




# Scheduling Processes Without Sudden Termination

Johann Eder<sup>(✉)</sup>, Marco Franceschetti, and Josef Lubas

Department of Informatics-Systems, Universität Klagenfurt, Klagenfurt, Austria  
{johann.eder,marco.franceschetti,josef.lubas}@aau.at

**Abstract.** Dynamic controllability is the most general criterion to guarantee that a process can be executed without time failures. However, it admits schedules with an undesirable property: starting an activity without knowing its deadline. We analyze the specific constellations of temporal constraints causing such a sudden termination. Consequently, we introduce the somewhat stricter notion of semi-dynamic controllability, and present necessary and sufficient conditions to guarantee that a process can be executed without time failures and without sudden termination. A sound and complete algorithm for checking whether a process is semi-dynamically controllable complements the approach.

**Keywords:** Process scheduling · Contingent durations · Sudden termination · Controllability

## 1 Introduction

Modeling and verification of the temporal aspects of a business process are crucial for process management. Modeling temporal aspects includes defining deadlines, durations, and other temporal constraints [5]. Verification of the temporal qualities aims at determining, whether a given process model meets certain quality criteria, in particular, whether time failures can be avoided by defining adequate schedules for the dispatching of activities.

In recent years, there has been increasing awareness on the distinction between activities whose duration is under the control of an agent, and activities whose duration cannot be controlled, but merely observed at run time [22]. These uncontrollable durations are called *contingent*. A good example for activities with contingent durations is bank money transfers within the EU, which are guaranteed to take between 1 and 4 working days, but the client cannot control the actual duration. In a similar way, a service contract might covenant a visit by a technician within 24 hours but it is not controllable by the client when the technician will actually appear.

The existence of contingent activities in processes led to the formulation of *dynamic controllability* [7, 19, 22] as preferred criterion for temporal correctness of processes. Dynamic controllability requires the existence of a dynamic schedule

(execution strategy), which assigns timestamps for starting and finishing activities in a reactive manner in response to the observation of the actual durations of contingent activities at run time. Dynamic controllability is the most relaxed notion for guaranteeing that a process controller is able to steer the execution satisfying all temporal constraints. Consequently, several techniques have been developed to check the dynamic controllability of a process [3, 19].

Nevertheless, dynamic controllability might admit processes where each admissible dynamic schedule requires some activities to start, without knowing yet, when they need to complete. This leads to the subsequent *sudden termination* scheduling of their end-event. In Sect. 2 we give a precise demonstration and an example of this phenomenon. While for some activities this poses no problem (e.g. for waiting tasks), for other non-contingent activities a sudden termination is highly undesirable, unacceptable, or even impossible, in particular, for activities involving human actors or invoking uninterruptible external services [10, 12]. We call such activities *semi-contingent*, i.e. their duration between minimum and maximum duration can be chosen by the process controller but only until the activity starts. In Sect. 2 we give some examples for semi-contingent activities and the sudden termination problem.

The research question we address here is: *how to determine, whether a given process can be scheduled without the risk of a sudden termination of a task?*

Here we show that sudden termination can be identified by the presence of specific constraint patterns, and propose a technique to check, whether it is possible to (dynamically) schedule a process with the guarantee that no sudden termination will be forced at run time.

The contributions of this paper are the following:

- The discovery and characterization of the problem of sudden termination, which might arise in dynamically controllable processes
- The introduction of the notion of semi-contingent activities to model relevant temporal characteristics of activities
- The identification and characterization of patterns of temporal constraints and conditions, which are sufficient and necessary for the problem of sudden termination
- The definition of semi-dynamic controllability, which specializes the traditional notion of dynamic controllability to address semi-contingent activities and sudden termination
- A procedure to check semi-dynamic controllability

These results contribute to the development of a comprehensive framework to support the design, modeling, and analysis of business processes at design time and to monitor the time-aware execution of business processes at run-time.

The remainder of this paper is structured as follows: in Sect. 2 we illustrate the problem with the help of examples. In Sect. 3 we introduce a lean process model which allows the formulation of the problem, show how a specific pattern can induce the problem, and show how to solve it. In Sect. 4 we provide an implementation of a checking procedure. In Sect. 5 we discuss related works, and in Sect. 6 we draw conclusions.

## 2 Semi-contingent Activities and the Sudden Termination Problem

Dynamic controllability of processes distinguishes two kinds of activities: contingent and non-contingent activities. A process controller striving to meet all temporal constraints can control the duration of non-contingent activities but can only observe and not influence the duration of contingent activities. Dynamic controllability implicitly assumes that a non-contingent activity can be terminated spontaneously at any time (between minimum and maximum duration) without any earlier notice. We nickname this phenomenon as *sudden termination*. Such a behavior may be undesirable, unacceptable, or even impossible. Frequently, one can control the duration of an activity only until this activity starts. We call such activities *semi-contingent*. For an example, we could deliver a talk on temporal constraints for any time between 10 min and 2 hours. But we need to know beforehand for how long we can talk otherwise we would have to stop maybe right after the introduction. Other examples include:

- Ship products with express delivery or regular delivery
- Prepare a meal: from quick lunch to 4 course gourmet dinner
- Apply an expensive quick test or a budget slow test
- Assign more or less people to finish a task earlier or later
- etc. etc.

We explain the sudden termination problem with a small example. Figure 1 shows a simplified procurement process. The controller may choose the duration of semi-contingent task  $A$  for arranging shipment of goods between 5 and 7 days.  $A$  is bound to a contingent task  $R$  of receiving payment which lasts between 3 and 5 days. A lower-bound constraint demands that the end of  $A$  is at least 3 days after the end of  $R$  to allow a customer to cancel the order. The upper-bound constraint states that  $A$  has to finish within 4 days after  $R$  to guarantee timely delivery. Now, let's assume  $R$  should start at time point 10 and hence ends anytime between 13 and 15. Is there a choice for the start time and duration of  $A$  satisfying the constraints? After some trials you see: no!  $A$  might e.g. start at 12 and end between 17 and 19, but 19 is too late, if  $R$  ends at 13, and 17 is too early, if  $R$  ends at 15. And at 12 the end of  $R$  is unknown.

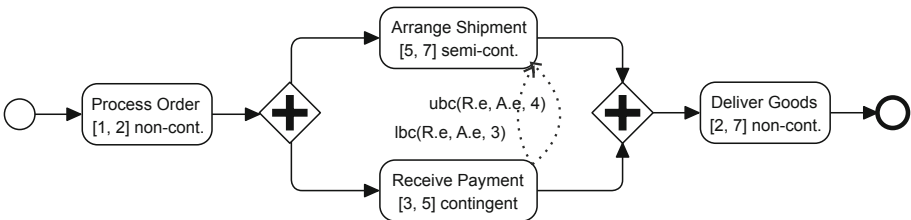


Fig. 1. Example process with a sudden termination problem

It is easy to see that it is not possible to determine a duration for  $A$  such that for all possible durations of  $R$  the constraints are satisfied. The end of the ongoing activity  $A$  can only be scheduled, when the end of  $R$  has been observed.

How should we treat semi-contingent activities for checking dynamic controllability? If we treat them as non-contingent activities, we risk sudden termination. If we treat them as contingent activities, we are unnecessarily strict and reject processes, which could be scheduled without sudden termination. Therefore, we introduce the notions of *semi-contingent activities* and *semi-dynamic controllability* to adequately deal with such activities.

### 3 Process Model and Semi-dynamic Controllability

#### 3.1 Process Model with Temporal Constraints

For most general applicability, here we introduce a minimal process model, which is sufficient to capture the patterns for which sudden termination may occur.

We consider the most common control flow patterns: sequence, inclusive and disjunctive splits, and the corresponding joins. To avoid design flaws, and according to the current state-of-the-art in this field, we assume that processes are acyclic and block-structured [6].

We consider activity durations, process deadline, and upper- and lower-bound constraints between events (start and end of activities). We measure time in *chronons*, representing, e.g., hours, days, ..., which have domain the set of natural numbers and are on an increasing time axis starting at *zero*. A duration is defined as the distance between two time points on the time axis.

Finally, we distinguish between non-contingent, semi-contingent, and contingent activities. The duration of contingent activities, by their nature, cannot be controlled, thus it cannot be known, when they will actually terminate. The process controller may however control the duration of non-contingent activities at any time. This means in particular, that they are allowed to start with no knowledge about the time when they have to end, thus conceding to be suddenly terminated. Semi-contingent activities, in contrast, require to know, at their start time, when they must terminate: this means the process controller can set their duration until the activity starts.

**Definition 1 (Process Model).** A process  $P$  is a tuple  $(N, E, C, \Omega)$ , where:

- $N$  is a set of nodes  $n$  with  $n.type \in \{start, activity, xor - split, xor - join, par - split, par - join, end\}$ . Each  $n \in N$  is associated with  $n.s$  and  $n.e$ , the start and end event of  $n$ . From  $N$  we derive  $N^e = \bigcup \{n.s, n.e \mid n \in N\}$ .
- $E$  is a set of edges  $e = (n1, n2)$  defining precedence constraints.
- $C$  is a set of temporal constraints:
  - duration constraints  $d(n, n_{min}, n_{max}, dur) \forall n \in N$ , where  $n_{min}, n_{max} \in \mathbb{N}$ ,  $dur \in \{c, sc, nc\}$ , stating that  $n$  takes some time in  $[n_{min}, n_{max}]$ .  $n$  can be contingent ( $dur = c$ ), semi-contingent ( $dur = sc$ ), non-contingent ( $dur = nc$ );

- *upper-bound constraints*  $ubc(a, b, \delta)$ , where  $a, b \in \mathbb{N}^e$ ,  $\delta \in \mathbb{N}$ , requiring that  $b \leq a + \delta$ ;
  - *lower-bound constraints*  $lbc(a, b, \delta)$ , where  $a, b \in \mathbb{N}^e$ ,  $\delta \in \mathbb{N}$ , requiring that  $b \geq a + \delta$ .
- $\Omega \in \mathbb{N}$  is the process deadline.

We now define the temporal semantics of the process model.

### 3.2 Temporal Semantics

We define the temporal semantics of temporally constrained process definitions by defining which scenarios are valid. A scenario is a run of the process (a process instance) with timestamps, when each event (starting and ending of process steps) occurred. A scenario is *valid*, if it satisfies all temporal constraints.

**Definition 2 (Valid Scenario).** *Let  $P(N, E, C, \Omega)$  be a process model. Let  $\sigma$  be a scenario for  $P$ , assigning to each time point  $t$  a timestamp  $\bar{t}$ .  $\sigma$  is a valid scenario for  $P$  iff:*

1.  $\forall (n1, n2) \in E, \overline{n1.e} \leq \overline{n2.s}$ ;
2.  $\forall d(n, n_{min}, n_{max}, [c|sc|nc]) \in C, \overline{n.s} + n_{min} \leq \overline{n.e} \leq \overline{n.s} + n_{max}$ ;
3.  $\forall ubc(a, b, \delta) \in C, \overline{b} \leq \overline{a} + \delta$ ;
4.  $\forall lbc(a, b, \delta) \in C, \overline{a} + \delta \leq \overline{b}$ ;
5.  $\overline{end.e} \leq \overline{start.s} + \Omega$ .

A schedule for a process states when each activity should be started and terminated. If a schedule exists, we call the process controllable. Controllability is often considered too strict, as it would not admit situations where, e.g. the time-point for the start of an activity depends on the observed duration of preceding contingent activities.

Dynamic controllability requires the existence of a dynamic schedule (or dynamic execution strategy), where the decision about starting and ending activities can be made based on the timestamp of all earlier events.

There are several techniques for checking the dynamic controllability of processes. We use here the technique of mapping a process model to a Simple Temporal Network with Uncertainty (STNU) and apply constraint propagation techniques which are proven to be sound and complete for checking dynamic controllability [3]. We present this technique in Sect. 4.

In Sect. 2 we gave an example of a process which is dynamically controllable, but suffers from the problem of sudden termination of a semi-contingent activity.

In the next section we explore a pattern of constraints, which may lead to the sudden termination of an activity. We use this pattern to formulate a new notion of controllability, which is somewhat stricter than dynamic controllability, and introduce a technique to verify a process model for such a notion.

### 3.3 Constellations for a Sudden Termination Problem

For the following considerations we assume, that a process is dynamically controllable, i.e. there is a dynamic schedule, such that no constraint is violated. We use the term *condition* or *constraint* in a process model  $P$  in the sense that such a constraint is either explicitly stated in  $P$  or that it can be derived from the constraints in  $P$  (see Sect. 4 for techniques to derive implicit constraints). We use  $\Phi$  as the set of all implicit and explicit constraints valid in  $P$ .

We now characterize precisely what constitutes a sudden termination problem for a semi-contingent activity, and observe which conditions have to be satisfied, such that a sudden termination problem might occur. We distinguish between a sudden termination constellation (STC) and a sudden termination pattern (STP). In a STP, constraints can only be satisfied with sudden termination, while in the more general STC, it might depend on the controller, whether a sudden termination actually occurs. In a STP, sudden termination needs to happen, while in a STC which is not a STP sudden termination is avoidable.

A STC requires (at least) 2 constraints. So let us consider two activities, one contingent ( $C$ ) and one semi-contingent ( $S$ ). A sudden termination means that the admissible times for ending  $S$  depend on the observation of the end of  $C$  and this is not known, when  $S$  starts. This requires that there is a constraint between the end events of  $C$  and  $S$ .

A sudden termination only occurs, if  $C$  and  $S$  have to be executed concurrently, i.e. it is not possible that they are executed sequentially. This requires an additional constraint. When  $S.e$  can always be executed before  $C.e$ , it cannot depend on the observed duration of  $C$ . In a similar way, if  $S$  can always start after  $C$  has ended, the duration of  $C$  is already known at the start of  $S$ .

**Definition 3 (Sudden Termination Constellation and Pattern).** *Let  $P$  be a dynamically controllable process. Let  $S$  and  $C$  be activities in  $P$ , with the duration constraints  $d(S, S.d_{min}, S.d_{max}, sc)$  and  $d(C, C.d_{min}, C.d_{max}, c)$ . Let  $\Phi$  be the set of constraints in  $P$ .*

*$S$  and  $C$  are in a **sudden termination constellation (STC)** iff*

$$\forall C.s \exists S.s \forall S.d_{min} \leq S.d \leq S.d_{max} \exists C.d_{min} \leq C.d \leq C.d_{max} : \neg \Phi$$

*$S$  and  $C$  are in a **sudden termination pattern (STP)** iff*

$$\forall S.s, C.s \forall S.d_{min} \leq S.d \leq S.d_{max} \exists C.d_{min} \leq C.d \leq C.d_{max} : \neg \Phi$$

*We use the notation  $STC_{S,C}$  to indicate that  $S$  and  $C$  are in a sudden termination constellation, and  $STP_{S,C}$  for a sudden termination pattern.*

If there is a STP in a process, sudden termination cannot be avoided, without changing the process. If there is a STC but not an STP, the execution of a process can be (dynamically) scheduled in a way to both observe all constraints and avoid sudden termination. We are now interested, whether it is possible for a process controller to schedule without sudden termination, i.e. whether it is possible to avoid that a STC becomes a STP.

### 3.4 Semi-dynamic Controllability

The constellation of constraints which renders two activities in a sudden termination pattern is compatible with the dynamic controllability of a process, i.e. there exist processes which are *dc* but have a pair of activities in a *STP*. So dynamic controllability is not strict enough, in the sense that it allows semi-contingent activities to start without knowing when to end. We define a specialization of dynamic controllability which recognizes the need to know the required end time for a semi-contingent activity at its start time.

**Definition 4 (Semi-dynamic Controllability).** *Let  $P$  be a process.  $P$  is semi-dynamically controllable (sdc) iff  $P$  is *dc*, and  $\nexists S, C \in P.N: STP_{S,C}$ .*

Semi-dynamic controllability is stricter than dynamic controllability, but not as strict as controllability. With semi-dynamic controllability we require a process to be dynamically controllable, and no semi-contingent activity is involved in a sudden termination pattern.

### 3.5 Basic Sudden Termination Pattern

First we consider and discuss the most fundamental constellation for a sudden termination problem: the end events of a contingent activity  $C$  and a semi-contingent activity  $S$  are connected with one upper- and one lower-bound constraint:  $ubc(C.e, S.e, w)$ , and  $lbc(C.e, S.e, v)$ .

In this constellation, there is only one uncertainty, which cannot be controlled: the actual duration of  $C$ . The constraint  $ubc(C.e, S.e, w)$  requires that  $S.e \leq C.e + w$ . For actual durations  $S.d$ , resp.  $C.d$ , this requires that  $S.s + S.d \leq C.s + C.d + w$ . The constraint  $lbc(C.e, S.e, v)$  requires that  $C.e + v \leq S.e$ .

A *Sudden Termination Pattern 1 (STP-1)* is defined as follows: A contingent activity  $C$  and a semi-contingent activity  $S$ , with constraints  $ubc(C.e, S.e, w)$  and  $lbc(C.e, S.e, v)$  as above are in a *STP-1*, iff there is no way to schedule the start of  $S$  and the start of  $C$ , such that the value for the end of  $S$  can be fixed, when  $S$  starts.

**Definition 5 (Sudden Termination Pattern: STP-1).** *Let  $P$  be a dynamically controllable process. Let  $S$  and  $C$  be activities in  $P$ , with the duration constraints  $d(S, S.d_{min}, S.d_{max}, sc)$  and  $d(C, C.d_{min}, C.d_{max}, c)$ .*

*Let  $ubc(C.e, S.e, w)$  and  $lbc(C.e, S.e, v)$  be constraints in  $P$ .*

*$S$  and  $C$  are in *STP-1*, iff*

$\forall S.s, C.s \forall S.d_{min} \leq S.d \leq S.d_{max} \exists C.d_{min} \leq C.d \leq C.d_{max} :$   
 $S.s + S.d > C.s + C.d + w$ , or  $S.s + S.d < C.s + C.d + v$ .

### 3.6 Characterization of STP

In the following, we derive conditions to check, whether a *STP* might occur for a given pair of activities  $S$  and  $C$ .

General preconditions for the existence of a sudden termination problem is that the following constraints do not hold in  $P$ :  $S.s > C.e, C.s > S.e$ .

**Theorem 1.** *Let  $P$  be a dynamically controllable process. Let  $S$  be a semi-contingent activity, and  $C$  be a contingent activity in  $P$ , with duration constraints  $d(S, S.d_{min}, S.d_{max}, sc)$  and  $d(C, C.d_{min}, C.d_{max}, c)$ . Let  $ubc(C.e, S.e, w)$  and  $lbc(C.e, S.e, v)$  be constraints in  $P$ .*

*A sudden termination of  $S$  cannot occur, iff  $C.d_{max} + v \leq S.e - C.s \leq C.d_{min} + w$  holds in  $P$ .*

*Proof.* We show that  $C.d_{max} + v < S.e - C.s \leq C.d_{min} + w$  is a necessary and sufficient condition that the activities  $S$  and  $C$  are not in a STP.

*Necessary condition:* we show that if the condition does not hold, a sudden termination might occur. If the condition does not hold then  $\nexists S.s, S.d, C.s$  with  $C.d_{max} + v < S.e - C.s \leq C.d_{min} + w$ . This is only possible, if  $C.d_{max} + v > C.d_{min} + w$ .

We now assume that  $C$  and  $S$  are in a STP. This means that  $\forall C.s, S.s$  there is no  $S.d$  such that  $\forall C.d$  the constraints hold:  $S.s + S.d \leq C.s + C.d + w$  and  $S.s + S.d \geq C.s + C.d + v$ . Hence  $\nexists S.d$  such that  $\forall C.d: C.d + v \leq S.s + S.d - C.s \leq C.d + w$ . Which requires in particular, that  $\nexists S.d$  to satisfy  $C.d_{max} + v \leq S.e - C.s \leq C.d_{min} + w$ .

*Sufficient condition:* We show that if the inequality holds, sudden termination does not occur. We show that  $\exists C.s, S.s, S.d$  such that  $\forall C.d_{min} \leq C.d \leq C.d_{max}$  the constraints are satisfied, i.e.  $C.s + C.d + v \leq S.s + S.d \leq C.s + C.d + w$ , which holds since  $\forall C.d_{min} \leq C.d \leq C.d_{max}: C.d + v \leq C.d_{max} + v \leq S.s + S.d - C.s \leq C.d_{min} + w \leq C.d + w$ .  $\square$

This theorem can now be used to establish conditions that a process model has to fulfill, such that it is dynamically controllable and a STP cannot occur. In particular, we can show that a STP cannot occur, when the process model includes a particular lower-bound constraint resp. upper-bound constraint between the start of  $S$  and the start of  $C$ .

**Theorem 2.** *Let  $P$  be a process. Let  $S$  be a semi-contingent activity, and  $C$  be a contingent activity in  $P$  with duration constraints  $d(S, S.d_{min}, S.d_{max}, sc)$  and  $d(C, C.d_{min}, C.d_{max}, c)$ . Let  $ubc(C.e, S.e, w)$  be a constraint in  $P$ .*

*A sudden termination of  $S$  cannot occur, iff constraints  $lbc(C.s, S.e, C.d_{max} + v)$  and  $ubc(C.s, S.e, C.d_{min} + w)$  hold in  $P$ .*

*Proof.* The constraints express exactly the condition in Theorem 1. Hence in a process model  $P$  including these constraints  $S$  and  $C$  cannot be in a STP.  $\square$

This theorem helps us to develop a procedure to check whether a process is semi-dynamically controllable, resp. transforming a process with a STC such that it avoids a STP.

### 3.7 Checking Semi-dynamic Controllability

We are now ready to apply this result for checking, whether a sudden termination problem can be avoided in the execution of a process. For each ST-constellation



in a process  $P$  (a semi-contingent activity  $S$ , and a contingent activity  $C$  with duration constraints  $d(S, S.d_{min}, S.d_{max}, sc)$  and  $d(C, C.d_{min}, C.d_{max}, c)$ , and the constraints  $ubc(C.e, S.e, w)$  and  $lbc(C.e, S.e, v)$ ) we include 2 additional constraints:  $lbc(C.s, S.e, C.d_{max} + v)$  and  $ubc(C.s, S.e, C.d_{min} + w)$ . Then we check the resulting process for dynamic controllability. If it is dynamically controllable, then (with Theorem 2) it is also semi-dynamically controllable.

### 3.8 Further STP Constellations

After a detailed characterization of the basic constellation, we enumerate in Table 1 all possible constellations between a contingent and a semi-contingent activity which could cause a sudden termination problem.

The first column shows the constraints specifying the constellation, the second column states the necessary and sufficient condition which has to be fulfilled such that a sudden termination does not occur, and the third column the constraints to add for checking the process for semi-dynamic controllability. The proofs for these conditions and constraints follow the structure of the proofs for the basic constellation but for space reasons they cannot be repeated here.

**Table 1.** Sudden termination constellations, with conditions to avoid sudden termination, and constraints to add to check for semi-dynamic controllability.

Constraints	Condition to avoid sudden termination	Constraints to add
$ubc(C.e, S.e, w)$ $lbc(C.e, S.e, v)$	$C.d_{max} + v \leq S.e - C.s \leq C.d_{min} + w$	$lbc(C.s, S.e, C.d_{max} + v)$ $ubc(C.s, S.e, C.d_{min} + w)$
$ubc(C.e, S.e, w)$ $lbc(C.s, S.e, v)$	$v < S.e - C.s \leq C.d_{min} + w$	$lbc(C.s, S.e, v)$ $ubc(C.s, S.e, C.d_{min} + w)$
$ubc(C.s, S.e, w)$ $lbc(C.e, S.e, v)$	$C.d_{max} + v < S.e - C.s \leq w$	$lbc(C.s, S.e, C.d_{max} + v)$
$ubc(C.e, S.e, w)$ $lbc(C.s, S.s, v)$	$S.d_{min} + v < S.e - C.s \leq C.d_{min} + w$	$lbc(C.s, S.s, S.d_{min} + v)$ $ubc(C.s, S.e, C.d_{min} + w)$
$ubc(C.e, S.e, w)$ $ubc(S.s, C.e, v)$	$C.d_{max} + S.d_{min} - v < S.e - C.s \leq C.d_{min} + w$	$lbc(C.s, S.e, C.d_{max} + S.d_{min} - v)$ $ubc(C.s, S.e, C.d_{min} + w)$
$ubc(C.e, S.e, w)$ $ubc(S.s, C.s, v)$	$S.d_{min} - v < S.e - C.s \leq C.d_{min} + w$	$lbc(C.s, S.e, S.d_{min} - v)$ $ubc(C.s, S.e, C.d_{min} + w)$

## 4 Checking Semi-dynamic Controllability with STNUs

In this section we show how we can kill two birds with one stone, and meet the two requirements of (1) deriving all implicit constraints, and (2) checking whether a process is semi-dynamically controllable, by mapping process models into temporal networks.

(1) The temporal constraints causing a STC or STP need not necessarily to be explicitly stated in the process model, but may be implicitly induced by the explicit temporal constraints in the process model. Therefore, we need to compute the set of all (implicit) constraints for identifying all possible STCs.

(2) Checking whether a process model is semi-dynamically controllable, requires applying some dynamic controllability checking procedure, as per Theorem 2 and Sect. 3.7.

#### 4.1 Mapping to STNUs

In previous works [9] we showed how to check whether a process model, such as the process in Fig. 1, is dynamically controllable by mapping it into a equivalent STNU (Simple Temporal Network with Uncertainty) [19].

In a nutshell, a STNU is a directed graph, in which nodes represent time points and edges represent constraints between time points. A special time point *zero* marks the reference in time, after which all other time points occur. Edges can be non-contingent or contingent. Non-contingent edges represent constraints which can be enforced by the execution environment by assigning appropriate values to the time points. Contingent edges (also called links) represent constraints which are guaranteed to hold, but the corresponding time point assignments cannot be controlled, only observed.

We use the notation  $(A, B, \delta)$  for non-contingent edges from  $A$  to  $B$  with weight  $\delta$ , which require that  $B \leq A + \delta$ ; and  $(A^C, l, u, C)$  for contingent links between  $A^C$  and  $C$ , which state that  $C$  occurs some time between  $l$  and  $u$  after  $A^C$ . For a detailed formalization on STNUs, we refer to [19].

Figure 2 shows the STNU derived by mapping the process model of Fig. 1. Note that in Fig. 2 we adopted the usual STNU notation with contingent edges dashed, inverted w.r.t. non-contingent edges, and labeled with the contingent time point name. For a more compact presentation, in the figure we did not include nodes resulting from the mapping of the par-split and par-join.

#### 4.2 Checking Dynamic Controllability

Several techniques have been developed to check the dynamic controllability of STNUs. Most of these techniques, such as [3], build on deriving implicit constraints from the existing ones. In particular, they work by propagating the existing constraints based on certain sets of propagation rules.

Constraint propagation can be applied until: (1) no new implicit edges can be derived, or (2) a negative cycle (in the usual sense of graph theory) is found. If constraint propagation stops in case (1), then the STNU, hence its originating process, is dynamically controllable. In case (2), instead, a constellation of contradicting constraints exists, and the STNU is not dynamically controllable. As an example, one can verify, by applying constraint propagation, that the STNU in Fig. 2 is dynamically controllable, since no negative cycle can be derived.

The effect of applying a constraint propagation procedure is twofold. First, the procedure derives all implicit constraints which hold if the explicit constraints hold. Second, it determines whether a STNU is dynamically controllable. We use these results, in combination with the results of Sect. 3, to design a procedure for checking, whether a given process model is semi-dynamically controllable.

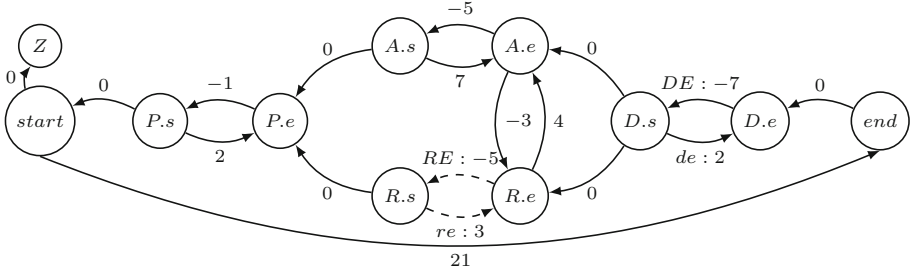


Fig. 2. STNU derived from the example process shown in Fig. 1.

### 4.3 Checking Semi-dynamic Controllability

Algorithm 1 shows the procedure we propose for checking the semi-dynamic controllability. A process model  $P$  is mapped into a STNU  $T$ . Additionally, we keep a data structure  $S_T$  containing STNU nodes representing semi-contingent activities, which is needed for identifying sudden termination patterns.

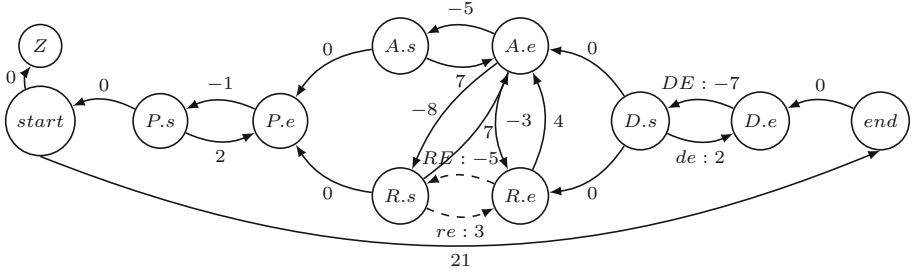
First,  $T$  is checked for dynamic controllability by applying a constraint propagation procedure  $check\_dc(T)$ , which as a side effect computes the closure of the set of constraints. If the procedure returns **True**, the process is *dc*. Then  $find\_stp(T, S_T)$  searches and returns all STPs. Then there is a loop (repeated as long as the network is *dc* and there are unresolved STPs) with three steps: (i) For each STP  $p$  found, edges corresponding to the constraints to resolve  $p$  to avoid a sudden termination problem are added to  $T$ . (ii) check for dynamic controllability and derive additional implicit constraints. (iii) search for unresolved STPs. If at the end of the iteration  $T$  remains *dc*, then it is also *sdc* and **True** is returned. One can verify (see the negative cycle introduced in Fig. 3 between  $R.s$  and  $A.e$ ) that the process of the running example is not *sdc*.

The correctness of the procedure trivially follows from the correctness of the existing constraint propagation procedures, and from the Theorems of Sect. 3.

## 5 Implementation and Evaluation

We implemented Algorithm 1, and performed experiments to evaluate its scalability.

We ran our experiments on a Windows 10 machine with an i7 CPU and 16GB of RAM. For the experiments we randomly generated a set 30 processes of different sizes in terms of number of process steps. We structured the test set into 5 subsets of processes: one for processes of size 10, one for 20, one for 30, one for 40, and one for 50. Each subset contained 10 processes. The smallest process contained 10 activities and 2 constraints, resulting in a STNU with 20 nodes. The largest process contained 50 activities and 10 constraints, resulting in a STNU with 100 nodes. We regard the range of process sizes used for the experiments as representative of most of the cases found in practical applications.



**Fig. 3.** STNU resulting from the application of Algorithm 1 to the process of Fig. 1.

We report the average measured execution times for the various process sizes in Fig. 4. On average, executing Algorithm 1 for a process of size 10 required 0.13s; for size 20, 0.83s; for size 30, 6.29s; for size 40, 18.94s; for size 50, 41.00s.

Our experiments showed that, despite the addition of new constraints to solve the STP, and the repeated execution of the dynamic controllability checking procedure, the required computation times are still acceptable for a design time check. Thus, we regard the proposed approach applicable for most practical applications which require design time checking for semi-dynamic controllability.

---

**Algorithm 1.** Check semi-dynamic controllability

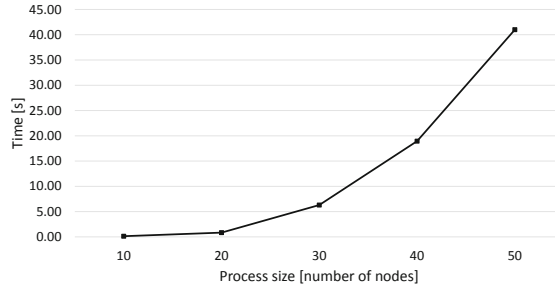
---

```

1: Input: Process  $P$ 
2:  $T := \text{map\_to\_STNU}(P)$ 
3:  $S_T := \text{get\_semicontingent\_nodes}(P, T)$ 
4: if ( $\neg \text{check\_dc}(T)$ ) then
5:   return False
6: else
7:    $STP := \text{find\_stp}(T, S_T)$ 
8:   while ( $STP \neq \emptyset$ ) do
9:      $p := \text{extract\_first}(STP)$ 
10:     $T := T \cup \text{compute\_resolving\_constraints}(p)$ 
11:    if ( $\neg \text{check\_dc}(T)$ ) then
12:      return False
13:    else
14:       $STP := STP \cup \text{find\_stp}(T, S_T)$ 
15:    end if
16:  end while
17: end if
18: return True

```

---



**Fig. 4.** Results of the evaluation.

## 6 Related Work

To the best of our knowledge, semi-contingent activities and hence semi-dynamic controllability have never been studied before. Related work, therefore, comprises of (a) formulation of temporal constraints in process models and (b) checking consistency and controllability of temporally constrained process models and (c) scheduling and monitoring of process execution.

General overviews of time management for business processes are provided in [5, 8, 11]. [16] presents a formalization of time patterns and their semantics, based on the analysis of recurring temporal constraints in process models, however, without considering semi-contingent activities.

Verification of deadlines and temporal constraints was subject of several previous works, such as [1, 4, 13, 18]. Similarly to our proposed approach based on STNUs, these approaches are based on network analysis, scheduling, and constraint networks. None of these works, however, addressed the problem of induced sudden termination of tasks.

More recent works combine the temporal aspects of a process definition with other constraining dimensions such as resource availability [23, 24]. While with our work we currently focus only on the temporal dimension, these works offer interesting possibilities to further investigate the problem of sudden termination with a holistic view on process models.

While we address the design time verification of process qualities here, considerable research (e.g., [14, 17, 20]) explored the pro-active monitoring of the compliance of process instances to their process model. However, to the best of our knowledge, all approaches to monitoring and compliance checking address the notion of satisfiability rather than dynamic controllability and do not consider the problem of sudden termination.

In contrast to all these approaches, process mining techniques [21] rely on the existence of a large number of cases (process logs) before they are able to provide scheduling and monitoring information. Thus they are not adequate for new or frequently changing processes, or processes with small number of instances [2].

For the implementation of our approach we map process definitions to Simple Temporal Networks with Uncertainty (STNUs) [19]. Considerable research

efforts have been devoted in the last decades both to developing different notions of controllability for temporal constraint networks, and to developing more expressive network models [15, 19, 22, 25]. With this work, we contribute to this field of research introducing a specialization of dynamic controllability to avoid a class of unwanted behaviors.

## 7 Conclusions

The presence of activities with uncontrollable duration requires a dynamic schedule to ensure that all temporal constraints are met. Dynamic scheduling, however, may entail not knowing, at activity start time, when an activity has to terminate. This leads to the sudden termination of an activity. Such a behavior is undesirable for a large number of activities.

With this paper we proposed semi-dynamic controllability as a new notion for temporal correctness. It requires a process to be dynamically controllable, and that the end time of activities, which cannot be interrupted, is known at their start time. We have shown how to verify whether a process is semi-dynamically controllable, based on patterns of temporal constraints. Our implementation of a verification procedure demonstrated that semi-dynamic controllability can be efficiently checked at design time. We regard semi-dynamic controllability as a desirable quality for process models, which process designers may easily check.

The results of the research reported here contribute to the development of a comprehensive set of tools supporting the design of business processes by checking relevant properties of process models at design time. The notions and techniques also provide the basis for monitoring of temporal properties of the process execution at run-time, enabling pro-active avoidance of time failures.

## References

1. Bettini, C., Wang, X., Jajodia, S.: Temporal reasoning in workflow systems. *Distrib. Parallel Databases* **11**(3), 269–306 (2002)
2. Breu, R., et al.: Towards living inter-organizational processes. In: 15th IEEE Conference on Business Informatics, pp. 363–366. IEEE (2013)
3. Cairo, M., Rizzi, R.: Dynamic controllability made simple. In: 24th International Symposium on Temporal Representation and Reasoning (TIME 2017), vol. 90 of LIPIcs, pp. 8:1–8:16 (2017)
4. Cardoso, J., Sheth, A., Miller, J., Arnold, J., Kochut, K.: Quality of service for workflows and web service processes. *J. Web Semant.* **1**(3), 281–308 (2004)
5. Cheikhrouhou, S., Kallel, S., Guermouche, N., Jmaiel, M.: The temporal perspective in business process modeling: a survey and research challenges. *SOCA* **9**(1), 75–85 (2014). <https://doi.org/10.1007/s11761-014-0170-x>
6. Combi, C., Gambini, M.: Flaws in the flow: the weakness of unstructured business process modeling languages dealing with data. In: Meersman, R., Dillon, T., Herrero, P. (eds.) OTM 2009. LNCS, vol. 5870, pp. 42–59. Springer, Heidelberg (2009). [https://doi.org/10.1007/978-3-642-05148-7\\_6](https://doi.org/10.1007/978-3-642-05148-7_6)

7. Combi, C., Posenato, R.: Controllability in temporal conceptual workflow schemata. In: Dayal, U., Eder, J., Koehler, J., Reijers, H.A. (eds.) BPM 2009. LNCS, vol. 5701, pp. 64–79. Springer, Heidelberg (2009). [https://doi.org/10.1007/978-3-642-03848-8\\_6](https://doi.org/10.1007/978-3-642-03848-8_6)
8. Combi, C., Pozzi, G.: Temporal conceptual modelling of workflows. In: Song, I.-Y., Liddle, S.W., Ling, T.-W., Scheuermann, P. (eds.) ER 2003. LNCS, vol. 2813, pp. 59–76. Springer, Heidelberg (2003). [https://doi.org/10.1007/978-3-540-39648-2\\_8](https://doi.org/10.1007/978-3-540-39648-2_8)
9. Eder, J., Franceschetti, M., Köpke, J.: Controllability of business processes with temporal variables. In: Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, pp. 40–47. ACM (2019)
10. Eder, J., Liebhart, W.: Workflow transactions. In: Workflow Handbook 1997, pp. 195–202. Wiley (1997)
11. Eder, J., Panagos, E., Rabinovich, M.: Workflow time management revisited. Seminal Contributions to Information Systems Engineering, pp. 207–213. Springer, Heidelberg (2013). [https://doi.org/10.1007/978-3-642-36926-1\\_16](https://doi.org/10.1007/978-3-642-36926-1_16)
12. Franceschetti, M., Eder, J.: Dynamic service binding for time-aware service compositions. In: 2019 IEEE 23rd International Enterprise Distributed Object Computing Workshop (EDOCW), pp. 146–151. IEEE (2019)
13. Guermouche, N., Godart, C.: Timed model checking based approach for web services analysis. In: ICWS 2009. IEEE International Conference on Web Services, pp. 213–221. IEEE (2009)
14. Hashmi, M., Governatori, G., Lam, H.-P., Wynn, M.T.: Are we done with business process compliance: state of the art and challenges ahead. Knowl. Inf. Syst. **57**, 1–55 (2018). <https://doi.org/10.1007/s10115-017-1142-1>
15. Lanz, A., Posenato, R., Combi, C., Reichert, M.: Controlling time-awareness in modularized processes. In: Schmidt, R., Guédria, W., Bider, I., Guerreiro, S. (eds.) BPMDS/EMMSAD -2016. LNBIP, vol. 248, pp. 157–172. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-39429-9\\_11](https://doi.org/10.1007/978-3-319-39429-9_11)
16. Lanz, A., Reichert, M., Weber, B.: Process time patterns: a formal foundation. Inf. Syst. **57**, 38–68 (2016)
17. Ly, L.T., Maggi, F.M., Montali, M., Rinderle-Ma, S., van der Aalst, W.M.: Compliance monitoring in business processes: functionalities, application, and tool-support. Inf. Syst. **54**, 209–234 (2015)
18. Marjanovic, O., Orlowska, M.E.: On modeling and verification of temporal constraints in production workflows. Knowl. Inf. Syst. **1**(2), 157–192 (1999). <https://doi.org/10.1007/BF03325097>
19. Morris, P.H., Muscettola, N.: Temporal dynamic controllability revisited. In: Proceedings of AAAI, pp. 1193–1198 (2005)
20. Pichler, H., Wenger, M., Eder, J.: Composing time-aware web service orchestrations. In: van Eck, P., Gordijn, J., Wieringa, R. (eds.) CAiSE 2009. LNCS, vol. 5565, pp. 349–363. Springer, Heidelberg (2009). [https://doi.org/10.1007/978-3-642-02144-2\\_29](https://doi.org/10.1007/978-3-642-02144-2_29)
21. van der Aalst, W.M., Schonenberg, M., Song, M.: Time prediction based on process mining. Inf. Syst. **36**(2), 450–475 (2011)
22. Vidal, T.: Handling contingency in temporal constraint networks: from consistency to controllabilities. J. Exp. Theor. Artif. Intell. **11**(1), 23–45 (1999)
23. Watahiki, K., Ishikawa, F., Hiraishi, K.: Formal verification of business processes with temporal and resource constraints. In: 2011 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 1173–1180. IEEE (2011)

24. Zavatteri, M., Combi, C., Viganò, L.: Resource controllability of workflows under conditional uncertainty. In: Di Francescomarino, C., Dijkman, R., Zdun, U. (eds.) BPM 2019. LNBP, vol. 362, pp. 68–80. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-37453-2\\_7](https://doi.org/10.1007/978-3-030-37453-2_7)
25. Zavatteri, M., Viganò, L.: Conditional simple temporal networks with uncertainty and decisions. *Theoret. Comput. Sci.* **797**, 77–101 (2019)