possibilities that differ from the chosen possibility in at most a small number of decision points, or discrepancies. Pesch and Tetzlaff (1996) apply a decomposition approach and use the BB method of Brucker *et al*. (1994) to optimally solve the subproblems generated. The most recent constraint based method (Nuijten and Le Pape 1998) applies various propagation methods and operation selection heuristics in order to dynamically determine whether to schedule an operation first or last.

3.3.1.1 COMPARATIVE ANALYSIS

Table 8 provides a comparison of the results achieved by some of these constraint satisfaction techniques from which it is apparent that only recently solutions of adequate quality have been achieved. Although Baptiste *et al*. (1995), Caseau and Laburthe (1995), Nuijten and Aarts (1996) and Pesch and Teztlaff (1996) produce good results they require extensive effort. (Note Pesch and Teztlaff (1996) have only attempted a limited number of benchmark problems.) The results of Caseau and Laburthe (1995) suggest the incorporation of other heuristics into the constraint satisfaction method improves results. This is further highlighted by Nuijten and Le Pape (1998) with their combination of approximation and exact techniques which in fact performs best. However as these techniques have close similarities with BB methods they are also extremely costly (c.f. Caseau and Laburthe (1995) for LA 29). It can therefore be concluded that much research has yet to be done if good constraint satisfaction approaches are to be developed for Π_J. One suggestion is that as the methods analysed are quite generic and can be easily applied to other scheduling problems in order to improve results and most importantly reduce computing times, and thus be competitive in both categories to the best Π_J methods, it is necessary to incorporate more problem specific information. In such a system the constraint satisfaction method can locate promising regions of the search space, by quickly obtaining a suboptimal solution a fixed percentage from optimum, (c.f. the results of Cheng and Smith (1997)), which can be exploited more efficiently and effectively by the problem specific method.

3.3.2 NEURAL NETWORKS (NNs)

Neural networks (NNs), are organised in a framework based on the brain structure of simple living entities. In these techniques information processing is carried out through a massively

interconnected network of parallel processing units. Their simplicity, along with their capability to perform distributed computing, as well as their propensity to learn and generalise has made neural networks a popular methodology, allowing them to be used in many real life applications (Zhang and Huang 1995). Cheung (1994) describes some of the main neural network architectures applied to solve scheduling problems: searching network (Hopfield net), error correcting network (Multi - Layer Perceptron), probabilistic network (Boltzmann machine), competing network and self - organising network. However as neural network applications for Π_J have mainly been implemented in the first two paradigms the focus here is on these two models only. An analysis and review of the application of neural networks in scheduling is also provided by Wang and Brunn (1995).

Searching networks such as Hopfield nets are autoassociative non-linear networks, which have inherent dynamics to minimise the system energy function or Lyapunov function. In many of the Hopfield based methods an underlying mathematical model is applied to map Π_J on to the neural network. The most common being the mixed integer linear programming (MIP) model of Manne (1960) (c.f. §2.2).

Error correcting networks, on the other hand, are trained on examples which take the form of a mapping f: $S \subset \Re^n \rightarrow \Re^m$, from some arbitrary bounded subset S of n-dimensional Euclidean space to m-dimensional Euclidean space. When an activity pattern is applied to the network, the error correcting rule adjusts the synaptic weights in accordance with the above mapping. Specifically the actual response of the network is subtracted from the desired target response to produce the error signal. Weights are adjusted so the actual network response moves closer to the desired response.

3.3.2.1 HOPFIELD NETWORKS

The Hopfield Network (Hopfield and Tank 1985) dominates neural network based scheduling systems and when applied to Π_J, the aim is to minimise the energy function, $\mathcal{E}$, which is based on makespan, subject to the various precedence and resource constraints. If the constraints are violated, a penalty value is produced which increases $\mathcal{E}$. In one of the earliest works, Foo and Takefuji (1988a, b) use an encoding similar to the TSP formulation of Hopfield and Tank (1985) to map Π_J onto a two dimensional matrix mn by (mn+1) neurons. To avoid the problems of local convergence a simulated annealing process is then applied to the model. A TSP matrix based encoding is also applied by Hanada and Ohnishi (1993).

In order to improve their earlier method Foo and Takefuji (1988c) construct an Integer Linear Programming Neural Network (ILPNN) by formulating Π_J as an MIP problem. The energy function is represented by the sum of starting times of all jobs and solutions are obtained by performing linear program and integer (binary) adjustments until convergence.

Van Hulle (1991) indicates that the approach of Foo and Takefuji (1988c) does not guarantee feasible solutions. As a result he translates the MIP formulation into a goal programming format which is then mapped onto a neural (goal programming) network. Willems and Rooda (1994) also use an MIP

formulation but try and overcome some of the above difficulties by reducing the search space of their ILPNN through precalculation.

In order to overcome the limitations faced by ILPNN systems Zhou *et al.* (1991) propose a Linear Programming Neural Network (LPNN) construction. They avoid the use of a quadratic energy function by implementing a linear function instead. This prevents the need for a conventional integer linear programming method which requires excessive control variables. Cherkassky and Zhou (1992) compare this neural model with three commonly used pdrs. The results show that except in one instance the neural network performs better. Other LPNN implementations include those of Chang and Nam (1993) and Gang and Shuchun (1994).

In their most recent work Foo *et al.* (1994, 1995) are also able to avoid the use of a local optimum converging quadratic energy function by the inclusion of non linear "*H*-amplifiers" which represent binary variables, into their hardware implementation. This allows the "*G*-amplifiers" which represent continuous variables, to be implemented as non-inverting amplifiers which obey a linear relationship. Other encoding strategies applied to the Hopfield network include a digital network simulation by Satake *et al.* (1994) and a three dimensional neural network box approach by Lo and Bavarian (1993).

The latest work on neural network based scheduling is by Sabuncuoglu and Gurgun (1996) who also apply a 3D structure, which is augmented with an external processor, that performs local search in the form of a threshold acceptance algorithm (c.f. §3.4.2.2). The threshold level is set at 10 % with regards to accepting disimproving moves. However swapping can result in non feasible solutions as it is not restricted to only critical operations (c.f. Van Laarhooven *et al*. 1992).

3.3.2.2 BACK -ERROR PROPAGATION NETWORKS

Since the work of Rumelhart *et al.* (1986) on the multi-layer perceptron (MLP) supervised learning by the back-error propagation (BEP) algorithm has become the most popular method of training neural networks. One of the earliest studies in BEP scheduling is by Remus (1990) who has developed several BEP neural models, which are compared with a linear regression rule.

Chryssolouris *et al*. (1991) have created a BEP application for the design of a manufacturing system which determines the number of resources required in each workcentre of a job-shop. Khaw *et al*. (1991) implement a BEP approach that receives variable data such as job orders, capacity availability and set-up times, while ouputting machine orders. The three layer network of Dagli and Huggahalli (1991) consists of an input layer, which receives data that has been preprocessed using Lawler's algorithm (Lawler 1973), a constraint recognition layer and a recovery layer. A four layer BEP model has been developed by Hoong *et al*. (1991) which provides a priority index, concerning the allocation of jobs to resources, while Cedimoglu (1993) applies an NN to simulate a simplified job-shop assembly and shop-floor linked together by stores.

In many of the above methods input data, used for training, is taken directly from the shop

floor. Another popular approach is to train the NN using results generated by pdrs. For example Watanabe *et al.* (1993) use the Slack Rule, while Kim *et al.* (1995) train an NN using the results achieved from the Apparent Tardiness Cost (ATC) rule (Vepsalainen and Morton 1987) and the Apparent Tardiness Cost with set-ups (ATCS) rule (Lee *et al.* 1992).

Another common construction is to combine the BEP model with other techniques. The most popular being a hybrid amalgamation with an expert system. One of the earliest examples of such a hybrid is by Rabelo and Alptekin (1989a, b, 1990a, b). Their neural network ranks priority rules and determines coefficients, and then sends these values to an expert system module which performs the required scheduling.

The BEP model in the approach of Dagli *et al*. (1991) receives data related to job and machine status from the expert system and classifies this information into job groups. This classification is sent to the Hopfield model which schedules the jobs according to a predetermined priority. Sim *et al.* (1994) embed sixteen BEP neural networks in an expert system. Each NN corresponds to an activation environment and is trained to recognise the individual contributions of various heuristics in the activation environment.

Instead of an expert system Dagli and Sittisathanchai (1995) combine a BEP neural network with a genetic algorithm (GNS). In GNS the BEP network maps the schedules constructed by the genetic algorithm to multiple objective values and is trained from the performance of an expert.

Most recently, Jain and Meeran (1998) indicate that traditional BEP models suffer from unsuccessful training and convergence to local optima solutions when faced with problems that involve complex input-output mappings. A modified BEP model that incorporates momentum, learning rate and jogging parameters overcomes these deficiencies. Their structure encodes the problem such that the requirement of neurons scales linearly, allowing large instances to be dealt with. However, as previous systems, it is unable to solve generic problems and provide a suitable input-output mapping, because new schedules can only be successful identified subject to the condition that they do not vary by more than 20 % from training data.

3.3.2.3 COMPARATIVE ANALYSIS

Table 9 indicates that only the technique of Sabuncuoglu and Gurgun (1996) provides adequate results to benchmark problems. As many of the Hopfield methods have been implemented in hardware execution times are dependent on the electrical components used and consequently results are produced instantaneously. As these models are encoded using a mathematical model they suffer from a requirement of excessive numbers of constraints, variables and interconnections, hence they can deal with small problems only and as they are often trapped in local minima they do not guarantee optimal solutions. In addition for several of the methods the problems have to be of a particular dimensionality in order to be solved (e.g. n has to equal m (Zhou *et al.* 1991)) and the system can malfunction if it is not applied to problems for which it is designed for. The discrepancies associated with BEP models

include an excessive requirement of neurons and training from non optimal data acquired either from an expert, existing information or pdrs. In addition, as the majority of BEP NNs are combined with other methods the optimisation is usually not performed by the BEP network but instead left to the other techniques in the system, with the BEP network used more for fast database retrieval. Consequently, neural networks are not currently considered to be competitive with the best heuristics for any class of optimisation problem (Osman and Kelly 1996b).

3.3.3 MISCELLANEOUS APPROACHES

Examples of miscellaneous AI approaches include a parallel distributed stochastic model by Lo and Hsu (1993) which has many similarities to the stochastic Hopfield Neural Network. A Vibrating Potential Method based on analogies from dynamics and statistical physics (Yokoi *et al*. 1994) where the physical potential of each job is observed as the interaction energy between operations on machines and an Ant System (AS) optimisation method for Π_J by Colorni *et al*. (1994). AS is a population based stochastic optimisation technique developed by Denebourg, Pasteels and Verhaeghe (1983). It models the behaviour that ant colonies have in finding the shortest path to their food source. The ant is a simple co-operating agent whose search performance becomes very effective when working collectively with many other simple agents, allowing poor local minima to be transcended. The ants are randomly positioned on the nodes of a disjunctive graph and follow a Monte Carlo procedure combined with a greedy heuristic in deciding which adjacent node to move to. However, the available results (table 10) are quite poor.

The results indicate that AI methods have generally performed poorly, while the computational effort required is high. In contrast, the final group of approximation techniques that shall be analysed – local search methods have achieved a greater degree of success.

3.4 Local Search Methods and Meta-heuristics

In order to derive an algorithmic solution for a given combinatorial optimisation problem $\mathcal{P}$, where $\mathcal{R}$ is the set of feasible solutions to $\mathcal{P}$, it is often necessary to define configurations, i.e. a finite set of solutions, a cost function to be optimised and a generation mechanism, which is a simple prescription to generate a transition from one configuration to another by a small change. Such methods are known as local search or neighbourhood search techniques (Aarts and Lenstra 1997).

In local search the generation mechanism outlines a neighbourhood for each configuration. A neighbourhood, $N(x)$ is a function which defines a simple transition from a solution x to another solution by inducing a change that typically may be viewed as a small perturbation.[5] Each solution $x' \in N(x)$ can be reached directly from x by a single predefined partial transformation of x called a move, and x is said to move to x' when such an transition is performed. (The term *solution* is conceived in the sense of one that satisfies certain structural requirements of a problem at hand, but it is not necessary

[5] Neighbourhoods that involve more substantial changes are sometimes fabricated, e.g. by randomised processes for creating "large-step" transformations in Markov-based methods and by strategic processes for generating influential transformations in tabu search.

that all requirements are feasible.) A symmetry is assumed to hold in the majority of neighbourhood searches where x' is a neighbour of x if and only if x is a neighbour of x'. The objective of these strategies is to progressively perturb the current configuration through a succession of neighbours in order to direct the search to an improved solution. Improvement is sought at each step by standard *ascent* methods, or in some (possibly) larger number of steps by more advanced methods. In cases where solutions may involve infeasibilities, improvement is often defined relative to a modified objective that penalises such infeasibilities.

In these methods the selection of a neighbour is dictated by the choice criteria. Several common selection procedures include: choosing the first lower cost neighbour found - this is known as first improvement and is applied in threshold algorithms. Select the best neighbour in the entire neighbourhood - this is known as best improvement and is done by the shifting bottleneck procedure and insertion algorithms. Choose the best of a sample of neighbours - tabu search adopts this criteria, providing it improves the current solutions present - this is the selection procedure applied in variable depth searches and by genetic algorithms.

From a general perspective, the solution to Π_J can be considered as a collection of local decisions concerning which operation to schedule next. Dorndorf and Pesch (1995) suggest that a framework should be constructed which navigates these local decisions through the search domain in order to determine a high quality global solution in a reasonable amount of time. In such a framework local decisions made by myopic problem specific heuristics are guided beyond local optimality by an underlying meta-strategy. This gives rise to iterated local search algorithms or meta-heuristics, which combine the problem specific properties of the local search with the generic properties of the meta-solver, allowing good solutions to be achieved. Meta-heuristic techniques are the most recent development in approximate search methods for solving complex optimisation problems (Osman and Kelly 1996a). Π_J meta-heuristics are based on the neighbourhood strategies developed by Grabowski *et al.* (1986, 1988), Matsuo *et al.* (1988), Van Laarhooven *et al.* (1992) and Nowicki and Smutnicki (1996).

Before detailing the individual meta-heuristic techniques, some of the general results achieved in local search with regards to complexity issues and theoretical and experimental analysis are described first. One of the earliest analytical works is by Evans (1987) who investigates the effectiveness of local search generating mechanisms from the state-space perspective of artificial intelligence.

Vaessens *et al.* (1995) present a template that captures most of the schemes proposed and they suggest that multi-level local search methods merit more investigation. Pirlot (1996) indicates that few serious comparative studies have been performed with regard to meta-solvers such as Simulated Annealing (SA), Tabu Search (TS) and Genetic Algorithms (GAs) and from his analysis GAs appear to be the weakest of these three methods both empirically and analytically. In a recent work Mattfeld *et al.*

(1998) analyse the structure of the fitness landscape of Π_J with respect to how it appears for an adaptive search heuristic. They indicate that adaptive search heuristics are suitable search techniques for Π_J, all that is required is an effective navigation tool.

In addition to such analytical reviews experimental studies have also been performed. The most recent (Vaessens *et al*. 1996) indicates that, on their comparative test bed (c.f. §4), none of the methods analysed could achieve a total MRE of less than 2 % within a total computer independent execution time of 100 secs. The other main type of analysis performed on local search algorithms concerns complexity issues. The main works in this area are by Johnson *et al*. (1988) who define the complexity class $\mathcal{PLS}$ (polynomial-time local search) and Yannakakis (1990, 1997) who sets a formal framework with regards to the complexity theory of local search, as well as defining the associated complexity issues more clearly.

3.4.1 PROBLEM SPACE BASED METHODS

Problem space based methods are bilevel heuristics that generate many different starting solutions, using fast problem specific constructive techniques, which are then improved by local search. Examples include searching in problem space and in heuristic space and greedy random adaptive search procedures.

3.4.1.1 SEARCH IN PROBLEM SPACE AND IN HEURISTIC SPACE

Storer *et al*. (1992) believe that many methods only tackle the problem of how to search the solution space, not where to search. Consequently they define search neighbourhoods in problem space and in heuristic space rather than in the more traditional solution space, which is based on swapping operations. Both definitions apply a fast base heuristic, *h*, in order to generate an appropriate neighbourhood of solutions. In problem space *h* is applied to perturbed versions of the original problem, while in heuristic space the search relies on the ability to define parameterised versions of *h*. A simple iterative improvement algorithm (c.f. §3.4.2.1) is then used to perform the neighbourhood search.

Storer *et al*. (1995) combine the fast base heuristic generated in Storer *et al*. (1992) with several iterated local search algorithms. Results indicate that the problem space genetic algorithm is clearly the best technique. The good results of the genetic approach are due to the fact that it is relatively easy to encode in heuristic space and problem space in comparison to simulated annealing and tabu search. As a result the simulated annealing and tabu search methods are concluded to lack sophistication.

3.4.1.2 GREEDY RANDOMISED ADAPTIVE SEARCH PROCEDURE (GRASP)

GRASP (Greedy Randomised Adaptive Search Procedure) is a problem space based method that consists of a constructive and an iterative phase. In the construction phase the solution is built one element at a time. All possible elements which can be chosen next are ordered in a candidate list with respect to a greedy function. A number of the best candidates are then placed in a restricted candidate list (RCL). The adaptive nature of GRASP is derived from its ability to update the values associated with

every element, at each iteration, based on the selection just made. While the probabilistic nature of the algorithm stems from the random selection of an element in the RCL.

In the iterative phase a local search procedure is applied, which successively replaces the current solution with a better one in its neighbourhood. This phase stops when no better neighbour can be found. Then the process returns to the construction phase and a new initial solution is built. The algorithm terminates once the desired overall solution or a predetermined number of iterations is achieved, and returns the best solution found so far. Resende (1997) presents an application of GRASP to Π_J. The RCL consists of the operations that when sequenced next would result in the lowest overall completion time of the partial sequence. The algorithm is run for no more than ten million iterations on 10 SGI R10000 processors in parallel.

3.4.1.3 COMPARATIVE ANALYSIS

Table 11 compares the results of these problem space based methods and indicates that only a limited number of such techniques have been constructed. The algorithms created by Storer *et al.* (1992) have only been applied to their own self generated benchmark problems and for some of the harder instances are quite poor. This is due to the fact that an iterative improvement method (c.f. §3.4.2.1) is used as the search technique. However results are achieved quite quickly and by searching in problem space and in heuristic space the results are better than if just a simple iterative improvement method is applied to the more traditional solution space derived by swapping critical operations. Nevertheless by applying other meta-search algorithms Storer *et al.* (1995) are able to achieve quite good solutions within a reasonable time frame on their own as well as the other benchmark problems. Even though GRASP has been applied successfully to several other $\mathcal{NP}$-Complete problems, the limited results available so far for Π_J are quite poor. It suggests that a deterministic structure is more appropriate as a system involving random components is not suited to the job-shop.

3.4.2 THRESHOLD ALGORITHMS

One of the most popular group of iterated local search methods are the threshold algorithms, which choose a new configuration if the difference in cost between the neighbour and the current solution is below a given threshold ($\mathcal{L}$), i.e. $f(x') - f(x) < \mathcal{L}$.

3.4.2.1 ITERATIVE IMPROVEMENT (IM)

The simplest example of a threshold algorithm is that of iterative improvement where thresholds are set at 0, therefore only better configurations are accepted. From an initial randomly generated schedule these methods direct the search to a local optima since the solution is at least as good or better than all the solutions in its neighbourhood. Once it reaches the local optima as it will not accept non improving moves it is trapped. IM is the simplest class of iterated local search technique, forming the basis of other more elaborate methods. While IM accepts the first improving solution in its neighbourhood, a simple variation of this known as steepest descent, evaluates all the moves in its neighbourhood and selects the one that provides the most improvement.

Aarts *et al*. (1994) apply a multi-start iterative improvement algorithm (MSIM) which terminates when a limit on the total running time is reached. This limit is the same for all the algorithms evaluated in their computational study (c.f. table 12). Their study compares MSIM, simulated annealing (SA), threshold accepting (TA) and genetic local search (GLS) when applied with the neighbourhoods of Van Laarhooven *et al*. (1992) and Matsuo *et al*. (1988). The MSIM method begins from a randomly generated sequence. The algorithm then iteratively improves the current solution with a better one in its neighbourhood. Once no better neighbour can be found, i.e. a local optimum is achieved, the search restarts from another randomly chosen initiation point. From which a local optimum is determined. This process is repeated until the termination criteria is satisfied and the best solution found is returned. (Note the similarities to GRASP.)

3.4.2.2 THRESHOLD ACCEPTING (TA)

In threshold accepting (TA) algorithms (Dueck and Scheuer 1990) the thresholds are non negative. $\mathcal{L}$ is initially set at a high value, readily accepting non improving moves, and then gradually decreased to zero so that only better configurations are chosen. The only application of TA to Π_J has been by Aarts *et al*. (1994) who determine suitable parameter values empirically by applying the neighbourhood of Van Laarhooven *et al*. (1992) with 30 different sets of thresholds to a 10×10 problem.

Two variants of TA include the Great Deluge Algorithm (GDA) and Record-to-Record Travel (RRT) (Dueck 1993). These methods also accept weaker solutions as long as they are above a certain threshold. The difference of GDA and RRT with TAs is that they only depend on the appropriate selection of a single parameter. As yet these methods have not been applied to Π_J.[6]

3.4.2.3 LARGE STEP OPTIMISATION

Large Step Optimisation, developed by Martin, Otto and Felten (1989, 1992), is a dual phase optimisation method consisting of a large step and then a small step. Small steps are most commonly performed by meta-heuristics, while large steps involve the application of problem specific techniques allowing local minima to be transcended. Glover and Laguna (1997) indicate it is a subclass of influential diversity and perturbation approaches as it propels a local optima sequence a greater distance than usual from its current location, to solutions of good quality. It is a relatively new technique with as yet only limited application to Π_J.

Testing performed by Lourenço (1993, 1995) indicates that of SA and IM small steps are best performed by SA, however SA takes longer than IM. Of the large step techniques analysed the best methods are the two random machine implementations. One based on the method specified in Carlier (1982) (2rand-car) and the other applying the Earliest Due Date Rule (2rand-edd). The results indicate that large steps with simulated annealing are better than just simulated annealing on its own, however

[6] Richard Bradwell, (Bradwell 1997), indicates that from his limited analysis on Π_J problems with GDA, it is able to provide solutions comparable to SA and as SA has a high computational requirement.

this difference becomes smaller as the amount of time allowed for both methods increases.

In the most recent work (Lourenço and Zwijnenburg 1996) small steps are performed using tabu search and large steps are applied using the 2rand-car technique. Results with an iteration based termination criteria indicate that large-step optimisation with tabu search performs better than just tabu search, which in turn is better than large step optimisation using simulated annealing. A strategy based on large-step optimisation has also been applied by Brucker *et al.* (1996, 1997), however, to a wider domain of scheduling problems.

3.4.2.4 COMPARATIVE ANALYSIS

Table 12 presents the results achieved by Aarts *et al.* (1994), large step optimisation, and job insertion. The TA algorithm is much better than the IM method although, the results of both are very poor even though they are run for the same time as the GLS and SA approaches. One of the reasons for this is that IM destroys a substantial characteristic of optimality (Mattfeld 1996). The insertion technique of Werner and Winkler (1995) is better than TA especially when incorporated with beam search but is not as good as the GLS and the SA methods. This is partly due to the fact that on some of the harder instances the SA and GLS techniques are run for between 2 and 15 hours. Even though the SA approach achieves excellent results 15 hours to solve problems with no more than 300 operations appears excessive. Despite this large run time of all these methods the best solutions, although limited, are achieved by the large step optimisation algorithm of Lourenço (1995). However it also appears to require a phenomenal computing effort. Nevertheless, due to the limitations of iterative improvement and the relative infancy of threshold acceptance and large-step optimisation, simulated annealing is by far the most popular threshold algorithm and has been applied extensively to Π_J.

3.4.2.5 SIMULATED ANNEALING (SA)

In simulated annealing (SA) the thresholds are positive and stochastic. SA is a random oriented search technique that was introduced as an analogy from statistical physics of the computer simulation of the annealing process of a hot metal until its minimum energy state is reached. It is based on the independent proposals of Kirkpatrick *et al.* (1983) and Cerny (1985) who adapt the work of Metropolis *et al.* (1953) to constraint optimisation problems. In SA the configurations are analogous to the states of a solid, while the cost function f and the control parameter c are equivalent to energy and temperature respectively.

A landmark contribution to neighbourhood functions for Π_J is provided by Van Laarhooven *et al.* (1992). It consists only of those moves that are achieved by reversing the processing order of an adjacent pair of critical operations, subject to the condition that these operations must be processed on the same machine. Such a neighbourhood is based on the following properties:

- If $x \in \mathcal{R}$ is a feasible solution, then swapping two adjacent critical operations that require the same machine can never lead to an infeasible solution;
- if the swap of two adjacent non critical operations leads to a feasible solution x', then the critical

path in x' cannot be shorter than the critical path in x (because the critical path in x still exists in x');

- starting with any feasible solution x, there exists some sequence of moves that will reach an optimal solution x^* (known as the connectivity property).[7]

Superimposed on this neighbourhood Van Laarhooven *et al*. (1992) construct a generic SA method which applies an annealing schedule governed by the size of the neighbourhood and bounded by $m(n\text{-}1)$. It is concluded to be robust, while being simple and easy to implement, as it does not require a deep insight into the structure of the problem.

Another important neighbourhood definition is provided by Matsuo *et al*. (1988) who proved that if two critical adjacent operations i and j are to be swapped then the move will never be immediately improving if both $\mathcal{MP}(i)$ and $\mathcal{MS}(j)$ are also on the critical path. This neighbourhood is embedded within their controlled search simulated annealing technique which utilises a good initial schedule (SBI). A look back and lookahead strategy are also applied to improve results.

Yamada *et al*. (1994) formulate a method called Critical Block Simulated Annealing (CBSA) which embeds a neighbourhood structure derived from critical blocks into an SA framework. Initial and final temperature values are defined in terms of an acceptance generated ratio and a reintensification or reannealing process is applied to improve the search. Yamada and Nakano (1995a, 1996a) augment CBSA with the active schedule generator of Giffler and Thompson (1960) and an iterative version of SBP called Bottle Repair (BR). BR is applied when a schedule chosen from the neighbourhood is rejected by the SA method. Most recently, Kolonko (1998) indicates that SA with regards to Π_J is not a convergent process and the standard Π_J neighbourhoods are not symmetric. Based on these results he presents a hybrid method which consists of a genetic algorithm superimposed on an SA scheme.

3.4.2.6 COMPARATIVE ANALYSIS

Although the work of Matsuo *et al*. (1988) and Van Laarhooven *et al*. (1992) is fundamental to local search approaches for Π_J, table 13 indicates that the results of these methods are quite poor. Only when other techniques are incorporated, e.g. genetic algorithms and SBP, does solution quality improve with Yamada and Nakano (1996a) and Kolonko (1998) able to achieve good results. However the main deficiencies of SA approaches are the excessive computing times to achieve good solutions (e.g. on some of the SWV instances Kolonko (1998) requires over 375 million iterations resulting in 44 hours of CPU time) and the highly problem dependent nature of the algorithm where several parameters have to be carefully selected. Possible reasons that these methods require such high times is because many runs have to be performed before good solutions are obtained and on the whole these algorithms are still quite generic applying less problem specific information than many other Π_J methods.

Due to their high computational requirement several strategies have been proposed to try and accelerate these algorithms. Johnson *et al*. (1989) suggest replacing the calculation of the acceptance

[7] However Kolonko (1998) proves that the connectivity property does not imply convergence to an optimum in this neighbourhoods.

probability with a table lookup approximation. Szu and Hartely (1987) have devised a method called fast simulated annealing (FSA) which permits occasional long jumps in order to speed up convergence. Peterson and Anderson (1987) propose a variant of SA called the mean field algorithm which is based on Ising spin glasses. The objective here is to optimise the magnetic energy of the system. Glover and Greenberg (1989) suggest amalgamating SA with AI to improve the move selection procedure which they believe should be based on human reasoning rather than on the criteria of accepting weaker moves with a decreasing probability. One of the most recent suggestions for improvement is by Sadeh and Nakakuki (1996) who construct the Focused Simulated Annealing (FoSA) search technique. FoSA iteratively identifies major inefficiencies in the current solution and attempts to make these inefficiencies more obvious to the search procedure by artificially inflating their values.

3.4.3 GENETIC ALGORITHMS (GAs)

Another generic optimisation technique modelled from the observation of natural processes are Genetic Algorithms (GAs). GAs are based on an abstract model of natural evolution, such that the quality of individuals builds to the highest level compatible with the environment (constraints of the problem) (Holland 1975, Goldberg 1989). The analysis presented here is loosely based on the classification given by Cheng *et al*. (1996) who index all the representation schemes applied in recent years into 9 categories:

1) Operation based	*4) Completion time based*	*7) Priority rule based*
2) Job based	*5) Random keys*	*8) Disjunctive graph based*
3) Job pair relation based	*6) Preference list based*	*9) Machine based*

These representation schemes can be grouped into two basic encoding approaches, direct and indirect. The direct approach encodes a Π_J schedule as a chromosome and the genetic operators are used to evolve these chromosomes into better schedules. Categories 1 to 5 are examples of this strategy. In the indirect approach a sequence of decision preferences is encoded into the chromosome, for example dispatching rules for job assignments, and the genetic operators are applied to improve the ordering of the various preferences. A Π_J schedule is then generated from the sequence of preferences. Categories 6 to 9 are examples of this method.

The earliest GA method applied to Π_J is by Davis (1985) which is an example of an indirect approach. His technique constructs a preferred ordering of operations for each machine. Such an approach is further extended by Falkenauer and Bouffouix (1991) who encode the operations to be processed on a machine as a preference list consisting of a string of symbols. Della Croce *et al*. (1995) also adopt this encoding strategy and crossover operator but with a look-ahead method in order to generate active schedules.

One of the more recent preference list based representations is by Kobayashi *et al*. (1995) where a chromosome is a string of symbols of length *n* and each symbol identifies the operation to be processed on a machine. Tamaki and Nishikawa (1992) apply an indirect representation based on the disjunctive graph. A chromosome consists of a binary string corresponding to an ordered preference list

of disjunctive edges.

One of the earliest direct approaches is by Nakano and Yamada (1991) who create a binary encoding based on the precedence relationship of operations on the same machine. A strategy called forcing is also adopted which modifies a chromosome if an operation can be left-shifted. Yamada and Nakano (1992) are able to improve on this work by defining a crossover operator called GA/GT based on the algorithm of Giffler and Thompson (1960). Here the chromosome is an ordered list of completion times of operations.

One of the more novel direct approaches is that of random keys (Bean 1994). For Π_J (Norman and Bean 1997) each gene (a random key) consists of two parts: an integer from the set $\{1, 2, \ldots, m\}$ and a fraction generated randomly from (0, 1). The integer part of the gene is the machine assignment while the fractional part, sorted in non decreasing order, determines the operation sequence on each machine.

Bierwirth (1995) creates a generalised-permutation genetic algorithm in order to improve existing methods. A chromosome represents a permutation of jobs. In another work Bierwirth *et al.* (1996) analyse three crossover operators, which respectively preserve the relative, position and absolute permutation order of operations. More recently Shi (1997) applies a crossover technique which randomly divides an arbitrarily chosen mate into two subsets, from which offspring are produced.

Despite the plethora of elaborate schemes applied to Π_J by GAs, the results they achieve are quite poor. Several works indicate that GAs are not well suited for fine tuning structures that are very close to optimal solutions (Dorndorf and Pesch 1995, Bierwirth 1995) as crossover operators generally lose their efficiency in order to generate feasible schedules. To overcome some of these problems a form of Evolutionary Computation known as Genetic Local Search (GLS) (or population based local search or memetic search) (Grefenstette 1987; Moscato 1989; Ulder *et al.* 1991) is applied which incorporates local search neighbourhoods into the GA strategy. Within the GLS framework a child conceived from the GA operators is used as the initial solution for the subsequent neighbourhood search which perturbs the offspring to the nearest locally optimal sequence. The local minima is then operated on in the next generation by the traditional genetic recombination operators.

The superiority of GLS over GAs is highlighted by Della Croce *et al.* (1994) who incorporate various neighbourhood structures into their genetic algorithm (Della Croce *et al.* 1995) and provide improved results (c.f. table 15). One of the most well known GLS works is by Dorndorf and Pesch (1995) who propose two approaches to solve Π_J. The first method guides a probabilistic combination of 12 pdrs and is called P-GA while the second approach, referred to as SB-GA, controls the selection of nodes for SBII. Results indicate that SB-GA is superior to P-GA. A similar chromosome evolution framework is applied by Pesch (1993) who applies a GLS strategy to control subproblem selections in his decomposition approach. The subproblems are solved by a constraint propagation approach which finds good solutions by fixing arc directions.

A detailed investigation of the application of evolutionary techniques to Π_J is provided by Mattfeld (1996). Based on this analysis three evolutionary algorithms GA1, GA2 and GA3 are created. The performance of GA3 is superior to GA2 which in turn is superior to GA1. This is because GA1 is a simple GLS algorithm only applying the neighbourhood of Dell'Amico and Trubian (1993) whereas GA2 incorporates distances between individuals by means of spatial populations, while GA3 provides further improvement by incorporating attitude inheritance, which allows individuals to adapt and change to their surroundings.

Yamada and Nakano (1995b) generate two parent schedules, as far apart as possible with respect to their disjunctive graph distance. Starting the search at the first parent they iteratively replace a solution in the current population with an improved sequence biased towards the second parent. Further improvements to this method have been made by Yamada and Nakano (1996b, c) where a biased stochastic replacement scheme that favours solutions that are closer to the second parent and a critical block neighbourhood strategy are applied. Note both these methods are based on the idea of path relinking in tabu search (Glover and Laguna 1997).

3.4.3.1 COMPARATIVE ANALYSIS

Table 14 analyses the results of many of the genetic techniques when applied to FT (06, 10, 20), while table 15 provides a comparative analysis of the best methods on a harder test bed of problems. The majority of GA approaches appear to give poor results due to the difficulties they have with crossover operators and schedule encoding. GLS methods apart from providing better solutions than GA techniques, in general achieve these results with smaller populations, in less generations and are more robust. Although per iteration they require more time, due to the neighbourhood search. The best GLS methods appear to be those integrated with other techniques, for example Mattfeld (1996) with spatial populations and attitude inheritance and Yamada and Nakano (1996b, c) with critical neighbourhoods and various forms of acceptance probabilities. However for very large and difficult instances it is concluded that even GLS cannot provide near optimal solutions in an acceptable time frame (Mattfeld 1996).

3.4.4 TABU SEARCH (TS)

The global iterative optimisation technique Tabu Search (TS) stems from general tenets of intelligent problem solving and is derived from the works of Glover (1977, 1986, 1989, 1990). In essence TS is a simple deterministic oriented search procedure that constrains searching and seeks to transcend local optimality by storing the search history in its memory. It forbids (makes tabu) moves in the neighbourhood having certain attributes, with the aim of guiding the search process away from solutions that (based on available information) appear to duplicate or resemble previously achieved solutions. The short term memory function enables "strategic forgetting" by only making the most recent *t* moves tabu. However tabu status of a move is not absolute. The aspiration criteria allows a tabu move to be selected if it attains a determined level of quality.

Medium and long term memory functions can also be applied to provide a wider exploration of the search space. Medium term or intermediate strategies are based on modifying choice rules to encourage moves and solutions historically found good where these schemes usually return to attractive parts of the search domain and intensify the search in these regions. Long term methods diversify the search into areas not previously explored. Often they are based on modifying choice rules to incorporate attributes into the solution that are not frequently used. More detailed descriptions are given in Glover and Laguna (1997).

Laguna *et al*. (1991, 1993) present some of the earliest TS approaches in scheduling. They create three tabu search strategies based on simple move definitions. Laguna and Glover (1993) apply target analysis to these two works and indicate the inclusion of job transfers in addition to job swaps improves solution quality, reduces computing time and allows larger problems to be solved.

Barnes and Laguna (1993) suggest six primary components that allow effective production scheduling by TS. They also emphasise the need for medium and long term memory schemes which should be coupled with a restricted tabu search structure. They note that in general insertion rather than swapping procedures are preferred as they provide a higher degree of perturbation and that the superposition of TS with other heuristics provides a fertile domain for future work.

Hara (1995) presents a technique known as Minimum Conflict Search (MCS), utilising restrictions such as those defined by the short term memory structure in tabu search. A minimum conflict is the minimal sufficient condition that is not optimal. In Π_J the critical path is indicated to be the minimum conflict. Sun *et al*. (1995) create a simple TS approach based on manipulating active chains (ACM). An active chain is a block on a critical path.[8]

In their TS algorithm Barnes and Chambers (1995) apply the largest tabu tenure in a specified range of short term memory values and each time a new best solution is found, it is stored for later retrieval in a master list. After a certain amount of time the sequence at the top of the list is then used as the starting point of a new search incorporated with the lowest short term memory value and then with all the others in increasing order. Once all memory values have been utilised the solution is taken off the list and the process continues for all of the remaining schedules.

A notable contribution is provided by Taillard (1994) as he incorporates a strategy that accelerates the search by preventing the need to recalculate the start times of all the operations in order to determine the cost of the move. However, this strategy is only effective when semi-active schedules are allowed and as it is unable to give the exact makespan of the move, it can only be considered as a fast estimation strategy. Nevertheless Taillard incorporates this scheme, with the neighbourhood of Van Laarhooven *et al*. (1992), into his TS algorithm. A parallel implementation of the algorithm is also provided with respect to the calculation of the longest paths. However he concludes that parallelisation

[8] Although not based on TS another work of note by these authors, (Sun and Lin 1994), is a dynamic backward scheduling strategy, which is a reversing transform of the normal sequencing instance, known as the forward scheduling problem.

is not suited for Π_J.

Dell'Amico and Trubian (1993) embed a novel initial solution method into their TS algorithm, which utilises a bi-directional property. Two neighbourhood structures are formulated. The first neighbourhood considers the possible inversion of up to three arcs involving i, $\mathcal{MP}(i)$ and $\mathcal{MP}(\mathcal{MP}(i))$ (resp. i, $\mathcal{MS}(i)$ and $\mathcal{MS}(\mathcal{MS}(i))$). If the reversal of anyone of the three arcs is tabu the whole move is forbidden and every time a move is accepted all the component swaps are included in the tabu list. The second neighbourhood structure is based on critical blocks and in order to accelerate the search Dell'Amico and Trubian adapt the estimation strategy of Taillard (1994) to their neighbourhoods.

Currently the best TS method, with respect to solution quality and time, is by Nowicki and Smutnicki (1996) who apply a highly constrained neighbourhood which divides a single critical path into blocks. If there are b blocks in the neighbourhood then for blocks $1 < l < b$ the first and last two operations of each block are swapped. While for block $l = 1$ ($l = b$) only the last (first) two block operations are swapped. The initial solution is generated from the constructive insertion method proposed by Werner and Winkler (1995). The tabu tenure is fixed and a solution recovery strategy is applied where a number of elite schedules are stored in a list. It is important to highlight that the CPU times reported for their algorithm do not include the time taken by the initial solution. As the initial solution has a complexity of $O(n^3m^5)$, Aarts (1996) suggests that the computing times are somewhat misleading. It also appears that the parameters are tuned to perform well on FT 10 and the problems of Lawrence (1984).

Aarts (1996) and Ten Eikelder *et al*. (1997) construct several sequential and parallel TS algorithms for Π_J. They devise a partial evaluation technique known as bowtie which accelerates the estimation strategy of Taillard (1994) by between (35-40) %. Their TS algorithm is as Nowicki and Smutnicki (1996) except that the tabu tenure changes randomly. Their analysis suggests that no more than about 20-30 parallel processors should be used in the implementation of the algorithm. However the speed up results of the parallel implementation are disappointing. There can also be problems with the tabu status of moves, as an arc which has not been reversed (and which might not have existed) can be made tabu. Although this results in a slight loss of accuracy it provides a considerable memory saving.

One of the most recent approaches (Thomsen 1997) has been able to improve the upper bounds of several hard benchmark problems by combining tabu search with a branch and bound algorithm. The BB strategy is used for diversification, however in order to maintain reasonable computing times, it is only performed heuristically, over a restricted number of iterations and to a limited depth in the search tree. The neighbourhood of Nowicki and Smutnicki (1996) is used as the branching scheme.

Several tabu search approaches have also been applied to generalisations of Π_J. Examples include Hurink *et al*. (1994) for Π_J with multi-purpose machines. Brucker and Krämer (1995) for multi-processor tasks on dedicated processors. Dauzère-Pérès and Paulli (1997) for the general multi-

processor Π_J, Brucker and Neyer (1998) for the multi mode Π_J which is a combination of the multi-processor task and multi-purpose machine Π_J and Baar *et al*. (1997) for the resource constrained project scheduling problem.

3.4.5 COMPARATIVE ANALYSIS

The computational study presented in table 16 indicates that in general TS provides the best results of all the techniques, where each TS approach is able to generate good schedules within reasonable computing times. The weakest results are achieved by Sun *et al*. (1995) and are attributed to the fact that this is a very simple TS approach with only a fixed short term memory. Nowicki and Smutnicki (1996) and Thomsen (1997) obtain the best results, the former method in very fast computing times which highlights the strength of their neighbourhood. Although the latter method is better, due to the incorporation of a branch and bound algorithm, it requires substantially more computing time. Nevertheless it highlights the suitability of hybrid methods and combining tabu search with other techniques (c.f. Barnes and Laguna (1993)). However tabu search, like the majority of local search methods, requires many parameters to be carefully fine tuned and appropriately adjusted for each problem, where such decisions are very hard to make.

4. BEST METHOD COMPARISONS

Applegate and Cook (1991) denote the seven problems which they could not solve: LA (21, 27, 29, 38); ABZ (7, 8, 9) as computational challenges as they are much harder than FT 10 and until recently their optimal solutions were unknown, even though every algorithm had been tried on them. Boyd and Burlingame (1996) also note that these seven instances are orders of magnitude harder than those which have already been solved. Consequently Boyd and Burlingame (1996) believe that solving such instances within 24 hours will require an algorithm based on a fundamentally new mathematical approach.

Vaessens *et al*. (1996) also indicate that LA (24, 25, 40) are challenging. As a result Vaessens *et al*. (1996) include these seven challenging instances of Lawrence (1984) as well as the remaining 15×15 problems LA (36, 37, 39), two smaller instances LA (2, 19) and FT 10 when comparing the performance of several methods. Because these 13 instances have also been applied by Balas and Vazacopoulos (1998) in their computational study of several techniques and as they provide a suitable comparative test bed, table 17 illustrates the performance of the best methods from each of the reviewed sections on these 13 problems.

Cherkassky *et al*. (1996) highlight some of the general difficulties encountered in comparing heuristic techniques. They suggest that it is more accurate to discuss comparisons between software implementations. However when making such comparisons it is well known that some machines are obviously more powerful than others and can therefore achieve solutions faster. In order to allow algorithmic comparisons between the results presented here and those of other methods, the Computer Independent Central Processing Unit (CI-CPU) time has been formulated (Vaessens *et al*. 1996). The

CI-CPU time takes into account the capabilities of the machine used and is based on the performance comparisons made by Dongarra (1998). Here TF × CPU = CI-CPU, where TF is the transformation factor. Nevertheless these factors have to be interpreted with a great deal of care.

As only limited computational results are available for some methods, i.e. solutions do not exist for all thirteen instances, mean values are used and comparisons are also made with the optimum, just purely for those instances attempted (table 17). Figure 3 illustrates the average MRE and CI-CPU times of the methods reviewed, where it is apparent that no one technique is clearly superior however approaches that employ certain features perform better. This is emphasised by the lower graph in figure 3 which focuses on the methods which have an average MRE of less than 1.2 %. In this graph only algorithms which have been applied to several instances in the test bed are shown. Hence techniques such as CL'94, SG'96 and PT'96, which have only been applied to one or two of the smaller test bed problems, are removed.

Enhancing the findings of Vaessens *et al.* (1996), two of the approaches in figure 3 which perform well, with respect to both solution quality and time, are the *SB-RGLSk* technique of Balas and Vazacopoulos (1998) and the TS algorithm of Nowicki and Smutnicki (1996). Balas and Vazacopoulos (1998) are able to attain an average MRE of 0.05 % in 999 CI-CPU secs, solving 11 out of the 13 problems optimally, while Nowicki and Smutnicki (1996) solve 7 out of the 13 problems optimally. Although their algorithm has a slightly higher average MRE, 0.20 %, than the SBP method it is considerably faster (on average nearly 10 times). Note that Blazewicz *et al.* (1996) considers Balas and Vazacopoulos (1998) to be a TS approach due to its variable depth formulation. The other TS strategies (Dell'Amico and Trubian 1993, Taillard 1994, Barnes and Chambers 1995) are all integrated with several other methods and are able to provide solutions of high quality within reasonable computing times. In comparison the remaining SBP techniques (Adams *et al.* 1988, Dauzère-Pérès and Lasserre 1993, Balas *et al.* 1995) are not competitive as they are not of hybrid construction. Such conclusions are also made by Ovacik and Uzsoy (1992, 1996) and Holtsclaw and Uzsoy (1996) who show that the performance of SBP clearly improves when incorporated with local search. Without local search SBP is concluded to be no better than pdrs, while requiring much more computing time.

Although SA is not a powerful Π_J technique, hybrid SA approaches are providing strong competition to other methods and the work of Yamada and Nakano (1995a, 1996a) is able to solve 7 of the 10 instances attempted optimally with an average MRE and time of 0.08 % and 132,961 CI-CPU secs respectively. The performance of pure GA implementations is also not very impressive. The GA method of Bierwirth (1995) provides poor results only achieving an average MRE of 3.10 % in 11,445 CI-CPU secs. In addition no instance is solved optimally. However the GLS approach of Yamada and Nakano (1996b, c) has allowed significant advances to be made. Their technique optimally solves 6 out of the 8 instances attempted and achieves an average MRE of 0.17 % within 161,977 CI-CPU secs.

Aside from their excellent solution performance these hybrids require phenomenal computing time. One reason is that they are both based on a randomised acceptance criteria (Sadeh *et al*. 1997) which suggests decisions are unpredictable and unsystematic with no reproducibility. These methods as well as techniques such as GRASP are also referred to as diffused strategies, (Glover and Laguna 1997) as they are designed to either "aimlessly" jump to a new point after each improving phase is completed or spend large periods of time in unattractive regions of the search space. The primary reason for this is that such techniques do not incorporate any memory (apart from the current and best solution) so they have no way of knowing whether it is worthwhile to search a particular region of the solution space, as good sequences have been found there or not to search an area as it has already been evaluated. However recently Sadeh *et al*. (1997) have incorporated "memory" type components into their SA search and have been able to provide good results, while reducing the required computing time.

Other approximation techniques analysed such as priority dispatch rules, threshold accepting and iterative improvement algorithms have produced extremely poor results, while some of the relatively newer approaches such as problem space methods, large step optimisation and artificial intelligence have the potential to be promising techniques. However further analysis is required, especially concerning the testing and tuning of parameter values and the application of more problem specific information, in order to make these methods more suitable for larger and harder problems. In the case of neural networks several authors conclude it is too early to make an assessment of their application to Π_J. Currently, results have only been provided by Sabuncuoglu and Gurgun (1996) which are solely applied to small problems. In addition their method requires excessive effort even for these small instances where many other approaches can solve similar problems in secs on personal computers.

Although branch and bound approaches are able to produce good solutions, the test bed of problems used in this analysis is just at the limit of the problems which can be solved by these methods. Hence none of these approaches can guarantee the solution to problems of slightly higher dimensionality, as the BB technique usually suffers from memory overflow. This occurs because for such instances the system is being completely overwhelmed by the size of the search tree generated. When incorporated with parallel techniques results are also very disappointing. As even on this test set Perregarrd and Clausen (1995) are unable to solve LA (27, 29) due to these limitations. Martin (1996) is able to cope better with this test set and solves all 13 instances optimally, the only method to do so, in a less average time than Perregaard and Clausen (1995). Thereby suggesting that a time orientation representation merits more consideration. Nevertheless, for instances of greater dimensionality Martin (1996) invariably suffers with the same problems. In order to avoid such difficulties it appears prudent to incorporate heuristics into the BB method (c.f. Nuijten and Le Pape (1998) and Thomsen (1997) who are able to achieve an average MRE of 0.12 and 0.05 % in 29,335 and 7,650 CI-CPU secs respectively).

The study indicates that the best results (with respect to time and solution quality) are achieved

from hybrid approximation algorithms involving TS, SBP, GLS and SA approaches as these methods equal and surpass the best solutions for the majority of available benchmark instances by combining several generic and problem specific techniques.

5. FUTURE WORK AND IMPORTANT POINTS TO CONSIDER

Despite the fact that an in depth review and comparative analysis is given here with regards to solution techniques for Π_J, there is much debate concerning how results of various methods, especially those of a heuristic nature, should be presented. Currently very few guide-lines are provided and, consequently no standard procedure is available. Not surprisingly, in many cases it is not clear what interpretation can be made from a set of reported results. In order to try and bridge this gap Barr *et al.* (1995) highlight some of the issues involved in designing and presenting computational experiments to test heuristics. They conclude that rigorous reporting of experimental studies is still in its infancy and there exists a need for larger, more accessible archives of benchmark problems, test beds and codes. Hooker (1995) suggests a more scientific framework should be applied with emphasis on what to measure and how it should be measured. Reeves (1993) also believes that a more rigorous experimental design is required as the choice of a heuristic for a given problem instance is not clear and depends on several factors.

While these are some of the main points of consideration concerning heuristic research other aspects also deserve attention. Even though from an empirical viewpoint local search approaches have shown to be the most appropriate approximation algorithms especially as problem dimensionality increases, theoretical studies have characterised the failings of these methods especially from a worst case perspective. Therefore it is essential to try and understand the theoretical finite-time performance of local search and when and why heuristic approaches are likely to succeed or fail. Other unresolved issues include determining the convergence properties of these approaches, their ability to guarantee optimal solutions and the complexity of finding a local optimum. Currently it is not possible to give a non trivial bound on the number of iterations needed to find a local minima.

There is also no formal method to suggest effective ways of combining meta-heuristic techniques. For example Yamada and Nakano (1995a, 1996a) believe that the most appropriate location for SBP in their SA algorithm is during the acceptance / rejection phase. However, an alternative arrangement might be more suitable. Hence investigations should be concentrated at suggesting how to assemble these components together and conversely how to dismantle a hybrid model and recombine it in such a way that it can become a new and more powerful search tool.

Due to the combinatorial nature of Π_J problems current methods have extreme difficulty in effectively searching the solution space for instances of dimensonality 15. Hence in order to improve existing techniques parallel approaches can be applied to simultaneously perform multiple searches of the neighbourhood. Unfortunately the most recent works applying such approaches have met with limited success (Taillard 1994, Perregaard and Clausen 1995, Boyd and Burlingame 1996, Ten

Eikelder *et al*. 1997 and Resende 1997).

We consider that AI methods such as neural networks have yet to show their true potential. Current NN approaches to Π_J (Sabuncuoglu and Gurgun 1996) are only suitable for small instances and require excessive computing time. Nevertheless if a suitable technique is to be created it will be of hybrid, possibly NN, construction and while being robust emphasis will also be placed on flexibility. By exploiting its inherent parallel distributed processing capability the NN will quickly prune the vast search space and permit an embedded meta-strategy to find the optimal schedule in the reduced solution domain.

Another area that merits exploration is the integration of optimisation and approximation methods such that convergence to strong LBs and proof of optimality are achieved from the exact part of the algorithm using a strong UB attained by the heuristic part of the algorithm. Results obtained by Caseau and Laburthe (1995), Thomsen (1997) and Nuijten Le Pape (1998) indicate their potential.

Based on the analysis performed in this chapter and complementing the conclusions of other recent reviews, this final section identifies particular attributes a good Π_J algorithm should have. Such a system should embody many of the features highlighted above as well as the positive characteristics existing within present techniques. The approach should be based mainly on approximation methods, therefore not suffering from exponential time and variable requirements. Application of bottleneck methods such as SBI (Adams *et al*. 1988) and insertion methods (Werner and Winkler 1995) will help to quickly achieve initial schedules of high quality. In addition powerful neighbourhood structures, for example critical blocks (Nowicki and Smutnicki 1996), should be adopted to select moves. However it need not necessarily be adjacent operations that are permuted within the block. A hybrid construction amalgamating several methods with a meta-strategy to guide a myopic local heuristic to the global optimum is currently the most suitable structure (where as yet no technique with more than two active search levels has been applied (Vaessens *et al*. 1995)). A mechanism to transcend poor local minima should be incorporated by allowing transitions to non improving solutions. However not with an acceptance probability that decreases with time (as in threshold algorithms) such that near the end of the search only improving moves are accepted. A deterministic, rather than a "randomised" method should be used to select the next move and the search should incorporate some form of memory.

The system should only generate active schedules in order to guarantee that the smallest solution space containing the optimal solution is searched. Baker (1974) indicates that with an objective of C_{max} heuristic techniques that produce semi-active schedules are better than those that produce active schedules (which may seem counter intuitive). As a result most techniques generate semi-active sequences (Brucker *et al*. 1994, Mattfeld 1996, Nowicki and Smutnicki 1996, Sabuncuoglu and Bayiz 1997). The reason for this is that up to now these methods have no or very poor diversification strategies and are therefore only able to provide intensification within a certain vicinity of the initial or elite solution. If the global optimum is in this area then it can be found but for many instances this is

not the case. As a result it is believed that some sort of diversification strategy should be applied to navigate the search to unexplored regions of the solution space from which the current methods can perform intensification in the search for the global minima. Also by applying such a diversification scheme poor local minima can easily be transcended hence the generation of active schedules will result in better solutions. The recent results of Thomsen (1997) who applies branch and bound, as the diversification strategy, superimposed on TS emphasises the suitability of such a framework.

6. CONCLUSIONS

The analysis indicates that investigations on Π_J are ever expanding and encompass a wide spectrum of approaches. To ensure that progress is made it is imperative that groups working on completely divergent approaches to a problem can have cross input. Thereby constructing an environment that encourages the congregation of diverse ideas and the creation of new concepts.

This paper has presented a concise review and computational comparison of the main job shop scheduling techniques over the last 40 years highlighting their corresponding strengths and weaknesses as well as detailing the requirement of some of the components that constitute a good Π_J system. In addition pointers for future work and directions for new applications and research are given, complementing and enhancing recent reviews. It is suggested that hybrid systems which incorporate meta-strategies of the form of TS and SBP as the driving mechanism for NNs and other such AI paradigms, in the development of new applications and improved methodologies, are most suitable and in order to improve current methods it is essential that they incorporate strong diversification schemes. Such models would therefore be appropriate not just for Π_J but also to solve production scheduling, network planning and other complex combinatorial problems.

Although considerable progress has been made in recent years the inherent difficulty faced by all of these methods is that no heuristic with a guaranteed performance has been developed so far. Consequently for most approximation methods there exist instances for which they perform badly as it is not completely known under which given circumstances a procedure is likely to succeed or fail. It is therefore apparent that if the current barriers within Π_J are to be surmounted rigorous experimentation and controlled analysis on hybrid approaches is essential.

7. REFERENCES

Aarts, B. J. M. (1996) A Parallel Local Search Algorithm for the Job Shop Scheduling Problem, *Masters Thesis*, Department of Mathematics & Computing Science, Eindhoven University of Technology, Eindhoven, The Netherlands, April.

Aarts, E. H. L., Van Laarhoven, P. J. M., Lenstra, J. K. and Ulder, N. L. J. (1994) A Computational Study of Local Search Algorithms for Job-Shop Scheduling, *ORSA Journal on Computing*, **6**(2), Spring, 118-125.

Aarts, E. H. L. and Lenstra, J. K. (eds) (1997) *Local Search in Combinatorial Optimization*, Wiley, Chichester.

Adams, J., Balas, E. and Zawack, D. (1988) The Shifting Bottleneck Procedure for Job-Shop Scheduling, *Management Science*, March, **34**(3), 391-401.

Akers, S. B. (1956) A Graphical Approach to Production Scheduling Problems, *Operations Research*, vol 4, 244-245.

Alvehus, M. (1997) The Shifting Bottleneck Procedure a Survey, Graduate Course Report, Linköping University, Linköping, Sweden, March.

Applegate, D. and Cook, W. (1991) A Computational Study of the Job-Shop Scheduling Problem, *ORSA Journal on Computing*, Spring, **3**(2), 149-156.

Ashour, S. (1967) A Decomposition Approach for the Machine Scheduling Problem, *International Journal of Production Research* **6**(2), 109-122.

Baar, T., Brucker, P. and Knust, S. (1997) Tabu-Search Algorithms for the Resource-Constrained Project Scheduling Problem, *MIC'97 2nd International Conference on Meta-heuristics*, Sophia-Antipolis, France, 21-24 July.

Baker, K. R. (1974) *Introduction to Sequencing and Scheduling*. John Wiley, New York.

Balas, E. (1969) Machine Scheduling via Disjunctive Graphs: An Implicit Enumeration Algorithm, *Operations Research*, vol 17, 941-957.

Balas, E., Lenstra, J. K. and Vazacopoulos, A. (1995) The One-Machine Problem with Delayed Precedence Constraints and its use in Job Shop Scheduling, *Management Science*, Jan, **41**(1), 94-109.

Balas, E., Lancia, G., Serafini, P., and Vazacopoulos, A. (1998) Job-Shop Scheduling with Deadlines, *Journal of Combinatorial Optimization*, **1**(4), 329-353.

Balas, E., and Vazacopoulos, A. (1998) Guided Local Search with Shifting Bottleneck for Job-Shop Scheduling, *Management Science*, Feb, **44**(2), 262-275.

Baptiste, P. and Le Pape, C. (1995) A Theoretical and Experimental Comparison of Constraint Propagation Techniques for Disjunctive Scheduling, in Mellish, C. S. *IJCAI'95 Proceedings of the 14th International Joint Conference on Artificial Intelligence*, Montréal, Québec, Canada, Aug 19-25, Morgan Kaufmann, vol 1, pp. 600-606.

Baptiste, P., Le Pape, C. and Nuijten, W. P. M. (1995) Constraint-Based Optimization and Approximation for Job-Shop Scheduling, *AAAI-SIGMAN Workshop on Intelligent Manufacturing Systems*, 14th International Joint Conference on Artificial Intelligence, Montréal, Québec, Canada, Aug 19, pp. 5-16.

Barker, J. R. and McMahon, G. B. (1985) Scheduling the General Job-Shop, *Management Science*, May, **31**(5), 594-598.

Barnes, J. W. and Laguna, M. (1993) A Tabu Search Experience in Production Scheduling, *Annals of Operations Research*, vol 41, 141-156.

Barnes, J. W. and Chambers, J. B. (1995) Solving the Job Shop Scheduling Problem Using Tabu Search, *IIE Transactions*, vol 27, 257-263.

Barr, R. S., Golden, B. L., Kelly, J. P., Resende, M. G. C. and Stewart, W. R. (Jr) (1995) Designing and Reporting on Computational Experiments with Heuristic Methods, *Journal of Heuristics*, Fall, **1**(1), 9-32.

Bean, J. (1994) Genetic Algorithms and Random Keys for Sequencing and Optimization, *ORSA J. Computing*, **6**(2), 154-160.

Bensana, E., Bel., G. and Dubois, D. (1988) OPAL: A Multi-Knowledge-Based System for Industrial Job-Shop Scheduling, *International Journal of Production Research*, **26**(5), 795-819.

Bhaskaran, K. and Pinedo, M. (1991) Dispatching, in Salvendy, G. (ed.) *Handbook of Industrial Engineering*, John Wiley and Sons, New York, chapter 83.

Bierwirth, C. (1995) A Generalized Permutation Approach to Job-Shop Scheduling with Genetic Algorithms, *OR Spektrum*, **17**(2-3), 87-92.

Bierwirth, C., Mattfeld, D. C. and Kopfer, H. (1996) On Permutation Representations for Scheduling Problems, in Voigt, H. M. *et al*. (eds) *PPSN'IV Parallel Problem Solving from Nature*, Springer-Verlag, Berlin, Heidelberg, Germany, pp. 310-318.

Blackstone, J. H. Jr, Phillips, D. T. and Hogg, G. L. (1982) A State of the Art Survey of Dispatching Rules for Manufacturing Job-Shop Operations, *International Journal Of Production Research*, Jan-Feb, vol 20, 27-45.

Blazewicz, J., Dror, M. and Weglarz, J. (1991) Mathematical Programming Formulations for Machine Scheduling: A Survey, *European Journal of Operational Research*, Invited Review, **51**(3), April 15, 283-300.

Blazewicz, J., Domschke, W. and Pesch, E. (1996) The Job-Shop Scheduling Problem: Conventional and New Solution Techniques, *European Journal of Operational Research*, **93**(1), 23rd August, 1-33.

Bowman, E. H. (1959) The Schedule-Sequencing Problem, *Operations Research*, vol 7, 621-624.

Boyd, E. A. and Burlingame, R. (1996) A Parallel Algorithm for Solving Difficult Job-Shop Scheduling Problems, Operations Research Working Paper, Department of Industrial Engineering, Texas A&M University, College Station, Texas 77843-3131, USA.

Bradwell, R. (1997) Personal Communication on the 17/10/97 with Mr Richard Bradwell, Department of Computer Science, University of Aberdeen, Aberdeen, Scotland, UK - rbradwel@csd.abdn.ac.uk.

Bratley, P., Florian, M. and Robillard, P. (1973) On Sequencing with Earliest Starts and Due Dates with Application to Computing Bounds for the ($n/m/G/F_{max}$) Problem, *Naval Res. Logist. Quart.*, vol 20, 57-67.

Brooks, G. H. and White, C. R. (1965) An Algorithm for Finding Optimal or Near Optimal Solutions to the Production Scheduling Problem, *Journal Of Industrial Engineering*, January, **16**(1), 34-40.

Brown, A. P. G. and Lomnicki, Z. A. (1966) Some Applications of the Branch-and-Bound Algorithm to the Machine Scheduling Problem, *Operational Research Quarterly*, **17**(2), 173-186.

Brucker, P. (1988) An Efficient Algorithm for the Job-Shop Problem with Two Jobs, *Computing*, vol 40, 353-359.

Brucker, P. and Jurisch, B. (1993) A New Lower Bound for the Job-Shop Scheduling Problem, *European Journal of Operational Research*, vol 64, 156-167.

Brucker, P., Jurisch, B. and Sievers, B. (1994) A Branch and Bound Algorithm for the Job-Shop Scheduling Problem, *Discrete Applied Mathematics*, vol 49, 109-127.

Brucker, P. and Krämer, A. (1995) Shop Scheduling Problems with Multiprocessor Tasks on Dedicated Processors, *Annals of Operations Research*, vol 57, 13-27.

Brucker, P., Hurink, J. and Werner, F. (1996) Improving Local Search Heuristics for Some Scheduling Problems Part I, *Discrete Applied Mathematics*, **65**(1-3), 7th March, 97-122.

Brucker, P., Hurink, J. and Werner, F. (1997) Improving Local Search Heuristics for Some Scheduling Problems Part II, *Discrete Applied Mathematics*, **72**(1-2), 10th January, 47-69.

Brucker, P. and Neyer, J. (1997) Tabu Search for the Multi-Mode Job-Shop Problem, *Operations Research Spektrum*, vol 20, 21-28.

Bullers, W. I., Nof, S. Y. and Whinston, A. B. (1980) Artificial Intelligence in Manufacturing Planning and Control, *AIIE Transactions*, Dec, 351-363.

Carlier, J. (1982) The One-Machine Sequencing Problem, *European Journal of Operational Research*, vol 11, 42-47.

Carlier, J. and Pinson, E. (1989) An Algorithm for Solving the Job Shop Problem. *Management Science*, Feb, **35**(2), 164-176.

Carlier, J. and Pinson, E. (1990) A Practical use of Jackson's Preemptive Schedule for Solving the Job-Shop Problem, *Annals of Operations Research*, vol 26, 269-287.

Carlier, J. and Pinson, E. (1994) Adjustment of Heads and Tails for the Job-Shop Problem, *European Journal of Operational Research*, Oct 27, **78**(2), 146-161.

Caseau, Y. and Laburthe, F. (1994) Improved CLP Scheduling with Task Intervals, in Van Hentenryck, P. (ed) *ICLP'94 Proceedings of the Eleventh International Conference on Logic Programming*, MIT Press.

Caseau, Y. and Laburthe, F. (1995) Disjunctive Scheduling with Task Intervals, LIENS Technical Report n° 95-25, Laboratoire d'Informatique de l'Ecole Normale Supérieure Département de Mathématiques et d'Informatique, 45 rue d'Ulm, 75230 Paris, France.

Cedimoglu, I. H. (1993) Neural Networks in Shop Floor Scheduling. *Ph. D. Thesis*, School of Industrial and Manufacturing Science, Cranfield University, U.K.

Cerny, V. (1985) Thermodynamical Approach to the Travelling Salesman Problem: An Efficient Simulation Algorithm, *Journal of Optimization Theory and Applications*, vol 45, 41-51.

Chang, S. H. and Nam, B. H. (1993) Linear Programming Neural Networks for Job-Shop Scheduling, *IJCNN International Joint Conference on Neural Networks*, Nagoya, Japan, 25-29 Oct, vol 2, 1557-1560.

Chang, Y. L, Sueyoshi, T. and Sullivan, R. S. (1996) Ranking Dispatching Rules by Data Envelopment Analysis in a Job-Shop Environment, *IIE Transactions*, **28**(8), 631-642.

Cheng, R., Gen, M., Tsujimura, Y. (1996) A Tutorial Survey of Job-Shop Scheduling Problems using Genetic Algorithms-I. Representation, *Computers & Industrial Engineering*, **30**(4), 983-997.

Cheng, C-C. and Smith, S. F. (1997) Applying Constraint Satisfaction Techniques to Job Shop Scheduling, *Annals of Operations Research*, vol 70, 327-357.

Cherkassky, V. and Zhou, D. N. (1992) Comparison of Conventional and Neural Network Heuristics for Job-Shop Scheduling, *Proceedings of the SPIE - International Society for Optical Engineering*, Orlando, 21-24 Apr, vol 2, **1710**(1), 815-825.

Cherkassky, V., Gehring, D. and Mulier, F. (1996) Comparison of Adaptive Methods for Function Estimation from Samples, *IEEE Transactions on Neural Networks*, July, **7**(4), 969-984.

Cheung, J. Y. (1994) Scheduling, in Dagli, C. H. (ed) *Artificial Neural Networks for Intelligent Manufacturing*, Chapman and Hall, London, Chapter 8, pp. 159-193.

Chryssolouris, G., Wright, K. and Pierce, J. (1988) A Simulator for Manufacturing Systems Based on Artificial Intelligence, *Symposium on Advances in Manufacturing Systems Engineering*, Nov 8 - Dec 2, Chicago, Ilinois, pp. 1-13.

Chu, C., Portmann, M. C. and Proth, J. M. (1992) A Splitting-Up Approach to Simplify Job-Shop Scheduling Problems, *International Journal of Production Research* **30**(4), 859-870.

Colorni, A., Dorigo, M., Maniezzo, V. and Trubian, M. (1994) Ant-System for Job-Shop Scheduling, *Belgian Journal of Operations Research, Statistics and Computer Science*, **34**(1), 39-54.

Conway, R. W., Maxwell, W. L. and Miller, L. W. (1967) *Theory of Scheduling*, Addison-Wesley, Reading Massachusetts.

Cook, S. A. (1971) The Complexity of Theorem Proving Procedures, *Proceedings of the Third Annual ACM*

Symposium on the Theory of Computing, Association of Computing Machinery, New York, pp. 151-158.

Crowston, W. B., Glover, F., Thompson, G. L., Trawick, J. D. (1963) Probabilistic and Parametric Learning Combinations of Local Job-Shop Scheduling Rules, ONR Research Memorandum, No. 117, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, PA 15213, USA.

Dagli, C. H. and Huggahalli, R. (1991) A Neural Net Architecture for Faster Dynamic Scheduling in Manufacturing Systems, *Proc IEEE Int Conf Rob Autom*, vol 3, Apr 9-11, pp. 2408-2413.

Dagli, C. H., Lammers, S. and Vellanki, M. (1991) Intelligent Scheduling in Manufacturing Using Neural Networks, *Journal of Neural Network Computing Technology Design and Applications*, **2**(4), 4-10.

Dagli, C. H. and Sittisathanchai, S. (1995) Genetic Neuro-Scheduler - A New Approach for Job-Shop Scheduling, *International Journal of Production Economics*, **41**(1-3), 135-145.

Dauzère-Pérès, S. and Lasserre, J. B. (1993) A Modified Shifting Bottleneck Procedure for Job-Shop Scheduling, *International Journal of Production Research*, April, **31**(4), 923-932.

Dauzère-Pérès, S. (1995) A Procedure for the One Machine Sequencing Problem with Dependent Jobs, *European Journal of Operational Research*, vol 81, 579-589.

Dauzère-Pérès, S. and Paulli, J. (1997) An Integrated Approach for Modeling and Solving the General Multiprocessor Job-Shop Scheduling Problem Using Tabu Search, *Annals of Operations Research*, vol 70, 281-306.

Davidor, Y., Yamada, T. and Nakano, R. (1993) The Ecological Framework II: Improving GA Performance at Virtually Zero Cost, *ICGA'5 5th International Conference on Genetic Algorithms*, pp. 171-176.

Davis, L. (1985) Job-Shop Scheduling with Genetic Algorithm, in Grefenstette J. J. (ed) *Proceedings of the 1st International Conference on Genetic Algorithms and their Applications*, Pittsburgh, PA, USA, Lawrence Erlbaum, pp. 136-140.

Della Croce, F., Menga, G., Tadei, R., Cavalotto, M. and Petri, L. (1993) Cellular Control of Manufacturing Systems, *European Journal of Operational Research*, vol 69, 498-509.

Della Croce, F., Tadei, R. and Rolando, R. (1994) Solving a Real World Project Scheduling Problem with a Genetic Approach, *Belgian Journal of Operations Research, Statistics and Computer Science*, **33**(1-2), 65-78.

Della Croce, F., Tadei, R. and Volta, G. (1995) A Genetic Algorithm for the Job Shop Problem, *Computers and Operations Research*, Jan, **22**(1), 15-24.

Dell'Amico, M. and Trubian, M. (1993) Applying Tabu Search to the Job-Shop Scheduling Problem, *Annals of Operations Research*, vol 41, 231-252.

Demirkol, E., Mehta, S. and Uzsoy, R. (1997) A Computational Study of Shifting Bottleneck Procedures for Shop Scheduling Problems, *Journal of Heuristics*, Winter, **3**(2), 111-137.

Denebourg, J. L., Pasteels, J. M. and Verhaeghe, J. C. (1983) Probabilistic Behaviour in Ants: A Strategy of Errors ?, *Journal of Theoretical Biology*, vol 105, 259-271.

Dongarra, J. J. (1998) Performance of Various Computers Using Standard Linear Equations Software, Technical Report CS - 89 - 85, Computer Science Department, University of Tennessee, Knoxville, TN 37996-1301 and Mathematical Sciences Section, Oak Ridge National Laboratory, Oak Ridge, TN 37831, Tennessee, USA, September 25.

Dorndorf, U. and Pesch, E. (1995) Evolution Based Learning in a Job-Shop Scheduling Environment, *Computers and Operations Research,* Jan, **22**(1), 25-40.

Dueck, G. and Scheuer, T. (1990) Threshold Accepting: A General Purpose Optimization Algorithm Appearing Superior to Simulated Annealing, *Journal of Computational Physics*, vol 90, 161-175.

Dueck, G. (1993) New Optimization Heuristics: The Great Deluge Algorithm and the Record-to-Record Travel, *Journal of Computational Physics*, vol 104, 86-92.

Erschler, J., Roubellat, F. and Vernhes, J. P. (1976) Finding Some Essential Characteristics of the Feasible Solutions for a Scheduling Problem, *Operations Research*, **24**(4), Jul-Aug, (Technical Note), 774-783.

Evans, J. R. (1987) Structural Analysis of Local Heuristics in Combinatorial Optimization, *Computers and Operations Research*, **14**(6), 465-477.

Falkenauer, E. and Bouffouix, S. (1991) A Genetic Algorithm for the Job-Shop, *Proceedings of the IEEE International Conference on Robotics and Automation*, Sacremento, California, USA, pp. 824-829.

Fang, H. L., Ross, P. and Corne, D. (1993) A Promising Genetic Algorithm Approach to Job-Shop Scheduling, Rescheduling and Open-Shop Scheduling Problems, *ICGA'5 Proceedings of the 5th International Conference on Genetic Algorithms*, Morgan Kaufmann Publishers, San Mateo, California, pp. 375-382.

Fisher, H. and Thompson, G. L. (1963) Probabilistic Learning Combinations of Local Job-Shop Scheduling Rules, in Muth, J. F. and Thompson, G. L. (eds) *Industrial Scheduling*, Prentice Hall, Englewood Cliffs, New Jersey, Ch 15, pp. 225-251.

Fisher, M. L. (1973a) Optimal Solution of Scheduling Problems using Lagrange Multipliers: Part I, *Operations Research*, vol 21, 1114-1127.

Fisher, M. L. (1973b) Optimal Solution of Scheduling Problems using Lagrange Multipliers: Part II,

Symposium on the Theory of Scheduling and its Applications, Springer, Berlin.

Fisher, M. L., Lageweg, B. J., Lenstra, J. K. and Rinnooy Kan, A. H. G. (1983) Surrogate Duality Relaxation for Job-Shop Scheduling, *Discrete Applied Mathematics*, **5**(1), 65-75.

Fisher, M. L. and Rinnooy Kan, A. H. G. (1988) The Design, Analysis and Implementation of Heuristics, *Management Science*, March, **34**(3), 263-265.

Fizzano, P., Stein, C., Karger, D. and Wein, J. (1997) Job Scheduling in Rings, *Journal of Parallel and Distributed Computing*, **45**(2), 122-133.

Florian, M., Trépant, P. and McMahon, G. (1971) An Implicit Enumeration Algorithm for the Machine Sequencing Problem, *Management Science Application Series*, August, **17**(12), B782-B792.

Foo, S. Y. and Takefuji, Y. (1988a) Stochastic Neural Networks for Solving Job-Shop Scheduling: Part 1. Problem Representation, in Kosko B. (ed.), *IEEE International Conference on Neural Networks*, San Diego, USA, 24-27 Jul, vol 2, pp. 275-282.

Foo, S. Y. and Takefuji, Y. (1988b) Stochastic Neural Networks for Solving Job-Shop Scheduling: Part 2. Architecture and Simulations, in Kosko B. (ed.), *IEEE International Conference on Neural Networks*, San Diego, USA, 24-27 Jul, vol 2, pp. 283-290.

Foo, S. Y. and Takefuji, Y. (1988c) Integer Linear Programming Neural Networks for Job-Shop Scheduling, in Kosko B. (ed.), *IEEE International Conference on Neural Networks*, San Diego, California, USA, 24-27 Jul, vol 2, pp. 341-348.

Foo, S. Y., Takefuji, Y. and Szu, H. (1994) Job-Shop Scheduling Based on Modified Tank-Hopfield Linear Programming Networks, *Eng. Appl. Artif. Intell.*, Jun, **7**(3), 321-327.

Foo, S. Y., Takefuji, Y. and Szu, H. (1995) Scaling Properties of Neural Networks for Job-Shop Scheduling, *Neurocomputing*, **8**(1), 79-91.

Fox, M. S. (1987) *Constraint - Directed Search: A Case Study of Job-Shop Scheduling*, Research Notes in Artificial Intelligence, Pitman Publishing, London.

Fox, M. S. and Sadeh, N. (1990) Why Is Scheduling Difficult ? A CSP Perspective, in Aiello L. (ed) *ECAI-90 Proceedings of the 9th European Conference on Artificial Intelligence*, August 6-10, Stockholm, Sweden, pp. 754-767.

Fox, M.S. and Sycara, K. (1990) Overview of cortes: A Constraint Based Approach to Production Planning, Scheduling and Control, *Proceedings of the Fourth International Conference on Expert Systems in Production and Operations Management*, May.

French, S. (1982) *Sequencing and Scheduling - An Introduction to the Mathematics of the Job-Shop,* Ellis Horwood, John-Wiley & Sons, New York.

Fukumori, K. (1980) Fundamental Scheme for Train Scheduling (Application of Range-Constriction Search), AI Memo No. 596, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, USA, September.

Gang, S. and Shuchun, W. (1994) Job-Shop Scheduling Solution with Neural Networks, *Chinese Journal of Electronics*, July, **3**(3), 48-52.

Garey, M. R., Johnson, D. S. and Sethi, R. (1976) The Complexity of Flow Shop and Job-Shop Scheduling, *Mathematics of Operations Research*, May, **1**(2), 117-129.

Garey, M. R. and Johnson, D. S. (1979) *Computers and Intertractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman, San Francisco.

Gere, W. S. Jr. (1966) Heuristics in Job-Shop Scheduling, *Management Science*, vol 13, 167-190.

Giffler, B. and Thompson, G. L. (1960) Algorithms for Solving Production Scheduling Problems, *Operations Research*, **8**(4), 487-503.

Glover, F. (1977) Heuristics for Integer Programming Using Surrogate Constraints, *Decision Sciences*, **8**(1), 156-166.

Glover, F. (1986) Future Paths for Integer Programming and Links to Artificial Intelligence, *Computers and Operations Research*, **13**(5), 533-549.

Glover, F. and Greenberg, H. J. (1989) New Approaches for Heuristic Search: A Bilateral Linkage with Artificial Intelligence, *European Journal of Operations Research*, **39**(2), 119-130.

Glover, F. (1989) Tabu Search - Part I, *ORSA Journal on Computing*, **1**(3), Summer, 190-206.

Glover, F. (1990) Tabu Search - Part II, *ORSA Journal on Computing*, **2**(1), Winter, 4-32.

Glover, F. and Laguna, M. (1997) *Tabu Search*, Kluwer Academic Publishers, Norwell, MA.

Goldberg, D. E. (1989) *Genetic Algorithms in Search Optimisation and Machine Learning*, Addison-Wesley, Reading, Massachusetts.

Gonzalez, T. and Sahni, S. (1978) Flowshop and Jobshop Schedules: Complexity and Approximation, *Operations Research*, vol 20, 36-52.

Grabot, B. and Geneste, L. (1994) Dispatching Rules in Scheduling: A Fuzzy Approach, *International Journal of Production Research,* April, **32**(4), 903-915.

Grabowski, J., Nowicki E. and Zdrzalka, S. (1986) A Block Approach for Single Machine Scheduling with

Release Dates and Due Dates, *European Journal of Operational Research*, August, **26**(2), 278-285.

Graham, R. L., Lawler, E. L., Lenstra, J. K. and Rinnooy Kan, A. H. G. (1979) Optimisation and Approximation in Deterministic Sequencing and Scheduling: A Survey, *Annals of Discrete Mathematics*, vol 5, 287-326.

Greenberg, H. (1968) A Branch and Bound Solution to the General Scheduling Problem, *Operations Research*, March, **16**(2), 353-361.

Grefenstette, J. J. (1987) Incorporating Problem Specific Knowledge into Genetic Algorithms, in Davis, L. (ed) *Genetic Algorithms and Simulated Annealing*, Pitman, pp. 42-60.

Hanada, A. and Ohnishi, K. (1993) Near Optimal Job-Shop Scheduling using Neural Network Parallel Computing, *IECON'93 Proceedings of the 19th Annual IEEE International Conference on Industrial Electronics, Control, and Instrumentation*, Maui, Hawaii, Nov 15-19, vol 1, pp. 315-320.

Hara, H. (1995) Job-Shop Scheduling by Minimal Conflict Search, *Japanese Artificial Intelligence Society*, January, **10**(1), 88-93.

Harvey, W. D. and Ginsberg, M. L. (1995) Limited Discrepancy Search, *IJCAI'95 International Joint Conference on Artificial Intelligence*, Montréal, Québec, Canada, pp. 607-613.

Haupt, R. (1989) A Survey of Priority Rule Based Scheduling, *OR Spektrum*, vol 11, 3-16.

Hefetz, N. and Adiri, I. (1982) An Efficient Optimal Algorithm for the Two-Machines Unit-Time Job-Shop Schedule-Length Problem, *Mathematics of Operations Research*, vol 7, 354-360.

Hoitomt, D. J., Luh, P. B. and Pattipati, K. R. (1993) Practical Approach to Job-Shop Scheduling Problems, *IEEE Trans Rob Autom*, Feb, **9**(1), 1-13.

Holland, J. H. (1975) *Adaptation in Natural and Artificial Systems*, The University of Michigan Press, Ann Arbor.

Holtsclaw, H. H. and Uzsoy, R. (1996) Machine Criticality Measures and Subproblem Solution Procedures in Shifting Bottleneck Methods: A Computational Study, *Journal of the Operational Research Society*, vol 47, 666-677.

Hooker, J. N. (1995) Testing Heuristics: We Have it all Wrong, *Journal of Heuristics*, Fall, **1**(1), 33-42.

Hoong, L. W., Yeo, K. T. and Sim, S. K. (1991) Application of Neural Network in a Job Shop Environment, *ICCIM'91 International Conference on Computer Integrated Manufacturing*, Sept 30- Oct 4, Singapore, pp. 559-562.

Hopfield, J. J., and Tank, D. W. (1985) Neural Computational of Decisions in Optimization Problems, *Biological Cybernetics*, vol 52, 141-52.

Hurink, J., Jurisch, B. and Thole, M. (1994) Tabu Search for the Job-Shop Scheduling Problem with Multi-Purpose Machines, *OR-Spektrum*, vol 15, 205-215.

Ignall, E. and Schrage, L. (1965) Application of the Branch and Bound Technique to some Flow-Shop Scheduling Problems, *Operations Research*, vol 13, 400-412.

Ivens, P. and Lambrecht, M. (1996) Extending the Shifting Bottleneck Procedure to Real-Life Applications, *European Journal of Operational Research*, vol 90, 252-268.

Jain, A. S., and Meeran, S. (1998) Job-Shop Scheduling Using Neural Networks, *International Journal of Production Research*, **36**(5), May, 1249-1272.

Jain, A. S. (1998) A Multi-Level Hybrid Framework for the Deterministic Job-Shop Scheduling Problem, *Ph. D. Thesis*, Dept. of APEME, University Of Dundee, Dundee DD1 4HN, Scotland, UK, October.

Jackson, J. R. (1955) Scheduling a Production Line to Minimise Maximum Tardiness, Research Report 43, Management Science Research Projects, University of California, Los Angeles, USA.

Jackson, J. R. (1956) An Extension of Johnson's Result on Job Lot Scheduling, *Naval Research Logistics Quarterly*, **3**(3), 201-203.

Jackson, J. R. (1957) Simulation Research on Job-Shop Production, *Naval Research Logistics Quarterly*, vol 4, 287-295.

Jeremiah, B., Lalchandani, A. and Schrage, L. (1964) Heuristic Rules Toward Optimal Scheduling, Research Report, Department of Industrial Engineering, Cornell University.

Johnson, D. S., Papadimitriou, C. H. and Yannakakis, M. (1988) How Easy is Local Search ?, *Journal of Computer and System Sciences*, **37**(1), August, 79-100.

Johnson, D. S., Aragon, C. R., McGeoch, L. A. and Schevon, C. (1989) Optimization by Simulated Annealing: An Experimental Evaluation; Part I, Graph Partitioning, *Operations Research*, **37**(6), Nov-Dec, 865-892.

Johnson, S. M. (1954) Optimal Two- and Three-Stage Production Schedules with Set-Up Times Included, *Naval Research Logistics Quarterly*, vol 1, 61-68.

Karp, R. M. (1972) Reducibility Among Combinatorial Problems, in Miller, R. E. and Thatcher, J. W. (eds) *Complexity of Computer Computations*, Plenum Press, New York, pp. 85-104.

Khaw, J., Siong, L. B., Lim, L., Yong, D., Jui, S. K. and Fang, L. C. (1991) Shop Floor Scheduling using a Three-Dimensional Neural Network Model, *International Conference on Computing Integrated Manufacturing*, Sept 30- Oct 4, Singapore, pp. 563-566.

Kim, S. Y., Lee, Y. H. and Agnihotri, D. (1995) A Hybrid Approach for Sequencing Jobs using Heuristic Rules and Neural Networks, *Production Planning and Control*, **6**(5), 445-454.

Kirkpatrick, S., Gelatt, C. D. Jr. and Vecchi, M. P. (1983) Optimization by Simulated Annealing, *Science*, **220**(4598), 13 May, 671-680.

Kobayashi, S., Ono, I. and Yamamura, M. (1995) An Efficient Genetic Algorithm for Job Shop Scheduling Problems, *Proceedings of the Sixth International Conference on Genetic Algorithms*, Morgan Kaufmann Publishers, San Francisco, CA, pp. 506-511.

Kolonko, M. (1998) Some New Results on Simulated Annealing Applied to the Job Shop Scheduling Problem, *to appear in the European Journal of Operational Research*.

Krüger, K., Shakhlevich, N. V., Sotskov, Y. N. and Werner, F. (1995) A Heuristic Decomposition Algorithm for Scheduling Problems on Mixed Graphs, *Journal of the Operational Research Society*, vol 46, 1481-1497.

Lageweg, B. J., Lenstra, K. and Rinnooy Kan, A. H. G. (1977) Job-Shop Scheduling by Implicit Enumeration, *Management Science*, December, **24**(4), 441-450.

Lageweg, B. J. (1982) Combinatorial Planning Models, *Ph. D. Thesis*, Mathematisch Centrum, Amsterdam, The Netherlands.

Lageweg, B. J. (1984) Private Communication with Peter J. M. Van Laarhooven; Emile H. L. Aarts and Jan Karel Lenstra discussing the achievement of a makespan of 930 for FT 10.

Laguna, M., Barnes, J. W. and Glover, F. W. (1991) Tabu Search Methods for a Single Machine Scheduling Problem, *Journal of Intelligent Manufacturing*, vol 2, 63-74.

Laguna, M., Barnes, J. W. and Glover, F. W. (1993) Intelligent Scheduling with Tabu Search: An Application to Jobs with Linear Delay Penalties and Sequence-Dependent Setup Costs and Times, *Journal of Applied Intelligence*, vol 3, 159-172.

Laguna, M. and Glover, F. (1993) Integrating Target Analysis and Tabu Search for Improved Scheduling Systems, *Expert Systems with Applications*, vol 6, 287-297.

Lawler, E. L. (1973) Optimal Sequencing of a Single Machine Subject to Precedence Constraints, *Management Science*, vol 19, 544-546.

Lawler, E. L., Lenstra, J. K., Rinnooy Kan, A. H. G. and Shmoys, D. B. (1993) Sequencing and Scheduling: Algorithms and Complexity, in Graves, S. C., Rinnooy Kan, A. H. G., Zipkin, P. H. (eds), *Handbook in Operations Research and Management Science*, Volume 4: Logistics of Production and Inventory, North Holland, Amsterdam.

Lawrence, S. (1984) Supplement to Resource Constrained Project Scheduling: An Experimental Investigation of Heuristic Scheduling Techniques, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, USA, October.

Lee, Y. H., Bhaskaran, K. and Pinedo, M. (1992) A Heuristic to Minimise the Total Weighted Tardiness with Sequence Dependent Setups, Technical Report, IEOR Dept, Columbia University, NY.

Lenstra, J. K., Rinnooy Kan, A. H. G. and Brucker, P. (1977) Complexity of Machine Scheduling Problems, *Annals of Discrete Mathematics*, vol 7, 343-362.

Lenstra, J. K. and Rinnooy Kan, A. H. G. (1979) Computational Complexity of Discrete Optimization Problems, *Annals of Discrete Mathematics*, vol 4, 121-140.

Lepape, C. (1985) SOJA: A Daily Workshop Scheduling System, *Expert System*, vol 85, 95-211.

Lo, C-C. and Hsu, C-C. (1993) A Parallel Distributed Processing Technique for Job-Shop Scheduling Problems, *IJCNN International Joint Conference on Neural Networks*, Nagoya, Japan, 25-29 Oct, vol 2, pp. 1602-1605.

Lo, Z-P. and Bavarian, B. (1993) Multiple Job-Shop Scheduling with Artificial Neural Networks, *Computers in Electrical Engineering*, **19**(2), Mar, 87-101.

Lomnicki, Z. A. (1965) A "Branch-and-Bound" Algorithm for the Exact Solution of the Three-Machine Scheduling Problem, *Operational Research Quarterly*, **16**(1), 89-100.

Lourenço, H. R. D. (1993) A Computational Study of the Job-Shop and the Flow-Shop Scheduling Problems, *Ph. D. Thesis* TR - 1060, School of Operations Research & Industrial Engineering (OR & IE), Cornell University, Ithaca, New York 14853-3801, July.

Lourenço, H. R. D. (1994) The One-Machine Scheduling Problem with Lags, Working Paper nº4/94, Departamento de Estatística e Investigação Operacional, Faculdade de Ciências, Universidade de Lisboa, Lisboa, Portugal, September 5.

Lourenço, H. R. D. (1995) Job-Shop Scheduling: Computational Study of Local Search and Large-Step Optimization Methods. *European Journal of Operational Research*, 8 June, vol 83, 347-364.

Lourenço, H. R. D. and Zwijnenburg, M. (1996) Combining the Large-Step Optimization with Tabu-Search: Application to the Job-Shop Scheduling Problem, in Osman, I. H. and Kelly, J. P. (eds) *Meta-heuristics: Theory and Applications*, Kluwer Academic Publishers, Boston, MA, USA, Chapter 14, pp. 219-236.

Manne, A. S. (1960) On the Job-Shop Scheduling Problem, *Operations Research*, vol 8, 219-223.

Martin, O., Otto, S. W. and Felten, E. W. (1989) Large-Step Markov Chains for traveling salesman problem, *Complex Systems*, vol 5, 299-326.

Martin, O., Otto, S. W. and Felten, E. W. (1992) Large-Step Markov Chains for TSP Incorporating Local Search Heuristics, *Operations Research Letters*, vol 11, 219-224.

Martin, P. D. (1996) A Time-Oriented Approach to Computing Optimal Schedules for the Job-Shop Scheduling Problem, *Ph. D. Thesis*, School of Operations Research & Industrial Engineering, Cornell University, Ithaca, New York 14853-3801, August.

Matsuo, H., Suh, C. J. and Sullivan, R. S. (1988) A Controlled Search Simulated Annealing Method for the General Job-Shop Scheduling Problem, Working Paper #03-04-88, Graduate School of Business, The University of Texas at Austin, Austin, Texas, USA.

Mattfeld, D. C., Kopfer, H. and Bierwirth, C. (1994) Control of Parallel Population Dynamics by Social-Like Behaviour of GA-Individuals, *PPSN'3 Proceedings of the Third International Conference on Parallel Problem Solving from Nature*, Springer-Verlag, pp. 15-25.

Mattfeld, D. C. (1996) *Evolutionary Search and the Job Shop: Investigations on Genetic Algorithms for Production Scheduling*, Physica-Verlag, Heidelberg, Germany.

Mattfeld, D. C., Bierwirth, C. and Kopfer, H. (1998) A Search Space Analysis of the Job Shop Scheduling Problem, *to appear in Annals of Operations Research*.

McMahon, G. B. and Florian, M. (1975) On Scheduling with Ready Times and Due Dates to Minimize Maximum Lateness, *Operations Research*, May-June, **23**(3), 475-482.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. and Teller, E. (1953) Equation of State Calculations by Fast Computing Machines, *The Journal of Chemical Physics*, **21**(6), June, 1087-1092.

Morton, T. E. (1990) Shifting Bottleneck Methods in Job Shop and Project Scheduling: Tutorial and Research Directions, Graduate School of Industrial Administration, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213, USA.

Morton, T. E. and Pentico, D. W. (1993) *Heuristic Scheduling Systems*, Wiley Series in Engineering and Technology Management, Wiley, New York.

Moscato, P. (1989) On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms, *C3P Report 826*, Caltech Concurrent Computation Program, Caltech, California, USA.

Nakano, R. and Yamada, T. (1991) Conventional Genetic Algorithm for Job-Shop Problems, in Kenneth, M. K. and Booker, L. B. (eds) *Proceedings of the 4th International Conference on Genetic Algorithms and their Applications*, San Diego, USA, pp. 474-479.

Nawaz, M., Enscore, E. E. Jr. and Ham, I. (1983) A Heuristic Algorithm for the *m*-Machine *n*-Job Flow-shop Sequencing Problem, *OMEGA The International Journal of Management Science*, **11**(1), 91-95.

Nemhauser, G. L. and Wolsey, L. A. (1988) *Integer and Combinatorial Optimisation*, John Wiley and Sons, New York.

Norman, B. A. and Bean, J. (1997) Random Keys Genetic Algorithm for Job Shop Scheduling, *Engineering Design and Automation*, vol 3, 145-156.

Nowicki, E. and Smutnicki, C. (1996) A Fast Taboo Search Algorithm for the Job-Shop Problem, *Management Science*, **42**(6), 797-813.

Nuijten, W.P.M., Aarts, E. H. L., Van Erp Taalman Kip, D. A. A. and Van Hee, K. M. (1993) Randomized Constraint Satisfaction for Job Shop Scheduling, *Proceedings of the IJCAI'93 Workshop on Knowledge Based Production Planning and Scheduling and Control*, pp. 251-262.

Nuijten, W. P. M. and Aarts, E. H. L. (1996) A Computational Study of Constraint Satisfaction for Multiple Capacitated Job Shop Scheduling, *European Journal of Operational Research*, vol 90, 269-284.

Nuijten, W. P. M. and Le Pape, C. (1998) Constraint-Based Job Shop Scheduling with ILOG SCHEDULER, *Journal of Heuristics*, March, **3**(4), 271-286.

Osman, I. H. and Kelly, J. P. (1996a) (eds) *Meta-Heuristics: Theory and Applications*, Kluwer Academic Publishers, Norwell, MA, USA.

Osman, I. H. and Kelly, J. P. (1996b) Meta-Heuristics: An Overview, in Osman, I. H. and Kelly, J. P. *Meta-Heuristics: Theory and Applications*, Kluwer Academic Publishers, Norwell, MA, USA, Chapter 1, pp. 1-21.

Ovacik, I. M. and Uzsoy, R. (1992) A Shifting Bottleneck Algorithm for Scheduling Semiconductor Testing Operations, *Journal of Electronics Manufacturing*, vol 2, 119-134.

Ovacik, I. M. and Uzsoy, R. (1996) Decomposition Methods for Scheduling Semiconductor Testing Facilities, *International Journal of Flexible Manufacturing Systems*, vol 8, 357-388.

Ow, P. S. (1986) Experiments in Knowledge-Based Scheduling, Technical Report, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA.

Ow, P. S. and Smith, S. F. (1988) Viewing Scheduling as an Opportunistic Problem-Solving Process, *Annals of Operations Research*, vol 12, 85-108.

Panwalkar, S. S. and Iskander, W. (1977) A Survey of Scheduling Rules, *Operations Research*, Jan-Feb, **25**(1), 45-61.

Perregaard, M. and Clausen, J. (1995) Parallel Branch-and-Bound Methods for the Job-Shop Scheduling Problem, Working Paper, University of Copenhagen, Copenhagen, Denmark.

Pesch, E. (1993) Machine Learning by Schedule Decomposition, Working Paper, Faculty of Economics and Business Administration, University of Limburg, Maastricht.

Pesch, E., Tetzlaff, U. A. W. (1996) Constraint Propagation Based Scheduling of Job Shops, *INFORMS Journal on Computing*, Spring, **8**(2), 144-157.

Peterson, C. and Anderson, J. R. (1987) A Mean Field Theory Learning Algorithm for Neural Networks, *Complex Systems*, vol 1, 995-1019.

Pinson, E. (1988) Le Problème de Job-Shop, *Thèse d'État*, L'Université Pierre et Marie Curie, Paris VI, France. (In French).

Pinson, E. (1995) The Job-Shop Scheduling Problem: A Concise Survey and Some Recent Developments, in Chrétienne P., Coffman, E. G. Jr, Lenstra, J. K. and Liu, Z. (eds) *Scheduling Theory and its Applications*, John Wiley & Sons Ltd, chapter 13, pp. 277-293.

Pirlot, M. (1996) General Local Search Methods, *European Journal of Operational Research*, vol 92, 493-511.

Potts, C. N. (1980) Analysis of a Heuristic for One Machine Sequencing with Release Dates and Delivery Times, *Operations Research*, **28**(6), 1436-1441.

Rabelo, L. C. and Alptekin, S. (1989a) Using Hybrid Neural Networks/Expert Systems for Intelligent Scheduling in Flexible Manufacturing Systems, *IJCNN International Joint Conference on Neural Networks*, Washington, Jun 18-22, vol 2, pp. 608.

Rabelo, L. C. and Alptekin, S. (1989b) Synergy of Neural Networks & Expert Systems for FMS Scheduling. In Stecke, K. E. and Suri, R. (eds) *Proceedings of the Third ORSA / TIMS FMS: Operation Research Models and Applications*, Amsterdam, pp. 361-366.

Rabelo, L. C. and Alptekin, S. (1990a) Adaptive Scheduling and Control using Artificial Neural Networks and Expert Systems for a Hierarchical/Distributed FMS Architecture. *Proceedings of Rensselaer's Second International Conference on Computer Integrated Manufacturing (IEEE)*, Troy, New York, May 21-23, pp. 538-545.

Rabelo, L. C. and Alptekin, S. (1990b) Synergy of Artificial Neural Networks and Knowledge-Based Expert Systems for Intelligent FMS Scheduling. *IJCNN International Joint Conference on Neural Networks*, San Diego, CA, Jun 17-21, vol 1, pp. 359-366.

Ramudhin, A. and Marier, P. (1996) The Generalized Shifting Bottleneck Procedure, *European Journal of Operational Research*, vol 93, 34-48.

Reeves, C. R. (1993) Evaluation of Heuristic Performance, in Reeves, C. R. (ed) *Modern Heuristic Techniques for Combinatorial Problems*, Blackwell Scientific Publications, Osney Mead, Oxford, England, chapter 7, pp. 304-315. (Re-issued 1995 by McGraw-Hill, London.)

Remus, W. (1990) Neural Network Models of Managerial Judgement, *23rd Annual Hawaii International Conference on System Science*, Honolulu, Hawaii, January 2-5, pp. 340-344.

Resende, M. G. C., (1997) A GRASP for Job Shop Scheduling, *INFORMS Spring Meeting*, San Diego, California, USA, May.

Rowe, A. J. and Jackson, J. R. (1956) Research Problems in Production Routing and Scheduling, *Journal of Industrial Engineering.*, vol 7, 116-121.

Roy, B. and Sussmann, B. (1964) Les Problèmes d'Ordonnancement avec Contraintes Disjonctives, Note D.S. no. 9 bis, SEMA, Paris, France, Décembre.

Rumelhart, D. E., Hinton, G. E. and Williams, R. J. (1986) Learning Internal Representations by Error Propagation in *Parallel Distributed Processing vol 1 and 2* (eds D. E. Rumelhart and J. L. McClelland), MIT Press, Cambridge, Massachusetts, pp. 318-362.

Sabuncuoglu, I. and Gurgun, B. (1996) A Neural Network Model for Scheduling Problems, *European Journal of Operational Research*, **93**(2), Sept, 288-299.

Sabuncuoglu, I. and Bayiz, M. (1997) A Beam Search Based Algorithm for the Job Shop Scheduling Problem, *Research Report: IEOR-9705*, Department of Industrial Engineering, Faculty of Engineering, Bilkent University, 06533 Ankara, Turkey (*to appear in the European Journal of Operational Research*).

Sadeh, N. (1989) Look-ahead Techniques for Activity-based Job-Shop Scheduling, *Thesis Proposal*, Technical Report, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, USA, December 18.

Sadeh, N. (1991) Look-Ahead Techniques for Micro-Opportunistic Job Shop Scheduling, *Ph.D. Thesis*, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, USA, March.

Sadeh, N., Sycara, K. and Xiong, Y. L. (1995) Backtracking Techniques for the Job-Shop Scheduling Constraint Satisfaction Problem, *Artificial Intelligence*, **76**(1-2), 455-480.

Sadeh, N. and Fox, M. S. (1996) Variable and Value Ordering Heuristics for the Job Shop Scheduling

Constraint Satisfaction Problem" *Artificial Intelligence*, **86**(1), 1-41.

Sadeh, N. and Nakakuki, Y. (1996) Focused Simulated Annealing Search - An Application to Job-Shop Scheduling, *Annals of Operations Research*, vol 63, 77-103.

Sadeh, N., Nakakuki, Y. and Thangiah, S. R. (1997) Learning to Recognise (Un)Promising Simulated Annealing Runs: Efficient Search Procedures for Job-Shop Scheduling and Vehicle Routing, *Annals of Operations Research* 75, 189-208.

Satake, T., Morikawa, K. and Nakamura, N. (1994) Neural Network Approach for Minimizing the Makespan of the General Job-Shop, *International Journal of Production Economics*, Jan, **33**(1-3), 67-74.

Schrage, L. (1970) Solving Resource-Constrained Network Problems by Implicit Enumeration: Non Preemptive Case, *Operations Research*, vol 18, 263-278.

Shi, G. (1997) A Genetic Algorithm Applied to a Classic Job-Shop Scheduling Problem, *International Journal of Systems Science*, **28**(1), 25-32.

Shmoys, D. B., Stein, C. and Wein, J. (1994) Improved Approximation Algorithms for Shop Scheduling Problems, *SIAM J Comput*, June, **23**(3), 617-632.

Sim, S. K., Yeo, K. T. and Lee, W. H. (1994) An Expert Neural Network System for Dynamic Job-Shop Scheduling, *International Journal of Production Research*, Aug, **32**(8), 1759-1773.

Slotnick, S. A., May, J. H. and Morton, T. E. (1992) FURNEX: Modeling Expert Scheduling on the Factory Floor, in Scherer, W. T. and Brown, D. E. (eds) *Proceedings of the Symposium on Intelligent Scheduling Systems*, pp. 277-286.

Smith, S. F., Fox, M. S. and Ow, P. S. (1986) Constructing and Maintaining Detailed Production Plans: Investigations into the Development of Knowledge-Based Factory Scheduling Systems, *AI Magazine*, **7**(4), 45-61.

Smith, W. E. (1956) Various Optimizers for Single Stage Production, *Naval Research Logistics Quarterly*, vol 3, 59-66.

Sotskov, Y. N. (1991) The Complexity of Shop-Scheduling Problems with Two or Three Jobs, *European Journal of Operational Research*, vol 53, 326-336.

Storer, R. H., Wu, S. D., and Vaccari, R. (1992) New Search Spaces for Sequencing Problems with Applications to Job-Shop Scheduling, *Management Science*, **38**(10), October, 1495-1509.

Storer, R. H., Wu, S. D. and Park, I. (1993) Genetic Algorithm in Problem Space for Sequencing Problems, in Fandel, G., Gulledge, T. and Jones, A. (eds), *Operations Research in Production Planning and Control: Proceedings of a Joint US/German Conference*, Springer Verlag, Berlin, Heidelberg, pp. 584-597.

Storer, R. H., Wu, S. D. and Vaccari, R. (1995) Problem and Heuristic Space Search Strategies for Job Shop Scheduling, *ORSA Journal on Computing*, **7**(4), Fall, 453-467.

Sun, D. K. and Lin, L. (1994) Dynamic Job-Shop Scheduling Framework: A Backward Approach, *International Journal of Production Research*, Apr, **32**(4), 967-985.

Sun, D. K., Batta, R. and Lin, L. (1995) Effective Job-Shop Scheduling Through Active Chain Manipulation, *Computers & Operations Research* **22**(2), 159-172.

Szu, H. and Hartley, R. (1987) Fast Simulated Annealing, *Phys. Lett. A.*, vol 122, 157-162.

Taillard, É. (1994) Parallel Taboo Search Techniques for the Job-Shop Scheduling Problem. *ORSA Journal on Computing*, **16**(2), 108-117.

Tamaki, H. and Nishikawa, Y. (1992) A Parallelled Genetic Algorithm Based on a Neighbourhood Model and its Application to the JobShop Scheduling, in Männer, R. and Manderick, B. (eds) *PPSN'2 Proceedings of the 2nd International Workshop on Parallel Problem Solving from Nature*, Brussels, Belgium, pp. 573-582.

Ten Eikelder, H. M. M., Aarts, B. J. M., Verhoeven, M. G. A. and Aarts, E. H. L. (1997) Sequential and Parallel Local Search Algorithms for Job Shop Scheduling, *MIC'97 2nd International Conference on Meta-heuristics*, Sophia-Antipolis, France, 21-24 July, pp. 75-80 (Extended Abstract).

Thomsen, S. (1997) Meta-heuristics Combined with Branch & Bound, *Technical Report*, Copenhagen Business School, Copenhagen, Denmark (in Danish).

Ulder, N. L. J., Aarts, E. H. L., Bandelt, H.-J., Van Laarhooven, P. J. M. and Pesch, E. (1991) Genetic Local Search Algorithm for the Travelling Salesman Problem, *Lecture Notes in Computer Science*, vol 496, 109-116.

Vaessens, R. J. M., Aarts, E. H. L. and Lenstra, J. K. (1995) A Local Search Template Revised Version, Department of Mathematics & Computing Science, Eindhoven University of Technology, Eindhoven, The Netherlands, January.

Vaessens, R. J. M. (1996) Operations Research Library of Problems, Management School, Imperial College London, Anonymous FTP site at ftp://mscmga.ms.ic.ac.uk/pub/jobshop1.txt.

Vaessens, R. J. M., Aarts, E. H. L. and Lenstra, J. K. (1996) Job Shop Scheduling by Local Search, *INFORMS Journal on Computing*, vol 8, 302-317.

Van De Velde, S. (1991) Machine Scheduling and Lagrangian Relaxation, *Ph. D. Thesis*, CWI Amsterdam, The Netherlands.

Van Hulle, M. M. (1991) A Goal Programming Network for Mixed Integer Linear Programming: A Case Study for the Job-Shop Scheduling Problem, *International Journal of Neural Networks*, **2**(3), 201-209.

Van Laarhooven, P. J. M., Aarts, E. H. L. and Lenstra, J. K. (1992) Job Shop Scheduling by Simulated Annealing, *Operations Research*, Jan-Feb, **40**(1), 113-125.

Vepsalainen, A. P. J. and Morton, T. E. (1987) Priority Rules for Job Shops with Weighted Tardiness Costs, *Management Science*, **33**(8), 1035-1047.

Vere, S. (1983) Planning in Time: Windows and Durations for Activities and Goals, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **5**(3), 246-267.

Viviers, F. (1983) A Decision Support System for Job-Shop Scheduling, *European Journal of Operational Research*, Sept, **14**(1), 95-103.

Wang, W. and Brunn, P. (1995), Production Scheduling and Neural Networks, in Derigs, U., Bachem, A. and Drexl, A. (eds.), *Operations Research Proceedings 1994*, Springer-Verlag, Berlin, pp. 173-178.

Watanabe, T., Tokumaru, H. and Hashimoto, Y. (1993) Job-Shop Scheduling using Neural Networks, *Control Engineering Practice*, Dec, **1**(6), 957-961.

Werner, F. and Winkler, A. (1995) Insertion Techniques for the Heuristic Solution of the Job-Shop Problem, *Discrete Applied Mathematics*, **58**(2), 191-211.

Williamson, D. P., Hall, L. A., Hoogeveen, J. A., Hurkens, C. A. J., Lenstra, J. K., Sevast'janov, S. V. and Shmoys, D. B. (1997) Short Shop Schedules, *Operations Research*, March - April, **45**(2), 288-294.

Willems, T. M. and Rooda, J. E. (1994) Neural Networks for Job-Shop Scheduling, *Control Eng. Practice*, Feb, **2**(1), 31-39.

Yamada, T. and Nakano, R. (1992) A Genetic Algorithm Applicable to Large-Scale Job-Shop Problems, in Männer, R. and Manderick, B. (eds) *PPSN'2 Proceedings of the 2nd International Workshop on Parallel Problem Solving from Nature*, Brussels, pp. 281-290.

Yamada, T., Rosen, B. E. and Nakano, R. (1994) A Simulated Annealing Approach to Job-Shop Scheduling using Critical Block Transition Operators, *IEEE ICNN'94 International Conference on Neural Networks*, Orlando, Florida, 26-29 June, vol 6, pp. 4687-4692.

Yamada, T. and Nakano, R. (1995a) Job-Shop Scheduling by Simulated Annealing Combined with Deterministic Local Search, *MIC'95 Meta-heuristics International Conference*, Hilton, Breckenridge, Colorado, USA, July 22-26, pp. 344-349.

Yamada, T. and Nakano, R. (1995b) A Genetic Algorithm with Multi-Step Crossover for Job-Shop Scheduling Problems, *GALESIA'95 Proceedings of the Int. Conf. on GAs in Eng. Sys.*, pp. 146-151.

Yamada, T. and Nakano, R. (1996a) Job-Shop Scheduling by Simulated Annealing Combined with Deterministic Local Search, *Meta-heuristics: Theory and Applications*, Kluwer Academic Publishers, MA, USA, pp. 237-248.

Yamada, T. and Nakano, R. (1996b) Scheduling by Genetic Local Search with Multi-Step Crossover, *PPSN'IV Fourth International Conference on Parallel Problem Solving from Nature*, Berlin, Germany, Sept 22-26, pp. 960-969.

Yamada, T. and Nakano, R. (1996c) A Fusion of Crossover and Local Search, *ICIT'96 IEEE International Conference on Industrial Technology*, Shanghai, China, Dec 2-6, pp. 426-430.

Yannakakis, M. (1990) The Analysis of Local Search Problems and their Heuristics, *Proceedings of the 7th Annual Symposium on Theoretical Aspects of Computer Science*, pp. 298-311.

Yannakakis, M. (1997) Computational Complexity of Local Search, in Aarts, E. H. L. and Lenstra, J. K (eds) *Local Search in Combinatorial Optimization*, Wiley, Chichester.

Yokoi, H., Kakazu, Y. and Minagawa, M. (1994) An Approach to the Autonomous Job-Shop Scheduling Problem by Vibrating Potential Method, *IEEE ICNN'94 International Conference on Neural Networks*, Orlando, Florida, 26-29 June, vol 6, pp. 3877-3882.

Zhang, H.-C. and Huang, S. H. (1995) Applications of Neural Networks in Manufacturing: A State-of-the-Art Survey, *International Journal of Production Research*, **33**(3), 705-728.

Zhou, D. N., Cherkassky, V., Baldwin, T. R. and Olson D. E. (1991) A Neural Network Approach to Job-Shop Scheduling, *IEEE Transactions on Neural Network*, **2**(1), 175-179.

	Processing Times			Machine Assignment		
	$\mathcal{M}_1$	$\mathcal{M}_2$	$\mathcal{M}_3$	1st	2nd	3rd
$\mathcal{J}_1$	5	8	2	$\mathcal{M}_1$	$\mathcal{M}_2$	$\mathcal{M}_3$
$\mathcal{J}_2$	3	9	7	$\mathcal{M}_3$	$\mathcal{M}_1$	$\mathcal{M}_2$
$\mathcal{J}_3$	1	10	7	$\mathcal{M}_1$	$\mathcal{M}_3$	$\mathcal{M}_2$
$\mathcal{J}_4$	7	4	11	$\mathcal{M}_2$	$\mathcal{M}_3$	$\mathcal{M}_1$

Table 1. Machine assignment and processing times for the 4×3 instance

Problem	*Size*	*Optimum*	*Makespan (CPU)*		*MRE (%)*	
			DMTCP ('93)	KSSW ('95)	DMTCP ('93)	KSSW ('95)
FT 06	6×6	55	58 (7)	55 (not given)	5.45	0.00
FT 10	10×10	930	980 (1585)	1054 (not given)	5.38	13.33
FT 20	20×5	1165	1296 (2593)	1379 (not given)	11.24	18.37
DMTCP ('93) – Della Croce *et al.* (1993)				KSSW ('95) – Krüger *et al.* (1995)		

Table 2. Results of the two mathematical formulations on FT 06, 10 & 20

Rule	**Priority Assignment**
Spt/Twkr	Smallest ratio of the processing time to total work remaining
Lrm	Longest remaining work excluding the operation under consideration
Mwkr	Most work remaining to be done
Mopr	Most number of operations remaining
Spt/Twk	Smallest ratio of processing time to total work
Fcfs	The operation that arrived the earliest is processed first (first come first served)
Lso	Longest subsequent operation
Spt	Shortest processing time - Also known as Sio (Shortest imminent operation) rule
Fhalf	More than one half of the total number of operations remaining, ie. first half preferred
Ninq	Next operation is on the machine with the fewest number of operations waiting

Table 5. Preference assignments of a set of priority dispatch rules

	Makespan Achieved			CPU Time (secs)			
Author	FT 06	FT 10	FT 20	FT 06	FT 10	FT 20	Machine Used
Namada & Yakano ('91) - NY'91	55	965	1215	NG	NG	NG	NG
Yamada & Nakano ('92) - NY'92	55	930	1184	NG	600	NG	SUN Sparcstation 2
Davidor *et al.* ('93) - DYN'93	55	930	1181	NG	300	1800	SUN Sparcstation 2
Fang *et al.* ('93) - FRC'93	55	949	1189	NG	1500	1500	SUN 4
Storer *et al.* ('93) - SWP'93	55	954	1180	NG	55	75	CDC 4340 - R3000
Pesch ('93) - 2J-GA	55	937	1193	8.1	100.0	95.3	VAX 8650
Pesch ('93) - 2JCP-GA	55	937	1175	11.4	146.9	121.1	VAX 8650
Pesch ('93) - 1MCP-GA	55	930	1165	8.5	104.4	101.0	VAX 8650
Mattfeld *et al.* ('94) - MKB'94	55	930	1165	NG	138	138	SUN Sparcstation 10
Della Croce *et al.* ('95) - DTV'95	55	946	1178	223	628	675	PC 486 (25MHz)
Bierwirth ('95) - B'95	55	936	1181	NG	135	147	SUN Sparcstation 10
Dorndorf and Pesch ('95) - DP'95	55	938	1178	19.7	106.7	95.7	DEC Station 3100
Yamada & Nakano ('95b) - YN'95	55	930	1165	NG	699	654	SUN Sparcstation 10
Mattfeld ('96) - Mf'96	55	930	1165	6	40	47	SUN 10/41
Yamada & Nakano ('96b,c) - YN'96	55	930	1165	NG	88	55	Dec Alpha 600 5/266
Norman & Bean ('97) - NB'97	55	937	1165	NG	300	300	SUN Workstation
Shi ('97) - S'97	55	930	1165	NG	NG	NG	SONY NWS - 3460

NG - This information is not given

Table 14. Results of several genetic techniques on the benchmark problems of Fisher and Thompson