Scheduling Algebra

Rob van Glabbeek[1], Peter Rittgen[2]

[1]Computer Science Department, Stanford University, CA 94305-9045, USA
rvg@cs.stanford.edu
[2]Institut für Wirtschaftsinformatik, Universität Koblenz-Landau, Rheinau 1, 56075 Koblenz, Germany
rittgen@uni-koblenz.de

Abstract. The goal of this paper is to develop an algebraic theory of process scheduling. We specify a syntax for denoting processes composed of actions with given durations. Subsequently, we propose axioms for transforming any specification term of a scheduling problem into a term of all valid schedules. Here a *schedule* is a process in which all (implementational) choices (e.g. precise timing) are resolved. In particular, we axiomatize an operator restricting attention to the *efficient* schedules. These schedules turn out to be representable as trees, because in an efficient schedule actions start only at time zero or when a resource is released, i.e. upon termination of the action binding a required resource. All further delay would be useless. Nevertheless, we do not consider resource constraints explicitly here. We show that a normal form exists for every term of the algebra and establish soundness of our axiom system with respect to a schedule semantics, as well as completeness for efficient processes.

Note. A slightly condensed version of this paper will appear in the proceedings of AMAST'98. An earlier version appeared as Arbeitsberichte des Instituts für Wirtschaftsinformatik 12, Universität Koblenz-Landau, Germany 1998. This work was supported by ONR grant N00014-92-J-1974.

Introduction

The problem of scheduling entails assigning an execution time (and sometimes a processor) to each of a set of actions with given durations, depending on a certain goal function (e.g. shortest schedule) and certain causal and resource constraints. The theory of scheduling has been investigated since the early 50s. The research so far can be divided into two categories: partitioning the set of all scheduling problems into (complexity) classes, and finding efficient algorithms for each class. Most of the problems being NP-complete, substantial effort has been spent on problem relaxations and heuristics to obtain algorithms that compute near-optimal schedules in polynomial time. But an axiomatization of the theory of scheduling still remains to be given. Here we outline such a calculus abstracting from both resource constraints and goal function and limiting the structure of the causal order. Our aim is to provide an axiom system that turns a process specification into a set of *efficient* (or *semi-active*) schedules [7]. These are the schedules that are potentially optimal for certain constraints and goal functions, provided the latter do not favor spending idle time. For simplicity, we consider only specifications in which no action has more than one immediate causal predecessor, i.e. the precedence graph has a multi-tree order.

Viewed from a different angle, our approach can be described as applying concurrency theory to scheduling. A lot of algebras of concurrent processes have been given, e.g. ACP [3], CCS [11] or CSP [10]. Hence, one might ask why we develop a new calculus instead of using an existing one. The answer lies in the specific requirements of processes needed for scheduling: A scheduling calculus must include the notion of time, the concept of durational actions and the possibility to delay an action arbitrarily.

Time has been incorporated into the calculi mentioned above (ACP [2], TCCS [12], TCSP [13]), but generally employing instantaneous actions. Durational actions are treated in [1, 9], for example, but these approaches do not allow actions to be delayed (or in the case of [9] only to wait for a synchronization partner). This means that in “$a \parallel b$” actions a and b are always executed simultaneously whereas in scheduling they might as well be run sequentially in either order. In e.g. [4, 6], and in various references quoted in [6], semantics are given that allow for arbitrary delays, but they are not accompanied by an axiomatization. Moreover, nowhere in the process algebra literature a concept of efficiency is formalized resembling the one of [7]. Hence, a new algebra is necessary. We call it *scheduling algebra* and develop its theoretical framework in the following sections.

An example process is given in fig. 1. Its algebraic term is: $a; (b \parallel c; d) \parallel e$.

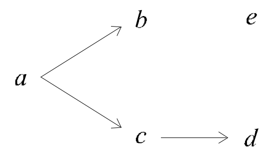

Fig. 1. Example process

To visualize a schedule, the so-called GANTT chart [5] is used (see fig. 2). It depicts every action as a box labeled with its identifier. The length of the box indicates the duration of the action, and the placement of the box shows exactly when it takes place. In the scenario of fig. 2, actions *a* and *d* start simultaneously (on different processors), whereas *c* is executed immediately after *a* (and *b* after *d*). For the additional action *e*, the arrows indicate possible placements.

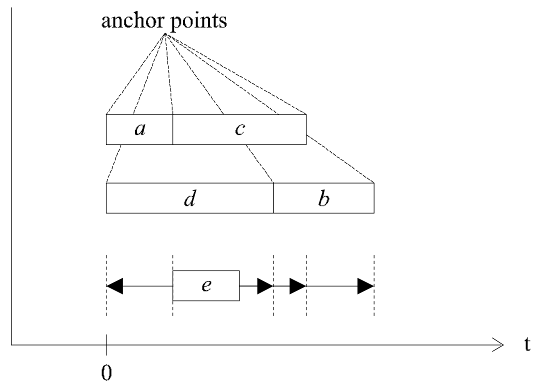

Fig. 2. GANTT diagram

Anchoring and efficiency

A full-fledged scheduling problem consists of a *specification*, determining a set of valid schedules, some *resource constraints*, saying that certain actions cannot overlap in time, and a *goal function*, specifying how satisfactory certain schedules are. The objective is then to find an optimal schedule among the valid schedules that satisfy the resource constraints. Here we deal with scheduling problems in which the goal function and resource constraints are unspecified. We are interested in describing the valid schedules, and in finding, among the possibly infinite assortment of valid schedules, a finite subset, as small as possible, that for any choice of resource constraints and goal function contains an optimal schedule. In this quest, as is usual in scheduling theory, we limit attention to goal functions that do not favor spending idle time: a valid schedule, meeting the resource constraints, cannot improve by moving an action forwards in time.

A schedule is called *efficient* (or *semi-active*) if it is valid and no actions can be moved backwards in time without violating validity or some hypothetical resource constraints. For the types of scheduling problems studied in this paper, as well as elsewhere in scheduling theory, every schedule can be "improved" to an efficient schedule by moving some of its actions backwards in time (cf. [7, theorem 2.1]). Depending on the goal function, this may not be a strict improvement, but it is never a regress. Thus the efficient schedules form the subclass of schedules sought above. However, in order to be of practical use, a more explicit characterization of the efficient schedules is needed.

We call a schedule *anchored* if each action starts either at time 0 or at the termination time of some other process. Fig. 2 depicts an anchored schedule. If the execution times for the actions *a* to *d* is already given as indicated, an additional action *e* may start at time 0 if it does not use a resource required by either *a* or *d*. If *e* conflicts with *a* but not with *d*, it may begin execution upon termination of *a* (as drawn in fig. 2). So in total we have five possibilities for the starting time of *e*: 0 or the termination time of *a*, *b*, *c* or *d*. We call these points in time *anchor points*.

It is easy to see that every anchored schedule *s* is efficient. If the resource constraints say that any two actions that do not overlap in *s* cannot overlap at all, it is impossible to preserve these resource constraints by moving actions backwards. Moreover, a goal function could be chosen that makes every schedule in which even one action takes place later than in s less attractive than s itself, no matter what happens to the other actions. Conversely, for a special class of scheduling problems it is shown in [7] that all efficient schedules are anchored, i.e. that every valid schedule can be turned into a valid anchored one by moving actions backwards in time. Here we are interested in scheduling problems for which the same holds. For these the efficient schedules are exactly the anchored ones, and our efficiency operator can be implemented as an anchoring operator.

Syntax

Let A be a set of *atomic actions* (external actions), and let T be the set of positive reals including zero. A pair (a, t) of $A \times T$ written as $a(t)$ is called a time-stamped action or activity. It indicates that action a takes t units of time (the duration of a). The elements t of T also denote internal time actions which can be seen as silent steps (τ, t). Process terms can be constructed using the constants

ε	denoting the empty process,
$a(t)^+ \in A \times T$	denoting the starting of action a with duration t,

the unary operators

$a(t)\cdot$	*Sequence*, $a(t)\cdot\ x$ meaning that x starts immediately after the execution of a, i.e. at time t,
$t\cdot$	*Time offset* for $t \in T$; in $t\cdot\ x$ the starting time of the actions in x is delayed by t time units and an anchor point at time t is added,
Δ	Arbitrary *delay operator*, meaning that the prefixed process can be postponed indefinitely,
[.]	*Delay elimination*, deleting every occurrence of the delay operator from the enclosed term by attaching the processes prefixed by Δ to every possible anchor point,

and the binary operators

$\cup$	*Choice*; $x \cup y$ denotes the execution of either x or y exclusively,
$\curlywedge$	*forking of time* ('Hühnerfuß' or 'chicken claw'), an urgent parallel composition; $x \curlywedge y$ means that the initial actions of both x and y start at the same time.

In addition, we use the following abbreviations:

$a(t);$	(*causal*) *precedence*, a unary operator expressing linear order, so $a(t); x$ says that x has to be executed some time (but not necessarily immediately) after completion of a. We have: $a(t); x = a(t)\cdot\ \Delta x$.
$\|\|$	*concurrence*, a binary operator expressing independence of processes. We have: $x \parallel y = \Delta x \curlywedge \Delta y$.

The binding precedence of the binary operators (from loose to tight) is: $\cup$, $\parallel$ and $\curlywedge$. The unary operators bind tightest.

Terms built from ε, $a(t)^+$, $a(t)\cdot$, $t\cdot$, $\curlywedge$ and [.] denote unique schedules. Choice between various schedules is introduced by the operators $\cup$ and Δ, and hence also by $a(t);$ and $\parallel$. The delay elimination simply prunes some choices introduced by Δ.

We take a, b to range over $A \times T \cup A$, t, u, v to range over T and x, y, z to range over all process expressions. For convenience, we sometimes abbreviate the expression $a(t)$ by a. Moreover, we leave out trailing ε's in terms $a(t)\cdot\ \varepsilon$ and $t\cdot\ \varepsilon$.

Example: The expression $a; b \parallel c; d$ denotes two concurrent tasks, one executing first a and then b, and the other running c followed by d. Example schedules for the efficient process $[a; b \parallel c; d]$ are shown in fig. 3.

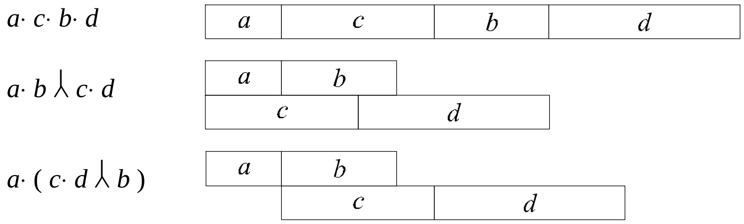

Fig. 3. Example schedules

As *specification terms* we consider processes P or $[P]$ that contain no further occurrences of [.] and no direct occurrences of the sequencing operator $a(t)\cdot$, although occurrences of $a(t);$ are permitted. Let us call an occurrence of $a(t)\cdot$ which can not be regarded as an occurrence of $a(t);$ *tight sequencing*. Tight sequencing can be encoded with the delay elimination operator, namely $a(t)\cdot\ b(t') = [a(t)\cdot\ \Delta b(t')] = [a(t);\ b(t')]$. For these specification terms the argument of [7] showing that all efficient schedules are anchored applies, so that the delay elimination operator indeed describes efficiency. This need not be true for general processes. Consider for example the process $\Delta\ a(1)\cdot\ b(1) \parallel c(2)$. A valid schedule for this process specification is shown in fig. 4. This schedule is efficient because it is the shortest schedule for the given specification if we assume that b and c may not overlap due to a resource conflict. But at the same time it is not anchored because the starting time of action a does not coincide with the finishing time of any other action.

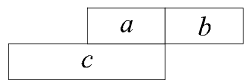

Fig. 4. An efficient schedule that is not anchored

Axiomatic System

The axioms of scheduling algebra are listed below. Please note that "+" refers to the real number addition.

<u>Basic Axioms</u>

$$x \cup (y \cup z) = (x \cup y) \cup z \qquad \text{A1}$$

$$x \cup y = y \cup x \qquad \text{A2}$$

$$x \cup x = x \qquad \text{A3}$$

$$(x \curlywedge y) \curlywedge z = x \curlywedge (y \curlywedge z) \qquad \text{A4}$$

$$x \curlywedge y = y \curlywedge x \qquad \text{A5}$$

<u>Zero Axioms</u>

$$0\cdot x = x \qquad \text{Z1}$$

$$x \curlywedge \varepsilon = x \qquad \text{Z2}$$

<u>Distributivity of choice</u>

$$t\cdot (x \cup y) = t\cdot x \cup t\cdot y \qquad \text{D1}$$

$$\Delta (x \cup y) = \Delta x \cup \Delta y \qquad \text{D2}$$

$$x \curlywedge (y \cup z) = x \curlywedge y \cup x \curlywedge z \qquad \text{D3}$$

Normal Form Axioms

$$a(t)\cdot x \;=\; a(t)^{+} \curlywedge t\cdot x \qquad \text{N1}$$

$$t\cdot x \curlywedge (t+u)\cdot y \;=\; t\cdot (x \curlywedge u\cdot y) \qquad \text{N2}$$

$$\Delta x \curlywedge \Delta y \;=\; \Delta(x \curlywedge \Delta y \cup y \curlywedge \Delta x) \qquad \text{N3}$$

Elimination of delay (anchor axioms)

$$[x \cup y] \;=\; [x] \cup [y] \qquad \text{AA1}$$

$$[a(t)^{+} \curlywedge x] \;=\; a(t)^{+} \curlywedge [x] \qquad \text{AA2}$$

$$[t\cdot x] \;=\; t\cdot [x] \qquad \text{AA3}$$

$$[\Delta x] \;=\; [x] \qquad \text{AA4}$$

$$[t\cdot x \curlywedge \Delta y] \;=\; [t\cdot x \curlywedge y] \cup t\cdot [x \curlywedge \Delta y] \qquad \text{AA5}$$

$$[\varepsilon] \;=\; \varepsilon \qquad \text{AA6}$$

Proposition
The following equation follows from applying N1, D1, D3 and N1 in the given order:

$$a\cdot (x \cup y) \;=\; a\cdot x \cup a\cdot y \qquad \text{P1}$$

The distributivity axioms and P1 indicate that we work in a linear-time setting. This is because we treat scheduling as a planning problem where all decisions (choices) are made prior to the execution of the process.

Note that we do not have the axioms $x \cup \varepsilon = x$ and $t\cdot u\cdot x = (t+u)\cdot x$. They are not sound because the terms on the left side introduce an additional empty schedule and an anchor point at t, respectively.

Example
Applying the axioms, the definitions and P1 in a suitable order, we can compute the schedules for an example process specification. The penultimate equality shows the term in *normal form* (see below); the last equality gives the term in *anchor form* (explained later). Recall that we abbreviate $a(t)$ by a, and that we leave out trailing ε's.

$$[\, a(1); b(3) \,\|\, c(2); d(1) \,] =$$

$$[\, \Delta(\, a(1)\cdot \Delta b(3)\,) \curlywedge \Delta(\, c(2)\cdot \Delta d(1)\,)\,] =$$

$$\begin{array}{llll}
a^{+} \curlywedge 1\,(\, b^{+} \curlywedge 3\,(\, c^{+} \curlywedge 2\,(\, d^{+} \curlywedge 1\,)\,)\,) & \cup & a^{+} \curlywedge 1\,(\, c^{+} \curlywedge b^{+} \curlywedge 2\,(\, d^{+} \curlywedge 1\,)\,) & \cup \\
a^{+} \curlywedge 1\,(\, c^{+} \curlywedge b^{+} \curlywedge 2\cdot 1\,(\, d^{+} \curlywedge 1\,)\,) & \cup & a^{+} \curlywedge 1\,(\, c^{+} \curlywedge 2\,(\, b^{+} \curlywedge 3\,(\, d^{+} \curlywedge 1\,)\,)\,) & \cup \\
a^{+} \curlywedge 1\,(\, c^{+} \curlywedge 2\,(\, b^{+} \curlywedge (\, d^{+} \curlywedge 1\cdot 2\,)\,)\,) & \cup & a^{+} \curlywedge 1\,(\, c^{+} \curlywedge 2\,(\, d^{+} \curlywedge 1\,(\, b^{+} \curlywedge 3\,)\,)\,) & \cup \\
c^{+} \curlywedge 2\,(\, a^{+} \curlywedge 1\,(\, b^{+} \curlywedge 3\,(\, d^{+} \curlywedge 1\,)\,)\,) & \cup & c^{+} \curlywedge 2\,(\, a^{+} \curlywedge 1\,(\, d^{+} \curlywedge (\, b^{+} \curlywedge 1\cdot 2\,)\,)\,) & \cup \\
c^{+} \curlywedge 2\,(\, a^{+} \curlywedge 1\,(\, d^{+} \curlywedge 1\,(\, b^{+} \curlywedge 3\,)\,)\,) & \cup & c^{+} \curlywedge 2\,(\, d^{+} \curlywedge a^{+} \curlywedge (\, 1\,(\, b^{+} \curlywedge 3\,)\,)\,) & \cup \\
c^{+} \curlywedge 2\,(\, d^{+} \curlywedge 1\,(\, a^{+} \curlywedge (\, 1\,(\, b^{+} \curlywedge 3\,)\,)\,)\,) & \cup & c^{+} \curlywedge a^{+} \curlywedge 1\,(\, b^{+} \curlywedge 1\cdot 2\,(\, d^{+} \curlywedge 1\,)\,) & \cup \\
c^{+} \curlywedge a^{+} \curlywedge 1\,(\, b^{+} \curlywedge 1\,(\, d^{+} \curlywedge 1\cdot 1\,)\,) & \cup & c^{+} \curlywedge a^{+} \curlywedge 1\cdot 1\,(\, b^{+} \curlywedge 3\,(\, d^{+} \curlywedge 1\,)\,) & \cup \\
c^{+} \curlywedge a^{+} \curlywedge 1\cdot 1\,(\, d^{+} \curlywedge 1\,(\, b^{+} \curlywedge 3\,)\,) & \cup & c^{+} \curlywedge a^{+} \curlywedge 1\cdot 1\,(\, d^{+} \curlywedge (\, b^{+} \curlywedge 1\cdot 2\,)\,) & =
\end{array}$$

$$\begin{array}{l}
a\cdot b\cdot c\cdot d \;\cup\; a\cdot c\cdot b\cdot d \;\cup\; a\cdot c\cdot d\cdot b \;\cup\; c\cdot a\cdot b\cdot d \;\cup\; c\cdot a\cdot d\cdot b \;\cup\; c\cdot d\cdot a\cdot b \;\cup \\
a\cdot (\, c\cdot d \curlywedge b\,) \;\cup\; a\cdot (\, b\cdot d \curlywedge c\,) \;\cup\; c\cdot (\, a \curlywedge d\cdot b\,) \;\cup\; a\cdot c\cdot (\, b \curlywedge d\,) \;\cup\; a\cdot b \curlywedge c\cdot d \;\cup \\
c\cdot a\cdot (\, b \curlywedge d\,) \;\cup\; a \curlywedge c\cdot (\, b \curlywedge d\,) \;\cup\; a \curlywedge c\cdot b\cdot d \;\cup\; a \curlywedge c\cdot d\cdot b \;\cup\; c \curlywedge a\cdot b\cdot d
\end{array}$$

Normal form theorem

All terms can be written in the form $\bigcup N$ where

$$
\begin{array}{lll}
N & ::= & a(t)^+ \curlywedge N \\
 & | & t\cdot N \qquad \text{for } t \neq 0 \\
 & | & \Delta N \\
 & | & t\cdot N \curlywedge \Delta N \qquad \text{for } t \neq 0 \\
 & | & \varepsilon
\end{array}
$$

This normal form theorem is proved by structural induction on terms. We show for every operator that the composition of terms in normal form again yields a term that can be expressed in normal form.

The choice operator distributes over every other operator (D1 - D3, P1, AA1) so that it can always be moved to the top level. This and associativity (A1) and commutativity (A2) allow us to denote every term as a union of choice-free subterms. Prefixing a normal term with Δ or $t\cdot$ (for $t\neq0$) is by definition normal, as is ε. Z1 takes care of terms $0\cdot N$. Using Z2, the constant $a(t)^+$ can be written as the normal form $a(t)^+ \curlywedge \varepsilon$. For $a(t)\cdot N$, axiom N1 leads to the normal form

$$a(t)\cdot N = a(t)^+ \curlywedge t\cdot N.$$

Delay elimination is straightforward with AA1 - AA6. It remains to be shown that *forking* of normal terms can be normalized. Let K, L, M and N be normal terms. Then

$$(a(t)^+ \curlywedge M) \curlywedge N \xrightarrow{\text{A4}} a(t)^+ \curlywedge (M \curlywedge N) \xrightarrow{\text{IH}} a(t)^+ \curlywedge N',$$
$$\varepsilon \curlywedge N \xrightarrow{\text{A5,Z2}} N,$$

where IH abbreviates induction hypothesis, and N' denotes the normalization of $M \curlywedge N$. For the remaining 3 normal forms we have to prove normalization only for the upper triangle of the following matrix (due to commutativity of forking), in which t, u>0:

$\curlywedge$	$u\cdot K$	ΔK	$uK \curlywedge \Delta L$
$t\cdot M$	N2 ($t\leq u$), IH	√	A4, N2 ($t\leq u$), IH
ΔM		N3, IH, IH	A5, A4, N3, IH, IH
$tM \curlywedge \Delta N$			A4, A4, A5, A4, A4, N2 ($t\leq u$), N3, IH, IH, IH

Table 1. Normal form proofs

The proof proceeds applying the laws stated modulo commutativity (√ means that the term is already in normal form).

Semantics

As a semantic model, we define schedules similar to the real-time execution sequences in [13]. Each expression maps to a set of possible schedules:

$$[\![.]\!]: Expr \to 2^{\mathbb{N}^{A\times T\times T} \times 2^T \times 2^{T-\{0\}}}$$

A schedule is a triple $\sigma = \langle\sigma_0; \sigma_1; \sigma_2\rangle$ where σ_0 is a multi-set over triples (action, duration, starting time), σ_1 is a set of starting and finishing times and σ_2 a set of anchor points (or finishing times), excluding 0. Because 0 is always an anchor point, there is no need to record it explicitly; suppressing it turns out to have technical advantages. Typically, we leave out the brackets delimiting σ_0, σ_1 and σ_2, i.e.

$$[\![\, a(1)\cdot b(2) \curlywedge a(1) \,]\!] = \{ \langle (a, 1, 0), (a, 1, 0), (b, 2, 1) ; 0, 1, 3 ; 1, 3 \rangle \}.$$

This schedule is interpreted as follows:

- $a(1)$ is started twice at time 0,
- $b(2)$ at time 1 (immediately after a),
- the starting times are 0 and 1 (for a and b respectively)
- and the anchor points are 1 and 3 (the finishing times of a and b respectively).

Because Z1 yields $a(1)\cdot b(2) = a(1)\cdot b(2)\cdot 0$ and the latter term has a starting time 3 (for the internal action of duration 0), 3 can be regarded as a potential starting time of the process $a(1)\cdot b(2)$ as well. For this reason we include in σ_1 not only the starting times, but also the finishing times of actions.

For two schedules σ and ρ, we write $\sigma\cup\rho$ for $\langle\, \sigma_0\cup\rho_0 \,;\, \sigma_1\cup\rho_1 \,;\, \sigma_2\cup\rho_2 \,\rangle$. The offset of a schedule by a fixed amount of time is defined by:

$$\sigma+u \;=\; \langle\, \{\!\!\{\, (a, t, p+u) \mid (a, t, p)\in \sigma_0 \,\}\!\!\} \,;\, \{\, p+u \mid p\in \sigma_1 \,\} \,;\, \{\, p+u \mid p\in \sigma_2 \,\} \,\rangle.$$

The general definition of the semantics is as follows:

$$[\![\, \varepsilon \,]\!] = \{\, \langle\, \varnothing \,;\, 0 \,;\, \varnothing \,\rangle \,\} \tag{1}$$

$$[\![\, a(t)^{+} \,]\!] = \{\, \langle\, (a, t, 0) \,;\, 0 \,;\, \varnothing \,\rangle \,\} \tag{2}$$

$$[\![\, a(t)\cdot P \,]\!] = \{\, \langle\, (a, t, 0) \,;\, 0 \,;\, \{t\}-\{0\} \,\rangle \cup (\sigma+t) \mid \sigma\in [\![P]\!] \,\} \tag{3}$$

$$[\![\, t\cdot P \,]\!] = \{\, \langle\, \varnothing \,;\, 0 \,;\, \{t\}-\{0\} \,\rangle \cup (\sigma+t) \mid \sigma\in [\![P]\!] \,\} \tag{4}$$

$$[\![\, \Delta P \,]\!] = \{\, \langle\, \varnothing \,;\, 0 \,;\, \varnothing \,\rangle \cup (\sigma+u) \mid \sigma\in [\![P]\!],\, u\in T \,\} \tag{5}$$

$$[\![\, [P] \,]\!] = \{\, \sigma\in [\![P]\!] \mid \sigma_1 \subset \sigma_2\cup\{0\} \,\} \tag{6}$$

$$[\![\, P\cup Q \,]\!] = [\![P]\!] \cup [\![Q]\!] \tag{7}$$

$$[\![\, P \curlywedge Q \,]\!] = \{\, \sigma\cup\rho \mid \sigma\in [\![P]\!],\, \rho\in [\![Q]\!] \,\} \tag{8}$$

Note that this definition ensures that $0 \in \sigma_1$, $0 \notin \sigma_2$ and $\sigma_2 \subset \sigma_1$, for all schedules σ.

The role of the elimination operator [.]

A term not containing [.] represents a set of schedules, both efficient and inefficient ones. Delay elimination restricts attention to the efficient schedules, thereby eliminating all schedules known not to be optimal under any goal function or constraints. On the syntactical level, this corresponds to the elimination of Δ's, which is achieved by the operator [.]. One might ask why this is not done implicitly, i.e. by giving every term p the semantics of $[p]$. The answer is that the following implication does not hold:

$$[\![\, [x] \,]\!] = [\![\, [y] \,]\!] \;\Rightarrow\; [\![\, [x \,||\, z] \,]\!] = [\![\, [y \,||\, z] \,]\!]. \qquad \lightning$$

A counterexample to the implication can be constructed easily by setting:

$$\begin{aligned} x &= a(1)\cdot b(1),\\ y &= a(1)\cdot \Delta\, b(1),\\ z &= c(1). \end{aligned}$$

We then get:

$$[\![\, [a(1)\cdot b(1)] \,]\!] \;=\; \{\, \langle\, (a, 1, 0), (b, 1, 1) \,;\, 0, 1, 2 \,;\, 1, 2 \,\rangle \,\} \;=\; [\![\, [a(1)\cdot \Delta\, b(1)] \,]\!],$$

but:

$$\langle\, (a, 1, 0), (b, 1, 2), (c, 1, 1) \,;\, 0, 1, 2, 3 \,;\; 1, 2, 3 \,\rangle \in [\![\, [a(1)\cdot \Delta\, b(1) \,||\, c(1)] \,]\!],$$
$$\langle\, (a, 1, 0), (b, 1, 2), (c, 1, 1) \,;\, 0, 1, 2, 3 \,;\; 1, 2, 3 \,\rangle \notin [\![\, [a(1)\cdot b(1) \,||\, c(1)] \,]\!].$$

Correctness

Axioms A1 to A3 are correct w.r.t. our schedule semantics because set union is associative, commutative and idempotent. Associativity and commutativity of multi-set union yield A4 and A5. Z1 and Z2 are explained by 0 and the empty set being the neutral elements of addition and (multi-)set union respectively, taking into account that 0 is already in σ_1 for every schedule σ. Correctness of D1 can be derived as follows:

$$\begin{aligned} [\![\, t\cdot (x\cup y) \,]\!] \;&=\; \{\, \langle\, \varnothing \,;\, 0 \,;\, \{t\}-\{0\} \,\rangle \cup (\sigma+t) \mid \sigma\in [\![x\cup y]\!] \,\}\\ &=\; \{\, \langle\, \varnothing \,;\, 0 \,;\, \{t\}-\{0\} \,\rangle \cup (\sigma+t) \mid \sigma\in [\![x]\!]\cup[\![y]\!] \,\} \end{aligned}$$

$$
\begin{aligned}
&= \{ \langle \varnothing ; 0 ; \{t\}-\{0\} \rangle \cup (\sigma+t) \mid \sigma\in \llbracket x \rrbracket \} \;\cup\; \{ \langle \varnothing ; 0 ; \{t\}-\{0\} \rangle \cup (\sigma+t) \mid \sigma\in \llbracket y \rrbracket \} \\
&= \llbracket\, t\cdot x \,\rrbracket \cup \llbracket\, t\cdot y \,\rrbracket \\
&= \llbracket\, t\cdot x \cup t\cdot y \,\rrbracket
\end{aligned}
$$

The proofs of D2 and D3 proceed analogously. For N1 - N3 and AA1 - AA6, we have:

$$
\begin{aligned}
\llbracket\, a(t)\cdot x \,\rrbracket &= \{ \langle (a, t, 0) ; 0 ; \{t\} - \{0\} \rangle \cup (\sigma + t) \mid \sigma\in \llbracket\, x \,\rrbracket \} \\
&= \{ \langle (a, t, 0) ; 0 ; \varnothing \rangle \cup \big(\langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup (\sigma + t) \big) \mid \sigma\in \llbracket\, x \,\rrbracket \} \\
&= \{ \langle (a, t, 0) ; 0 ; \varnothing \rangle \cup \rho \mid \rho\in \llbracket\, t\cdot x \,\rrbracket \} \\
&= \llbracket\, a(t)^{+} \curlywedge t\cdot x \,\rrbracket
\end{aligned}
$$

$$
\begin{aligned}
&\llbracket\, t\cdot x \curlywedge (t + u)\cdot y \,\rrbracket \\
&= \{ \sigma\cup\rho \mid \sigma\in \llbracket\, t\cdot x \,\rrbracket, \rho\in \llbracket\, (t+u)\cdot y \,\rrbracket \} \\
&= \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup (\pi + t) \cup \langle \varnothing ; 0 ; \{t+u\} - \{0\} \rangle \cup (\omega + t + u) \mid \pi\in \llbracket\, x \,\rrbracket, \omega\in \llbracket\, y \,\rrbracket \} \\
&\overset{[1]}{=} \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup (\pi + t) \cup \langle \varnothing ; t ; \{t+u\} - \{0\} \rangle \cup (\omega + t + u) \mid \pi\in \llbracket\, x \,\rrbracket, \omega\in \llbracket\, y \,\rrbracket \} \\
&\overset{[2]}{=} \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup (\pi + t) \cup \langle \varnothing ; t ; \{t+u\} - \{t\} \rangle \cup (\omega + t + u) \mid \pi\in \llbracket\, x \,\rrbracket, \omega\in \llbracket\, y \,\rrbracket \} \\
&= \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup (\pi + t) \cup (\langle \varnothing ; 0 ; \{u\} - \{0\} \rangle + t) \cup (\omega + t + u) \mid \pi\in \llbracket\, x \,\rrbracket, \omega\in \llbracket\, y \,\rrbracket \} \\
&= \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup \Big(\big(\pi \cup \langle \varnothing ; 0 ; \{u\} - \{0\} \rangle \cup (\omega + u) \big) + t \Big) \mid \pi\in \llbracket\, x \,\rrbracket, \omega\in \llbracket\, y \,\rrbracket \} \\
&= \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup \big((\pi \cup \rho) + t \big) \mid \pi\in \llbracket\, x \,\rrbracket, \rho\in \llbracket\, u\cdot y \,\rrbracket \} \\
&= \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup (\sigma + t) \mid \sigma\in \llbracket\, x \curlywedge u\cdot y \,\rrbracket \} \\
&= \llbracket\, t\cdot (x \curlywedge u\cdot y) \,\rrbracket
\end{aligned}
$$

$$
\begin{aligned}
&\llbracket\, \Delta x \curlywedge \Delta y \,\rrbracket \\
&= \{ \sigma\cup\rho \mid \sigma\in \llbracket\, \Delta x \,\rrbracket, \rho\in \llbracket\, \Delta y \,\rrbracket \} \\
&= \{ \langle \varnothing ; 0 ; \varnothing \rangle \cup (\pi + t)\cup(\omega + u) \mid \pi\in \llbracket\, x \,\rrbracket, \omega\in \llbracket\, y \,\rrbracket, t, u\in T \} \\
&\overset{[1]}{=} \{ \langle \varnothing ; 0, t ; \varnothing \rangle \cup (\pi + t)\cup(\omega + u) \mid \pi\in \llbracket\, x \,\rrbracket, \omega\in \llbracket\, y \,\rrbracket, t \le u \} \cup \\
&\qquad \{ \langle \varnothing ; 0, u ; \varnothing \rangle \cup (\pi + t)\cup(\omega + u) \mid \pi\in \llbracket\, x \,\rrbracket, \omega\in \llbracket\, y \,\rrbracket, t \ge u \} \\
&\overset{[3]}{=} \{ \langle \varnothing ; 0 ; \varnothing \rangle \cup \Big(\big(\pi \cup \langle \varnothing ; 0 ; \varnothing \rangle \cup (\omega + v) \big) + t \Big) \mid \pi\in \llbracket\, x \,\rrbracket, \omega\in \llbracket\, y \,\rrbracket, t, v\in T \} \cup \\
&\qquad \{ \langle \varnothing ; 0 ; \varnothing \rangle \cup \Big(\big(\omega \cup \langle \varnothing ; 0 ; \varnothing \rangle \cup (\pi + w) \big) + u \Big) \mid \pi\in \llbracket\, x \,\rrbracket, \omega\in \llbracket\, y \,\rrbracket, u, w\in T \} \\
&= \{ \langle \varnothing ; 0 ; \varnothing \rangle \cup \big((\pi \cup \rho) + t \big) \mid \pi\in \llbracket\, x \,\rrbracket, \rho\in \llbracket\, \Delta y \,\rrbracket, t\in T \} \cup \\
&\qquad \{ \langle \varnothing ; 0 ; \varnothing \rangle \cup \big((\omega \cup \sigma) + u \big) \mid \sigma\in \llbracket\, \Delta x \,\rrbracket, \omega\in \llbracket\, y \,\rrbracket, u\in T \} \\
&= \{ \langle \varnothing ; 0 ; \varnothing \rangle \cup (\psi + t) \mid \psi\in \llbracket\, x \curlywedge \Delta y \,\rrbracket, t\in T \} \cup \{ \langle \varnothing ; 0 ; \varnothing \rangle \cup (\zeta + u) \mid \zeta\in \llbracket\, \Delta x \curlywedge y \,\rrbracket, u\in T \} \\
&= \llbracket\, \Delta(x \curlywedge \Delta y) \,\rrbracket \cup \llbracket\, \Delta(y \curlywedge \Delta x) \,\rrbracket \\
&= \llbracket\, \Delta(x \curlywedge \Delta y \cup y \curlywedge \Delta x) \,\rrbracket
\end{aligned}
$$

$$
\begin{aligned}
\llbracket\, [x \cup y] \,\rrbracket &= \{ \sigma\in \llbracket\, x \cup y \,\rrbracket \mid \sigma_1 \subset \sigma_2 \cup \{0\} \} \\
&= \{ \sigma\in \llbracket\, x \,\rrbracket \cup \llbracket\, y \,\rrbracket \mid \sigma_1 \subset \sigma_2 \cup \{0\} \} \\
&= \{ \sigma\in \llbracket\, x \,\rrbracket \mid \sigma_1 \subset \sigma_2 \cup \{0\} \} \cup \{ \sigma\in \llbracket\, y \,\rrbracket \mid \sigma_1 \subset \sigma_2 \cup \{0\} \} \\
&= \llbracket\, [x] \,\rrbracket \cup \llbracket\, [y] \,\rrbracket \\
&= \llbracket\, [x] \cup [y] \,\rrbracket \\
\llbracket\, [a(t)^{+} \curlywedge x] \,\rrbracket &= \{ \sigma\in \llbracket\, a(t)^{+} \curlywedge x \,\rrbracket \mid \sigma_1 \subset \sigma_2 \cup \{0\} \} \\
&= \{ \langle (a, t, 0) ; 0 ; \varnothing \rangle \cup \rho \mid \rho\in \llbracket\, x \,\rrbracket, \rho_1\cup\{0\} \subset \rho_2\cup\varnothing\cup\{0\} \} \\
&= \{ \langle (a, t, 0) ; 0 ; \varnothing \rangle \cup \rho \mid \rho\in \llbracket\, x \,\rrbracket, \rho_1 \subset \rho_2\cup\{0\} \} \\
&= \{ \langle (a, t, 0) ; 0 ; \varnothing \rangle \cup \sigma \mid \sigma\in \llbracket\, [x] \,\rrbracket \} \\
&= \llbracket\, a(t)^{+} \curlywedge [x] \,\rrbracket \\
\llbracket\, [t\cdot x] \,\rrbracket &= \{ \sigma\in \llbracket\, t\cdot x \,\rrbracket \mid \sigma_1 \subset \sigma_2 \cup \{0\} \} \\
&= \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup (\rho + t) \mid \rho\in \llbracket\, x \,\rrbracket, (\rho_1 + t)\cup\{0\} \subset (\rho_2 + t)\cup\{t, 0\} \} \\
&= \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup (\rho + t) \mid \rho\in \llbracket\, x \,\rrbracket, \rho_1 \subset \rho_2\cup\{0\} \} \\
&= \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup (\rho + t) \mid \rho\in \llbracket\, [x] \,\rrbracket \} \\
&= \llbracket\, t\cdot [x] \,\rrbracket \\
\llbracket\, [\Delta x] \,\rrbracket &= \{ \sigma\in \llbracket\, \Delta x \,\rrbracket \mid \sigma_1 \subset \sigma_2 \cup \{0\} \}
\end{aligned}
$$

---

[1] We always have that $0\in\pi_1$, hence $t\in\pi_1+t$.

[2] We leave it to the reader to check $(\{t\} - \{0\})\cup(\{t+u\} - \{0\}) = (\{t\} - \{0\})\cup(\{t+u\} - \{t\})$.

[3] In the first line we write $t+v$ for u. In the second we replace t by $u+w$.

$$
\begin{aligned}
&= \{ \langle \varnothing ; 0 ; \varnothing \rangle \cup (\rho + u) \mid \rho \in \llbracket x \rrbracket, u \in T, (\rho_1 + u) \cup \{0\} \subset (\rho_2 + u) \cup \{0\} \} \\
&=^4 \{ \langle \varnothing ; 0 ; \varnothing \rangle \cup \rho \mid \rho \in \llbracket x \rrbracket, \rho_1 \subset \rho_2 \cup \{0\} \} \\
&=^5 \{ \rho \mid \rho \in \llbracket x \rrbracket, \rho_1 \subset \rho_2 \cup \{0\} \} \\
&= \llbracket [x] \rrbracket
\end{aligned}
$$

$\llbracket [t{\cdot}x \curlywedge \Delta y] \rrbracket$

$$
\begin{aligned}
= \;& \{ \sigma \in \llbracket t{\cdot}x \curlywedge \Delta y \rrbracket \mid \sigma_1 \subset \sigma_2 \cup \{0\} \} \\
= \;& \{ \rho \cup \omega \mid \rho \in \llbracket t{\cdot}x \rrbracket, \omega \in \llbracket \Delta y \rrbracket, \rho_1 \cup \omega_1 \subset \rho_2 \cup \omega_2 \cup \{0\} \} \\
= \;& \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup (\psi + t) \cup (\pi + u) \\
& \mid \psi \in \llbracket x \rrbracket, \pi \in \llbracket y \rrbracket, u \in T, (\psi_1 + t) \cup (\pi_1 + u) \cup \{0\} \subset (\psi_2 + t) \cup (\pi_2 + u) \cup \{t, 0\} \} \\
= \;& \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup (\psi + t) \cup (\pi + u) \\
& \mid \psi \in \llbracket x \rrbracket, \pi \in \llbracket y \rrbracket, (\psi_1 + t) \cup (\pi_1 + u) \subset (\psi_2 + t) \cup (\pi_2 + u) \cup \{t, 0\}, \boldsymbol{u < t} \} \\
\cup \;& \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup (\psi + t) \cup (\pi + u) \\
& \mid \psi \in \llbracket x \rrbracket, \pi \in \llbracket y \rrbracket, (\psi_1 + t) \cup (\pi_1 + u) \subset (\psi_2 + t) \cup (\pi_2 + u) \cup \{t, 0\}, \boldsymbol{u \geq t} \} \\
=^6 \;& \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup (\psi + t) \cup \pi \mid \psi \in \llbracket x \rrbracket, \pi \in \llbracket y \rrbracket, (\psi_1 + t) \cup \pi_1 \subset (\psi_2 + t) \cup \pi_2 \cup \{t, 0\} \} \\
\cup \;& \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup (\psi + t) \cup (\pi + t + v) \\
& \mid \psi \in \llbracket x \rrbracket, \pi \in \llbracket y \rrbracket, (\psi_1 + t) \cup (\pi_1 + t + v) \subset (\psi_2 + t) \cup (\pi_2 + t + v) \cup \{t, 0\}, \boldsymbol{v \in T} \} \\
= \;& \{ \omega \cup \pi \mid \omega \in \llbracket t{\cdot}x \rrbracket, \pi \in \llbracket y \rrbracket, \omega_1 \cup \pi_1 \subset \omega_2 \cup \pi_2 \cup \{0\} \} \\
\cup \;& \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup \Big(\big(\psi \cup (\pi + v) \big) + t \Big) \\
& \mid \psi \in \llbracket x \rrbracket, \pi \in \llbracket y \rrbracket, \big(\psi_1 \cup (\pi_1 + v) \big) + t \subset \Big(\big(\psi_2 \cup (\pi_2 + v) \cup \{0\} \big) + t \Big) \cup \{0\}, \boldsymbol{v \in T} \} \\
= \;& \{ \sigma \mid \sigma \in \llbracket t{\cdot}x \curlywedge y \rrbracket, \sigma_1 \subset \sigma_2 \cup \{0\} \} \\
\cup \;& \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup \big((\psi \cup \rho) + t \big) \mid \psi \in \llbracket x \rrbracket, \rho \in \llbracket \Delta y \rrbracket, (\psi_1 \cup \rho_1) + t \subset (\psi_2 \cup \rho_2 \cup \{0\}) + t \} \\
= \;& \llbracket [t{\cdot}x \curlywedge y] \rrbracket \\
\cup \;& \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup \big((\psi \cup \rho) + t \big) \mid \psi \in \llbracket x \rrbracket, \rho \in \llbracket \Delta y \rrbracket, \psi_1 \cup \rho_1 \subset (\psi_2 \cup \rho_2) \cup \{0\} \} \\
= \;& \llbracket [t{\cdot}x \curlywedge y] \rrbracket \cup \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup (\sigma + t) \mid \sigma \in \llbracket x \curlywedge \Delta y \rrbracket, \sigma_1 \subset \sigma_2 \cup \{0\} \} \\
= \;& \llbracket [t{\cdot}x \curlywedge y] \rrbracket \cup \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup (\sigma + t) \mid \sigma \in \llbracket [x \curlywedge \Delta y] \rrbracket \} \\
= \;& \llbracket [t{\cdot}x \curlywedge y] \rrbracket \cup \llbracket t{\cdot} [x \curlywedge \Delta y] \rrbracket \\
= \;& \llbracket [t{\cdot}x \curlywedge y] \cup t{\cdot} [x \curlywedge \Delta y] \rrbracket
\end{aligned}
$$

$\llbracket [\varepsilon] \rrbracket = \{ \langle \varnothing ; 0 ; \varnothing \rangle \mid \{0\} \subset \varnothing \cup \{0\} \} = \llbracket \varepsilon \rrbracket$

Completeness w.r.t. Δ-free terms

In this section, we will prove completeness of the axiomatic system for Δ-free terms. The normal form theorem allows restriction to Δ-free normal forms, which, for example, arise from normalizing arbitrary terms of the form $[P]$. Each choice-free subterm of such a term corresponds uniquely to a set of only one schedule.

Lemma 1. Let N be a term of the form

$$N \;::=\; a(t)^+ \curlywedge N \;\mid\; t{\cdot}\, N \;\mid\; \varepsilon, \text{ where } t \neq 0.$$

Then $|\, \llbracket N \rrbracket \,| = 1$.

Proof by structural induction:

$$
\begin{aligned}
&|\, \llbracket \varepsilon \rrbracket \,| \;=\; |\, \{ \langle \varnothing ; 0 ; \varnothing \rangle \} \,| \;=\; 1. \\
&\llbracket t{\cdot}\, N \rrbracket \;=\; \{ \langle \varnothing ; 0 ; \{t\} - \{0\} \rangle \cup (\sigma + t) \mid \sigma \in \llbracket N \rrbracket \}.
\end{aligned}
$$

So if $|\, \llbracket N \rrbracket \,| = 1$, then $|\, \llbracket t{\cdot}\, N \rrbracket \,| = 1$.
The case $a(t)^+ \curlywedge N$ proceeds likewise.

---

[4] Since $0 \in \rho_1$ and $0 \notin \rho_2$, we have $u \in (\rho_1 + u) \subset (\rho_2 + u) \cup \{0\}$ but $u \notin \rho_2 + u$, hence $u \in \{0\}$.

[5] Because always $0 \in \rho_1$.

[6] Since $0 \in \pi_1$ and $0 \notin \pi_2$, we have $u \in (\pi_1 + u) \subset (\psi_2 + t) \cup (\pi_2 + u) \cup \{t, 0\}$ but $u \notin (\psi_2 + t) \cup (\pi_2 + u) \cup \{t\}$ provided $u < t$, hence $u = 0$. Otherwise write $u = t + v$.

Lemma 2. Let N, M be of the form stipulated above, then

$$\llbracket N \rrbracket = \llbracket M \rrbracket \ \Rightarrow\ \text{A4, A5} \vdash N = M.$$

Proof by structural induction:

$$N = a_1(t_1)^+ \curlywedge a_2(t_2)^+ \curlywedge \ldots \curlywedge a_n(t_n)^+ \curlywedge N' \text{ with } N' = \varepsilon \text{ or } N' = t\cdot N'' \text{ with } t>0, \text{ and}$$
$$M = b_1(u_1)^+ \curlywedge b_2(u_2)^+ \curlywedge \ldots \curlywedge b_m(u_m)^+ \curlywedge M' \text{ with } M' = \varepsilon \text{ or } M' = u\cdot M'' \text{ with } u>0.$$

Let σ be the unique schedule in $\llbracket N\rrbracket = \llbracket M\rrbracket$. We have $N' = \varepsilon \Leftrightarrow \sigma_2 = \emptyset \Leftrightarrow M' = \varepsilon$.

In case $N' = M' = \varepsilon$ we have

$$\sigma_0 = \{\!\{(a_1, t_1, 0), (a_2, t_2, 0), \ldots, (a_n, t_n, 0)\}\!\} = \{\!\{(b_1, u_1, 0), (b_2, u_2, 0), \ldots, (b_m, u_m, 0)\}\!\}.$$

Hence $\{\!\{\, a_1(t_1)^+, a_2(t_2)^+, \ldots, a_n(t_n)^+ \,\}\!\} = \{\!\{\, b_1(u_1)^+, b_2(u_2)^+, \ldots, b_m(u_m)^+ \,\}\!\}$, so in particular $n = m$. It follows that A4, A5 $\vdash N = M$.

In case $N' = t\cdot N''$ and $M' = u\cdot M''$ we have

$$\llbracket N \rrbracket = \{ \langle (a_1, t_1, 0), (a_2, t_2, 0), \ldots, (a_n, t_n, 0)\ ;\ 0\ ;\ t \rangle \cup (\sigma{+}t) \mid \sigma\in \llbracket N''\rrbracket \} \text{ and}$$
$$\llbracket M \rrbracket = \{ \langle (b_1, u_1, 0), (b_2, u_2, 0), \ldots, (b_m, u_m, 0)\ ;\ 0\ ;\ u \rangle \cup (\rho{+}u) \mid \rho\in \llbracket M''\rrbracket \}.$$

Again it follows that $\{\!\{\, a_1(t_1)^+, a_2(t_2)^+, \ldots, a_n(t_n)^+ \,\}\!\} = \{\!\{\, b_1(u_1)^+, b_2(u_2)^+, \ldots, b_m(u_m)^+ \,\}\!\}$, so in particular $n = m$. Furthermore, we have $u = t$ because t is the least anchor point in the unique schedule of $\llbracket N\rrbracket$, and so is u for $\llbracket M\rrbracket$. Finally, $\llbracket N''\rrbracket = \llbracket M''\rrbracket$, and therefore by induction A4, A5 $\vdash N'' = M''$. Hence A4, A5 $\vdash N = M$.

Theorem. Let K, L be Δ-free normal forms with $\llbracket K\rrbracket = \llbracket L\rrbracket$ then

$$\text{A1, \ldots, A5} \vdash K = L.$$

Proof: Using A1, A2, A3, it suffices to show that for every choice-free subterm N of K there is a choice-free subterm M of L such that A4, A5 $\vdash N = M$ (and vice versa). By lemma 1, $\llbracket N\rrbracket = \{\sigma\}$, so $\sigma \in \llbracket K\rrbracket = \llbracket L\rrbracket$. Again by lemma 1, there is a choice-free subterm M of L such that $\llbracket M\rrbracket = \{\sigma\} = \llbracket N\rrbracket$. Now, by lemma 2, A4, A5 $\vdash$ $N = M$.

Anchor Form

For every t-free term $[P]$ we can eliminate $[.]$ without introducing time offsets.

$$P \text{ is } t\text{-free} \ \Rightarrow\ \exists Q\colon Q = [P] \text{ where } Q \text{ contains neither } [.] \text{ nor } t\cdot$$

We call this the *anchor form* of a term because all schedules are given by just anchoring actions to the endpoints of others. The GANTT chart can be drawn easily from this form.

Using a straightforward structural induction, one can establish that for every schedule σ of a t-free term we have

$$\forall t\in\sigma_2\colon \exists(a, u, v)\in\sigma_0\colon t = u + v \text{ with } u > 0. \qquad (*)$$

Furthermore, employing N2, Z2 and again N2, we obtain

$$t\cdot (u\cdot x \curlywedge v\cdot y) \ =\ t\cdot \varepsilon \curlywedge (t{+}u)\cdot x \curlywedge (t{+}v)\cdot y.$$

Every term $[P]$ can be written in a Δ-free normal form, which, using structural induction, Z1 and the law above, can be converted to a sum of terms of the form

$$u_1\cdot a_1(t_1)^+ \curlywedge u_2\cdot a_2(t_2)^+ \curlywedge \ldots \curlywedge u_k\cdot a_k(t_k)^+ \curlywedge u_{k+1}\cdot \varepsilon \curlywedge u_{k+2}\cdot \varepsilon \curlywedge \ldots \curlywedge u_n\cdot \varepsilon.$$

Assuming P is t-free, (*) implies that for every $u_i > 0$ there exists a $j \in \{1, \ldots k\}$ such that $u_i = u_j + t_j$ and $t_j > 0$. We will remove step by step the time offsets $u_i\cdot$ in the term above, starting with the largest, until only time offsets $0\cdot$ remain. The intermediate terms encountered during this process will always remain in the form

$$u_1\cdot a_1(t_1)^+ \curlywedge u_2\cdot a_2(t_2)^+ \curlywedge \ldots \curlywedge u_k\cdot a_k(t_k)^+ \curlywedge u_{k+1}\cdot N_{k+1} \curlywedge u_{k+2}\cdot N_{k+2} \curlywedge \ldots \curlywedge u_n\cdot N_n \qquad (**)$$

where for every $u_i > 0$ there exists a $j \in \{1, \ldots k\}$ such that $u_i = u_j + t_j$, $t_j > 0$ and the N_i contain neither time offsets nor occurrences of $[.]$ or Δ.

Let $v > 0$ be the maximum of $\{ u_1, u_2, \dots u_n \}$, and let $j \in \{1, \dots k\}$ be such that $v = u_j + t_j$ and $t_j > 0$. Using A4-5, we write all subterms $u_i \cdot a_i(t_i)$ and $u_i \cdot N_i$ with $u_i = v$ as well as $u_j \cdot a_j(t_j)$ at the right of the formula, thus obtaining an expression of the form

$$u'_1 \cdot a'_1(t'_1)^+ \curlywedge u'_2 \cdot a'_2(t'_2)^+ \curlywedge \dots \curlywedge u'_l \cdot a'_l(t'_l)^+ \curlywedge u'_{l+1} \cdot N'_{l+1} \curlywedge u'_{l+2} \cdot N'_{l+2} \curlywedge \dots \curlywedge u'_m \cdot N'_m \curlywedge K$$

with $K = u_j \cdot a_j(t_j)^+ \curlywedge v \cdot M_1 \curlywedge v \cdot M_2 \curlywedge \dots \curlywedge v \cdot M_h$. Using axioms N2 and Z1, K can be rewritten as $u_j \cdot a_j(t_j)^+ \curlywedge v \cdot (M_1 \curlywedge M_2 \curlywedge \dots \curlywedge M_h)$ and, with N1, as $u_j \cdot a_j(t_j) \cdot (M_1 \curlywedge M_2 \curlywedge \dots \curlywedge M_h)$. This brings the term again in the form (**) and we can continue with the next-largest member of $\{ u_1, u_2, \dots u_n \}$. In the end, all remaining u_i's will be 0 and Z1 turns the term into the required form.

Example

Consider the term $[\ a;\ (b \parallel c) \parallel\ d\ ]$ where all actions consume one unit of time. Transformation of this term into its anchor form yields 19 possible schedules. They are shown in fig. 5 grouped into the three basic orders "*a*; (*b* || *c*)", "*a*; *b*; *c*" and "*a*; *c*; *b*" with the independent action *d* attached to an anchor point or inserted in front of or in between the other actions.

This example might suggest that it is possible to convert a *t*-free term [*P*] into its anchor form without considering the duration of its actions. However, this is not possible in general: the anchor form of the term $[\ a(1);\ b(3) \parallel c(2);\ d(1)\ ]$ that has been calculated in a previous example contains the schedule $a(1) \curlywedge c(2) \cdot (\ b(3) \curlywedge d(1)\ )$, but not $a(1) \cdot (\ b(3) \curlywedge d(1)\ ) \curlywedge c(2)$.

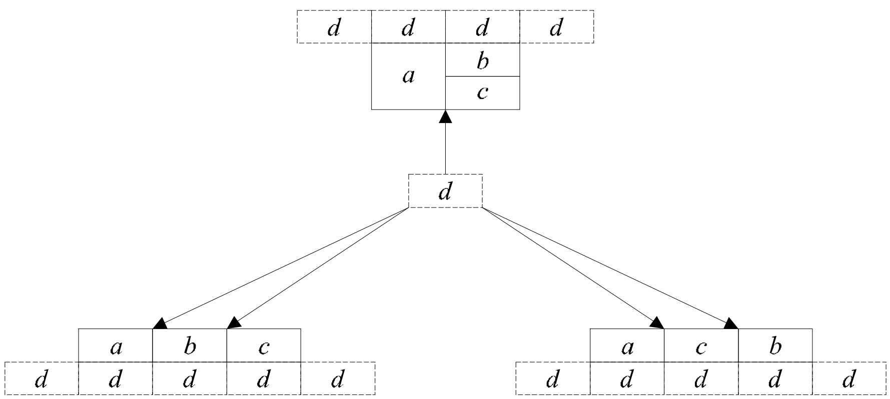

Fig. 5. All schedules

Term Rewriting

We show that modulo associativity and commutativity of choice and Hühnerfuß our axioms, when read from left to right, form a confluent and (strongly) normalizing rewrite system, provided that we add the rewrite rule $[a(t)^+] \rightarrow a(t)^+$ (which is derivable from the axiom system). If we drop the rewrite rule A3 (the idempotence of choice), the normal forms are exactly the normal forms in the sense of our normal form theorem, except that all subterms $a(t)^+ \curlywedge \varepsilon$ are replaced by $a(t)^+$ (so $a(t)^+$ is a normal form for the rewrite system). In the presence of A3 the same normal forms apply, but without pairs of summands that are equal modulo A4 and A5.

It is easy to check that such normal forms allow no further rewriting; the proof of the normal form theorem (but omitting the reverse application of Z2) shows essentially that all other terms can be rewritten into a normal form. Thus the rewrite system is normalizing. Here we omit the proof that it is even strongly normalizing; however, it involves only standard term rewriting techniques.

Confluence for terms of the form [*P*] follows immediately from the proof of our completeness theorem for such terms (because only the axioms A1-5 are involved in the completeness proof for Δ-free normal forms). Confluence for the entire rewrite system can be established by the Knuth-Bendix method, skipping pairs of overlapping redexes that lay within a term of the form [P]. In the table 2, we review the relevant pairs of overlapping redexes.

A3 with D1, D2 and D3	trivial
Z1 with D1 and N2 (in two ways)	trivial
Z2 with D3 and Z2	trivial
D1 with N2 (in two ways)	trivial with D3
D2 with N3 (in two ways)	trivial with D3
D3 with D3	trivial
N2 with itself	OK
N3 with itself	OK

Table 2. Overlapping redexes

An implementation of the system without A3 can be found at the URL

`http://www.uni-koblenz.de/~rittgen/SA.html.`

This applet generates the normal form for the axiom system given any term of your chosing.

Conclusion

In this paper, we have proposed a calculus for durational actions, namely scheduling algebra. The constants and operators of this algebra allow to specify process terms as a multi-tree order over actions with a certain duration. A normal form exists for each term which for Δ-free terms is unique modulo A1-5. Using the axioms of this calculus, the set of all efficient schedules [P] can be "computed" for any process P without tight sequencing. The operator [.] does so by generating a subterm (schedule) for any possible attachment of actions to anchor points, thus eliminating all time delays Δ.

We established soundness of the axiomatic system w.r.t. a schedule semantics and showed that the algebra is complete for Δ-free terms (sets of schedules), such as arise from terms of the form [P].

Currently, our calculus is designed for multi-tree precedence. To extend it to arbitrary orders (with more than one predecessor to an action), synchronization must be present in the algebra (as follows from the existence of partial orders which are not "series-parallel" [8]). This is left for future research.

References

1. Aceto, L.; Murphy, D.: *Timing and Causality in Process Algebra*, Acta Informatica 33, 1996, pp. 317-350
2. Baeten, J.C.M.; Bergstra, J.A.: *Real Time Process Algebra*, Formal Aspects of Computing 3(2), 1991, pp. 142-188
3. Bergstra, J.A.; Klop, J.W.: *Process Algebra for Synchronous Communication*, Information and Control 60, 1984, pp. 109-137
4. Chen, X.J.; Corradini, F.; Gorrieri, R.: *A Study on the Specification and Verification of Performance Properties*, Proceedings of AMAST '96, Lecture Notes in Computer Science 1101, Springer, Berlin, 1996, pp. 306-320
5. Clark, W.: *The Gantt Chart: A Working Tool of Management*, Pitman, London, 1935
6. Corradini, F.: *On Performance Congruences for Process Algebras*, Information and Computation 145(2), 1998, pp. 191-230
7. French, S.: *Sequencing and Scheduling: An Introduction to the Mathematics of the Job-Shop*, Ellis Horwood, Chichester, 1982
8. Gischer, J.L.: *The Equational Theory of Pomsets*, Theoretical Computer Science 61, 1988, pp. 199-224
9. Gorrieri, R.; Roccetti, M.; Stancampiano, E.: *A Theory of Processes with Durational Actions*, Theoretical Computer Science 140(1), 1995, pp. 73-94
10. Hoare, C.A.R.: *Communicating Sequential Processes*, Prentice Hall, Englewood Cliffs, 1985
11. Milner, R.: *A Calculus of Communicating Systems*, Lecture Notes in Computer Science 92, Springer, Berlin, 1980
12. Moller, F.; Tofts, C.: *A Temporal Calculus of Communicating Systems*, in: Baeten, J.C.M.; Klop, J.W.: Proceedings of CONCUR '90, Lecture Notes in Computer Science 458, Springer, Berlin, 1990, pp. 401-415
13. Reed, G.M.; Roscoe, A.W.: *A Timed Model for Communicating Sequential Processes*, Theoretical Computer Science 58, 1988, pp. 249-261