Time prediction on multi-perspective declarative business processes

Andres Jimenez-Ramirez[1] · **Irene Barba**[1] · **Juan Fernandez-Olivares**[2] · **Carmelo Del Valle**[1] · **Barbara Weber**[3]

Abstract Process-aware information systems (PAISs) are increasingly used to provide flexible support for business processes. The support given through a PAIS is greatly enhanced when it is able to provide accurate time predictions which is typically a very challenging task. Predictions should be (1) multi-dimensional and (2) not based on a single process instance. Furthermore, the prediction system should be able to (3) adapt to changing circumstances and (4) deal with multi-perspective declarative languages (e.g., models which consider time, resource, data and control flow perspectives). In this work, a novel approach for generating time predictions considering the aforementioned characteristics is proposed. For this, first, a multi-perspective constraint-based language is used to model the scenario. Thereafter, an optimized enactment plan (representing a potential execution alternative) is generated from such a model considering the current execution state of the process instances. Finally, pre-dictions are performed by evaluating a desired function over this enactment plan. To evaluate the applicability of our approach in practical settings we apply it to a real process scenario. Despite the high complexity of the considered problems, results indicate that our approach produces a satisfactory number of good predictions in a reasonable time.

Keywords Flexible process-aware information systems · Time prediction · Constraint programming · Planning and scheduling · Constraint-based process models · Decision support systems

This research has been supported by the Spanish MINECO R&D Projects Pololas TIN2016-76956-C3-2-R and PLAN MINER TIN2015-71618-R.

✉ Andres Jimenez-Ramirez
ajramirez@us.es

[1] Depto. Lenguajes y Sistemas Informáticos, University of Seville, Seville, Spain

[2] Depto. Ciencias de la Computación e Inteligencia Artificial, University of Granada, Granada, Spain

[3] Technical University of Denmark, Lyngby, Denmark

1 Introduction

Businesses are increasingly interested in improving the quality and efficiency of their processes and in aligning their information systems in a process-centered way [32]. In such context, process-aware information systems (PAISs) [8] have emerged to provide a more dynamic and flexible support for business processes (i.e., BPs). BPs can be defined as sets of activities[1] which are performed in coordination in an organizational and technical environment [43] and which jointly achieve a business goal.

The support provided through a PAIS is greatly enhanced when it is able to provide accurate time predictions (i.e., predictions related to the completion time of a running process instance) since these predictions constitute a valuable tool when managing processes [29]. In fact, there exist many process scenarios for which temporal aspects are of utmost importance [44], and hence, reliable time predictions are crucial for any PAIS [41]. Specifically, such predictions allow process managers to (1) anticipate time problems, (2) proactively avoid time constraint violations, and (3) make decisions about the relative process priorities and timing constraints when significant or unexpected delays occur [10].

In such a context, time predictions must be provided while taking into account a set of basic requirements [33]: (1) the forecast must be highly accurate, (2) the prediction must take place nearly instantaneously, (3) the prediction functionality must be easy to use, and (4) the prediction may not interfere with the efficient operation of the PAISs. Therefore, time prediction represents a very challenging task and, even more, if the following desirable characteristics are considered:

- **The predictions should not be based on a single process instance** Typically, activities in a BP compete for limited resources which are shared between all the process instances which are executed in the PAIS. Therefore, predictions which are evaluated over an isolate process instance may lack accuracy since some of the resources might be assigned to a different instance. These resources are then not available for such instance. For this, in order to provide accurate predictions they must consider multiple process instances and resources [33,37].
- **The predictions should be multi-dimensional** In modern cooperative business, time is of utmost importance. Therefore, process models should be able to take this dimension into consideration [44]. Moreover, the resource perspective of BPs—which refers to the link between the activities defined in the processes and the entities that carry out the work related to them [45]—is significant to the efficiency and effectiveness of a process [40], and hence, should be also considered when designing a BP model [38]. Likewise, it becomes essential to take into account the data perspective since data constraints influence the possible executions of activities and, in turn, the execution of activities results in certain data constraints that should be met. Thus, a multi-perspective time prediction methodology, including all these aspects, would be desirable, i.e., besides predicting the remaining time of a specific instance, other relevant issues that can be also predicted for improving the management of running instances (e.g., start and end times of process activities, use of resources, and critical activities).
- **The prediction system should be able to adapt to changing circumstances** Many real scenarios might be subject to input uncertainty, e.g., the arrival time of clients is not well known or a resource became unavailable during the process enactment [39]. Therefore, in general, BPs are designed considering different alternatives to cope with such situations

[1] Similarly to [27], the term *activity* is used in this paper to express the smallest unit of work in a BP. In other previous works (e.g., [7]), however, the term *task* is used instead.

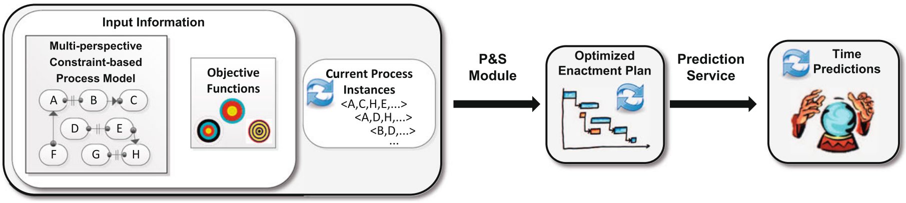


Fig. 1 Overview of the proposed approach for generating time predictions

when they are enacted. Accordingly, a related prediction system should also consider such alternatives during the enactment to increase the supported flexibility.

- **The prediction system should be able to deal with declarative models** Flexible PAISs [32] are required to allow companies to rapidly adjust their BPs to changes. In such a context, declarative BP models (e.g., constraint-based models) are increasingly used allowing their users to specify *what* has to be done instead of *how* [27], and hence, offering a high flexibility to end users. Many enactment plans related to the same constraint-based process model typically exist, and each of these plans presents different values for relevant objective functions (e.g., overall completion time).

Although there exist some approaches related to time prediction (e.g., [9,23,29,34,37,41,42]), they neglect some of the aforementioned characteristics. Especially in the last one, whereas there exist solid time prediction techniques for imperative models (e.g., [9,23,24]), little work has been conducted for declarative models [41].

In this work, an approach for generating time predictions of running process instances related to a multi-perspective constraint-based process model is proposed, i.e., which considers time, resource, data and control flow perspectives. The generated predictions are based on information extracted from both a constraint-based process model and the current state of partially executed process instances.

Note that while constraint-based process models offer a high flexibility to end users, they also increase the challenge of performing accurate predictions in such uncertain scenarios. In this work, we consider that all the decisions that the process stakeholders make about the way to execute it are generally aligned to the optimization of relevant objective functions (e.g., among all the available activities, which one should be executed next in order to minimize the overall completion time of the current process instances?). For this, instead of using any possible enactment plan which is compliant with the constraint-based process model, we propose to base the predictions on an optimized enactment plan that is generated from such a model at the current execution state.

Figure 1 depicts the main contribution of this work. Starting from (1) a multi-perspective constraint-based process model, (2) an objective function, and (3) the execution state of multiple running process instances, an optimized enactment plan is automatically generated. The generation of such plan is carried out by solving a planning and scheduling (P&S) problem in which, on the one hand, the activities to be executed are selected and ordered considering all the constraints, resource requirements, and resource availabilities—what constitutes a planning problem [15]. On the other hand, attribute values like start time are assigned to activities—what constitutes a scheduling problem [6]. For solving this P&S problem, we base on a previous work [21] where we propose a constraint-based approach in which the process model and the objective function are represented as a constraint optimization problem (i.e., COP). The generated enactment plan is obtained as an optimized solution to this problem and is, in turn, used for performing the predictions. Thereafter, the predictions are performed through the evaluation of desired functions over the optimized enactment plan.

Note that the decisions which are taken by the process stakeholders might not be aligned with the enactment plan which has been used to generate the predictions due to unexpected events (e.g., a resource became unavailable). Nonetheless, at run-time this plan is updated—if necessary—considering the current state of the running process instances, and therefore, the predictions are also updated as the execution of the process proceeds.

The proposed approach has several advantages that are worth nothing. First, when generating the optimized enactment plan, multiple process instances as well as the allocation of resources are considered. Secondly, since the optimized enactment plans are updated according to the running process instances, it is possible to deal with unexpected factors and hence, to adapt to changing circumstances. Third, besides predicting the remaining time of a specific process instance, the proposed approach allows the prediction of other relevant issues. Finally, declarative models are considered as starting point.

In previous work, we presented an approach for generating optimized enactment plans from constraint-based specifications [21]. In addition, we applied such technique to provide recommendations [2] and generate imperative BP models [1]. However, this paper significantly extends these previous works by:

1. Introducing a novel method to generate predictions from declarative specifications. Such a method is built upon the constraint-based tool developed in [21].
2. Performing a case study considering a benchmark and a real scenario in order to validate the effectiveness and suitability of the proposal.

Moreover, “Appendix A” of this paper provides further implementation details of the constraint-based tool which are not included in previous works.

The rest of the paper is organized as follows: Sect. 2 introduces backgrounds, Sect. 3 details the proposal for providing predictions, Sect. 4 explains a real example where the current proposal is applied, Sect. 5 deals with the evaluation, Sect. 6 presents a critical discussion, and Sect. 7 includes some conclusions and future work.

2 Background

This section introduces the related concepts which are used in the remainder of this paper. Section 2.1 provides backgrounds regarding constraint-based BP models. Section 2.2 gives an overview of planning, scheduling, and constraint programming. Section 2.3 summarizes previous related work on time prediction on business processes.

2.1 Constraint-based BP models

Different paradigms for process modeling exist, e.g., imperative and declarative. Irrespective of the chosen paradigm, desired behavior must be supported by the process model, while forbidden behavior must be prohibited [25,27]. While imperative process models specify exactly *how* things have to be done, declarative models only focus on *what* should be done. In our proposal, we use the constraint-based language Declare[2] [27,28] for the BP control flow specification. Declare is based on constraint-based process models.

[2] DECLARE is one of the most referenced and used declarative process modeling languages.

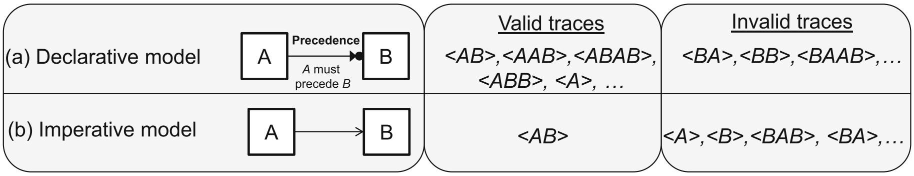


Fig. 2 Increased flexibility of declarative models versus imperative models

Definition 2.1 A **constraint-based process model** $CM = (A, C_{BP})$ consists of a set of activities A, and a set of constraints C_{BP} prohibiting undesired execution behavior. Each activity a ∈ A can be executed arbitrarily often if not restricted by any constraints.

Constraints can be added to a Declare model to specify forbidden behavior, restricting the desired behavior. For this, Declare proposes an open set of templates which can be divided into 4 groups:

1. **Existence** templates: unary template (i.e., it involves only one activity) concerning the number of times one activity is executed, e.g., Exactly(N, A) specifies that A must be executed exactly N times.
2. **Relation** templates: positive binary templates used to impose the presence of a certain activity when some other activity is performed, e.g., Precedence(A, B) specifies that to execute activity B, activity A needs to be executed before.
3. **Negation** templates: negative templates used to forbid the execution of activities in specific situations, e.g., NotCoexistence(A,B) specifies that if B is executed, then A cannot be executed, and vice versa.
4. **Choice** templates: n-ary templates expressing the need of executing activities belonging to a set of possible choices, e.g., ExactlyChoice(N, S) specifies that exactly N activities of the set of activities S must be executed.

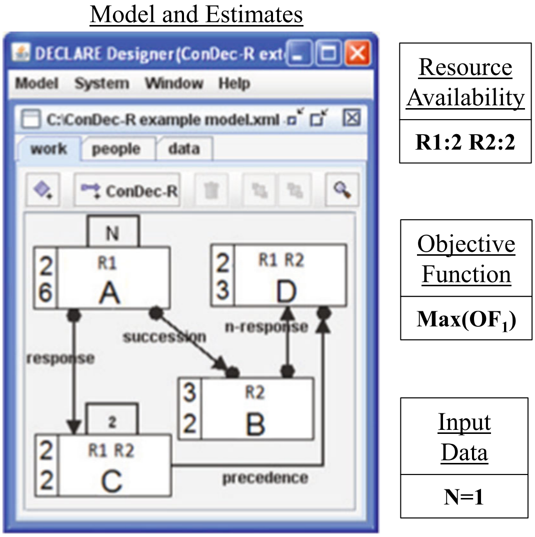
Example 2.1 Figure 2a shows a constraint-based BP model where traces[3] <AAB>, <AB>, <ABAB>, <ABB>, <A> are some of the valid ways of executing such model, while traces <BA>, <BB>, <BAAB> are invalid since A must precede B. In contrast, Fig. 2b shows an imperative model where there is only one valid execution trace, <AB>.

There are different ways to execute a constraint-based process model while fulfilling the constraints, i.e., there are several related enactment plans.[4] The different valid execution alternatives, however, can greatly vary in respect to their quality, i.e., how well different performance objective functions can be achieved. Such objective functions of the BPs are the functions to be optimized during the BP enactment, e.g., minimization of the overall completion time.

In order to allow dealing with more realistic problems compared to Declare, and motivated by requirements described in the literature [22,25,27,44], in previous works we extended

[3] For the sake of clarity regarding the examples of Fig. 2, traces represent sequences of only completed events of activity executions, i.e., no parallelism is considered. Nonetheless, as stated in Definition 2.4, the current approach deals with the identifier of the activity as well as with the start time, end time and the resource which executes the activity.

[4] Although imperative models allow for several choices, in general, all the execution paths should be explicitly specified. In contrast, declarative models specify constraints, and therefore, these models typically allow for more variants.



Resource Availability

R1:2 R2:2

Objective Function

Max(OF$_1$)

Input Data

N=1

Fig. 3 A simple ConDec-R process model

Declare to ConDec-R [20] by adding resource reasoning as well as temporal and data constraints,[5] resulting in ConDec-R process models. For this, ConDec-R supports activities with an open set of attributes and alternative resources.

Definition 2.2 A **BP activity** $BPAct = (a, Res, Atts)$ represents a BP activity called a, which can be performed by any resource included in Res, and which has a set $Atts$ of attributes associated (e.g., duration and profit) which is composed of tuples <att, val>.

Definition 2.3 A **ConDec-R process model** $CR = (BPActs, Data, C_{BP}, AvRes, OF)$ related to a constraint-based process model $CM = (Acts, C'_{BP})$ is composed of (1) a set of BP activities $BPActs$ associated with $Acts$, (2) problem data information $Data$, (3) a set of ConDec-R constraints C_{BP} including the constraints of C'_{BP} and the constraints which relates activities included in $BPActs$ and the data included in $Data$, (4) a set of available resources $AvRes$ which is composed of tuples (role,#role) which includes for each role (i.e., role) the number #role of available resources, and (5) an objective function OF to be optimized.[6]

Figure 3 shows a simple ConDec-R process model where: $BPActs = \{(A, < R1 >, << att_1, 2 >, < att_2, 6>>), (B, < R2 >, << att_1, 3 >, < att_2, 2 >>), (C, < R1, R2>, << att_1, 2 >, < att_2, 2>>), (D, < R1, R2 >, << att_1, 2 >, < att_2, 3>>)\}$;[7] $Data = \{<N, 1>\}$; $C_{BP} = \{exactly(1, A), succession(A, B), response(A, B), negate\text{-}response$

[5] ConDec-R directly supports the most common workflow resource pattern, i.e., the role-based distribution [35], which also supports our case study. Furthermore, ConDec-R allows to specify temporal constraints in a similar way as [25,44], i.e., all the Declare constraints are extended to support time intervals that indicate the time frame within which activities shall be performed. Moreover, ConDec-R includes data constraints in a similar way as [25].

[6] Note that a ConDec-R process model considers multiple perspectives and, hence, is a multi-perspective declarative process model.

[7] For the sake of simplicity, (1) all the BP activities of the example of Fig. 3 have the same attributes—which is a common situation—i.e., att_1 and att_2 and (2) the graphical representation depicts the room only for 2 attributes. Nonetheless, as stated in Definition 2.2, the number of attributes can be different for each one.

Enactment Plan

{**actId:** A_1, **st:** 0, **et:** 6, **res:** $R2_1$}
{**actId:** B_1, **st:** 7, **et:** 9, **res:** $R2_2$},
{**actId:** C_1, **st:** 3, **et:** 5, **res:** $R2_2$},
{**actId:** C_2, **st:** 6, **et:** 8, **res:** $R2_1$},
{**actId:** D_1, **st:** 0, **et:** 3, **res:** $R1_1$}

⬌

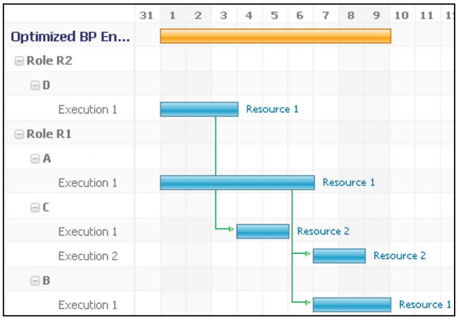


Fig. 4 Enactment plan example as a Gantt chart

(B, C), $precedence(C, D)$, $exactly(2, B)\}$; $AvRes = \{(R1, 2), (R2, 2)\}$; and $OF = maximize(OF_1)$.

2.2 Planning, scheduling, and constraint programming

The area of scheduling includes problems in which it is necessary to determine an enactment plan for a set of activities related to temporal and resource constraints (in our context the control flow constraints). In scheduling problems, several objective functions are usually considered to be optimized, in most cases related to temporal measures, or considering the optimal use of resources. In a wider perspective, in artificial intelligence (AI) planning [15], the activities to be executed are not established a priori; hence, it is necessary to select them from a set of alternatives and to establish an ordering. Thus, the objective of P&S is to find an enactment plan which fulfills the temporal and resource constraints while considering the optimization of some objective function. Such enactment plans are commonly represented as Gantt Charts [13] (cf. Fig. 4).

Definition 2.4 An **enactment plan** $EP = (pId, Acts)$ is composed by an identifier (i.e., pId) and a set of activities (i.e., $Acts$) which are executed without preemption. Each activity $act \in Acts$ consists of a tuple $<actId, st, et, res>$ where: $actId$ is an unique identifier of the activity, st and et state the start time and end time of the activity in the enactment plan, respectively, and res identifies the resource where the activity is allocated.[8]

In such a context, constraint programming (CP) [31] supplies a suitable framework for modeling and solving problems involving P&S aspects [36]. In order to solve a problem through CP, it needs to be modeled as a constraint satisfaction problem (CSP).

Definition 2.5 A **CSP** $P = (V, D, C_{CSP})$ is composed of a set of variables V, a set of domains D which is composed of the domain of values for each variable $var_i \in V$, and a set of constraints C_{CSP} between variables, so that each constraint represents a relation between a subset of variables and specifies the allowed combinations of values for these variables.

A solution to a CSP consists of an assignment of values to the CSP variables.

[8] Note that, since activities are executed without preemption and the same resource cannot be used to perform more than one activity in parallel, there are implicit precedence relations between the activities which are executed by the same resource since our approach does not allow a resource doing multiple activities in parallel.

Definition 2.6 A **solution** $S = <(var_1, val_1), (var_2, val_2), ...(var_n, val_n)>$ for a CSP $P = (V, D, C_{CSP})$ is an assignment of a value $val_i \in dom_i$ to each variable $var_i \in V$.

A solution is **feasible** when the assignments variable-value satisfy all the constraints. In a similar way, a CSP is feasible if at least one feasible solution for this CSP exists. From now on, S^{var} refers to the value assigned to variable var in a solution S.

Similar to CSPs, constraint optimization problems (COPs) require solutions that optimize an objective function.

Definition 2.7 A COP $P_o = (V, D, C_{CSP}, o)$ related to a CSP $P = (V, D, C_{CSP})$ is a CSP which also includes an objective function o to be optimized.

A feasible solution S for a COP is **optimal** when no other feasible solution exists with a better value for the objective function o.

Constraint programming allows to separate the models from the algorithms, so that once a problem is modeled in a declarative way as a CSP, a generic or specialized constraint-based solver can be used to obtain the required solution. Furthermore, constraint-based models can be extended in a natural way, maintaining the solving methods. Several mechanisms are available for solving CSPs and COPs [31], which can be classified as search algorithms (i.e., for exploring the solution space to find a solution or to prove that none exists) or consistency algorithms (i.e., filtering rules for removing inconsistent values from the domain of the variables). In turn, search algorithms can be classified as complete search algorithms (i.e., performing a complete exploration of a search space which is based on all possible combinations of assignments of values to the CSP variables) and incomplete search algorithms (i.e., performing an incomplete exploration of the search space so that, in general, to get a feasible or an optimal solution is not guaranteed). In this work, we apply P&S to generate the best possible enactment plan from the a constraint-based process model through a complete search algorithm.

Since many COPs present NP complexity [14], optimized solutions are considered.

Definition 2.8 Let $Sols$ be the set of all the solutions of a COP P_o and let $Sols_t \subseteq Sols$ be the subset of the solutions already explored at certain time t. Then, a solution $sol_1 \in Sols_t$ is **optimized** if it can be ensured that it is optimal regarding only the subset $Sols_t$.

2.3 Time prediction on business processes

Time prediction[9] represents a valuable tool for any PAIS [41] since there exist many process scenarios for which time is of utmost importance [44].

Many proposals related to time prediction can be found in the literature. Such proposals base their predictions either on: (1) applying data mining techniques for analyzing logs of past process execution [33,41,42], (2) applying certain techniques over the process model [9,24,37] (e.g., simulation, critical path method, or queuing network analysis), and (3) both sources of information [23,34] (i.e., event logs and process design).

Regardless of the source information which is analyzed for performing the predictions, such predictions should not be based on a single process instance in isolation, but consider multiple process instances and resources [33,37]. However, while only few proposals deal with this issue (e.g., [24,34]), most proposals (e.g., [9,23,41,42]) do not pay attention to the influence that the execution of multiple instances competing for shared resource has on the related predictions.

[9] Some works refer to time prediction as case prediction (e.g., [33]).

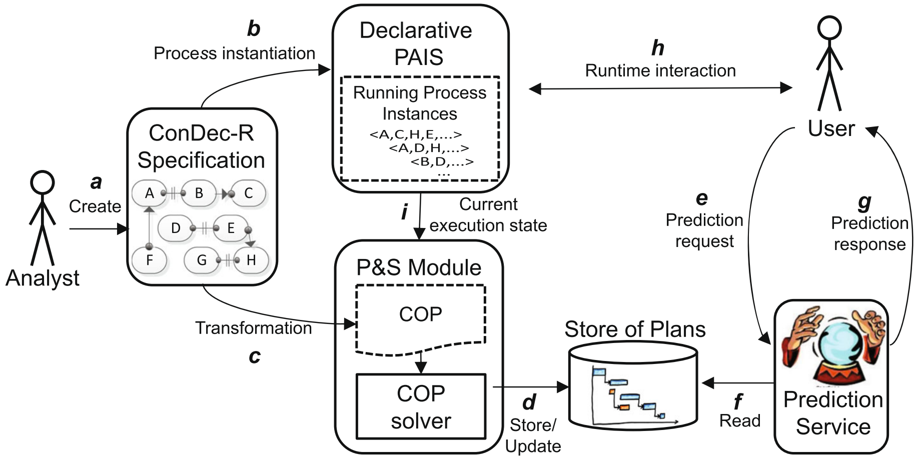


Fig. 5 Overview of our proposal for generating time predictions based on optimized enactment plans

Moreover, since flexible PAISs [32] are required more and more, the prediction system should be able to adapt to changing circumstances (e.g., a resource became unavailable during the process enactment) [23,34,37].

Although most proposals on time predictions are only focused on predicting the remaining time that is needed to complete the handling of a specific instance, there are other relevant issues that can be also predicted to support and improve the management of running instances: (1) start and end times of process activities, (2) use of resources, or (3) critical activities (i.e., activities whose delay implies a greater overall completion time). In such respect, existing proposals could be used for dealing with some of these issues, but, to the best of our knowledge, there is not any previous proposal able to deal with all the aforementioned issues.

3 Method for generating time predictions

This section details the method which is proposed for generating time predictions. Figure 5 shows an overview of the proposed approach, and its main components are explained as follows:

- **ConDec-R Specification** is the multi-perspective constraint-based model (cf. Definition 2.1) created by an analyst through the ConDec-R language (cf. Definition 2.3).
- **Declarative PAIS** is the component which takes a declarative specification and allows the user for instantiating such specification and interacting with the running process instances.
- **P&S Module** is a constraint-based tool (i.e., it relies on a COP solver) which is in charge of generating optimized BP enactment plans from a declarative specification. In addition, the P&S Module allows considering that the current execution state (i.e., partial traces) of the running process instances is reproducible in the generated plan (i.e., the plan includes the current execution state).
- **Store of Plans** is just the shared place where both the P&S Module and the Prediction Service manage the plans.

- **Prediction Service** is the system which allows the user to ask for predictions. Each time the user asks for a prediction related to any magnitude (e.g., remaining time until completion of all running instances), this service considers the most recent optimized enactment plan for producing the prediction response.

Therefore, the Prediction Service allows to predict values not only over a single process instance but over the whole set of instances which is planned to be executed within a certain timeframe.

To initialize the components, in a first step, a multi-perspective constraint-based model is created by an analyst through the ConDec-R language (cf. step a in Fig. 5). Therefore, the control flow, resource requirements, estimates of the activities (e.g., the duration), and resource availabilities are specified. Estimates can be obtained by interviewing business experts or by analyzing past process executions (e.g., by calculating the average values of the parameters to be estimated from event logs). Moreover, both approaches can be combined to get more reliable estimates. Second, the Declarative PAIS takes such specification as input to allow users to execute related running process instances (cf. step b in Fig. 5). And third, the same specification is transformed and passed to the P&S Module to initialize it (cf. step c in Fig. 5). For this, the elements of the ConDec-R model (i.e., BP activities, constraints, resources, data and objective function) are transformed into the elements of the COP (i.e., variables, domains, constraints and objective function). Thereafter, the resulting COP can be solved using a search algorithm to obtain the solution of the COP which is directly considered as the optimized enactment plan (cf. step d in Fig. 5).[10]

After these initial steps, two different processes run in a parallel way:

- **The user requests a prediction**: At run-time, the user may request a prediction for the running process instances (cf. step e in Fig. 5). Thereafter, as stated in Algorithm 1, the prediction service registers that a user requested a prediction through the method `registerPendingPrediction`. Then the algorithm gets the existing optimized enactment plan from the store or waits for it in case that no plan is there yet (cf. step f in Fig. 5). As soon as a new plan is calculated by the P&S module, such a plan is returned by the method `getPlanOrWait`. After that, the prediction is generated (cf. step g in Fig. 5) using: (1) the optimized enactment plan and (2) a measurement function[11] `Mf` which states the magnitudes to extract from the plan. After the prediction is performed, the method `deregisterPendingPrediction` informs the system that the prediction has been served.

Algorithm 1: Providing Predictions

```
input : Measurement Function $Mf$
        PlanStore $store$
output: Prediction pred

$registerPendingPrediction()$;
EnactmentPlan $ep \leftarrow store.getPlanOrWait()$;
$pred \leftarrow Mf(ep, currentTime())$;
$deregisterPendingPrediction()$;
```

Definition 3.1 Let *EP* be an enactment plan and *T* a time, then a **Measurement function** Mf(EP,T) is a function that produces some related measurements, e.g., accumulated profit

[10] A more formal description of the transformation as well as deep implementation details is stated in "Appendix A."

[11] This definition of measurement is an adaptation of the one given by van der Aalst in [41].

until time T, remaining profit from time T, time until completion. Formally, $Mf \in (C \times N) \rightarrow M$, where C is the set of possible enactment plans, N is the set of possible time stamps, and M is the set of possible measurement values (e.g., some time duration).

Note that, similarly to [41], this definition of measurement can be used for generating both predictive and non-predictive values. That is, a predictive value is obtained when the measurement function looks beyond T, e.g., expected time until completion. In contrast, non-predictive values are those which only require information of the enactment plan regarding the elapsed time (i.e., before T), e.g., accumulated profit.

Example 3.1 On the one hand, a predictive measurement function related to the remaining completion time would be described as follows:

$$Max_{act \in EP.Acts}(act.et) - T$$

That is, it is measured by calculating the end time of the last activity in the plan—which is the total completion time—and then, subtracting the time T when the prediction is asked.

In addition, a measurement function to predict the expected profit would be described as follows:

$$\sum_{act \in EP.Acts} \mathit{profit}(act)$$

That is, it is calculated by summing the attribute profit of all the activities which exist in the plan.

On the other hand, a non-predictive measurement function related to the accumulated profit would be described as follows:

$$\sum_{act \in EP.Acts | act.et < T \mathit{profit}(act)}$$

That is, the sum of the attribute profit of all the activities whose end time is before the time T when the prediction is asked.

In the above formulas, $EP.Acts$ refers to the activities which are enacted in the enactment plan EP, $act.et$ represents the value of the et variable of the activity act, and $profit(act)$ is the value of the attribute profit of the BP activity associated with act in the model.

- **Process instances are executed by authorized users** (cf. step h in Fig. 5): As the execution of the process proceeds, events are generated (i.e., starting or finishing an activity by a resource) and constitute the current execution state which is sent to the P&S Module (cf. step i in Fig. 5). This module is in charge of updating the optimized enactment plan which is in the store by: (1) removing it if does not support the current state, i.e., the current partial trace is not a subtrace of such a plan, (2) continuously generating new optimized enactment plans in order to improve and replace the existing one, i.e., allowing continuous replanning. In this step, the solver considers the information of the current execution state obtained from the declarative PAIS. Note that this allows to adapt the predictions to changing circumstances.
  The behavior of the P&S Module is described in Algorithm 2. In the first line, a solver is created to look for solutions (i.e., optimized enactment plans) compliant with the ConDec-R process model. Each time the solver finds a solution which is better than the previous one, the solver updates the plan in the store. It is important to notice that, since predictions are requested on demand (cf. Algorithm 1), the time between the user interacts with the PAIS and the Prediction Service takes the optimized enactment plan from

the store for generating a prediction is unknown. For this, the solver is executed without establishing a time limit, i.e., the `startGeneratingPlans` method launches the solver which goes on optimizing the plan, always compliant with the `currentState`, until `stopGeneratingPlans` is invoked. The algorithm is executed until all running instances have been completed (line 9 in Algorithm 2). While the solver is running, the algorithm waits until no predictions are pending (cf. line 5 in Algorithm 2), i.e., the Prediction Service is not waiting for a plan in the store. After that, it waits and listens to the changes in the current execution state that the P&S module may receive (line 6 in Algorithm 2). Whenever a change is received, the solver is stopped (line 7 in Algorithm 2) since the change may invalidate the solutions which are being generated and the optimized enactment plan in the store is updated (line 8 in Algorithm 2). Specifically, the `removePlanIfNotCompliant` method removes the plan from the store if the current execution trace does not match, i.e., it is not reproducible in the plan.

Algorithm 2: Run-time Monitoring

```
  input : ConDec-R Specification $cr$
          PlanStore $store$
1 Solver $solver \leftarrow newSolver(cr, store)$;
2 Set<Event> $currentState \leftarrow \emptyset$;
3 repeat
4     $solver.startGeneratingPlans(currentState)$;
5     $waitIfPendingPredictions()$;
6     $currentState \leftarrow waitForChangesInCurrentState()$;
7     $solver.stopGeneratingPlans()$;
8     $store.removePlanIfNotCompliantWith(currentState)$;
9 until $!CompleteTrace(cr, currentState)$;
```

Despite the NP complexity of the considered problems, in general, replanning is less time-consuming than initial planning, since most of the information about previous generated plans can usually be reused. In the context of the current approach for the P&S Module, CSP variable values become known as execution proceeds.

From the point of view of the proposed approach, the complexity of performing a prediction over a given declarative model depends on the number and diversity of its execution alternatives, i.e., on the size of the search space. To be more precise, such complexity is related to the number of BP activities, the number and type of constraints among these activities, and the percentage of the plan that is already executed.

4 A real example: a beauty salon of Seville

This section introduces a real example from a beauty salon that is used to validate the current proposal in the considered case study.

The considered business has expanded quickly in the last years involving more staff, services, and complex constraints which resulted in problems related to the management of the salon. In particular, long waiting time for clients and a lack of information for the manager are causing problems, affecting customer satisfaction and profit of the business.

Since our approach generates predictions based on the state of the beauty salon, the aforementioned problems can be detected in advance, and therefore, the manager of the salon can react to overcome them.

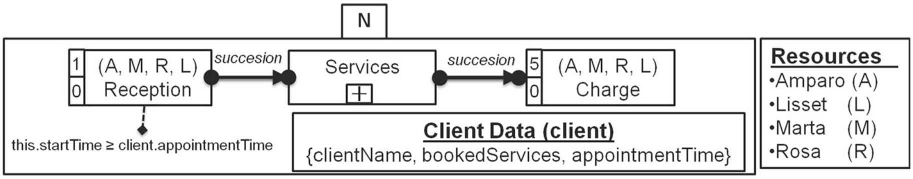


Fig. 6 ConDec-R Model for the beauty salon problem (top level process)

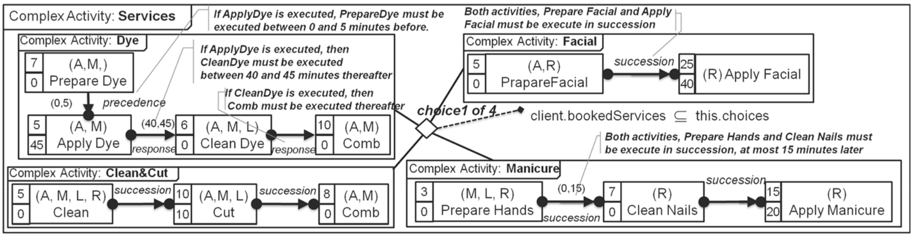

Complex Activity: Services
If ApplyDye is executed, PrepareDye must be executed between 0 and 5 minutes before.
Both activities, Prepare Facial and Apply Facial must be execute in succession
Complex Activity: Dye
7
0
(A,M,)
Prepare Dye
If ApplyDye is executed, then CleanDye must be executed between 40 and 45 minutes thereafter
(0,5)
precedence
If CleanDye is executed, then Comb must be executed thereafter
5
45
(A, M)
Apply Dye
(40,45)
response
6
0
(A, M, L)
Clean Dye
response
10
0
(A,M)
Comb
Complex Activity: Facial
5
0
(A,R)
PrapareFacial
succession
25
40
(R) Apply Facial
choice1 of 4
client.bookedServices ⊆ this.choices
Both activities, Prepare Hands and Clean Nails must be execute in succession, at most 15 minutes later
Complex Activity: Clean&Cut
5
0
(A, M, L, R)
Clean
succession
10
10
(A,M, L)
Cut
succession
8
0
(A,M)
Comb
Complex Activity: Manicure
3
0
(M, L, R)
Prepare Hands
(0,15)
succession
7
0
(R)
Clean Nails
succession
15
20
(R)
Apply Manicure

Fig. 7 ConDec-R Model for some of the services which are offered

The considered beauty salon offers various services[12] like dye, clean&cut, manicure, and facial services. Clients are required to make appointment calls so the number of clients and its booked services are known at the beginning of the day. There are several full-time employees identified by A, R, L, and M, and each activity can be performed by certain employees only. In addition, each activity has an average estimated duration and a profit which is obtained after their execution. The manager of the salon wants to plan and schedule a working day with several clients considering that the waiting time (WT) of the clients has to be minimized and distributed uniformly among all the clients (objective function):

$$WT = \frac{\sqrt{\sum_{c \in C} ((S^{et(c)} - c.appT) - (\sum_{b \in c.served} b.estimate))^2}}{C.size}$$

where C is the set of clients, S is the considered solution, $S^{et(c)}$ is the time when the client c has finished, $c.appT$ is the appointment time of c, *c.served* is the set of services which are applied to c (i.e., included in the enactment plan), and *b.estimate* is the estimated duration for service b.

Typically, as illustrated in Fig. 6, a client visit starts with the reception in the beauty salon. After that, the staff applies some services to the client and, finally, the client is charged. Complex activity *Services* is composed of other activities[13] (e.g., dye, clean&cut, facial, and manicure, cf. Fig. 7), while *Reception* and *Charge* are BP activities (cf. Definition 2.2). For each BP activity two attributes are considered: (1) estimated activity duration and (2) profit.[14] Moreover, the set of alternative resources which can perform the BP activity is also included, e.g., the activity *Reception* of Fig. 6 has an estimated duration of 1 min and a profit of 0 and can be performed by A, R, M, or L.

[12] For the sake of clarity, the depicted scenario is a subset of the actual beauty salon, i.e., the salon offers more services and has more employees.

[13] In a similar way to PSL [30], ConDec-R allows hierarchical modeling (i.e., complex activities aggregate activities).

[14] As can be seen in Figs. 6 and 7, the profit of the services is associated with one of the activities of the related services.

Table 1 Case study research questions

Id	Research question
MQ1	Is the proposed approach useful for generating predictions?
AQ1	Can the proposed approach propose accurate predictions?
AQ2	Can the proposed approach generate predictions nearly instantaneously?

The current problem deals with N clients (cf. Existence constraint in Fig. 6)—each one representing an instance of the model—which come to the salon at different times and with different bookings during a working day.

Such information is included in the data perspective (cf. $Client\text{-}Data$ element in Fig. 6). Through the data perspective, it is also modeled that activity $Reception$ cannot start before the client appointment time. Moreover, a data constraint is used (in conjunction with the choice constraint) to ensure that all the services the client has booked are selected in the generated enactment plans.[15]

5 Empirical evaluation

This section provides an empirical study for the proposed approach. Specifically, the purpose of this study is the evaluation of the whole approach in terms of its suitability to provide predictions. In this section, the case study protocol for the software engineering field proposed by [5] is followed to improve the rigor and validity of the proposed study. Such protocol suggests the following sections: background, design, case selection, case study procedure and data collection, analysis and interpretation and validity evaluation.

Background Taking the purpose of the proposed study into account, a main research question (MQ1) is defined (cf. Table 1). Specifically, $MQ1$ assesses whether the current approach can be useful to provide predictions by satisfying the aforementioned requirements [33]. For this, $MQ1$ is divided into two additional questions: (1) $AQ1$ checks whether the predictions which are obtained by our approach are accurate, and (2) $AQ2$ evaluates the immediacy of our approach for generating accurate predictions.

Design The object of study is the method which is proposed for generating predictions from ConDec-R specifications. For this, a $holistic$ design which considers the overall proposal as a whole is carried out. In this design, the architecture described in Sect. 3 is established for addressing $MQ1$ (i.e., $AQ1$ and $AQ2$). In such respect, the interaction between the user and the declarative PAIS (cf. Fig. 5) is simulated as detailed later in the case study procedure.

This case study is run on a Intel(R) Core (TM) CPU i7-3517U, 1.90 GHz, 10 GB memory, running Windows 7. In this work, we consider the constraint-based system IBM ILOG CPLEX Optimization Studio (CPLEX) [18] for implementing the constraint-based approach detailed in "Appendix A".[16] CPLEX provides for efficient search algorithms as well as effi-

[15] As an example, two optimized BP enactment plans for the beauty salon problem with different concurrent clients can be found at http://azarias.lsi.us.es/Predict/PlansBeautySalon.pdf.

[16] CPLEX has been the selected tool for the current approach due to its maturity—it is the successor of ILOG Solver, the market leader in the last decade [31]. Although it is a proprietary software, it can be freely accesses for the academic community and it is currently used in many papers, e.g., [46,47].

Table 2 Quantified variables for the holistic design which are obtained by applying the proposed approach with a time limit equal to lt seconds and after the $sp\%$ of a reference process enactment is executed with $lt \in \{1, 5, 10\}$ and $sp \in \{0, 25, 50, 75\}$

Variable	Description
Min^{WT}	Average of the waiting time of the reference solution which represents minimum waiting time
Max^{WT}	Average of maximum waiting time which is achieved
$Pred_{lt,sp}$	Average of the prediction of the waiting time
$NPred_{sp}$	Average of the prediction of the waiting time when no optimization is performed
$Err_{lt,sp}$	Average of the error made in the prediction $Pred_{lt,sp}$. This error is measured as $abs(\|Pred_{lt,sp} - Ref^{WT}\|)/(range^{WT})$, where Ref is the reference enactment plan, and $range^{WT}$ is the range of possible values of WT
$NErr_{sp}$	Average of the error made in the prediction $NPred_{sp}$. It is measured similarly to $Err_{lt,sp}$

Table 3 Generic synthetic models with 10 and 20 activities and a varying number of constraints

Model	#Acts	Description
A10	10	Includes 10 activities and 7 constraints
B10	10	Extends A10 by including 3 additional constraints
A15	15	Includes 15 activities and 12 constraints
B15	15	Extends A15 by including 3 additional constraints
A20	20	Includes 20 activities and 17 constraints
B20	20	Extends A20 by including 6 additional constraints

cient high-level objects and constraints to deal with temporal constraints, resource allocation, and optimization. This leads to an efficient management of the problems to be solved. After the application of the aforementioned holistic design, the generated information (i.e., the predictions) is analyzed to answer the research questions (cf. Table 1).

The data described in Table 2 are quantified for each ConDec-R model which is considered following the case study procedure.

Case selection For this case study, the beauty salon problem is studied. We consider this is a good and suitable case since it fulfills the following selection criteria: (1) it has been created for **an actual business**, (2) the business has grown up and now it has **scheduling problems** (i.e., involves resource allocation, complex constraints and the optimization some objective function), and (3) it manages several performance **measures which can be used to generate predictions** (e.g., resource usage, waiting times, profit, completion time, etc.).

In addition, in order to extend the external validity, a set of synthetic models has been created taking some important characteristics into account. First, correctness, i.e., the ConDec-R models must represent feasible problems without any conflict (i.e., there are some traces that satisfy the model). Second, representativeness, i.e., the ConDec-R models must represent problems which are similar to actual BPs with different constraints and sizes. Consequently, we considered models of medium-size (i.e., including 10–20 activities) which comprise all basic types of ConDec-R templates, i.e., existence, relation, and negation. For this, 6 generic

Table 4 Description of the acronyms

Acronym	Description
NC	Number of clients in the beauty salon during the day
T	Appointment time of each client
NS	Average number of booked services by client during the day
S	Booked services for each client

test models are considered with 10, 15 and 20 activities, respectively, and a varying number of constraints (cf. Table 3).

Case study procedure and data collection The execution of the study is planned as follows.

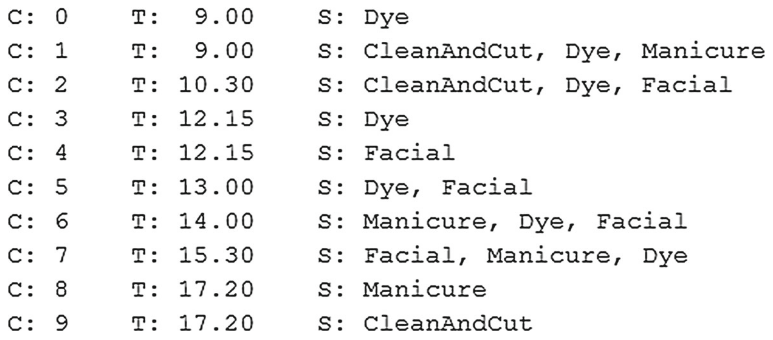
First, the business is selected—according to the selection criteria—and it is modeled as a ConDec-R model.

After that, on the one hand, different configurations are generated related to the beauty salon problem. Each configuration specifies the number of clients (NC) which are considered and the average number of booked services (NS). According to the information which is provided by the manager of the salon (i.e., there are normally between 10 and 20 clients per day and a client typically books one or two services), we consider values $\{1, 1.5, 2\}$ for NS and the values $\{10, 15, 20\}$ for NC. Based on this information, to average the results, a collection of 30 ConDec-R models is randomly generated for each pair $<NC, NS>$ by varying the booked services of each client (S) and their appointment times (T). In summary, $30*3*3 = 270$ different ConDec-R related to the beauty salon is considered. Figure 8 shows an example of a problem file of the beauty salon generated for 10 clients (i.e., $NC = 10$) with 2 booked services in average (i.e., $NS = 2$).[17] To clarify, Table 4 summarizes the different acronyms which appear above.

On the other hand, for the synthetic problems, Fig. 10 shows the ConDec-R representation of the generic models A10, B10, A15, B15, A20 and B20. There are some activities that are involved in an existence constraint, which means that such activities must be repeated several times. We have considered 15, 30 and 60 repetitions, i.e., $N \in \{15, 30, 60\}$. Regarding the number of available resources, in turn, for all the generated test models, two available resources of two kinds of roles (i.e., R1 and R2) are considered. Moreover, random durations and resource requirements are considered for each activity since these aspects have a great influence on the complexity of the search of optimal solutions. This is due to the considered problems are extensions of typical scheduling problems. Specifically, in order to average the results over a collection of randomly generated ConDec-R models, 30 instances are randomly generated for each specific ConDec-R model by varying activity durations between 1 and 10 and role of required resources between R1 and R2. In summary, $6 * 3 * 30 = 540$ different synthetic ConDec-R models are considered for evaluating predictions. In this case, the objective function and the prediction function are both related to the overall completion time, i.e., the time spent to complete all the instances.

Thereafter, for each beauty salon problem (i.e., same activities, relations and resources but different booked services and appointment times) and for each synthetic problem, we proceed as depicted in Fig. 9:

[17] The set of problems which are used for the empirical evaluation is available at http://azarias.lsi.us.es/Predict/ObjectsBeautySalon.zip.



```
C: 0    T:  9.00   S: Dye
C: 1    T:  9.00   S: CleanAndCut, Dye, Manicure
C: 2    T: 10.30   S: CleanAndCut, Dye, Facial
C: 3    T: 12.15   S: Dye
C: 4    T: 12.15   S: Facial
C: 5    T: 13.00   S: Dye, Facial
C: 6    T: 14.00   S: Manicure, Dye, Facial
C: 7    T: 15.30   S: Facial, Manicure, Dye
C: 8    T: 17.20   S: Manicure
C: 9    T: 17.20   S: CleanAndCut
```

Fig. 8 Example of a problem file of the empirical evaluation generated for 10 clients (i.e., $NC = 10$) with 2 booked services in average (i.e., $NS = 2$). For each client (i.e., C), its appointment time (i.e., T) and its booked services (i.e., S) are stated

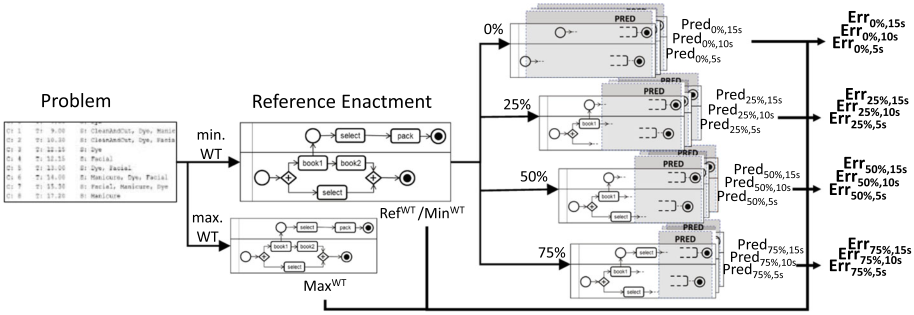


Fig. 9 Case study procedure

1. An optimized BP enactment plan is generated by the proposed approach for minimizing the waiting time when establishing the time limit of the solver equal to 5 min. This plan is then selected for being the reference process enactment, i.e., to simulate the behavior of a potential user.[18] In this step, the Ref^{WT} is obtained which is the same as Min^{WT}.
2. In order to calculate a tentative maximum value of the waiting time—which is necessary for calculating $range^{WT}$—another search is performed for 5 min considering the maximization of the waiting time as objective function and then, Max^{WT} is obtained.
3. The next steps evaluate different predictions using the proposed approach (cf. Sect. 3). For this, the initial parts of the reference plan are considered as the current state of the instances (cf. Algorithm 2). The reasons for taking initial parts of this best plan are to (1) easily get feasible traces and (2) be able to compare the quality of the enactment plans which are generated in few seconds versus the ones which are generated in 5 min. Specifically, 0, 25, 50 and 75% of the reference plan are considered as already executed. For each percentage, the solver is given different time limits to generate the plans (i.e., the time which exists between the $startGeneratingPlans$ method is invoked in Algorithm 2 and the related prediction is requested by Algorithm 1) which are used by the Prediction Service to provide the predictions. Specifically, 5, 10 and 15 s. are used.
Therefore, the values of $Pred_{tl,sp}$ are obtained for $tl \in \{5, 10, 15\}$ and $sp \in \{0, 25, 50, 75\%\}$.

[18] Note that, as previously mentioned, the process stakeholders which are involved want the optimization of the average waiting time of the clients when executing the model.

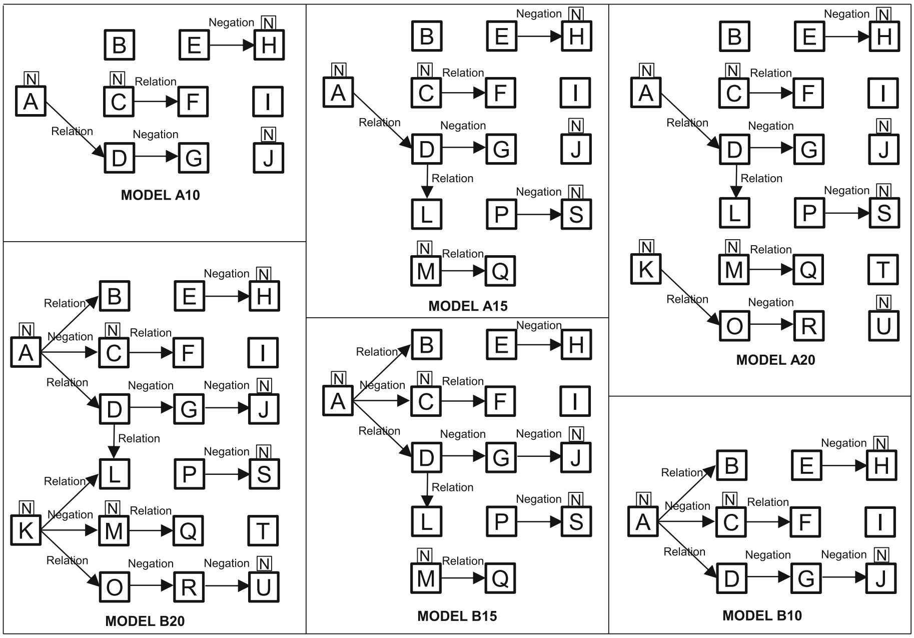


Fig. 10 Generic synthetic ConDec-R models

4. In addition, in order to illustrate the effectiveness of the method in the beauty salon scenario, the values of $NPred_{sp}$ are obtained by considering the first solution which is obtained by the solver, i.e., a non-optimized solution.
5. After that, the $Err_{tl,sp}$ and the $NErr_{sp}$ values are calculated using the above data.

Finally, the analysis and interpretation of the collected data is conducted and the validity of the case study procedure is studied.

The values for the response variables are included in Table 5.

Analysis and interpretation The data which are collected are analyzed to answer the research question and to draw conclusions (cf. Tables 5 and 6). In order to address $MQ1$, sub-questions $AQ1$ and $AQ2$ need to be answered (cf. Table 1).

As expected, the ranges $[Min, Max]$ are narrower as the complexity of the problem increases since fewer options to allocate the activities exist. Moreover, when the prediction (cf. column $Pred$) is closer to the reference value (cf. column Min) the average error is lower (cf. column Err) (Fig. 10).

Regarding the accuracy of the solutions, the average of the error increases as the complexity of the problems increases in both the beauty salon problems and the synthetic problems. Specifically, the problems which entail the highest complexity are those related to the configuration $<NC = 20, NS = 2, tl = 5s>$—in the beauty salon—and $<Model = A20, N = 60>$—in the synthetic problems—in which the value for Err is nearly 50% when 0% of the reference plan is known (i.e., $sp = 0\%$). Although it is not a good value, this can be explained by the fact that the process enactment has not been started yet, and the time limit for solving a rather complex problem is very low (i.e., only 5 s). Moreover, it is the only case where the obtained prediction is really close to the one obtained by the trial predictor. Nonetheless, as can be observed, for all the configurations where $sp \geq 25\%$, Err is lower

Table 5 Quantified variables for the experiment

NC	NS	Min	Max	Time	sp = 0%		sp = 25%		sp = 50%		sp = 75%	
					Err (%)	Pred	Err (%)	Pred	Err (%)	Pred	Err (%)	Pred
10	1	1.8	509.5	Trial	41.6	213.2	15.2	79.5	8.8	46.5	2.6	15.3
				5	4.5	24.9	3.0	17.2	0.9	6.2	0.1	2.1
				10	3.1	17.5	2.1	12.7	0.7	5.2	0.0	2.0
				15	2.5	14.6	1.6	9.9	0.5	4.3	0.0	2.0
	1.5	1.5	489.6	Trial	33.5	165.0	9.1	46.2	4.8	25.0	0.9	6.0
				5	4.0	21.4	2.8	15.2	0.7	4.7	0.1	1.9
				10	2.7	14.7	2.0	10.7	0.5	4.1	0.0	1.6
				15	2.2	12.5	1.4	8.0	0.4	3.6	0.0	1.5
	2	1.9	437.7	Trial	13.2	59.4	7.4	34.4	5.0	22.9	0.8	5.5
				5	5.6	26.3	3.0	15.3	2.0	10.7	0.0	1.9
				10	4.5	21.6	2.7	13.8	1.3	7.7	0.0	1.8
				15	3.2	15.9	2.2	11.8	1.2	7.0	0.0	1.8
15	1	8.4	467.1	Trial	32.0	151.2	28.1	137.8	13.8	72.7	2.6	20.2
				5	11.6	62.4	7.4	42.6	2.3	18.6	0.1	8.7
				10	8.8	49.0	4.3	28.4	1.2	13.7	0.1	8.5
				15	6.1	36.6	2.7	20.9	0.8	11.0	0.0	8.3
	1.5	10.7	450.2	Trial	29.7	141.1	24.5	118.4	15.0	76.5	2.7	22.1
				5	14.8	75.8	9.9	54.3	2.9	23.6	0.1	10.9
				10	9.7	53.3	6.4	38.9	1.9	18.9	0.1	10.4
				15	7.0	41.4	5.3	33.8	1.1	15.5	0.1	10.3
	2	14.3	401.9	Trial	35.9	153.4	29.1	128.0	16.0	76.2	2.9	25.8
				5	18.2	84.6	13.9	67.2	3.5	27.9	0.3	14.7
				10	11.6	59.4	8.1	45.4	1.9	21.5	0.1	13.9
				15	9.0	49.3	6.0	36.9	1.2	19.0	0.1	13.9
20	1	24.7	436.4	Trial	40.7	192.3	35.4	170.4	18.8	103.1	3.2	36.5
				5	28.7	143.0	20.1	107.0	6.4	51.2	1.2	29.5
				10	18.1	99.7	11.6	72.4	4.1	41.6	0.6	26.9
				15	12.4	75.8	9.2	62.6	3.4	38.2	0.3	25.2
	1.5	34.1	419.4	Trial	44.6	206.1	35.1	169.1	19.5	108.9	3.1	47.1
				5	34.4	166.3	26.7	135.9	9.7	71.2	0.7	36.1
				10	26.9	136.1	18.5	104.4	6.2	57.7	0.3	34.0
				15	21.9	117.2	14.4	89.0	4.7	51.8	0.3	33.2
	2	66.2	388.8	Trial	51.3	239.3	35.4	190.0	19.8	137.6	5.5	84.2
				5	49.8	236.3	33.9	177.6	19.8	137.4	2.1	73.0
				10	45.7	213.6	11.7	106.5	8.8	102.3	2.1	72.2
				15	38.9	182.5	9.3	91.5	6.0	87.9	2.1	72.2

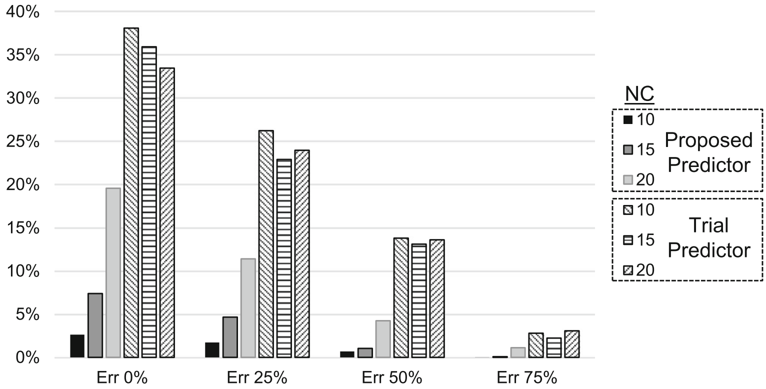


Fig. 11 Average error committed by the predictors grouped by NC

than 20% when 10 s or more are provided as time limit. Moreover, the average error is lower than 10% in most cases, which can be considered good solutions overall if it is compared with the trial predictor. A similar behavior is observed with the synthetic problems in which the error is lower and stays below 7% for configuration with $N = 15$ regardless of the model. As can be seen in Fig. 11—which shows the average error which is committed by both predictors grouped by the number of clients—the error of the proposed prediction is considerably lower than the trial one. For this, $AQ1$ can be answered as true when some information is known about the process enactment, but might be questioned for early predictions.

Regarding the immediacy of the predictions (i.e., $AQ2$), as expected, the accuracy of the predictions increases as the time limit increases. In addition, for most of the problems, the error in the prediction considering $time = 10$ is lower than 12%, that can be considered a good result. Thereafter, $AQ2$ can be answered as true since 10 s can be considered a reasonable time limit for the considered scenario.

Validity evaluation This section evaluates if the results of the proposed case study are valid and not biased. To be more precise, three types of validity are addressed in this section: construct, internal and external.

Firstly, regarding the construct validity, it has to be addressed in how far the measures which have been used are appropriate to address the research questions which have been planned. Three different threats are identified related to the acquisition of the data. The first threat is related to how the problems have been randomly generated in both designs. In these designs, unsolvable problems were not considered in order to evaluate the algorithm better. This is checked considering a simple rule: the generated appointment time of a client plus the time which her booked services consume cannot overpass the closing time of the beauty salon. Due to the parallelism which may exist because of the temporal constraints (i.e., a client can be served by different employees at the same time), this rule leaves out some problems which might be solvable. To mitigate this threat, a more elaborated algorithm can be performed to avoid eliminating problems which may be solvable. Secondly, the complexity of the problems which are generated is controlled only by varying the number of clients and her booked services. Although we consider that the beauty salon is a suitable business due to its complexity, different ways of controlling this complexity can be applied to mitigate this threat, e.g., by changing the type of constraints. The third threat concerns the data collected (cf. Table 2). To the best of our knowledge, the metrics are good enough for addressing AQ1

Table 6 Error values for the synthetic experiment for $tl = 5s$

Model	N	$Err_{0\%}$ (%)	$Err_{25\%}$ (%)	$Err_{50\%}$ (%)	$Err_{75\%}$ (%)
A10	15	0.0	0.0	0.0	0.0
	30	1.2	0.1	0.0	0.0
	60	43.2	31.3	13.8	0.2
A15	15	0.3	0.0	0.0	0.0
	30	18.2	12.3	4.9	0.1
	60	42.2	41.0	32.8	13.2
A20	15	3.2	1.1	0.0	0.0
	30	41.4	33.3	24.3	5.7
	60	45.5	40.9	33.8	12.7
B10	15	0.0	0.0	0.0	0.0
	30	2.5	0.0	0.0	0.0
	60	32.2	21.8	3.4	0.0
B15	15	0.3	0.0	0.0	0.0
	30	19.2	10.2	4.0	0.1
	60	28.9	25.8	12.8	2.7
B20	15	6.5	0.4	0.0	0.0
	30	29.8	20.1	4.7	0.7
	60	33.2	22.2	11.6	2.8

and AQ2. To mitigate this threat, a consolidated definition of accuracy of a prediction and a way of measuring it could be defined.

Regarding the internal validity, the main threat is that the obtained results related to the immediacy of the proposal could be biased. This is because its interpretation can be subjective since it depends on the business which is analyzed. To mitigate it, other business experts can be consulted in order to state what is a appropriate period for considering a prediction to be instantaneous.

Finally, the external validity considers in how far the obtained results could be generalized to any business. This generalization is threatened by the fact that the beauty salon was the unique real scenario which was studied. Although a set of synthetic models of a range of complexities has been included in the experiment, other real scenarios can be considered to replicate this study in order to mitigate this threat.

6 Discussion and limitations

The Declare language [27] has been extended in several works [19,25,26,44]. In fact, ConDec-R is based on the time extensions defined in [25,44] where it is possible to define time lags over the different Declare constraints. Furthermore, the data-aware extension which has been proposed in [26] is considered in the current approach. This way, with the proposed declarative language, the considered problems can be modeled in an easy way, since it is based on high-level constraints. Moreover, realistic problems can be managed, e.g., the beauty salon detailed in Sect. 4.

Regarding proposals on time prediction, a probabilistic time-aware workflow system for time prediction is presented in [11]. However, the focus of [11] is more on scheduling, and, unlike the current approach, assumes that the workflow is known beforehand and stable [41]. Moreover, both [11,12] provide design-time support (i.e., before the enactment time), whereas the proposed approach provides run-time support as well (i.e., support during the enactment of instances). Similarly, [42] proposes a service that predicts the completion time of process instances by using nonparametric regression. In addition, [41] proposes the application of process mining to an event log in order to obtain a transition system. In a related way, [4] presents an approach for predicting process remaining time based on query catalogs. Such catalogs are groups of partial traces (annotated with additional information about each partial trace) that have occurred in an event log, and are then used to estimate the remaining time of new executions of the process. Although the interaction between process instances and the availability of resources constitute important factors for predicting the remaining time until completion, the proposals presented in [4,37,41,42] do not consider the number of instances being executed and the resources available at a specific point of time during process enactment when making this prediction. Furthermore, unlike in our approach, the predictions cannot be adapted to changing circumstances (e.g., a resource became unavailable during the process enactment). Similar to our approach, [34,37] consider the enactment of multiple instances and resource availabilities when making the prediction. Moreover, in such approaches the predictions can be adapted to changing circumstances. Nevertheless, as opposed to the current approach, [34,37] do not perform optimization over objective functions and do not start from a declarative model.

However, our approach also presents a few limitations. The predictions are generated by considering estimated values for the number of instances to execute, and hence, our proposal is only appropriate for processes in which this number is known a priori. In a related way, activity attributes and resource availability need to be estimated. As a real example, the beauty salon problem is detailed and an extensive empirical evaluation is carried out with the goal of supporting the contributions of our proposal. Moreover, some of our previous works also dealt with this kind of scenarios (e.g., [1] describes a travel agency problem and [3] considers computer support for clinical guidelines as an application example). Nevertheless, if the actual values of the estimates deviate from the estimated values during the execution of the model, P&S techniques can be applied to replan the activities and to update the predictions at run-time by considering the actual values of the estimates.

In addition, motivated by the requirements of the considered scenarios, the data perspective which is considered in the current approach mainly includes data constraints which can be applied to input data and activity relations. However, more advanced features like dynamic data or data-flow perspective have been left out since they are not part of the design requirements of the considered scenarios and will be addressed in future work when applying our proposal to BPs with different characteristics.

7 Conclusions and future work

In this work, an approach for generating time predictions of running process instances related to a multi-perspective constraint-based process model is proposed. For performing such predictions, we propose generating optimized enactment plans from a multi-perspective constraint-based process model and from the current state of partially executed process instances by considering a given objective function. This approach has several advantages regarding previous related work: (1) multiple process instances as well as the allocation of resources are considered, (2) is able to adapt to changing circumstances, and (3) besides predicting the remaining time of a specific process instance, it allows the prediction of other relevant issues. To evaluate the applicability of our approach in practical settings, we applied it to a real process scenario. Despite the high complexity of the considered problems, results indicate that our approach produces a satisfactory number of good solutions in a reasonable time.

As for future work, we will consider to deal with both paradigms imperative and declarative for the specification of the BPs. Furthermore, it is planned to consider the information from past process executions as additional input data for providing more accurate predictions. Additionally, we intend to apply it to real scenarios from other domains. Finally, further aspects of data perspective (e.g., the data-flow) are planned to be considered in future versions of ConDec-R.

A. Method for generating optimized BP enactment plans

In this work, we consider the constraint-based system IBM ILOG CPLEX Optimization Studio (CPLEX) [18] together with the CPLEX CP Optimizer complement [17] since they provide for efficient mechanisms to deal with scheduling problems as well as temporal constraints. This appendix explains how COPs with a single objective function are created for a multi-perspective constraint-based specification (cf. A.1) and then modeled in CPLEX (cf. Section A.2: Global constraints using CPLEX).

A.1. From ConDec-R model to a COP

This section explains how to transform the ConDec-R model as COP which can be solved using a search algorithm to obtain a set of optimized enactment plans.[19]

Figure 12 depicts the different steps of the method. The elements of the multi-perspective constraint-based specification (cf. Fig. 12a) need to be transformed into a COP (cf. Fig. 12b).

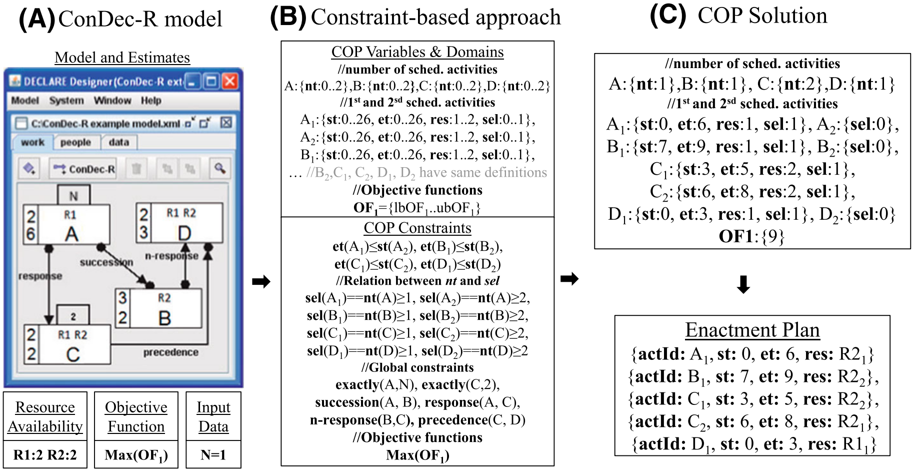
Definition A.1 A **scheduling activity** $sa = (st, et, res, sel)$ related to a BP activity $BPAct = (a, Res, Atts)$ represents a specific instance of $BPAct$, where st and et are CSP variables indicating the start and the end times of such execution sa, respectively, $res \in Res$ is a CSP variable representing the resource used for its execution, and sel is a CSP variable indicating whether or not sa is selected to be executed.

For this, BP activities—which can be executed arbitrarily often if not restricted by any constraint—are modeled as sequences of optional scheduling activities. This is required since each execution of a BP activity is considered as one single activity which needs to be allocated to a specific resource and temporarily placed in the enactment plan, i.e., stating values for its start and end times. For each BP activity, nt_{MAX} scheduling activities exist, which are added to the CSP model as decision variables.

In a formal way, a ConDec-R process model $CM(BPActs, Data, C_{BP}, AvRes, OF)$ (cf. Definition 2.3) is transformed into a COP $P_o(V, D, C_{CSP}, OF)$ (cf. Definition 2.7) where:

1. The set of variables V comprises all CSP variables related to the possible executions of the BP activities (i.e., the scheduling activities) plus the CSP variables related to the objective function, i.e., $V = \{nt(act), act \in BPActs\} \cup \{st(act_i), et(act_i), res(act_i), sel(act_i), i \in [1..\, nt_{MAX}(act)], act \in BPActs\} \cup OFs$.

[19] The transformation method has been already introduced, discussed and evaluated in previous works (The reader is referred to [21] for deeper details of the method).



Resource Availability	Objective Function	Input Data
R1:2 R2:2	Max(OF_1)	N=1

COP Variables & Domains

//number of sched. activities
A:{**nt**:0..2},B:{**nt**:0..2},C:{**nt**:0..2},D:{**nt**:0..2}
//1st and 2sd sched. activities
A_1:{**st**:0..26, **et**:0..26, **res**:1..2, **sel**:0..1},
A_2:{**st**:0..26, **et**:0..26, **res**:1..2, **sel**:0..1},
B_1:{**st**:0..26, **et**:0..26, **res**:1..2, **sel**:0..1},
… //B_2,C_1, C_2, D_1, D_2 have same definitions
//Objective functions
OF_1={$lbOF_1$..$ubOF_1$}

COP Constraints

et(A_1)≤**st**(A_2), **et**(B_1)≤**st**(B_2),
et(C_1)≤**st**(C_2), **et**(D_1)≤**st**(D_2)
//Relation between *nt* and *sel*
sel(A_1)==**nt**(A)≥1, **sel**(A_2)==**nt**(A)≥2,
sel(B_1)==**nt**(B)≥1, **sel**(B_2)==**nt**(B)≥2,
sel(C_1)==**nt**(C)≥1, **sel**(C_2)==**nt**(C)≥2,
sel(D_1)==**nt**(D)≥1, **sel**(D_2)==**nt**(D)≥2
//Global constraints
exactly(A,N), **exactly**(C,2),
succession(A, B), **response**(A, C),
n-response(B,C), **precedence**(C, D)
//Objective functions
Max(OF_1)

//number of sched. activities
A:{**nt**:1},B:{**nt**:1}, C:{**nt**:2},D:{**nt**:1}
//1st and 2sd sched. activities
A_1:{**st**:0, **et**:6, **res**:1, **sel**:1}, A_2:{**sel**:0},
B_1:{**st**:7, **et**:9, **res**:1, **sel**:1}, B_2:{**sel**:0},
C_1:{**st**:3, **et**:5, **res**:2, **sel**:1},
C_2:{**st**:6, **et**:8, **res**:2, **sel**:1},
D_1:{**st**:0, **et**:3, **res**:1, **sel**:1}, D_2:{**sel**:0}
OF1:{9}

Enactment Plan

{**actId:** A_1, **st:** 0, **et:** 6, **res:** $R2_1$}
{**actId:** B_1, **st:** 7, **et:** 9, **res:** $R2_2$},
{**actId:** C_1, **st:** 3, **et:** 5, **res:** $R2_2$},
{**actId:** C_2, **st:** 6, **et:** 8, **res:** $R2_1$},
{**actId:** D_1, **st:** 0, **et:** 3, **res:** $R1_1$}

Fig. 12 Generating optimized BP enactment plans from a ConDec-R model

2. D is composed of the domains of each CSP variable, where $UB(var)$ and $LB(var)$ represent the upper and lower bounds of the domain of var, respectively.
3. C_{CSP} is composed of the resource constraints and the global constraints related to C_{BP} together with the constraints which are inherent to the proposed model, which are listed as follows:
   (a) A specific execution of a BP activity precedes the next execution of the same activity, i.e, $\forall act \in BPActs\ \forall i : 1 \leq i < nt(act) : et(act_i) \leq st(act_{i+1})$.
   (b) The nt variable is directly related to the sel variables of the associated scheduling activities, i.e., $\forall act \in BPActs\ \forall i : 1 \leq i \leq UB(nt(act)) : sel(act_i) == (nt(act) >= i)$.
   (c) The start and the end times of each scheduling activity are related by the estimated duration of the associated BP activity, i.e., $\forall act \in BPActs\ \forall i : 1 \leq i \leq nt(act) : st(act_i) + duration(act) == et(act_i)$.
4. The optimization of the considered objective function is explicitly stated, i.e., $optimize(OF)$.[20]

Resource constraints are not explicitly stated since most constraint-based systems (e.g., IBM ILOG CPLEX Optimization Studio [18]) provide high-level mechanisms specific to scheduling which allow for an efficient management of shared resources.

The solution which is obtained by solving the COP is directly considered as the optimized enactment plan (cf. Definition 2.4) since the solution contains all the required information for the enactment plan (cf. Fig. 12c).

As stated, the proposed approach is based on complete search algorithms, which explore a search tree for the COP problem based on all possible combinations of assignments of values to the COP variables. However, since the generation of optimal plans presents NP complexity [14], it is not possible to ensure the optimality of the generated plans for all cases. Therefore, the developed algorithm will look for the best solution during a given time limit.

[20] The optimization can be either maximization or minimization.

The developed constraint-based approach, however, allows solving the considered problems in an efficient way, as demonstrated in Sect. 5.

A.2. Global constraints using CPLEX

The CPLEX implementation covers: (1) the decision variables related to the COP (cf. Section A.2.1: Decision variables) and (2) the constrains over such variables using global constraints (cf. Section A.2.2: Constraints).

A.2.1. Decision variables

To state the scheduling model, CPLEX provides a high-level object which can be used to encapsulate the CSP variables of each scheduling activity a_i (cf. Definition A.1). Such object is called *interval* variable. An interval variable has a start (which corresponds to the CSP variable $st(a_i)$), an end (which corresponds to the CSP variable $et(a_i)$) and a length (which corresponds to the estimated duration of the corresponding BP activity). An interval variable allows for these values to be variable within the model. Moreover, an interval variable may be optional, and whether or not an interval is present in the solution is represented by a decision variable (that corresponds to the CSP variable $sel(a_i)$). In addition, for supporting hierarchical activities, interval variables which do not require a resource and which wrap the related scheduling activities are used. Therefore, in order to model the scheduling activities in CPLEX, a three-dimensional array of interval variables is created. The first dimension corresponds to the different BP activities that exist in the model plus the hierarchical activities. The second dimension is related to the number of possible repetitions of such BP activities. And finally, the third dimension is related to the number of process instances (e.g., number of clients, patients, etc.).

Regarding the management of resources, CPLEX includes cumulative function expressions, which can be used to model resource usage functions over time. These functions can be computed as a sum of interval variable demands on a resource over time. A cumulative function expression can be constrained to model limited resource capacity by constraining that the function be $\leq$ the capacity, that is stated considering the resource availability. In addition, CPLEX provides a mechanism to model the fact that an alternative set of roles can perform an activity. For this, an array of optional intervals can be associated with the execution of an activity. To be more precise, an array of interval variables is created for each scheduling activity. The size of that array is the number of alternative roles that can execute the related BP activity. In addition, a cumulative function variable is created for each role existing in the model.

Lastly, the management of multiple instances is carried out in CPLEX through a high-level object called sequence. A sequence represents an aggregate of intervals which are execute on sequence, e.g., by the same resource, in the same instance, etc. Therefore, some efficient constraints and expressions over sequences can be easily included in the model. Specifically, for each process instance, the intervals related to its scheduling activities are aggregated in a sequence variable.

A.2.2. Constraints

To increase the efficiency of the search for solutions of the COP, the proposed approach models each COP constraint with a global constraint according to a well-known catalog of

Table 7 Relation between the features of ConDec-R and the elements provided by CPLEX

	Precedence constraints	Alternative constraints	Span constraints	Cumulative expressions	Data separation	Decision variable
Existence constraints						✓
Relation constraints	✓					✓
Negation constraints						✓
Choice constraints		✓				
Temporal constraints	✓		✓			✓
Data constraints					✓	✓
Durations						✓
Alternative resource		✓				
Resource Availability				✓		
Hierarchy			✓			✓

global constraints [16].[21] Such global constraints are implemented through efficient filtering rules by using high-level objects and high-level constraints provided by CPLEX. To be more precise, the proposed global constraints are based on establishing high-level constraints on the interval, sequence and cumulative function variables which were stated above. The high-level constraints provided by CPLEX are detailed as follows:

- precedence constraints, which ensure the relative positions of intervals in the solution,
- no overlap constraints, which ensure that positions of intervals within a sequence in the solution are disjointed in time,
- span constraints, which ensure that one interval covers the intervals included in a set of intervals,
- alternative constraints, which ensure that exactly one of a set of intervals be present in the solution, and
- cumulative expression constraints, which restrict the bounds on the domains of cumulative function expressions.

This way, all supported features of ConDec-R are modeled through the high-level constraints and objects provided by CPLEX as follows (cf. Table 7):

- The existence constraints are modeled through the presence decision variable of intervals.
- The relation constraints involve several high-level constraints. To be more precise, the relative order between activities of such constraints is modeled using precedence constraints. The presence decision variables are used to state whether or not the constraint has to take place. For example, the constraints *precedence(A,B)* and *response(A,B)* state a similar order relation between *A* and *B*. However, the constraint must be fulfilled only if *A* is present or *B* is present, respectively.
- The negation constraints involve the same modeling mechanisms as existence and relation constraints.
- Choice constraints are modeled through alternative constraints. Given a set of intervals related to some activities, such constraints allow stating the number of these intervals that may be present in the solution.

[21] Note that this catalog is independent of any constraint-based language but followed by many of them.

- Temporal constraints involve a temporal aspect in the existence, relation and negation constraints. For this, the span constraints are used to state the periods of time which affect the constraints.
- Data constraints use the data separation features of CPLEX models. Such feature allows to have a separated file where the input data are specified. Then, these data can be easily included in the model affecting the decision variables which are needed.
- The constraints related to the duration of the activities are modeled through the length decision variable of the intervals.
- The alternative resource constraints are modeled through the alternative constraints to indicate that one interval of a set must be present if some activity is executed. Then, such set of intervals are related to the number of available resources.
- The resource availability is modeled using cumulative expression to limit the units of a role that can work in parallel.
- To allow for a hierarchical reasoning, the intervals which are created for each hierarchical activity wrap all the related activities through span constraints.

With the suggested global constraints, the constraints contained in the ConDec-R specification can be easily included. Moreover, such constraints increase the efficiency of the search for solutions since the related filtering rules remove inconsistent values from the domains of the variables during the search process. In the CSP model, initial estimates considering activity durations and existence constraints are made for upper and lower bounds of variable domains, and these values are refined during the search process.

References

1. Barba I, Del Valle C, Weber B, Jimenez-Ramirez A (2013) Automatic generation of optimized business process models from constraint-based specifications. Int J Cooper Inf Syst 22(2):1350009
2. Barba I, Weber B, Del Valle C, Jimenez-Ramirez A (2013) User recommendations for the optimized execution of business processes. Data Knowl Eng 86:61–84
3. Barba I, Lanz A, Weber B, Reichert M, del Valle C (2012) Optimized time management for declarative workflows. In: Enterprise, business-process and information systems modeling, volume 113 of LNBIP. Springer, Berlin, pp 195–210
4. Bolt A, Sepúlveda M (2013) Process remaining time prediction using query catalogs. In: Proceedings of BPI, pp 54–65
5. Brereton P, Kitchenham B, Budgen D (2008) Using a protocol template for case study planning. In: Proceedings of EASE 2008. BCS-eWiC
6. Brucker P, Knust S (2006) Complex scheduling (GOR-publications). Springer, Secaucus, NJ
7. Business Process Model and Notation (BPMN), Version 2.0. http://www.omg.org/spec/BPMN/2.0/ (2011). Accessed 01 May 2014
8. Dumas M, van der Aalst WMP, ter Hofstede AH (eds) (2005) Process-aware information systems: bridging people and software through process technology. Wiley-Interscience, Hoboken, NJ
9. Eder J, Panagos E, Rabinovich M (1999) Time constraints in workflow systems. In: Advanced information systems engineering, volume 1626 of LNCS, pp 286–300
10. Eder J, Pichler H (2002) Duration histograms for workflow systems. In: Proceedings of IFIP TC8/WG8.1 working conference on engineering information systems in the internet context, pp 239–253
11. Eder J, Pichler H (2005) Probabilistic calculation of execution intervals for workflows. In: Proceedings of TIME 2005, pp 183–185
12. Eder J, Pichler H, Gruber W, Ninaus M (2003) Personal schedules for workflow systems. In: Proceedings of BPM, pp 216–231
13. Gantt HL (1913) Work, wages, and profits. Engineering Magazine Co., New York
14. Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of NP-completeness. W. H. Freeman & Co., New York, NY
15. Ghallab M, Nau D, Traverso P (2004) Automated planning: theory and practice. Morgan Kaufmann, Amsterdam

16. Global Constraint Catalog. http://sofdem.github.io/gccat/. Accessed 20 June 2017
17. IBM. CPLEX CP Optimizer. http://www-01.ibm.com/software/commerce/optimization/cplex-cp-optimizer (2016). Accessed 11 July 2016
18. IBM. IBM ILOG CPLEX Optimization Studio. http://www-03.ibm.com/software/products/en/ibmilogcpleoptistud/ (2016). Accessed 11 July 2016
19. Jimenez-Ramirez A, Barba I, Weber B, Del Valle C (2015) Generating optimized configurable business process models in scenarios subject to uncertainty. Inf Softw Technol 57:571–594
20. Jimenez-Ramirez A, Barba I, del Valle C, Weber B (2013) Generating multi-objective optimized business process enactment plans. In: Advanced information systems engineering. Volume 7908 of LNCS. Springer, Berlin, pp 99–115
21. Jimenez-Ramirez A, Barba I, Del Valle C, Weber B (2013) OptBPPlanner: automatic generation of optimized business process enactment plans. In: Proceedings of ISD. Springer US, pp 429–442
22. Lanz A, Weber B, Reichert M (2012) Time patterns for process-aware information systems. Requir Eng 19:113–141
23. Leitner P, Wetzstein B, Rosenberg F, Michlmayr A, Dustdar S, Leymann F (2010) Runtime prediction of service level agreement violations for composite services. In: Service-oriented computing. ICSOC/ServiceWave 2009 workshops, volume 6275 of LNCS. Springer, Berlin, pp 176–186
24. Marzolla M, Mirandola R (2007) Performance prediction of web service workflows. In: Software architectures. components, and applications, volume 4880 of LNCS. Springer, Berlin, pp 127–144
25. Montali M (2009)Specification and verification of declarative open interaction models: a logic-based approach. Ph.D. thesis, Department of Electronics, Computer Science and Telecommunications Engineering, University of Bologna
26. Montali M, Chesani F, Mello P, Maggi FM (2013) Towards data-aware constraints in declare. In: Proceedings of the 28th annual ACM symposium on applied computing. SAC '13, pp 1391–1396
27. Pesic M (2008) Constraint-based workflow management systems: shifting control to users. Ph.D thesis, Eindhoven University of Technology, Eindhoven
28. Pesic M, Schonenberg MH, Sidorova N, van der Aalst WMP (2007) Constraint-based workflow models: change made easy. OTM Conf 1:77–94
29. Polato Mi, Sperduti A, Burattin A, de Leoni M (2014) Data-aware remaining time prediction of business process instances. In: Proceedings of IJCNN, pp 816–823
30. Process Specification Language project. http://www.nist.gov/psl/ (1977). Accessed 1 May 2014
31. Rossi F, van Beek P, Walsh T (eds) (2006) Handbook of constraint programming. Elsevier, London
32. Reichert M, Weber B (2012) Enabling flexibility in process-aware information systems. Springer, Berlin
33. Reijers H (2006) Case prediction in bpm systems: a research challenge. J Korean Inst Ind Eng 33:1–10
34. Rozinat A, Wynn MT, van der Aalst WMP, ter Hofstede AHM, Fidge C (2008) Workflow simulation for operational decision support using design, historic and state information. In: Proceedings of BPM, vol. 5240, pp 196–211
35. Russell N, van der Aalst WMP, ter Hofstede AHM, Edmond D (2005) Workflow resource patterns: identification, representation and tool support. In: Proceedings of CAiSE, pp 216–232
36. Salido MA (2010) Introduction to planning, scheduling and constraint satisfaction. J Intell Manuf 21(1):1–4
37. Schellekens B (2009) Cycle time prediction in staffware. Master's thesis, University of Technology, Eindhoven
38. Semmelrodt Franziska, Knuplesch D, Reichert M (2014) Modeling the resource perspective of business process compliance rules with the extended compliance rule graph. In: Enterprise, business-process and information systems modeling. Springer, pp 48–63
39. Souki M (2011) Operating theatre scheduling with fuzzy durations. J Appl Oper Res 3:177–191
40. Stroppi LJR, Chiotti O, Villarreal PD (2011) A bpmn 2.0 extension to define the resource perspective of business process models. In: XIV Iberoamerican conference on software engineering, pp 25–38
41. van der Aalst WMP, Schonenberg MH, Song M (2011) Time prediction based on process mining. Inf Syst 36(2):450–475
42. van Dongen BF, Crooy RA, van der Aalst WMP (2008) Cycle time prediction: when will this case finally be finished? In: Proceedings of CoopIS 2008 5331(I), pp 319–336
43. Weske M (2007) Business process management: concepts, languages, architectures. Springer, Berlin
44. Westergaard M, Maggi FM (2012) Looking into the future: using timed automata to provide a priori advice about timed declarative process models. In: International conference on cooperative information systems (CoopIS 2012)
45. Zur Muehlen M (2004) Organizational management in workflow applications-issues and perspectives. Inf Technol Manag 5(3–4):271–291

46. Zhang Y, Su R, Li Q, Cassandras CG, Xie L (2017) Distributed flight routing and scheduling for air traffic flow management. IEEE Trans Intell Transp Syst 18(99):1–12
47. Zhang X, Margellos K, Goulart P, Lygeros J (2013) Stochastic model predictive control using a combination of randomized and robust optimization. In: 52nd IEEE conference on decision and control, pp 7740–7745