Internationales Begegnungs- und
Forschungszentrum für Informatik

Schloß Dagstuhl

Seminar Report

J. Blazewicz, K. Ecker, W. Kubiak, D. Trystram

# Scheduling in Computer and Manufacturing Systems

June 2 - 6, 1997

# Overview

During the week of June 2 - 6, 1997, the Seminar on Scheduling in Computer and Manufacturing Systems was organized by J. Blazewicz (Posen), K. Ecker (Clausthal), W. Kubiak (St. Johns), D. Trystram (Grenoble). Participants came from Universities or Reseach Centers from Greece, Germany, France, Poland, Italy, Belarus, Austria, USA, Canada, The Netherlands and Mexico.

Altogether 37 lectures covered a large area of scheduling problems with various aspects of scheduling, both in algorithmic approaches and applications. Areas of particular interest were task scheduling under consideration of communication and shop problems.

The participants appreciated the outstanding local organization and the environment including all Dagstuhl facilities which not least enabled a successful workshop.

In the name of all participants,

J. Blazewicz, K. Ecker, W. Kubiak, D. Trystram

# E-Mail Addresses


| | |
|---|---|
| Foto Afrati | afrati@softlab.ece.ntua.gr |
| Jörg Apsel | apsel@informatik.tu-clausthal.de |
| Evripidis Bampis | bampis@lami.uni-evry.fr |
| Jacek Blazewicz | blazewic@sol.put.poznan.pl |
| Peter Brucker | peter@mathematik.Uni-Osnabrueck.de |
| Paolo Dell'Olmo | dellolmo@iasi.rm.cnr.it |
| Andreas Drexl | Drexl@bwl.uni-kiel.de |
| Moshe Dror | mdror@pl.ecp.fr |
| Maciej Drozdowski | maciej-d@pozn1v.put.poznan.pl |
| Klaus H. Ecker | ecker@informatik.tu-clausthal.de |
| Gerd Finke | gerd.finke@imag.fr |
| Lucian Finta | lucian.finta@sophia.inria.fr |
| Valery Gordon | gordon@newman.basnet.minsk.by |
| Alain Guinet | guinet@gprhp.insa-lyon.fr |
| Andreas Jakoby | jakoby@informatik.mu-luebeck.de |
| Adam Janiak | Janiak@ict.pwr.wroc.pl |
| Klaus Jansen | jansen@dm3.uni-trier.de |
| Emmanuel Jeannot | ejeannot@lip.ens-lyon.fr |
| Joanna Jozefowska | JJozef@sol.put.poznan.pl |
| Hans Kellerer | kellerer@edvz.kfunigraz.ac.at |
| Roger Kieckhafer | rogerk@cse.unl.edu |
| Mikhail Y. Kovalyov | koval@newman.basnet.minsk.by |
| Axel Krings | krings@cs.uidaho.edu |
| Marek Kubale | kubale@sunrise.pg.gda.pl |
| Wieslaw Kubiak | wkubiak@morgan.ucs.mun.ca |
| Jan Karel Lenstra | jkl@win.tue.nl |
| Rolf H. Möhring | moehring@math.tu-berlin.de |
| Dritan Nace | dnace@utc.fr |
| Jurek Nawrocki | nawrocki@put.poznan.pl |
| Edgar Nett | nett@gmd.de |
| Klaus Neumann | neumann@wior.uni-karlsruhe.de |
| Bernard Penz | Bernard.Penz@ensgi.inpg.fr |
| Erwin Pesch | e.pesch@uni-bonn.de |
| Antonio Rodriguez | arodrig@cicese.mx |

| | |
|---|---|
| Tadeusz Sawik | tsawik@novell.zarz.agh.edu.pl |
| Günter Schmidt | gs@itm.uni-sb.de |
| V. Sridharan | suhas@clemson.edu |
| Frederik Stork | stork@math.tu-berlin.de |
| Eugeniusz Toczylowski | e.toczylowski@ia.pw.edu.pl |
| Denis Trystram | trystram@imag.fr |
| Marinus Veldhorst | marinus@cs.ruu.nl |
| Egon Wanke | wanke@cs.uni-duesseldorf.de |
| Scott Yeomans | AS000147@orion.yorku.ca |
| Joris van de Klundert | j.vandeklundert@ke.unimaas.nl |
| Steef van de Velde | S.L.VanDeVelde@wb.utwente.nl |

# Abstracts

**Machine Scheduling with Availability Constraints**
Günter Schmidt and Eric Sanlaville

We will give a survey on results related to scheduling problems where machines are not continuously available for processing. We will deal with single and multi machine problems and analyze their complexity. We survey $\mathcal{NP}$-hardness results, polynomial optimization and approximation algorithms. We also distinguish between on-line and off-line formulations of the problems. Results are concerned with criteria on completion times and due dates.

**Scheduling for Client-Server Applications**
Maciej Drozdowski, J.Blazewicz and P.Dell'Olmo

In this presentation we analyze the problem of deterministic scheduling of applications (programs) in a client-server environment. We assume that the client reads data from the server, processes it, and stores the results on the server. This paradigm can also model a wider class of parallel applications. The goal is to find the shortest schedule. It is shown that the general problem is computationally hard. However, any greedy heuristic delivers solutions not worse than twice the optimum when tasks are preallocated, and three times the optimum when tasks are not preallocated. A number of polynomially solvable cases are listed.

**Scheduling with Ready Times and Sequence Dependent Processing Times on a Single Machine**
Paolo Dell'Olmo

In this talk we give an analysis of upper and lower bounds for two problems with change-over times and release dates of jobs. In the first problem we consider as an objective function the mean-flow time of the schedule. Relation with the so called delivery problem is discussed. Three lower bounds based on a relaxation of the mathematic formulation of the scheduling problem are presented. A suitable heuristic is derived from the nearest insertion heuristic of the traveling salesman problem. The second problem consider the maximum completion time as objective function. For this a lower bound based on linear programming formulation is derived. An experimental analysis of the problem bounds is performed for different assumptions for the internal in which the ready times of the jobs are distributed.

**Robotic Flow-Shop Scheduling**
Joris van de Klundert

We consider a robotic flow shop model in which a single robot is responsible for the transportation of parts between machines. For reasons of simplicity, when the shop is to produce a large number of identical parts, the robot usually performs repeatedly a fixed sequence of activities. This sequence of activities is called a 1-unit cycle if each execution of the sequence results in the production of exactly one part. It has been conjectured that 1-unit cycles yield optimal production rates for 3-machine robotic flow shops. We establish the validity of this conjecture.

**On the One-Cycle Conjecture in Robotic Cells**
Gerd Finke and Nadia Braune

We consider a robotic cell, consisting of a flow-shop in which the machines are served by a single central robot. We concentrate on the case where only one part type is produced and want to analyze the conjecture of Sethi, Sriskandarajah, Sorger, Blazewicz and Kubiak.

This well-known conjecture claims that the repetition of the best one-unit production cycle will yield the maximum throughput rate in the set of all possible robot moves. The conjecture holds for two and three machines, but the existing proof by van de Klundert for the three-machine case is extremely tedious.

We adopt the theoretical background developed by Crama and van de Klundert. Using a particular state graph, the k-unit production cycles are represented as special paths and cycles for which general properties and bounds for the m-machine robotic cell can be obtained. By means of these concepts, we shall give a concise proof for the validity of the conjecture for the three-machine case.

### Heuristics for Job-Shop and Flow-Shop Scheduling with Stochastic Precedence Constraints

Klaus Neumann

We consider a job shop problem where stochastic precedence constraints for the jobs are given by a so-called OR network (i.e. a special GERT network all of whose nodes have OR entrance and deterministic or stochastic exit). The objective function to be minimized is the expected makespan. Such job shop problems may occur in mixed-model make-to-order production.

We introduce the concept of an aggregate schedule, which represents a deterministic scheduling policy from which the sequences of jobs processed on the individual machines and their start times can be obtained for each network realization in a unique way.

Since the problem of finding an optimal aggregate schedule is $\mathcal{NP}$-hard, we restrict ourselves to computing an approximate aggregate schedule by some heuristic. It is already known that the shifting bottleneck heuristic exploiting the construction of an appropriate aggregate disjunctive graph can be used for determining an approximate aggregate schedule.

In this presentation we adapt the Giffler-Thompson priority rule method to our stochastic job shop problem. An experimental performance analysis shows that a particular MEWR (most expected work remaining) rule outperforms all other priority rules considered. Moreover, that priority rule method is much faster than the shifting bottleneck heuristic and provides more accurate results for larger problem instances.

Finally, we briefly discuss the flow shop problem with stochastic OR network precedence constraints. The same machine sequence for all jobs permits us to combine several operations beyond a node with stochastic exit in one so-called action, which results in a reduction of the computational effort for finding an approximate solution.

### Minimizing Mean Completion Time for Discrete Continuous Scheduling Problems

Joanna Jozefowska

Discrete-continuous problems of scheduling independent, nonpreemptable jobs on parallel machines to minimize mean completion times are considered. A model of a job is assumed in which processing rate of a job depends on the amount of the renewable continuous resource allotted to this job at a time. The problem consists of two interrelated subproblems:

(i) to sequence jobs on machines and

(ii) to allocate the continuous resource among jobs already sequenced.

For convex processing rates of jobs an optimal schedule can be found by scheduling all jobs on one machine with total resource amount in the SPT order. It is shown that for concave processing rates the optimal allocation of the continuous resource for a given sequence of jobs can be found by solving a system of nonlinear equations. Moreover, this allocation does not depend on the vector of processing demands. For the processing rates of jobs being concave power functions a feasible sequence leading to an optimal schedule can be found in polynomial time.

### Single Machine Due Date Assignment and Scheduling Problems

Valery Gordon

Single machine scheduling and due date assignment problems subject to precedence constraints are considered. Due dates are determined according to SLK rule: for each job a due date is obtained by adding a positive slack q to the processing time or to the processing time plus release date. Two types of problems are considered:

(a) The objective function to be minimized is a sum of maximum tardiness and aq, where $aq > 0$ is the cost per unit time of slack allowance; release dates are given and preemption is allowed.

(b) The objective function is non-decreasing in both slack allowance and earliness penalties; all jobs are simultaneously available for processing, and there is no tardy job in a feasible schedule (joint results with V.Strusevich, Greenwich University, UK).

For problem (a), an optimal schedule can be found in $O(n^2)$ time for arbitrary precedence constraints, and the optimal slack allowance is easily obtained depending on the value of a. Problem (b) is $\mathcal{NP}$-hard for arbitrary precedence constraints even for penalties being total earliness. If the precedence constraints are defined by a series-parallel graph, the problem is shown to be polynomially solvable when penalties are the sum of either linear or exponential functions.

**Non-Deterministic Scheduling in an Abstract Parallel Machine Using the U Model**
Antonio Rodriguez

A machine is presented as an implementation of partial evaluation mixed computation that generates dynamic non-deterministic scheduling problems. The Abstract Network Machine is defined in a work of A. Stepanov et al. (95). The strategies for solving this scheduling problems are based on artificial intelligence, which views them as restriction satisfaction problems. Knowledge based approaches include predictive and reactive nodes.

The relaxation of the precedence constraints permits the early generation of tasks, thus reducing the schedule length. The use of a multi-agent system for handling communication on the network is proposed. Heuristics for agents must be greedy in order to compensate the communication costs, an analytical algorithm is not recommended because of the dynamical characteristic of ANM computation.

**Scheduling Cyclic Tasks with Binary Periods**
Jurek Nawrocki, A.Czajka and W.Complak

Hard-real time systems consist of concurrent tasks and each task has a deadline. Thus, a problem arises how to schedule such tasks before deadlines. One of the approaches is pre-run-time scheduling. The idea is that at system design time a timetable is generated showing which task should be run at a given moment. This timetable is used at run-time by a dispatcher procedure. The problem is that sometimes the timetable can be very big, even for a small number of tasks. To reduce its size it is proposed to replace original periods with binary ones in such a way that a binary period is not greater than its original counterpart. During the presentation it has been shown how to find all the interesting binary estimations. Moreover, time complexity of the problem of scheduling tasks with binary periods has been investigated and it was proved that the problem is strongly $\mathcal{NP}$-complete. In the end a very fast heuristic strategy was presented which, according to simulation experiments, is very effective.

**Efficiency and Stability in Non-Preemptive Schedules**
Roger Kieckhafer, J.S. Deogun, A.W. Krings, L.R. Dondeti and B.N. Masti

Non-preemptive static priority list scheduling is a simple, low-overhead approach to scheduling precedence-constrained tasks in real-time multiprocessor systems. However, it is vulnerable to anomalous timing behavior caused by variations in task durations. Specifically, 'reducing' the duration of one task can 'delay' the starting time of another task. This phenomenon, called 'Scheduling Instability', can make it difficult or impossible to guarantee real-time deadlines. Recent research has shown that instability can be prevented by limiting how far the run-time dispatcher is allowed to scan into the priority list. In addition to being stable, several of the dispatchers have a complexity less than that of the original dispatcher. Furthermore, simulations have shown that for representative real-time workloads, these simple, low-overhead dispatchers perform nearly as well as a complex 'minimally stabilized' dispatcher.

This presentation continues this work by addressing two other forms of list scheduling overhead. First, we address the 'linkage' overhead incurred when a task is completed, and all of its immediate successors must be identified and updated. This overhead is constrained by transforming the precedence graph to bound the number of immediate successors any task may have. Second, we address the 'contention' overhead caused by the necessity to maintain mutual exclusion on the dispatcher code itself. This overhead is addressed by imposing static task-to-processor allocation at system design-time. For both cases, the conditions necessary to prevent instability are identified and applied to the dispatcher. As before, simulations have shown only minimal reductions in processor utilization due to either of these effects.

**Run-Time Feasibility of Hard Real-Time Systems Containing Coupled Tasks**
Axel Krings, M.H. Azadmanesh

This talk addresses the problem of guaranteeing stability and run-time feasibility in real-time systems containing coupled tasks, in the context of non-preemptive priority scheduling. Instability is the result of so-called multiprocessor timing anomalies, where deadlines can be missed due to the reduction in task durations. Such reductions can also result in runtime infeasibilities of coupled tasks due to the inherent inter-task timing constraints.

A scheduling environment and general algorithm are presented that avoid both phenomena at run-time. Furthermore, the theoretical background is provided allowing the design of algorithms of varying complexity.

**Scheduling a Batch Processing Machine**
Mikhail Y. Kovalyov

The problem of scheduling $n$ jobs on a batch processing machine to minimize regular scheduling criteria that are non-decreasing in the job completion times is addressed. A batch processing machine is a machine that can handle up to $b$ jobs simultaneously. The jobs that are processed together form a batch, and all jobs in a batch start and complete at the same time.

The processing time of a batch is equal to the largest processing time of any job in the batch. We analyze two variants: the unbounded model, where $b \geq n$; and the bounded model, where $b < n$.

For the unbounded model, we give a characterization of a class of optimal schedules, which leads to a generic dynamic programming algorithm that solves the problem of minimizing an arbitrary regular cost function in pseudopolynomial time. The characterization leads to more efficient dynamic programming algorithms for specific cost functions: a polynomial algorithm for minimizing the maximum cost, an $O(n^3)$ time algorithm for minimizing the number of tardy jobs, an $O(n^2)$ time algorithm for minimizing the maximum lateness, and an $O(n \log n)$ time algorithm for minimizing the total weighted completion time. Furthermore, we prove that minimizing the weighted number of tardy jobs and the total weighted tardiness are $\mathcal{NP}$-hard problems.

For the bounded model, we derive an $O(n^{b(b\Gamma 1)})$ time dynamic programming algorithm for minimizing total completion time when $b > 1$; for the case with $m$ different processing times, we give a dynamic programming algorithm that requires $O(b^2 m^2 2^m)$ time. Moreover, we prove that due-date based scheduling criteria give rise to NP-hard problems. Finally, we show that an arbitrary regular cost function can be minimized in polynomial time for a fixed number of batches.

This is a joint work of Peter Brucker, Andrei Gladky, Han Hoogeveen, Mikhail Y. Kovalyov, Chris Potts, Thomas Tautenhahn and Steef van de Velde.

**Resource Constrained Scheduling**
'Sri' V. Sridharan

Much of the existing jobshop scheduling literature concerns itself almost exclusively with non-delay (ND) schedules. ND schedules may not be appropriate for a number of cases. We present a taxonomy of scheduling problems and identify conditions when tactically delayed (TD) schedules may be appropriate. TD schedules are defined as those which contain deliberate idle times between jobs. TD procedures are defined as those which permit TD schedules when appropriate. After reviewing the extant literature dealing with the so called E/T problem, we reach the conclusion that the approach taken to solve these problems – the approach of first finding the best sequencing and then inserting idle times – is inadequate and cannot guarantee optimality. Recognizing the NT-completeness of the problem, we propose a new heuristic approach called the DT approach which is based on decision theory. The DT approach is a dispatch like approach which takes a global view and is criterion independent. We present DT procedures for dynamic one-machine problems with regular and non-regular penalty functions. They permit TD schedules when appropriate. Experimental investigation indicates that the DT-TD procedures outperform several existing heuristics and find the optimum solution for a large fraction of problems studied. We believe that the DT approach is an attractive alternative whose effectiveness can be further improved by obtaining better estimates of job completion times.

This is joint work with Z. Zhou and J.J. Kanet.

**Approximation Results for Wavelength Allocation in Optical Networks**
Klaus Jansen

The wavelength allocation problem in optical networks with directed tree topology has recently received a substantial amount of attention. Wavelengths must be assigned to connection requests which are represented as directed paths, and it is required that paths receive different wavelengths if they share a directed link. The goal is to minimize the number of different wavelengths used. The best previously known approximation algorithm for this $\mathcal{NP}$-hard problem routes any set of paths with 7/4L wavelengths, where L is the maximum load on a directed link. We give an improved algorithm that uses at most 5/3L wavelengths. This is provably the best ratio that can be achieved by any greedy algorithm for this problem.

This is joint work with T. Erlebach (München), C. Kaklamanis (Patras) and G. Persiano (Salerno).

### Computational Complexity Analysis of Single Machine Scheduling Problems with Job Release Dates and Processing Times Dependent on Resources

Adam Janiak

Single machine scheduling problem was considered with either job release dates or job processing times depend on continuously divisible resources. The process of preheating ingots in soaking pits and next hot-rolling them on belooming mill was given as an practical example of application of the considered models. There were considered linear and nonlinear models of job release dates and processing times and the following optimality criteria: a) minimization of the makespan s.t. a given constraint on the total resource consumption is met, b) minimization of the total resource consumption s.t. a given common deadline is not violated and c) bicriterial approach searching for the Pareto compromise points between the makespan and the total resource consumption.

These problems were considered with and without additional precedence constraints. The computational complexity analysis of all the considered problems and their special cases was done.

### Linear Time Bin-Packing Algorithm

Hans Kellerer

In 1985 Mertel published a linear time algorithm with a 4/3 asymptotic worst-case ratio for the one-dimension bin packing problem. The algorithm is based on a linear classification of the sizes of the items and thereafter - according to the number of elements in certain subclasses - pairing the items in a clever way. In his paper Martel mentioned a natural generalization of this algorithm, which should lead to a 5/4 algorithm. Although this seemed to be very simple, the improvement had not been realized until now. We present an algorithm which uses the ideas of Martel and has a 5/4 asymptotic ration. Solving appropriate linear programs reduces the complexity of the proof.

**Simultaneous Loading, Routing and Assembly Plan Selection in a FAS**
Tadeusz Sawik

The paper presents integer programming formulations and a heuristic solution approach for a bicriterion machine loading, assembly routing and assembly plan selection in a flexible assembly system. The problem objective is to simultaneously determine an allocation of assembly tasks among the stations and to select assembly plans and assembly routes for a mix of products so as to balance station workloads and minimize total transportation time. In the approach proposed, first the station workloads are balanced using a linear relaxation-based heuristic and then assembly plans and routes are selected for all products, based on a network flow model. An illustrative example is provided and some computational results are reported.

**Sequencing JIT Mixed-Model Assembly Lines under Station Load- and Part Usage-Constraints**
Andreas Drexl

In the past several years, there has been growing interest in the problem of sequencing mixed-model assembly lines. So far most research concentrated on the 'Level scheduling problem' while only some papers have been published on the 'Car sequencing problem'. The subject of the level scheduling problem is to derive 'smooth' production schedules while the car sequencing problem takes sequencing constraints related to station loads and part usages into account. The former in general is a difficult optimization problem while for the latter even the feasibility problem is $\mathcal{NP}$-complete.

This paper presents a nonlinear integer programming model which covers both the balancing requirements of level scheduling and the constraints of car sequencing. For the solution of the problem we present a set partitioning/column generation approach. Solving the LP-relaxation of this model by column generation provides tight lower bounds for the optimal objective function value.

**Approximation Algorithms for Scheduling Problems with Communication Delays**
Rolf Möhring

In the last few years, multi-processor scheduling with interprocessor communication delays has received increasing attention. This is due to the more realistic constraints in modeling parallel processor systems.

We develop approximation algorithms for minimizing the makespan and the weighted sum of completion times for 'small' communication delays, arbitrary precedence constraints, and parallel identical processors.

The common underlying idea of our algorithms is to compute first a schedule by LP-relaxation and rounding that regards all constraints except for the processor restrictions. This schedule is then used to construct a provable good feasible schedule for a limited number of processors by a suitable generalization of Graham's list scheduling rule. We thus obtain a simple $\frac{7}{3}$-approximation algorithm for the makespan, and a 6.14232-approximation algorithm for the weighted sum of completion times.

Complementing these results, we obtain new polynomial-time exact algorithms and $\mathcal{NP}$-completeness results on the class of series-parallel precedence constraints.

This is joint work with Markus Schäffter and Andreas Schulz, Berlin.

**Scheduling in Order-Picking Systems**
Egon Wanke

An order-picking system is used to pick articles into bins for delivery. Is consist of roller conveyers that cyclically connects so-called picking stations. The transportation system automatically pushes out the bins to the picking stations where they can be stopped without blocking the flow of bins on the main conveyer. The corresponding scheduling problem is called "conveyer job shop scheduling". The buffer for all jobs is the conveyer itself. That is, if machine $P_j$ is busy when job $J_i$ wants to start its task $t_{i,j}$ then $J_i$ has to wait a complete turn around time until it gets the next chance.

We show that it is possible to schedule in polynomial time the jobs given in sequence of jobs one after the other such that each job $J_i$ finishes its processing on the last machine as early as possible and starts its processing on the first machine as late as possible. The possibility to keep a job on a machine after its processing is also taken into account. The polynomial time solution is based on the computation of a set of idle time intervals $I_j$ and a set of availability time intervals $A_i$ for each machine $P_j$. The size of each $I_j$ is bounded by the number of jobs and the size of each $A_i$ is at most $|A_i\Gamma_1| + 2 \cdot |A_i|$.

### A Priori Knowledge in Hard Real-Time Systems

Jörg Apsel

Real-time systems are being increasingly used in several applications which are time critical in nature. Scheduling real-time tasks within certain time constrains have to be done not only correctly but also with low complexity. Hence, in applications with unknown arrival of requests dynamic feasibility tests are used instead of static ones, because they are able to reduce the number of processing steps. Moreover a priori knowledge about number of processors and processor utilization are needed to provide fault-tolerance in multiprocessor systems. In our paper we present both, a dynamic feasibility test for a task system with independent tasks and identical processors and a priori knowledge about schedule and processor characteristics.

### Structural Approach for Production Scheduling Problems with Multiple Resources and Jobs

Eugeniusz Toczylowski

A structural framework for modeling general scheduling problems in the context of FMS scheduling is presented. The formulation of the problem considered in the paper is a generalization of the job-shop scheduling problem, where various resource types and limitations may be taken into account.

In the second part of the paper we show how to exploit a particular feature of many realistic scheduling problems, where many identical jobs belonging to $N$ types of jobs are to be scheduled. The approximate two-level approach is based on solving at the aggregate level the allocation problem, which provides a decomposition of the desired general schedule into a set of elementary, extremal fixed cyclic schedules.

**Tabu Search for the Multi-Mode Job-Shop Problem**
Peter Brucker, Jürgen Neyer

In a multi-processor-tasks job-shop problem (MPTJSP) there is a machine set associated with each operation. To process the operation all machines are needed for the whole processing period. The objective is to find a schedule which minimizes the makespan. In a multi-mode job-shop problem (MMJSP) there is a set of machine sets associated with each operation. One has to assign a machine set to each operation and to solve the resulting MPTJSP such that the resulting makespan is minimized. For the MMJSP a tabu-search algorithm is presented. Computational results are reported.

**What is the Most Generic Model: Job-Shop or Flow-Shop?**
Alain Guinet

We study the problem of scheduling N independent jobs in a job-shop environment. Each job must be processed on M machines according to individual routes. The objective is to minimize the maximum completion time of the jobs. First, the job-shop problem is reduced to a flow-shop problem with job precedence constraints. Then, a set of flow-shop algorithms are modified to solve it. To evaluate the quality of these heuristics, several lower bounds on the optimal solution and job-shop algorithms have been computed and compared with the flow-shop algorithm solutions for 3040 problems. The flow-shop heuristics appear especially promising for job-shop problems with "Flow-like" properties (e.g. two job-shop in series).

**Dynamic Resource Scheduling in Distributed Real-Time Systems**
EDGAR NETT

Real-time computing is becoming an important technology for many applications such as process control, service robotics and flexible manufacturing. Traditionally, real-time systems have been realized as isolated, embedded systems, dedicated to provide the requested service for single, specific applications like automatic brake systems in vehicles or control systems for nuclear plants. Unfortunalty, the solutions provided here do not longer suffice for more complex systems as they are encountered in many of the above mentioned applications. The most distinguishing requirement is the need to dynamically adapt to changing conditions. This talk presents the basic ideas of our approach, called TaskPair-Scheduling. Its usefulness is sketched by means of a robot application, called GMD-Snake.

**Linear Programming Bounds for Minimizing Total Completion Time in a Two-Machine Flowshop**
STEEF VAN DE VELDE

The $\mathcal{NP}$-hard problem of minimizing the sum of $C_j$ in a two-machine flowshop looks deceivingly simple - but state-of-the-art algorithms cannot even solve all problems with 25 jobs. This poor performance is due to the lack of strong lower bounds. The best lower bounds known so far, which require $O(n^3)$ time to compute, show an average relative gap of about 1.8 % from the optimal solution value, where n is the number of jobs.

We propose a series of powerful linear programming bounds that are derived from a formulation that is based on positional completion times. We derive the Lagrange problem and show how the Lagrange dual problem can be solved directly as a linear program. The simplest bound in this series, which requires the solution of a linear program with $O(n^2)$ variables and $O(n)$ constraints shows an average of only 0.8 %

**Scheduling Interval Orders with Large Communication Delays**
Foto Afrati

The general case of scheduling with unit execution time and communication delay $> 1$ is known to be $\mathcal{NP}$-complete even in the case where infinitely many processors are available. We investigate the same problem for a special case of directed acyclic graph, namely the internal order graph. For this class of graphs, there are polynomial algorithms both for the case of no communication and the case of unit communication delay; in fact in both cases a similar methodology is applied. The case, though, with large communication delays is dramatically different. We had to exploit fine properties of internal order graphs in order to develop polynomial algorithm which finds the optimal schedule in this case. Our algorithm constructs an auxiliary graph and reduces the problem to a shortest graph problem.

**An Approach to Assign Spare-Capacities in Restorable Networks**
Dritan Nace, Jacques Carlier and Jean Luc Lutton

Future services of telecommunication will demand a fault-tolerant network with complete survivability. In this context, developing fast and efficient rerouting algorithms which can reconfigure the network in case of failures will have a great impact in the quality of services. This problem is related very closely with this of designing reliable networks where an economical spare-capacity assignment is strongly required. This paper presents the method used for assigning the spare capacities under different scenarios.

We consider the problem that consists in computation of spare capacities. Spare capacities are needed to reroute traffic demands when failures occur in a network. Usually for most of realistic networks, spare capacities are dimensioned such that one can reroute all traffic demands for every single failure of a network component. These network components are transmission systems, cables and nodes. Up to now it is impossible to optimize simultaneously the topology, the edge dimensioning, the routing and the spare capacity for rerouting in a network. So we proceed in two steps. First we determine the topology, the edge dimensioning and the routing of traffic demands. Then we compute the spare capacities and reroutings using the topology and routings found in the first step. For this second step, we have developed an algorithm which is based on linear programming methods. It finds the optimal continuous solution and can treat many options : link or end-to-end rerouting, recovery or not recovery of the capacities used by the failed demands, maximum length of rerouting paths. We modelize the spare capacity assignment problem as a non-simultaneous multicommodity flow problem. We use the "edge-path" multicommodity formalism. This leads to very large linear programs even for medium size problems (30 to 50 nodes). So we propose a decomposition method which allows separate treatments of the failures. To each failure corresponds a simultaneous multicommodity flow problem. We solve these problems using the Dantzig-Wolfe decomposition method which is nothing else than a path generation method. We have used the CPLEX linear programming code to implement this double decomposition method. It allows solving medium size problems (40 nodes) in less than 5ms CPU time (HP735-135) and solving large problems (100 nodes) in less than one hour.

The obtained solution by resolving this LP problem has always a number of variables (flows and spare-capacities) with continuous values even that there are also a lot of variables with integer values. In fact, for SDH and SONET Networks, both flows and spare-capacities must have integer values, when for ATM networks only the spare-capacities must have integer values. To deduce a near-optimal admissible solution we propose an heuristic which gives very good results.

This work was supported by the CNET (Centre National d'Etudes de Telecommunications) contract 94 1B 068.

**Compact Open Shop Scheduling of Unit Time Operations**
Marek Kubale

We consider the following problem: Given $m$ specialized machines and $n$ jobs. Each job consists of $m$ operations and each operation must be performed by precisely one dedicated machine. Execution time is either 0 or 1 and the order in which operations of a job are performed is immaterial (open shop). Moreover, we assume that operations of a job are performed in a no-wait manner and machines work without inserted idle times. The aim is to find a schedule with minimum possible length. Thus our problem can be described as $O|p_{ij} \in \{0,1\}, no - wait\&idle|C_{max}$. Unfortunately, not always such a solution is possible.

The problem considered can be modeled as a bipartite graph coloring problem. Then a solution to this problem corresponds to a consecutive coloring of the edges of the associated bipartite graph using the minimum possible number of colors. Since the latter problem is $\mathcal{NP}$-hard, we focus on identifying special cases which can be solved in polynomial time with respect to $m$ and $n$. This concerns such highly structured graphs as: complete bipartite, trees and cacti, regular and (2,delta)-regular, multidimensional grid graphs, etc.

**Scheduling Some Precedence Graphs on Uniform Processors with Communication Delays**

Bernard Penz

The first studies concerning the scheduling problem on uniform processors (of different speeds) have been done many years ago. There is today a regain of interest concerning this problem because of the popularization of networks of work-stations of the same type. These scalable systems may include several micro-processors of different generations which only differ by their clock speeds (or frequency). A good way to model these systems is to consider uniform processors.

The first problem considered in the talk can be expressed as follows Q2 |pj=1, Ci, j=1, chains | Cmax, according to the standard notation. This is a joint work with W. Kubiak and D. Trystram. For this problem, we present an optimal polynomial time algorithm for scheduling chains on two uniform processors with anti-speed 1 and a (a integer greater or equal than 2.

The second problem is Q2 | pj=1,Ci,j=1,complete in-tree | Cmax. This is a joint work with J. Blazewicz, F. Guinand and D. Trystram. For this particular structure of precedence constraints, we give an optimal polynomial time algorithm. The restriction for the value of a is the same than previously.

**On the Efficiency of Dynamic Scheduling for Automatic Parallelization**

Emmanuel Jeannot and Michel Cosnard

Static scheduling of task graphs has two major drawbacks. It requires the task graph to be in memory at compile time. Hence, since large task graphs (in the order of million tasks) are too big to fit into memory, it is not possible to schedule such large task graphs. Static scheduling techniques also require the analysis to be performed each time the value of the program parameters change. Hence, it is not possible to build generic parallel programs.

In this talk we present a new approach, which allow to handle large task graphs, and to build generic programs. It uses a new model called the Parameterized Task Graph (PTG). The PTG is a problem size-independent representation of the sequential program. A tool, PlusPyr, is able to do the symbolic analysis of the program, and to build the PTG. In our approach, the output of our automatic parallelization tool is an SPMD program composed of a supervisor part, and as many executors as processors in the parallel machine.

The supervisor schedules the tasks using the Parameterized Task Graph Dynamic Scheduler (PTGDS). The executors execute the schedule on their respective processors. We present theoretical results concerning our approach. The time complexity, and a performance bound (at least 2 for coarse grained DAG) are proved for PTGDS. We also show that, if the average number of operations performed by each task is sufficiently large, then the scheduling overhead is negligible with respect to the parallel time. Our experimental results show that PTGDS performed well on a fixed number of processors, and the memory cost of our algorithm is very low with respect to size of the task graph.

In conclusion we can say that this approach is efficient when dealing with coarse grain task graphs, and when tasks durations are important.

**Scheduling with Precedence and/or Resource Constraints**

Evripidis Bampis, Jean-Claude König

We consider a task graph model where the computational problem is represented by a directed acyclic graph (dag). The basis of our analysis is the communication delay between the time some result has been produced on some processor and the time this information can be used by another processor. This communication delay, denoted by $c$ is the fundamental parameter of a parallel architecture. Given this value, the problem that we have to solve is how to schedule the dag on the processors under the following communication model:

If task $i$ starts its execution at time $t$ on processor $P_1$ and task $j$ is a predecessor of $i$ in the dag, then either $j$ starts its execution not later than time $t-p_j$ on processor $P_1$ or not later than time $t-p_j-c$ on some other processor, where $p_j$ represents the execution time of task $j$.

Many papers studied the complexity of this problem in the case where for every task of the dag $p_j = 1$, and $c = 1$ $(P|prec, c = 1, p_j = 1|C_{max})$. This problem was first addressed by Rayward-Smith who established the $\mathcal{NP}$-hardness of the problem. Picouleau considered the same problem and showed that the problem of deciding if a dag can be scheduled on at most 3 units of time is polynomial. He had also considered the case where the number of processors is unbounded $(\overline{P}|prec, c = 1, p_j = 1|C_{max})$ and he established the $\mathcal{NP}$-completeness of the problem of deciding if an arbitrary dag can be scheduled in at most 8 time units.

Independently Hoogeveen Lenstra and Veltman, and Saad showed that this last problem becomes $\mathcal{NP}$-complete for $C_{max} = 4$. A corollary of this result is that there will never exist a polynomial-time algorithm with performance bound smaller than $\frac{5}{4}$, unless $P = NP$. Similar results have been established for the case where the number of processors is unbounded and the performance bound becomes $\frac{7}{6}$.

In this talk, we are interested in the case where the execution time of the tasks is equal to one and the communication delay is a constant value greater than 1; does there exist a schedule of length $L$?

**Labelled-Delay Scheduling**
Andreas Jakoby, Maciej Liskiewicz, Rüdiger Reischuk

A great deal of effort has been done to study efficient scheduling algorithms for parallel programs represented as directed acyclic graphs. Such a graph represents the elementary computation steps (tasks) of a program on a given input data and dependencies among them. It is well known that the scheduling problem for this model is $\mathcal{NP}$-complete, in general. Hence one can expect that more difficult problems arise when a more realistic model (allowing e.g. conditional branching) for parallel programs is assumed.

In this paper we study a scheduling problem for algorithms containing the common for parallel programs `ALT` and `PAR` constructors as in OCCAM language. One of our main results says that even if the values for all `ALT` guards depend on scheduling policy only, the problem to find an optimal schedule for it is very hard, namely $\mathcal{PSPACE}$-hard.

The scheduling problem we consider, assumes unit execution time of each task and that there is communication delay between the time some information is produced at a processor and the time it can be used by another processor.

| input | output | complexity |
|---|---|---|
| `ALT` | `ALT`, `PAR` | $\mathcal{NL}$ |
| `PAR` | `ALT` | $\mathcal{NP}$-complete |
| `PAR` | `PAR` | $\mathcal{NP}$-complete |
| `PAR` | `ALT`, `PAR` | $\mathcal{NP}$-complete |
| `ALT`, `PAR` | `ALT` | $\mathcal{NP}$-complete |
| `ALT`, `PAR` | `PAR` | $\mathcal{BH}_2$-hard |
| `ALT`, `PAR` | `ALT`, `PAR` | $\mathcal{PSPACE}$-hard |

**Batching to Minimize Job Flow Times on One Machine**
Erwin Pesch

We consider the problem where a given number of items of different types are to be processed on a single machine. The items are supposed to be processed in batcher which become available when the last item of the batch is finished. Hence flow times are the same for all items in a batch and correspond to the completion time of the batch. Only items of the same type can be batched together. There is a setup time whenever a batch of a certain item type is going to be processed. The problem is to find an optimal schedule with respect to the batching and sequencing decisions that minimizes the total item flow time. The single type case can easily be solved. Efficient heuristics are provided for the multiple type case.

**Tree Precedence Scheduling: The Strong - Weak Distinction**
Moshe Dror, W. Kubiak and J.Y-T. Leung

In this talk we formally introduce the distinction between strong and weak precedence relation in scheduling for the case of trees. We demonstrate that this distinction in precedence relation for trees (as demonstrated in earlier work for chains) is a proper one in the sense that some problems are solvable in polynomial time if weak tree relation is assumed and are $\mathcal{NP}$-hard in the case of strong tree relations. For some other problems, both weak tree and strong tree relations are $\mathcal{NP}$-hard, and for yet other problems both weak and strong tree relations are polynomially solvable. Since the distinction between strong and weak tree precedence was not clearly recognized in the past, this work establishes the existence of new problem categories in scheduling.