Proceedings of the DARPA Workshop on Innovative Approaches to Planning, Scheduling & Control, 1990.

The CORTES Project: A Unified Framework for Planning, Scheduling and Control

Morgan Kaufmann Pub. Co.

Mark S. Fox and Katia P. Sycara

Center for Integrated Manufacturing Decision Systems
The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Abstract

We present an overview of CORTES, an integrated framework for production planning, scheduling and control (PSC). CORTES's approach to PSC problems departs from others in the hypotheses it explores: *Generality Hypothesis*: There exists a single approach that can optimize decision making across a wide variety of PSC problems. *Flexibility Hypothesis*: The same approach can be used for both planning, predictive scheduling and reactive control. *Uncertainty Hypothesis*: In order to provide the appropriate level of precision in PSC, reasoning about uncertainty must be an integral part of the PSC approach. *Scale Hypothesis*: Large PSC problems, that contain thousands of activities, resources and constraints, must be solved in a qualitatively different manner than small PSC problems. CORTES uses Constrained Heuristic Search to make PSC decisions. In this paper, we describe CORTES, its architecture, problem solving method, and functions including modeling, planning, scheduling, distributed scheduling, dispatching, and uncertainty management.

1. Introduction

Our research explores the role of constraints in solving planning, scheduling and control (PSC) problems. It is generally believed that to efficiently construct optimizing solutions to large PSC problems, a fundamental understanding of ***problem structure*** and ***properties*** is required. It is our conjecture that knowledge of domain constraints will lead to this understanding. The goal of the CORTES project is to operationalize this conjecture.

CORTES is a distributed system for production planning, scheduling and control. CORTES is designed to be composed of an integrated set of modules distributed across many workstations and connected by a communication network. The overall architecture is shown in Figure 1-1.

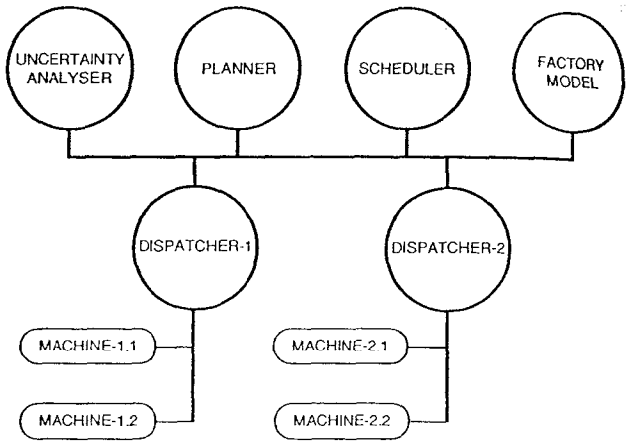

Figure 1-1: The CORTES Architecture

CORTES represents a departure from previous approaches to solving PSC problems in the hypotheses it explores:

1. ***Generality Hypothesis:*** There exists a single approach that can optimize decision making across a wide variety of PSC problems. Previously, PSC approaches were tailored to the particular production environment, with the "common wisdom" being that there does not exist a single approach, short of enumeration, that applies to all PSC problems. We believe that there does exist a single approach that may be generally applied to PSC problems, that also provides very good results and is computationally efficient.
2. ***Flexibility Hypothesis:*** The same approach can be used for both planning, predictive scheduling and reactive control. Traditionally, planning, scheduling and control approaches have tended to be separate and unrelated in approach. For example, in actual production environments,Manufacturing Resource Planning (MRP) tends to be used for

planning, scheduling approaches can range from dispatch approaches to knowledge based scheduling, and control tends to be ignored. In AI, planning algorithms tend to be generative, scheduling is constraint directed, and control is reactive.

3. ***Uncertainty Hypothesis:*** In order to provide the appropriate level of precision in PSC, reasoning about uncertainty must be an integral part of the PSC approach. Production environments contain a plethora of stochastic events that increase the uncertainty with which a schedule may be executed. For example, unexpected events such as personnel not showing up for work, machine failures, failure of resources, etc, may quickly invalidate a schedule. Consequently, PSC approachs must take a pro-active approach in mitigating the effects of uncertainty.

4. ***Scale Hypothesis:*** Large PSC problems, that contain thousands of activities, resources and constraints, must be solved in a qualitatively different manner than small PSC problems. The point is that in large PSCs, the aggregate behavior of the system be optimized, as opposed to any individual entity or job. Optimizing each decision is computationally expensive. Instead, many decisions must be made at the aggregate level using statistical summaries of underlying requirements.

CORTES is evolutionary in its approach in that it can be viewed as a continuation of the line of constraint directed scheduling systems developed at Carnegie Mellon University [Fox & Smith 84, Smith et al. 86, Fox 90]. It departs from the approach of these previous systems in its use of Constrained Heuristic Search (CHS) as its underlying problem solving paradigm [Fox 89].

In the reminder of the paper, we first review the Constrained Heuristic Search (CHS) problem solving paradigm. We then describe the functionality of each of the modules in figure 1-1 and how this is provided using CHS.

2. Constrained Heuristic Search

Our approach to both planning and scheduling is based upon a problem solving paradigm we call Constrained Heuristic Search (CHS)[1] CHS views problem solving as a constraint optimization activity. CHS combines the process of constraint satisfaction (CSP) [Mack 87] with heuristic search (HS). CHS retains heu search's synthetic capabilities and extends it by addin structural characteristics of constraint satisfa techniques. In particular, our model adds to the defin of a problem space [Newell & Simon 76], compose states, operators and an evaluation function, by refini state to include:

1. **Problem Topology**: Provides a structural characterization of a problem.

2. **Problem Textures**: Provide measures of a problem topology that allows search to be focused in a way that reduces backtracking.

3. **Problem Objective**: Defines an objective function for rating alternative solutions that satisfy a goal description.

This model allows us to (1) view problem solving constraint optimization, thus taking advantage of the techniques, (2), incorporate the synthetic capabilities heuristic search, thus allowing the dynamic modification the constraint model, and (3) extend constraint satisfactio to the larger class of optimization problems. In th following, problem topology and textures are defined.

2.1. Problem Topology

We define problem topology as a constraint graph G, composed of variables V and constraints C:

V is a set of variables $\{v_1, v_2, ..., v_m\}$
C is a set of constraints $\{c_1, c_2, ..., c_n\}$

Each variable in N may be a vector of variables whose domains may be finite/infinite and continuous/discrete. Constraints are n-ary predicates over variables vertices.

We extend the topology to allow constraints to be grouped into a modified conjunctive normal form:

$[s_1 \text{ AND } s_2 \text{ AND } ... \text{ AND } s_o]$

where each s_i denotes a set of constraints where either only one or at least one constraint must be satisfied.

We distinguish between two types of problem topologies:

Definition 1: A *completely structured problem* is one in which all non-redundant vertices and edges are known a priori.

This is true of all CSP formulations and in this case CHS reduces to either a CSP or a COP (i.e., optimization problem).

Definition 2: A *partially structured problem* is one in which not all non-redundant vertices and

[1] This section is a composed of excerpts from [Fox 89].

edges are known prior to problem solving.

This definition tends to be true of problems in which synthesis is performed resulting in new variables and constraints (e.g. the generation of new subgoals during the planning process).

Operators in CHS have many roles: refining the problem by adding new variable and constraint vertices, reducing the number of solutions by reducing the domains of variables (e.g., assigning a value to a variable vertex), or reformulating the problem by relaxing constraints or omitting constraints and/or variables.

Our intent is to distinguish topologies that lead to significant changes in problem solving quality and efficiency. Examples include:

- The decomposability of constraint graphs into unconnected or lossely connected subgraphs, allowing the problem solver to focus on one set of variables and constraints before attending to another.
- Graph width which combined with arc-consistency will guarantee backtrack free search [Freuder 82].
- Contention graphs which identify the degree of contention that exists among variables for the same values.

2.2. Problem Textures

Focus of attention in search is concerned with the ability of the search algorithm to opportunistically decide where the next decision is to be made [Erman et al. 80]. In CHS, for search to be well focused, that is to decide where in the problem topology an operator is to be applied, there must be features of the topology that differentiate one subgraph from another, and these features must be related to the goals of the problem. We have identified and are experimenting with seven such features that we call problem *textures* [Sadeh & Fox 88]. Below we define these textures for CHSs where all solutions are equally preferred, i.e., the Problem Objective rates all solutions to the constraints equally acceptable.

(Variable) Value Goodness: the probability that the assignment of that value to the variable leads to an overall solution to the CHS (i.e. to a fully consistent set of assignments). This texture is related to the value ordering heuristics [Haralick & Elliott 80] which look for the least constraining values. Value ordering heuristics are meant to reduce the chance of backtracking. In the case of discrete variables, the goodness of a value is the ratio of complete assignments that are solutions to the CHS and have that value for the variable over the total number of possible assignments.

Constraint Tightness: Constraint tightness refers to the contention between one constraint or a subset of constraints with all the other problem constraints. Consider a CHS A and a subset C' of constraints in A. Let B be the CHS obtained by omitting C's constraints in A. The constraint tightness induced by C' on A is defined as the probability that a solution to B is not a solution to A. In the case of discrete variables, this is the ratio of solutions to B that are not solutions to A over the total number of solutions to B.

Variable Tightness with respect to a set of constraints: Again consider a CHS A, a subset C' of constraints, and the CHS B obtained by omitting C' in A. A variable V's tightness with respect to the set of constraints C' is defined as the probability that the value of V in a solution to B does not violate C'. In the case of discrete variables, this is simply the ratio of solutions to B in which V's value violates C' (i.e. at least one of the constraints in C') over the total number of solutions to B.

Constraint Reliance: This measures the the importance of satisfying a particular constraint. Consider a constraint c_i. We defined CHS B as being CHS A - $\{c_i\}$. Given that constraints can be disjunctively defined, the reliance of CHS A on a constraint c_i is the probability that a solution to CHS B is not a solution to A. In the case of discrete variables, constraint reliance is defined as the ratio of the number of solutions to CHS B that are not a solution to CHS A to the number of solutions to CHS B. The larger the value, the greater the reliance the problem has on satisfying the particular constraint.

Variable Tightness: Consider a variable v in a CHS A. Let C' be the set of constraints involving v and B be the CHS obtained by omitting C' in A. v's tightness with respect to C' is simply called v's tightness. Hence the tightness of a variable is the probability that an assignment consistent with all the problem constraints that do not involve that

variable does not result in a solution. Alternatively one can define *variable looseness* as the probability that an assignment that has been checked for consistency with all the problem constraints, except those involving that variable, results in a fully consistent assignment. Notice that if one uses a variable instantiation order where v is the last variable, v's tightness is the *backtracking probability*. Variable looseness/tightness can be identified with variable ordering heuristics [Haralick & Elliott 80, Freuder 82] which instantiate variables in order of decreasing tightness.

These textures generalize the notion of constraint satisfiability or looseness defined by [Nadel 86] and apply to both CHSs (and CSPs) with discrete and continuous variables. Notice that, unless one knows all the CHS's solutions, the textures that we have just defined have to be approximated. Textures may sometime be evaluated analytically [Sadeh & Fox 88]. A brute force method to evaluate any texture measure consists in the use of Monte Carlo techniques. Such techniques may however be very costly. In general, for a given CHS, some textures are easier to approximate than others, and some are also more useful than others. Usually the texture measures that contain the most information are also the ones that are the most difficult to evaluate. Hence there is a tradeoff. Each domain may have its own approximation for a texture measure.

2.3. Problem Objectives

Many problems, for example scheduling, are optimization problems and not simply satisfaction problems. The notion of what is best becomes important. Rather then defining what is best in an evaluation function or an objective function, our goal is to embed objectives directly in the constraint graph so that it can be both propagated and used to make local decisions. For example,

- Disjunctive constraint sets may have preferences associated with each disjunct.
- Start times, commonly found as a variable in scheduling constraint graphs, can have preferences associated with each alternative time.

In our work we have extended the textures to take into account the Problem Objectives, using Bayesian probabilities to approximate the likelihood that a variable results in an optimal value [Sadeh & Fox 88].

2.4. CHS Problem Solver

The CHS model of problem solving is a combination of constraint satisfaction and heuristic search. Search is performed in the problem space where each state contains a problem topology. The problem solving model we propose contains the following elements:

- An initial state is defined composed of a problem topology, i.e., the PSC activity, time and capacity constraint graph,
- Constraint propagation is performed within the state,
- Texture measures and the problem objective are evaluated for the state's topology,
- Operators are matched against the state's topology, and
- A variable node/operator pair is selected and the operator is applied, i.e., a resource or start time is assigned to an activity.

The application of an operator results in either adding structure to the topology, further restricting the domain of a variable, or reformulating the problem (e.g., relaxation).

It is our belief, which is supported by experimentation, that this approach is powerful enough to solve a variety of PSC problems. Domains in which it has been applied include, job shop scheduling [Sadeh 90], cell scheduling and transportation planning [Sathi et al. 90]. Secondly, the opportunism inherent in the approach, allows the approach to be applied to both predictive planning and scheduling, and reactive control.

2.5. Representation

The main conceptual primitives in the CORTES representation are *activities*, *resources*, *production units*, *states*, and *constraints* [Sathi et al. 85, Fox 88]. These primitives provide an extensible framework that can be used to represent the relevant aspects of manufacturing environments. In addition, these primitives are represented at various levels of conceptual abstraction depending on the granularity of knowledge. For example, an *operation* is a specialization of an activity. An important component of the representational framework is the *relations* that connect the primitives and their instantiations. The main types of relations are temporal and causal.

In general, there are five types of constraints that a scheduler should take into consideration. These five types are domain-independent and help structure constraints in many kinds of scheduling domains (e.g., factory scheduling, transportation scheduling).

- Physical constraints. Physical constraints include, number of machines, fixtures, setup and run times for each operation.

- Organizational constraints. Examples of organizational constraints include meeting due dates, reducing Work in Process, increase machine utilization, and enhance throughput.
- Preferential constraints. Examples of preferential constraints include preference for using a particular machine for an operation (perhaps because of its speed or accuracy), or using a particular human operator (perhaps because of his skill).
- Enablement constraints. These refer to constraints, the fulfillment of which creates a state that enables the execution of an activity. For example, a process plan embodies enablement constraints.
- Availability constraints. These constraints refer to the availability of particular resources at scheduling time. For example, a machine may become unavailable because of breakdown, the assignment of a third shift makes extra resources available for scheduling.

In the model, we treat explicitly two types of constraints, *required constraints* and *preferential constraints* [Fox 83]. The degree of satisfaction of a preferential constraint is expressed by a *utility function* ranging between 0 and 1. A value of 0 utility is non-admissible; a value of 1 is optimal. Variables can be constrained by more than one constraint. The utility value associated with a variable is calculated by taking the weighted sum (with constraint importance as the weight) of the utilities of all the constraints that affect the variable.

Constraints differ in *importance*. A particular constraint could have different importance depending on the context in which it is applied. The importance of a constraint is specified by a value between 0 and 1. An importance of 0 implies that the constraint should not be considered, and 1 signifies maximum importance. The actual level of importance is relative to the importance of the other constraints under consideration. The measure of importance of a constraint may be viewed as a weight that can be combined with a constraint's utility value to form a weighted combination of utilities. Constraints also differ in *relevance*. Depending on the context, a constraint may be more relevant than others.

3. Scheduler

The detailed scheduler is an activity-based scheduler [Sadeh 90], where the activities are the operations that must be scheduled according to a process plan that specifies a partial ordering among these operations. Each operation requires one or several resources for each of which there may be one or several alternatives. Sched is viewed as a constrained heuristic search problem v solution is a schedule that satisfies the many technolo temporal, organizational, and preference constraints are imposed both by the characteristics of the job itself and the environment.

The scheduler models a problem as a constraint g where there are two types of nodes: activities resources. An activity is an 4-tuple defining its start duration, and resources it is to use. With each activity associate utility functions that map each possible start and each possible resource alternatives onto a utility v (i.e. preference). These utilities [Fox 83, Sadeh & 88] arise from global organizational goals such as redu order tardiness (i.e. meeting due dates), reducing c earliness (i.e. finished good inventory), reducing c flowtime (i.e. in-process inventory), using acc machines, performing some activities during some s rather than others, etc. A resource is a 3-tuple definin total capacity, available capacity over time, and activities that are scheduled to use it.

We distinguish between two types of constra activity temporal constraints and capacity constraints. activity temporal constraints together with the order rel dates and latest acceptable completion dates restrict the of acceptable start times of each activity. The capa constraints restrict the number of activities that a reso can be allocated to at any moment in time to the capacit that resource. For the sake of simplicity, we only cons resources with unary capacity in this paper. Typically limited capacity of the resources induces interacti between orders competing for the possession of the sa resource at the same time.

The schedule is built incrementally by iterativ selecting an activity and assigning a start time a resource(s) to it, propagating temporal and capac constraints and checking for constraint violations. constraint violations are detected the system backtrac Search is focused via a set of *variable* and *value* orderi heuristics so as to minimize backtracking and optimi schedule quality.

The variable ordering heuristic assigns a *critical measure* to each unscheduled activity; *the activity with t highest criticality is scheduled first.* The value orderi heuristic attempts to leave enough options open to t activities that have not yet been scheduled in order reduce the chances of backtracking. This is done b assigning a *goodness* measure to each possible reservatic of the activity to be scheduled. Both activity criticality an value goodness are composed of *texture measures.* Th

next two paragraphs briefly describe both of these measures[2].

A critical activity is one whose resource requirements are likely to conflict with the resource requirements of other activities. [Sadeh & Fox 88, Sadeh 90] describes a technique to identify such activities. The technique starts by building for each unscheduled activity and for each appropriate time interval a probabilistic *activity demand* that denotes the probability that the activity will require a resource at that time interval. Clearly activities with many possible start times and resource reservations tend to have smaller demands at any moment in time, while activities with fewer possible reservations tend to have higher ones. In a second step, the activity demands for each resource are aggregated over time to form a demand profile for a resource. The demand profile expresses likely contention for the resource over time. The percentage contribution of an activity's demand to the aggregate demand for a resource over a highly contended-for time interval is the *activity reliance*.

To choose the next activity to schedule, the scheduler focuses on the resource/time interval with the highest aggregate demand. The activity with the the highest reliance on the resource is picked to be scheduled next, since it is the activity that is most likely to be involved in contention for the resource.

The particular start time assigned to the chosen activity is picked using either of two strategies:

1. **A Least Constraining Value Ordering Strategy (LCV)**: This heuristic attempts to select the reservation that is the least likely to prevent other activities to be scheduled.
2. **A "Greedy" Value Ordering Strategy (GV)**: At the other extreme, a reservation can be chosen that maximizes the preference of the activity for the resource/time interval.

Experimental results have demonstrated the effectiveness of this approach for problems where resource contention is an issue [Sadeh 90].

4. Distributed Scheduling

As part of the CORTES project, we are investigating how to manage scheduling when distributed across multiple schedulers [Sycara 90]. In particular, we are investigating how schedulers, which possess their own resources, coordinate their decisions when they require possessed by others. Due to the size of the sc problem, we distinguish between coordinatior strategic level versus the tactical level. Our assumes that each scheduler develops schedule Constrained Heuristic Search. At the strategic le coordination of large numbers of activities r resources outside of a particular scheduler is perfor communicating statistical summaries of aggregate textures. These demands are used to bias a sche reservations so that it does not require ar scheduler's resources during a period of high dema the strategic level, we expect that coordination pr will be reduced but not removed. It is the role tactical level to negotiate resource allocations that not be handled strategically.

The textures play four important roles in distri search: (1) they focus the attention of an agent to gl critical decision points in its local search space, (2) provide guidance in making a particular decision decision point, (3) they are good predictive measures c impact of local decisions on system goals, and (4) the used to model beliefs and intentions of other agents. development of the presented texture measures is the r of extensive experimentation in the single agent setting. have completed the implementation of a distributed tes and are currently performing experiments involv multiple agents. Experimental results from the distribu scheduling testbed are presented in [Sycara 90].

5. Planning

We are currently investigating the integration planning with scheduling[3]. In previous planners, planni has been an end unto itself. Any feasible plan is consider a success, with only very inflexible criteria for plan qualit such as minimizing the total number of actions. In th context of the CORTES project, the planner will b producing process plans to be used by the scheduler. Th quality of these plans is defined by the quality of th schedules that the scheduler can produce using them. Thus there is a strong need for a constraint language to use in communicating with the scheduler to determine what sorts of plans would be good.

Current state-of-the-art planners are constraint-directed, domain-independent, hierarchical, nonlinear, and support replanning [Wilkins 88]. We intend to include these capabilities, and extend them where appropriate.

[2] For a more complete description of these measures, the reader is referred to [Sadeh & Fox 88, Sadeh 90].

[3] See [Frederking & Chase 90] for more details.

Planners already exist that use constraints on planning variables to increase the power of their representation and to reduce arbitrary decisions that can lead to unnecessary backtracking. In addition to making wider use of constraints, we will make this planner be truly constraint-*directed* by developing measures of criticality for goal ordering and operator selection. This will provide a domain-independent representation for the domain-dependent heuristics that focus attention in the search for a plan.

The planner will always support planning at different levels of abstraction, and the re-use of plans in support of reactive planning.

6. Uncertainty Analyzer

Uncertainty is a fact of life in most job shop scheduling environments. Sources of uncertainty include: Demand change (seasonal, forecast error, cancel orders, expedition), Inventory Policy (raw material arrival pattern, safety stock policy) Machine failure, Change of time duration (transit, set-up, processing), Yield, and Quality (Tool wear, precision). Uncertainty increases as the planning horizon is extended, and its the amount and sources of uncertainty change over time.

The presence of uncertainty means that it is very unlikely that a detailed predictive schedule that assigns precise start and finishing times on resources for activities is going to be adhered to. This characteristic imposes two requirements on schedulers: (a) A scheduler should be able to represent and reason about degrees of uncertainty, and (b) a scheduler should be able to react to unexpected events on the factory floor. The inability of a scheduler to reason about uncertainty almost always results in a schedule being invalid at the time it is released to the production floor.

CORTES manages uncertainty in three stages. In the first stage, the Uncertainty Analysis module monitors and records the stochastic events. It develops over time a model of the sources and characteristics of uncertainty. Once a valid model is constructed, the Uncertainty Analysis module passes the information to the Scheduler. In the second stage, the scheduler uses the uncertainty models to reduce the precision of its schedules. Precision can be reduced by increasing the durations of activities, overlapping activity temporal intervals, or assigning activities to resource aggregates rather than to specific resources. In the third stage, the Dispatcher control module, is able to react more flexibly to stochastic events by taking advantage of the imprecision inserted in the schedule by the scheduler; it can start an activity earlier or later or assign an activity to another resource in an aggregate (i.e., work center). The Dispatcher's task is to dispatch jobs to machines and monitor machine and job execution status. The Dispatcher notes deviations from the schedule and resource unavailability and communicates this information to the Scheduler, Uncertainty Analyzer and factory floor.

Two approaches have typically been utilized to address the problem of temporal uncertainty. One approach is based on the idea of dividing the time horizon into time zones using progressively coarser time units to describe events in the future. For example, a time unit of one hour may be used to project a schedule over a one week horizon; a time unit of a day mey be used to project a schedule from a one-week to a one-month horizon and so on. Although this approach recognizes the fact that events that are further int the future are less accurately predictable, it has been criticized [Kerr 89] as suffering from the presence of discontinuous boundaries between time zones and the difficulty of handling orders whose processing crosses a zone boundary. A second approach to handling uncertainty is the use of probability distributions to describe schedule parameters. This approach has the disadvantage [Kerr 89] that probability is concerned with the combination and manipulation of independent random variables whereas many of the probabilistically described scheduling parameters are not independent (e.g., processing times of different jobs on a particular machine could depend on some characteristic of the machine)[4].

The CORTES uncertainty analyzer represents uncertainty in terms of fuzzy logic [Zadeh 85, Kaufmann 85, Prade 79]. The present version [Chiang & Fox 90] focuses on uncertainty concerning machine failures. The mean time between failure and mean duration of the failure are assumed known. It is also assumed that once a machine is fixed after a failure, processing resumes at the point of interruption with no rework necessary. In other words, machine failure causes a variation in processing time only and not in scheduling order. The time between machine failures and the failure duration are used to express uncertainty in processing time. Instead of being random variables of known distribution, the duration of failure and time between failures may be only approximately known. This approximate information on the procesing time bounds is expressed in terms of fuzzy numbers of *Type-1*, where a real number that is approximately known is expressed as a confidence interval of upper and lower bounds. Fuzzy bound values may be ther result of subjectively known processing characteristics

[4]Handling variable dependence through the use of conditional or joint probability distributions poses severe estimation problems.

described by a shop operator, or of known distributions described by shop statistics. An extension of type-1 fuzzy representation of uncertainty in operation duration is *Type-2* representation where the lower and upper bounds of a confidence interval, instead of being ordinary numbers are fuzzy numbers that themselves have intervals of confidence.

Uncertainty bounds work in a similar manner as earliest start time/latest start time and earliest finish/latest finish time. The bounds can be viewed as slack to protect against uncertainty. The mean processing time is reserved for the operation and the slack time is reserved for protection against uncertainty. Once an operation is ready for processing, a dispatcher should follow the schedule within the prescribed bounds. We ran experiments [Chiang & Fox 90] to compare various cost measures, such as tardiness, work-in-process, and delayed orders, under various processing duration representation schemes (type-2 fuzzy representation, fixed processing time given in the process plan, mean processing time considering machine failure duration and time between failures) and under different cost structures and shop loads. The general result is that type-2 bounds give sufficient protection against uncertainty in processing time with less investment in the planned cost (planned cost= planned tardiness+planned work-in-process+ planned lateness)[5]. For a more detailed description of the experiments, see [Chiang & Fox 90].

7. Conclusion

In this paper, we have given an overview of the CORTES integrated framework for production planning, scheduling and control (PSC) system. CORTES's approach to PSC problems departs from others in the hypotheses it explores:

1. Generality Hypothesis: There exists a single approach that can optimize decision making across a wide variety of PSC problems.
2. Flexibility Hypothesis: The same approach can be used for both planning, predictive scheduling and reactive control.
3. Uncertainty Hypothesis: In order to provide the appropriate level of precision in PSC, reasoning about uncertainty must be an integral part of the PSC approach.
4. Scale Hypothesis: Large PSC problems, that contain thousands of activities, resources and constraints, must be solved in a qualitatively different manner than small PSC problems.

The CORTES project is investigating all fo assumptions in parallel. We have experimental data acro a variety of PSC problems that support the generali assumption. The flexibility assumption is currently beir tested by our integration of PSC functions. The uncertain assumption is supported by the ease with which we hav adapted CHS to account for uncertainty. The Scal assumption remains to be tested.

[5] To protect against uncertainty the planned operation duration is longer, more planned work-in-process exists and orders are planned to arrive late. Hence, the more protection we design into the bounds, the higher the planned cost.

References

[Chiang & Fox 90] Chiang, W-Y., and Fox, M.S. Protection Against Uncertainty In a Deterministic Schedule. In *Proceedings of the Fourth International Conference on Expert Systems in Production and Operations Management*, May, 1990. Submitted for publication.

[Erman et al. 80] Erman, L.D., Hayes-Roth, F., Lesser, V.R., and Reddy, D.R. The Hearsay-II Speech Understanding System: Integrating Knowledge to Resolve Uncertainty. *ACM Computing Surveys*12(2):213-253, 1980.

[Fox 83] M. Fox. *Constraint-Directed Search: A Case Study of Job-Shop Scheduling.* PhD thesis, Department of Computer Science, Carnegie-Mellon University, 1983.

[Fox 88] Fox, M., and Sycara. K. *Knowledge-Based Logistics Planning and its Application in Manufacturing and Strategic Planning.* Technical Report First Interim Report, submitted to RADC, CMU Robotics Institute, December, 1988.

[Fox 89] Mark S. Fox, Norman Sadeh, and Can Baykan. Constrained Heuristic Search. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, Pages 309-315. 1989.

[Fox 90] Fox, M.S.
Constraint Guided Scheduling: A Short History of Scheduling Research at CMU.
Computers and Industry, 1990.
To Appear.

[Fox & Smith 84] Fox, M.S., and Smith, S.
ISIS: A Knoweldge-Based System for Factory Scheduling.
*International Journal of Expert Systems*1(1):25-49, 1984.

[Frederking & Chase 90]
Frederking, R.E., and Chase, L.L.
Planning in a CIM Environment: Research Towards a Constraint-Directed Planner.
In *Proceedings of the Fourth International Conference on Expert Systems in Production and Operations Management*, May, 1990.
Submitted for publication.

[Freuder 82] Freuder, E.C.
A Sufficient Condition for Backtrack-free Search.
*Journal of the ACM*29(1):24-32, 1982.

[Haralick & Elliott 80]
Haralick, R.M., and Elliott, G.L.
Increasing Tree Search Efficiency for Constraint Satisfaction Problems.
*Artificial Intelligence*14(3):263-313, 1980.

[Kaufmann 85] Kaufmann, A. and Gupta, M.
Introduction to Fuzzy Arithmetic: Theory and Applications.
Van Nostrand Reinhold, New York, N.Y., 1985.

[Kerr 89] Kerr, R.M., and Walker, R.N.
A Job Shop Scheduling System Based on Fuzzy Arithmetic.
In *Proceedings of the 2nd Interational Conference on Expert systems and Leading Edge in Production and Operations Management*, Pages 433-450. Hilton Head Island, S.C., May, 1989.

[Mackworth 87] Mackworth, A.K.
Constraint Satisfaction,
In Shapiro, S., *Encyclopedia of Artificial Intelligence*. J. Wiley & Son, 1987.

[Nadel 86] Nadel, B.A.
The General Consistent Labeling (o Constraint Satisfaction) Proble
Technical Report DCS-TR-170, Department of Computer Scienc Laboratory for Computer Resea Rutgers University, New Brunsv NJ 08903, 1986.

[Newell & Simon 76]
Newell, A., and Simon, H.A.
Computer Sciences as Empirical Inquiry: Symbols and Search.
*Communications of the ACM*19(3):113-126, 1976.

[Prade 79] Prade, H.
Using fuzzy set theory in a schedulir problem.
*Fuzzy Sets and Systems*2(2):153-165 1979.

[Sadeh 90] Norman Sadeh, and Mark Fox.
Focusing Attention in an Activity-ba Job Shop Scheduler.
In *Proceedings of the Forth International Conference on Exp Systems in Production and Operations Management*, 1990.

[Sadeh & Fox 88] Sadeh, N., and Fox, M.S.
Preference Propagation in Temporal Constraints Graphs.
Technical Report , Intelligent System Laboratory ,The Robotics Institut Carnegie Mellon University, Pittsburgh, PA 15213, 1988.
CMU-RI-TR-89-2.

[Sathi et al. 85] Sathi, A., Fox, M.S., and Greenberg,
Representation of Activity Knowledg for Project Management.
*IEEE Transactions on Pattern Analys and Machine Intelligence*PAMI-7(5): 531-552, September, 1985.

[Sathi et al. 90] Sathi, N., Fox, M.S., Goyal, R., and Kott, A.
CORAL: An Order Configuration and Resource Allocation System.
Technical Report , Carnegie Group Inc Five PPG Place, Pittsburgh PA 15219,
1990.
In preparation.

[Smith et al. 86] Smith, S., Fox, M.S., and Ow, P.S.
Constructing and Maintaining Detailed Production Plans: Investigations into the Development of Knowledge-Based Factory Scheduling Systems.
*AI Magazine*7(4):45-61, Fall, 1986.

[Sycara 90] Katia Sycara, Steve Roth, Norman Sadeh, Mark Fox.
Managing Resource Allocation in Multi-Agent Time-Constrained Domains.
In *Proceedings of the 1990 Darpa Workshop on Innovative Approaches to Planning, Scheduling and Control*, 1990.

[Wilkins 88] Wilkins, D.E.
Practical Planning.
Morgan Kaufmann Publishers Inc., 1988.

[Zadeh 85] Zadeh, L.
Fuzzy Sets.
*Information and Control*8(338), 1985.