HYBRID TEMPORAL AND NON-TEMPORAL KNOWLEDGE FOR SCHEDULING

Stephan Becker
Technical University of Hamburg-Harburg
Department of Process Automation Techniques
D-21071 Hamburg
Germany
becker@tu-harburg.d400.de

Abstract

Scheduling is an essential task within production planning. It is strongly determined by the notion of constraints. One important class of them are temporal constraints. They might be quantitative, as due dates, or qualitative, as the logical relationship of orders. The requirements implied by scheduling are matched to the features of well-known formalisms for temporal reasoning. Other relevant constraints are originally non-temporal, but directly influence the temporal arrangement of orders. The use of a hybrid formalism to treat both qualitative and quantitative, both temporal and non-temporal data is strongly desirable. This article provides a detailed investigation on the interdependency between qualitative and quantitative temporal knowledge. The integration of temporal knowledge into a constraint-system beyond mere temporal reasoning is outlined. The result is TIC II, a hybrid formalism which is capable to deal with both quantitative and qualitative temporal and non-temporal constraints. Short scenarios which typically are part of scheduling tasks are used to illustrate the representation of hybrid constraints. Although TIC II is domain-independent, it specifically supports scheduling tasks.

1. Introduction

An important part of any production management system is scheduling. It determines when and by which resources (machines, tools, material, personnel, etc.) orders should be accomplished on the factory floor. Today's market demands for a large variety of products, an increasing number of orders of decreasing lot sizes is the consequence. The synchronization of competing orders is essential for modern production. Data uncertainty and manufacturing complexity are two opposing forces, which heavily determine a factory's organization and, in specific, the scheduling process (Meyer 90). Several sources of data uncertainty can be identified:

- unreliable information: "Sub-assembly is expected to be finished at 11 a.m."
- incomplete information: "Final assembly takes 3 hours. Its start depends on the finish of sub-assembly."
- dynamically changing information: "Machine M has broken down."
- contradictory information: "Sub-assembly needs machine M, but M is broken down."
- imprecise information: "Sub-assembly will be finished after machine M is repaired. The repair of M will take 1 - 2 hours."

Starting-point of scheduling is the decision frame (Meyer 90), which states all demands on the schedule and thus covers all admissible schedule. The task of scheduling is to refine the decision frame in order to identify one or more extensions (= schedules). As scheduling takes place in a complex and rapidly changing environment, it is an ongoing, incremental process highly depending on the latest data in order to react on unexpected events, which might disturb the current schedule. (Becker, Parr 94) characterize the scheduling process by 3 basic steps:

- Structuring: A process model represents the orders to be produced. Each order is decomposed into elementary tasks. Models of the products' design and the resources' layout enforce the tasks' logical relationship and substantially constrain the number of admissible schedules. The central matter of structuring is to expose the intrinsic constraints.

- Dispatching: It determines precisely when and where the tasks will be executed. Available quantitative information is exploited here. The main objective is to approximate an optimal solution, which allows the foreman to organize the next shift.
- Execution: The schedule is often affected by unexpected events like machine break-downs, delays, urgent customer orders and others. They need to be integrated into the schedule. At worst some of the initial constraints have to be re drawn.

An adequate representation of time is essential for successful scheduling. At least it should enable statements being
- qualitative: "job A has to preceed job B",
- quantitative: "job A has to start at 10.00 a.m.",
- incomplete: "job A's duration is 2 h, its start and end point depend on jobs B and C",
- indefinite: "job A precedes or follows job B", "job A takes between 1 and 2 hours".

2. Constraints on Scheduling

2.1 Qualitative Temporal Constraints

(Allen 83) introduces a formalism called interval-algebra (IA) on the treatment of qualitative temporal knowledge. In short, IA offers 13 temporal primitives (fig. 2.1) to arrange time intervals. E.g. "A happens during B". The indefinitivity of two intervals' relation is expressed by a disjunction on the primitives, e.g. "C overlaps or meets D".

The nature of time implies a further constraint on each 3 intervals, for example: "If A precedes B and B precedes C, then also A precedes C." The complete knowledge is compiled by a table (known as transitivity-table) of 13×13 = 169 rules (Allen 83). It is referred to as interval-logic constraint (ILC) later. Local consistency of each 3 intervals is achieved by application of a propagation algorithm.

While the complexity of the decision on local consistency is $O(n^3)$, where n is the number of intervals, the test on full consistency is NP-complete (Villain et al. 89). Therefore various formalisms based on time points have been developed. Instead of representing the temporal extent of an activity by an interval, time-point logic models only start- and end-point. Thus the relationship of two periods is expressed by 4 relations. Possible primitive relations between time-points are <, = and >. Disjunctive relations cover all 2^3-1 = 8 combinations in case of the full time-point algebra (PA) (Vilain et al. 89). The exclusion of ≠ produces its continuous subset (CPA) (van Beek, Cohen 90). A transitivity-table is used to constrain the relations between each three time-points. Both PA and CPA show polynomial behaviour: $O(n^4)$ and $O(n^3)$, respectively. (Meiri 91) provides a detailed investigation on their complexity.

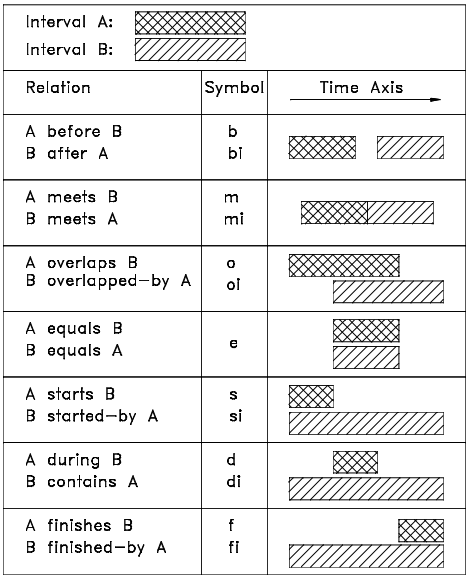

Fig. 2.1: Allen's temporal primitives

Unfortunately, point-based logic lacks of expressivity: While IA knows The number of possible consistent disjunctive relations between two periods is 2^{13}-1 = 8191 in case of IA. It reduces to 187 within PA and to only 82 in case of CPA. Assume orders A and B to be exclusive. Then A either follows or precedes B. This constraint cannot be stated by a point-based logic (van Beek, Cohen 90). As this is an important constraint, point-based logic does not meet the requirements on scheduling. In fact, it is not surprising that a sufficient tool to solve scheduling-tasks is NP-complete: Scheduling itself is (Garey, Johnson 79)! A promising approach is followed at the Technical University of Hamburg-Harburg to speed-up IA by hardware-support (Jakob 92). Recent results are published in (Jakob 94a) and (Jakob 94b). A

second approach is based on the problem-structure and the use of heuristic search-strategies (Becker, Heuer 95).

2.2 Quantitative Temporal Constraints

Schedules show precise chronological information, namely the orders' start, finish and duration. Initially often an admissible period can be stated. Dispatching decides on suitable and definite time points.

(Rit 86) defines the set of possible occurrences (SOPO). A SOPO holds all options for mapping a certain activity onto the time-axis. Minimal (-) and maximal (+) values delimit intervals of admissible values for start (S), finish (F) and duration (D): S = [S-, S+], F = [F-, F+], D = [D-, D+]. Off course, these 6 data constrain each other by the interval-algebraic (Moore 66) equation (1): S + D = F. Equation (1) is useful to compute missing chronological data. To illustrate the interdependency (Rit 86) maps the SOPO into a system of co-ordinates. S is mapped onto the abscissa and F onto the ordinate. A temporally definite activity (S- = S+, F- = F+) is a point on the shadowed plane P, which covers all temporal events (fig. 2.2). In general, a SOPO is a set of points, a (possibly degenerated) hexagon.

The qualitative relation between two activities implies a further constraint on their respective SOPOs (Rit 86). Assume activities A and B to be ordered by r_{AB}, which can be any of Allen's 13 primitive temporal relations. Let A's chronological data be definite: A's SOPO is a point. Depending on r_{AB}'s value, B's SOPO can now be restricted to a specific sector of P. Such sectors exist for all of Allen's primitives. They are called certain areas (fig. 2.3). This concept is generalized to chronologically indefinite activities (with hexagonal SOPOs). Fig. 2.4 shows the certain area with respect to "during" of an exemplary SOPO. Examples with respect to all 13 primitives are provided by (Becker 88). The information on certain areas is used to constrain SOPOs by a related propagation algorithm (Rit 86).

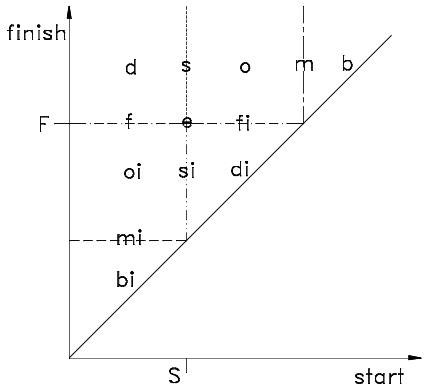

Fig. 2.3: The certain areas of the definite SOPO marked by point (S, F). The areas are sectors (for d, o, b, oi, di, bi), lines (for s, m, si, mi) or a point (for e).

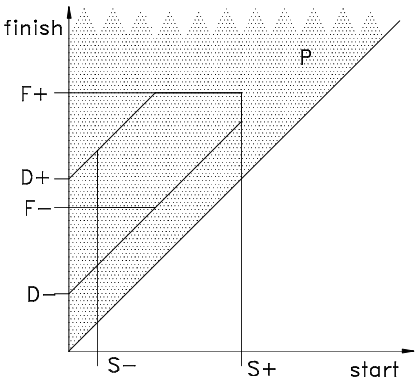

Fig. 2.2: A SOPO states minimal (-) and maximal (+) values for start (S), finish (F) and duration (D). Because an activity's start must be ≤ its finish, chronological data is restricted to the shaded plane P.

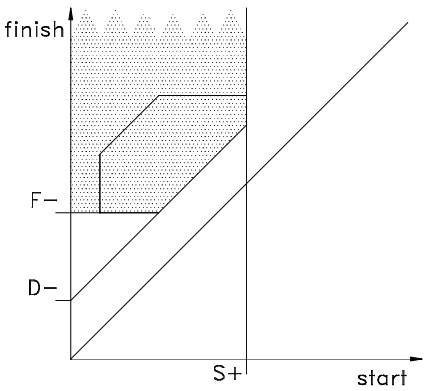

Fig. 2.4: The SOPO of an activity A. If A happens during a 2nd activity B, then B's SOPO is constrained to the certain area of A (shaded).

2.3. Combined Temporal Constraints

The recognition that scheduling is determined by both qualitative and quantitative constraints motivated the development of several hybrid formalisms (Huber, Becker 1988; Kautz, Ladkin 1991; Meiri 1991; Winklhofer 1994). Meiri's (Meiri 92) framework of general temporal networks (GTN) is based on the notion of relations between time-points and -intervals:

- quantitative distance between two points (q),
- qualitative point-to-point relations (pp),
- qualitative point-to-interval relations (pi) and vice versa (ip) and
- qualitative interval-to-interval relations (ii).

q-relations are defined by sets of closed intervals. The other relations are disjunctive sets of temporal primitives. In specific, pp-relations stem from (Vilain et al. 89) and ii-relations from (Allen 83). A propagation algorithm constrains triples of qualitative relations by the respective transitivity table. Intersection, union and composition of sets of intervals constraint triples of q-relations. The qualitative and the quantitative fragment of GTN are linked by the constraints q-relations imply onto pp-relations and vice versa.

Formalisms based on Rit's SOPOs are Zeitkern[1] (Winklhofer 94) and TIC[2] (Huber, Becker 88). SOPOs hold start, end and duration of an activity as a single entity, a triple of intervals. Zeitkern represents a concave period as a set of such triples and thereby keeps for each fragment the coherence of start, finish and duration. This information is lost within GTN: Admissible intervals for start, finish and duration of the fragments are separated into three sets of intervals. In so far, Zeitkern extends GTN.

Despite their rich expressiveness, these approaches do not support all temporal constraints (Winklhofer 1994):

- Assume orders A, B, C and D. The only admissible sequences are ABCD and DCBA, all other permutations are prohibited by the product's layout. None of the above formalisms is able to express this constraint.

The only constraint on the admissibility of tupels of relations is defined by the various transitivity tables and is applied implicitly within the propagation algorithm. No means exists to define further dependencies within tupels of relations.

[1]Zeitkern ≈ time-kernel, in English

[2]TIC = Temporal Inference Component

2.4 Non-Temporal Constraints

A case-specific analysis of an industrial scheduling task detected and classified a variety of relevant constraints (Fox 83). Although many of them are not originally temporal, they all influence the design of admissible schedules. Not always they can be circumscribed in terms of temporal relations and constraints. An important example is the limitation of resource capacity. Assume an exclusive resource. It has a capacity of 1. Two orders competing for this resource can easily be sequenced by constraining their qualitative relation to (b bi). Such a simple solution is not available if the resource's capacity is limited to $n > 1$.

- Assume 2 equivalent machines, i.e. a resource of capacity 2. 3 orders compete for the resource. None of the mere temporal formalisms is able to allow the in-parallel production of any 2 orders, but to prevent full parallelity.

It is not possible to constrain the qualitative relation of orders in dependence of their resource allocation. The main reason is that the non-temporal concept "resource allocation" is not modelled.

As a result, the solution of scheduling tasks require a more powerful approach than mere temporal reasoning. The theory of general, application independent constraint satisfaction problems (CSP) promises an appropriate basis. The basic steps of scheduling, i.e. structuring, dispatching and monitoring, are matched very well by the constraint-net paradigm (Becker, Parr 94).

3. Temporal Reasoning within CSP

The formalism presented below views at temporal reasoning as a specific CSP. It allows to describe both temporal and non-temporal problems within one homogeneous constraint net which is solved by one propagation-algorithm. Specifically, the formalism unifies Allen's temporal logic and Rit's SOPOs. This

requires the translation of Allen's formalism into the general CSP-view of variables and constraints. The constraints implicit to both Allen's and Rit's specific propagation algorithms need to be made explicit in order to allow the algorithm to be independent of the area of application. A number of direct constraints which link and refine the quantitative and the qualitative fragment is identified. The capabilities of the resulting formalism go far beyond those of mere temporal reasoning. The formalism is implemented within the shell TIC II.

3.2 Qualitative Temporal Knowledge

Representation and solution of CSPs are based on graph-notation. The usual view is to decide on labels for indefinite variables, which are represented as nodes. A variable's domain puts a first, unary constraint on its instantiation. Further constraints describe restrictions implied by other variables. These constraints are explicitly represented by arcs. Labels conflicting with constraints are retracted. Their removal may lead to the removal of further labels. This process is called propagation. It terminates, when no more changes occur or if inconsistency is detected. In case of success the result is a n-tuple of sets. Each set holds all admissible labels for one of the constraint-net's n variables. A result for the initial problem is a n-tuple of singletons from these sets. Even in case of successful propagation such a n-tuple may not exist, because propagation algorithms do only check local consistency. In that case, the net is still not satisfiable. Binary constraint-satisfaction is described by e.g. (Mackworth 77; Mackworth, Freuder 85; Davis 87; Dechter 92).

The constraint-net mechanisms employed by Allen or Rit (and others, as van Beek or Meiri) deviate from this view: Temporal relations are arcs, although they are the decision variables. These arcs are labelled by temporal primitives. The graph's nodes symbolize temporal intervals, which represent the occurrence of activities. Propagation establishes the local consistency of each three labelled arcs.

Two distinct types of temporal constraints can be identified at this point: At first, there are the constraints on the relationship of pairs of temporal intervals. They are defined by the initial labelling of the arcs, i.e. the temporal relations. These constraints are formulated to reflect preferences or restrictions with regard to the specific circumstances of a specific scheduling problem. This type of constraints is represented declaratively and is integrated into the problem's description, the graph. At second, there is another, application independent constraint. It is represented by the transitivity table, which excludes certain combinations of temporal relations due to the physical nature of time. In case of the original representation of Allen's temporal logic, this second constraint is integrated into the propagation algorithm and thereby is represented procedurally.

In order to combine qualitative and quantitative temporal logic (independent of the specific approach) and nets of arbitrary (non-temporal) constraints homogeneously, a uniform representation must be used. Although Allen's and Rit's formalisms had been successfully combined within TIC (Huber, Becker 88) already, their heterogeneity prevented a uniform view and the representation of other than temporal constraints. This recognition lead to the development of TIC II. There, Allen's temporal logic has been conversed: The temporal relations are viewed as decision variables and therefore are represented as nodes. The physical time-constraint is made declarative and handled as a ternary (hyper-) arc. (Becker 92) informs about details of the conversion, examples are to be found in chapter 4.1. The next sections describe the combination of qualitative and quantitative temporal knowledge and its integration into general CSP.

3.3 Quantitative Temporal Knowledge

A SOPO consists of 6 components, stating a minimal (-) and a maximal (+) value for start (S), duration (D) and finish (F). E.g., A.F+ denotes activity A's latest finish. The influence on A's SOPO implied by a second activity B and its relation to A is computed via the concept of certain areas:

Example 3.1:

A before B ⇒ A.F+ < B.S+ and B.S- > A.F-.

The intervals S, D and F of both A and B now have to be narrowed until they satisfy the consequence. Thereby A's latest finish can be decremented and B's earliest start can be incremented (fig. 3.1).

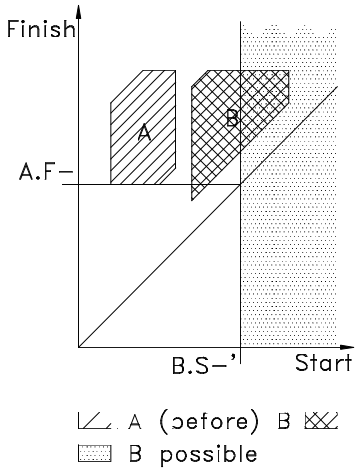

Fig. 3.1: The SOPOs of activities A and B. The assertion "A happens before B" constrains B's SOPO to the shaded area: B cannot start before A's earliest finish. B's earliest start will be updated to the new value B.S-'.

A r B	Constraints on SOPO(B) B.S-	B.S+	B.F-	B.F+	B.D-	B.D+
b	> A.F-		*>a.f-*			
bi		*<a.s+*		< A.S+		
m	≥ A.F-	≤ A.F+	*>a.f-*			
mi		*<a.s+*	≥ A.S-	≤ A.S+		
o	> A.S-	< A.F+	> A.F-			
oi		< A.S+	> A.S-	< A.F+		
s	≥ A.S-	≤ A.S+	> A.F-		> A.D-	
si	≥ A.S-	≤ A.S+	*>a.s-*	< A.F+		< A.D+
d		< A.S+	> A.F-		> A.D-	
di	> A.S-	*<a.f+*	*>a.s-*	< A.F+		< A.D+
f		< A.S+	≥ A.F-	≤ A.F+	> A.D-	
fi	> A.S-	*<a.f+*	≥ A.F-	≤ A.F+		< A.D+
e	≥ A.S-	≤ A.S+	≥ A.F-	≤ A.F+	≥ A.D-	≤ A.D+

Tab 3.1: Constraints on the SOPO of an activity B implied by the qualitative temporal relation to another activity A.

Assume a situation A r_{AB} B. Then tab. 3.3.1 compiles the constraints implied by the single primitive relations on B's SOPO. Further, weaker constraints than those implied by the certain areas can be concluded by use of equation (1). For example: If B.S- > A.F- is valid, then also B.F- > A.F-. In tab. 3.1, they are printed in italics. The total constraint on B's SOPO is the total row dedicated to r_{AB}. An empty entry indicates the absence of constraints on the specific SOPO-component: It can be related to any of A's SOPO-components by the trivial constraint, i.e. the disjunction of >, = and <. The constraint on A's SOPO is achieved by inversion of r_{AB}: A r_{AB} B ⇔ B r_{BA} A.

If A and B are related by a non-primitive relation, each SOPO component may be affected by a disjunction of several constraints resulting from the different primitives. The result for the specific component is the weakest, common constraint. Now the italic entries of tab. 3.1 become important:

Example 3.2:

A meets or overlaps B ⇒

A: A.F- ≥ B.S-, A.F+ < B.F+, A.S+ < B.S+ (2).

B: B.S- > A.S-, B.S+ ≤ A.F+, B.F- > A.F- (3).

Satisfaction of these constraints might require to increase A's earliest finish, B's earliest start and B's earliest finish and to decrease A's latest finish, B's latest start and A's latest start. The existence of constraints (2) and (3) results from equation (1).

The constraints implied by the single primitive relations combine to the trivial constraint only in those cases, where a non-primitive relation contains both "before" and "after". 2^{11} = 2048 out of 2^{13} -1 = 8191 such relations exist. Therefore 6143 consistent relations imply non-trivial constraints.

The concept of certain areas allows also to deduce information about the qualitative temporal relation of two activities A and B. If a primitive relation r_{AB} is assumed to relate A with B, and B's SOPO is completely outside of A's certain area with regard to r_{AB}, then r_{AB} cannot be valid. Table 3.2 states conditions for the incompatibility of primitive relations with SOPOs.

Example 3.3:

A.F- > B.S+ or A.F+ < B.S+ ⇒ not (A meets B).

A cannot meet B, if A's earliest finish is later than B's latest start or if A's latest finish is earlier than B's earliest start. In other words: If A finishes before B can starts or if A still goes on, when B has to start, A cannot meet B.

IF at least one of these conditions is satisfied						THEN NOT (A r B)
A.S-	A.S+	A.F-	A.F+	A.D-	A.D+	
		≥ B.S+				b
	≤ B.F-					bi
		> B.S+	< B.S-			m
> B.F+	< B.F-					mi
≥ B.S-		≥ B.S+	≤ B.S-			o
≥ B.F+	≤ B.S-		≤ B.F-			oi
> B.S+	< B.S-	≥B.F+		≥ B.D+		s
> B.S+	< B.S-		≤ B.F-		≤ B.D-	si
	≤ B.S-	≥ B.F+		≥ B.D+		d
≥ B.+			≤ B.F-		≤ B.D-	di
	≤ B.S-	> B.F+	< B.F-	≥ B.D+		f
≥ B.S+		> B.F+	< B.F-		≤ B.D-	fi
> B.S+	< B.S-	> B.F+	< B.F-	> B.D+	< B.D-	e

Tab. 3.2: Constraints on the qualitative temporal relation of two activities implied by their SOPOs.

4. TIC II: A Hybrid Constraint-Based Shell

TIC II[3] implements temporal reasoning within nets of arbitrary constraints, which are defined in terms of the interface language NICOLE[4]. NICOLE hides the required transformation of temporal knowledge into CSP-notation. TIC II's generic parser enables the user to define further interface languages to connect TIC II with other modules or to support other application specific constraints. While the user thinks in categories as orders, allocations, resources or logical relationships of orders, the internal representation is more abstract. It is completely formulated in terms of variables and constraint. In this sense, NICOLE bridges the gap between a user-oriented view of a scheduling-problem and its graph-theoretical counterpart. The reminder of this chapter illustrates this transformation by small examples.

[3]TIC II = Temporal Inferences and Constraint Nets

[4]NICOLE = Network Initiation and Construction Language

4.1 Variables

Within TIC II, a node represents a variable, whose domain is restricted by unary constraints. Imprecise knowledge corresponds to a wide domain instead of a single value. TIC II supports two subclasses of variables: atoms and aggregates.

- Atom-variables are straight forward. Two types of domains are possible:
  1. closed numeric intervals: Ignorance about the borders is symbolized by "∗". For instance, the assertion that a task may finish from now on, but latest at time-point 100, is represented by (∗ 100).
  2. finite sets of numbers or symbols: The sets hold the admissible labels for the variables The relation of two exclusive orders is constrained by {before, after}.
- Aggregate-variables facilitate to build complex structures of several components. Both an aggregate as a whole and its single components may be constrained in relation to other variables. For instance, it is possible to delimit the value of each component and additionally to delimit their sum. An aggregate has not an own value, but manages several components, which variables for their own. One example is the SOPO (fig. 4.3.6). A second purpose of aggregates is to link intentionally related concepts, though they might be not directly constrained by each other, e.g. the ORDER (fig. 4.4.1).

4.2 Constraints

A constraint explicates the local interdependency of n variables. It is defined by a set of n-tuples or in terms of functions:

- Tuple-Constraints: Each set of n-tuples explicitly lists the admissible combinations of labels for n mutually dependent variables. The interval-logic constraint is a set of 169 triples. Each one specifies one admissible labelling for 3 temporal relations between 3 activities.
- Function-Constraints: Following (Hyvönen 89), the admissible values for n mutually dependent variables are computed by (n-1)-ary solution function. In case of the variables' domains being intervals, the basic operations follow Moore's interval arithmetic extended by the use of ∗. Examples are equation (1), the

dependency between qualitative and quantitative temporal data or any metric relation between the SOPOs of intervals.
- Condition-Constraints: The conditional interdependence of n variables $V_1,...,V_n$ is represented by up to n templates
  "if <condition> then V_i := <domain> else V_i := <domain>", $1 \le i \le n$,
  where <condition> denotes a predicate on constants or up to n variables, and <domain> denotes the supported labels of V_i by reference to a set or an interval, a further variable or to a a function.

4.3 Representation of Temporal Knowledge

The temporal knowledge presented so far can be modelled easily within TIC II:
- An activity's SOPO is an aggregate-variable consisting of 3 interval-variables (fig. 4.1).
- The interdependency of the 6 SOPO data is realized by a function-constraint (SOPO-Constraint, SC) on the aggregate (fig. 4.1).
- Qualitative temporal relations are represented by tuple-variables, which are labelled by the admissible subsets of primitives (fig. 4.2).
- The interval-logic constraint (= Allen's 169 rules) is a ternary tuple-constraint (fig. 4.2).
- Each two SOPOs and their temporal relation are constrained by a function-constraint (Combined temporal Constraint, CC), which realizes the complete version of tables 3.1 and 3.2. Each of the tree variables' domain is updated by a solution function on the remaining two variables (fig. 4.3).
- Metric constraints describe relations among the SOPO data of different activities. This is represented by function-constraints employing interval-arithmetic operations (fig. 4.4).
- Arbitrary tuple-constraints restrict the admissibility of the labellings of arbitrary tuples of relations (fig. 4.5).

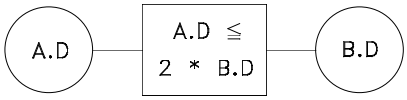

Fig. 4.4: An interval-arithmetic constraint assures A's duration being maximally twice as long as B's.

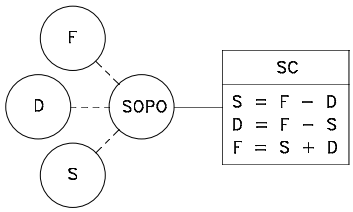

Fig. 4.1: The pattern of a SOPO-constraint (SC). The atoms holding start (S), duration (D) and finish (F) are linked to the aggregate (the SOPO) by component-arcs. SC is linked to the SOPO by a constraint-arc.

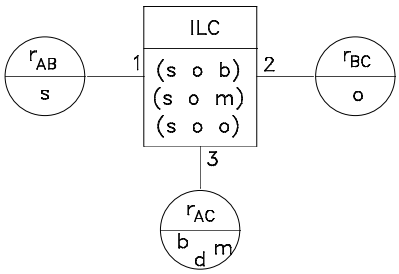

Fig. 4.2: An example for 3 intervals A, B and C being arranged by 3 set-variables r_{ij}. Their labels are options for the temporal relation I r_{ij} J. ILC constrainsr the admissible values for the r_{ij}. In this case, ILC supports 3 combinations. Propagation will remove label d for r_{AC}.

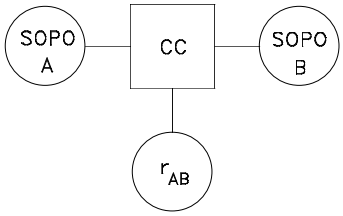

Fig. 4.3: The constraint implied by combined qualitative and quantitative temporal knowledge (CC) links the SOPOs of A and B with their relation r_{AB}. CC is a functional constraint. It implements tables 3.1 and 3.2.

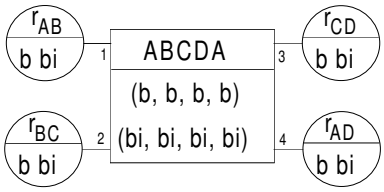

Fig. 4.5: Orders A, B, C and D are sequenced to either ABCD or DCBA by constraint ABCD on their relations. ABCD names the admissible labelings for each relation.

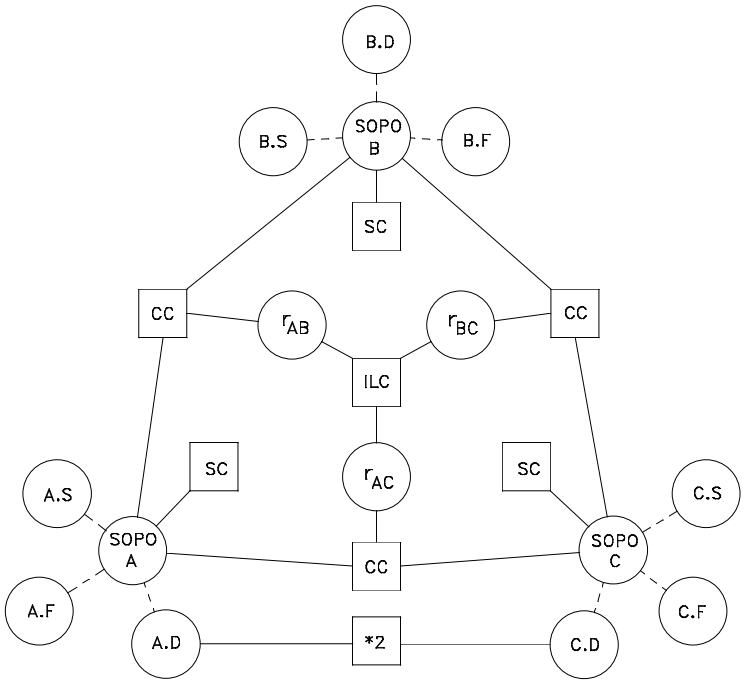

Fig. 4.6: Various temporal constraints on 3 orders A, B and C. The logical relationship of each two orders i and j is constrained by a qualitative temporal relation r_{ij}. ILC implements the transitivity-table on triples of relations. Order i's chronological data is constrained by SOPO i, which aggregates i's start (i.S), duration (i.D) and finish (i.F). Equation (1) is assured by the SOPO constraint (SC). Relation r_{ij} and the SOPOs of i and j are constrained by the combined temporal constraint CC. C's duration is constrained to be twice as long as A's (*2).

```
(NICOLE
   (NET  (NAME    "fig 4.3.6"))
   (STRUCTURE
      (CLUSTER
        (NAME "B-A-C")
        (SOPO  (NAME       "A")
               (DURATION (10 *)))
        (SOPO  (NAME       "B")
               (START      (0 *))
               (FINISH     (* 100)))
        (SOPO  (NAME       "C")
               (DURATION (40 50))
        (RELATION (FROM       "A")
                  (TO         "B")
                  (RELATION s d f))
        (RELATION (FROM       "C")
                  (TO         "B")
                  (RELATION s d f))
        (RELATION
           (FROM       "C")
           (TO         "A")
           (RELATION b bi m mi)))
      (GENERAL
        (FUNC-CONS
          (NAME   "*2")
          (ON     "C.D")
          (FUNC   mult A.D 2)))))
```

Fig. 4.7: A NICOLE description corresponding to fig. 4.3.6. Intervals A and C are defined to happen exclusive to each other. Both A and C are covered by B. Durations are defined for A and C, where A takes half the time of C. B is limited by an earliest start and a latest finish.

The interval-logic-constraint, the combined-constraint and the SOPO-constraints reside in the nature of time. They are inserted into a constraint net by NICOLE automatically. All other constraints must be defined by the user.

Fig. 4.6 shows the constraints on an arrangement of 3 chronologically linked events. A corresponding NICOLE description is given in fig. 4.7.

4.4 Representation of Hybrid Knowledge

- An order is an aggregate, which consists of a SOPO and a resource allocation. An order combines temporal and non-temporal concepts (fig. 4.8).
- The allocation of an order to a resource is hold by a set-variable, whose labels symbolize all admissible resources. The capacity of a resource determines the number of different labels it is represented by. In this sense, a resource of capacity n has n exclusive slots for the execution of tasks. Once an order is allocated to a definite resource (slot), all other labels are removed (fig. 4.9).
- An order's duration may depend on the resource it is dispatched to. This is expressed by a condition-constraint (fig. 4.9).
- A number of orders depending on the same resource of limited capacity imply a constraint. It is represented by a condition-constraint among the orders' SOPOs, their allocations and relations. It forces the relation

of two orders to exclusivity, in case they are dispatched to the same resource slot. Fig. 4.10 shows an example. Chronological data is suppressed. The corresponding NICOLE fragment is given by fig. 4.11.

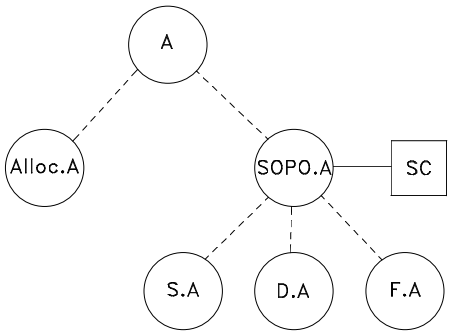

Fig. 4.8: Order A is an aggregate of a SOPO and the order's Allocation. No direct constraints exist between A's components, A just combines intentionally related concepts. The component SOPO.A is an aggregate again, while Alloc.A is an atom.

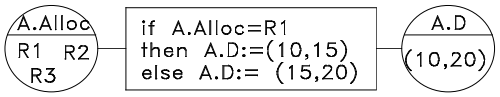

Fig. 4.9: Order A's resource allocation (A.Alloc) and its duration (A.D) are variables. A can be produced optionally on resources R1, R2 or R3. Production on R1 takes less time than on R2 or R3.

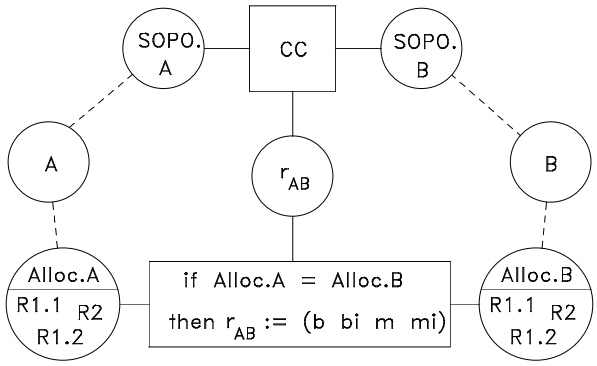

Fig. 4.10: Both resources R1 (2 slots) and R2 (exclusive) can produce A and B. Allocation to the same resource forces orders A and B to happen sequentially: Their relation is conditionally constrained to exclusivity and their SOPOs are adjusted via CC. Allocation to different resources does not trigger the conditional constraint.

```
(NICOLE
   (NET  (NAME "fig 4.10"))
   (STRUCTURE
      (CLUSTER
         (NAME A-B)
         (SOPO  (NAME A)
                <SOPO-data>)
         (SOPO  (NAME B)
                <SOPO-data>))
      (RESOURCE (NAME R1)
                (CAPACITY 2)
                (CLAIMS   A   B))
      (RESOURCE (NAME R2)
                (CAPACITY 1)
                (CLAIMS   A   B))))
```

Fig. 4.11: Orders A and B need resources R1 or R2. R1 can handle two orders in parallel, therefore 2 labels are used to symbolize R1. R2 is exclusive and is represented by 1 lable.

5. An Example for Scheduling

The following (simple) example illustrates how TIC II supports structuring, dispatching and execution of a scheduling task.

Assume orders A, B and C. The orders are partinioned into sub tasks according to the resource requirements. Each order's sub tasks are to be produced in sequence. Tab. 5.1 shows the tasks' allocation to machines, their due dates and durations. Each of the resources R1 and R2 consists of 2 machines, i.e. they can execute maximally two tasks at a time. Resources R3 and R4 are exclusive.

Task	Earliest Start	Duration	Latest Finish	Allocation
A.1	0	14	*	R1
A.2	*	11	*	R3
A.3	*	6	55	R2
B.1	0	13	*	R1
B.2	*	14	*	R4
B.3	*	22	52	R2
C.1	0	8	*	R1
C.2	*	24	*	R2
C.3	*	14	55	R2

Tab. 5.1: The initial knowledge about the 9 tasks of the orders A, B and C. Missing or irrelevant quantitative information is indicated by a "*"

- Structuring: The means the explication of all constraints on the task. TIC II parses the NICOLE description produced by the user and constructs a corresponding constraint net. Fig. 5.1 shows a simplyfied view. The arrows indicate the only qualitative temporal relations of the net. I.e., the temporal net is not completely connected.
- Dispatching: Propagation either (1) detects inconsistency and leads to re-structuring or (2) establishes local consistency among the constraints. The quantitative temporal knowledge on the tasks is completed by propagation of the constraints, in specific of those implied by time-logic: ILC, CC and SC. Propagation does not add any further qualitative temporal relations. Therefore Allen's qualitative time-logic applies only within the boxes. All other temporally related intervals are restricted by the CC and SC constraint only. The quiecent net implements the decision frame, which delimits all further decisions on the variables' evaluation. Fig. 5.2 illustrates the resulting time frames for the tasks. A feasible solution is searched within the decision frame (fig. 5.3). The absence of any solution would indicate an over-constrained problem and would requires re-structuring.
- Execution: Once a schedule is established, its execution is monitored. The constraint net is updated by the chronological data of the production process. Unexpected events or deviations from the schedule lead to new constraints. In that case a new solution is searched for the remaining tasks. Assume the latest finish date of order A to be changed to time-point 50 at time-point 20. Fig. 5.4 shows a solution for the new situation.

6. Outlook

The task of scheduling goes far beyond the capabilities of mere temporal logic. Scheduling even exceeds the capabilities of CS. It does not suffice to find any valid solution: Scheduling is an optimization problem (Fox, Sadeh 90). The quality of a schedule has direct impact on the efficiency of production. On the other hand, scheduling is an NP-complete problem. (Becker, Heuer 95) present efficient heuristic strategies to solve scheduling tasks.

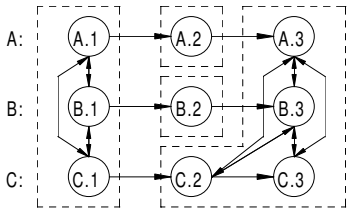

Fig. 5.1: Logical relationship of the 9 tasks. An arrow indicates qualitative precedence ("before"). Tasks which are constrained by allocation to the same resource (cp. fig. 4.10) are surrounded by a box. The temporal constraints correspond to fig. 4.1 and 4.2.

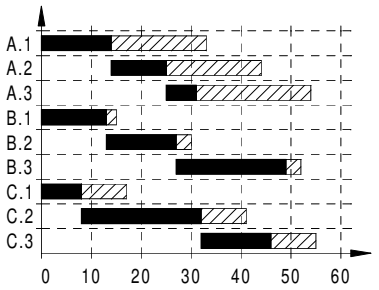

Fig. 5.2: The time frames for the tasks. The execution of a task (black box) can be realized within the hatched box. The current positions of the black boxes are not admissible: E.g. tasks A.1, B.1 and C.3 cannot be produced all in-parallel.

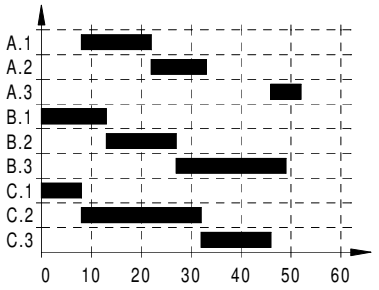

Fig. 5.3: An admissible schedule.

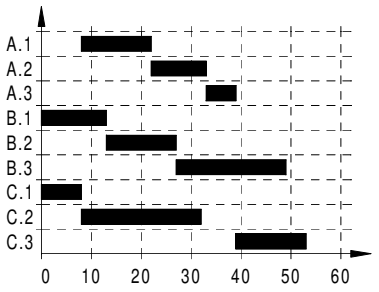

Fig. 5.4: The change of A's latest due date to 50 causes the re-ordering of A.3 and C.3.

Current research concerns the integration of TIC II as monitoring facility into the shop-controller CCC[5] (Becker, Parr 94). CCC's objective is to schedule, coordinate and control workstations within different work cells of a factory. The aim is to replace exhaustive re-scheduling in case of deviations from an initial schedule by modifications within a constraint-net.

Acknowledgement: Volker Schwalm at Technical University of Hamburg-Harburg supported the completion of table 3.1.

7. References

Allen, J. F. (1983), Maintaining Knowledge about Temporal Intervals, *Comm. of the ACM, 26*, pp. 832 - 843.

Becker, S. (1988), *Konzeption und Implementation einer temporalen Inferenzkomponente - Anwendung auf Planungsprobleme in der Fließproduktion*, MS-thesis, Dept. of Computer Science, University of Hamburg. (in German)

Becker, S. (1992), Production Scheduling with Qualitative and Quantitative Constraints, *Proc. IEEE Int. Workshop on Emerging Technologies and Factory Automation ETFA '92*, Melbourne, pp. 83 - 88.

Becker, S., Parr, R. (1994), CCC: An Approach to Cooperative Scheduling, *Proc. of the 3rd Int. Conf. on Automation Technology Automation '94*, Taipei, Vol. 4, pp. 249 - 256.

Dechter, R. Constraint Networks, in: Shapiro, S.C. (ed), *Encyclopaedia of Artificial Intelligence*, 2nd Edition, Vol. 1, John Wiley & Sons, Inc., New York, pp. 276 - 285.

Dechter, R., Meiri, I., Pearl, J. (1991), Temporal Constraint Networks, *Artificial Intelligence 49*, pp. 61-95.

Fox, M. S. (1983), *Constraint-Directed Search: A Case Study of Job-Shop Scheduling*, PhD thesis, University Microfilm International, Ann Arbor.

Fox, M. S., Sadeh, N., (1990), Why is Scheduling Difficult? A CSP Perspective, *Proc. 9th European Conf. on AI ECAI '90*, Stockholm, pp. 754 - 766.

Huber, A., Becker, S. (1988) Production Planning Using a Temporal Planning Component, *Proc. 8th European Conf. on AI ECAI '88*, Munich, pp. 188 - 190.

Huber, A. (1990), *Wissensbasierte Überwachung und Planung in der Fertigung*, Erich Schmidt Verlag, Berlin (in German).

Hyvönen, E. (1989), Constraint Reasoning based on Interval Arithmetic, *Proc. Int. Joint Conf. on AI IJCAI-89*, Michigan, pp. 1193 - 1198.

Jakob, J.-I. (1992), The Temporal Inference Unit - Implementing a Coprocessor for Manufacturing and Artificial Intelligence, *Proc. IEEE Int. Workshop on Emerging Technologies and Factory Automation ETFA '92*, Melbourne, pp. 196 - 200.

Jakob, J.-I. (1994a), Benchmarking Constraints and Temporal Logic Using Manufacturing Schedules, *Proc. 3rd Int. Conf. on Automation Technology Automation '94*, Taipei, Vol. 4, pp. 241 - 248.

Jakob, J.-I. (1994b), Optimizing Constraints for Speed: Benchmarking Temporal Logic, *Proc. 2nd Int. Conf. on Intelligent Systems Engineering ISE (IEE)*, Hamburg, pp. 189 - 194.

Mackworth, A. K. (1977), Consistency in Networks of Relations, *Artificial Intelligence 8*, pp. 99-118.

Mackworth, A. K., Freuder, E. C. (1985), The Complexity of Some Polynomial Network Consistency Algorithms for Constraint Satisfaction Problems, *Artificial Intelligence 25*, pp. 65-73.

Meiri, I. (1991), Combining Qualitative and Quantitative Constraints in Temporal Reasoning, *Proc. 9th Nat. Conf. on AI AAAI-91*, Anaheim, pp. 260 - 267.

Meiri, I. (1992), *Temporal Reasoning: A Constraint Based Approach*, Technical Report R-173, Dept. of Computer Science, University of California.

Meyer, W. (1990), *Expert Systems in Factory Management: Knowledge-Based CIM*, Ellis Horwood Ltd., Chichester.

Moore, R. E. (1966), *Interval Arithmetic*, Prentice-Hall, Englewood Cliffs.

Rit, J.-F. (1986), Propagating Temporal Constraints for Scheduling, *Proc. 5th Nat. Conf. on AI AAAI-86*, Philadelphia, pp. 383-388.

van Beek, P. (1989), Approximation Algorithms for Temporal Reasoning, *Proc. 11th Int. Joint Conf. on Artificial Intelligence IJCAI-89*, Detroit, pp. 1291-1296.

van Beek, P., Cohen, R.(1990), Exact and approximate reasoning about temporal relations, Computational Intelligence, 6, pp. 132 - 144.

Vilain, M, Kautz, H., van Beek, P.G. (1989) Constraint Propagation Algorithms for Temporal Reasoning: A revised Report, in: Weld, D.S., de Kleer, J. (eds.), *Readings in Qualitative Reasoning about Physical Systems*, pp. 373 - 381, Morgan Kaufman, San Mateo.

Winklhofer, A. (1994), *Report Temporale Planung und Terminplanung*, FR-1994-003, April 1994, Bayrisches Forschungszentrum für Wissensbasierte Systeme, Munich.

[5]Coordination, Control and Constraint Nets