

Optimizing Process Model Redesign

Akhil Kumar¹✉ and Paronkasom Indradat²

¹ Smeal College of Business, Penn State University,
University Park, PA 16802, USA
akhil@psu.edu

² Department of Industrial Engineering, Penn State University,
University Park, PA 16802, USA
pxi5005@psu.edu

Abstract. In recent years there has been considerable interest in business process redesign. A process model may be redesigned by combining various tasks and services according to best practices so as to satisfy predefined business rules and constraints to achieve a specific purpose. This purpose may be stated in terms of functional goals (such as desired or acceptable process behavior) and non-functional goals like cost, time and quality of service. There are many ways to redesign a process instance by applying improvements such as: making a task optional, replacing a task by another faster task (or service), task postponement, task combination, task splitting, task restructuring, etc. Given many such alternatives, there is no systematic way of evaluating their costs and benefits, and the tradeoffs among them. We describe a novel approach based on a formal model to optimize the “benefits” or net effects of a redesign with respect to a baseline design and show how it can be used to evaluate and compare alternative models at both design and run time.

1 Introduction

Organizations are constantly trying to improve their business processes to make them more efficient in terms of time, cost, quality and flexibility [6, 11, 13], and also to deal with exigencies. Thus, under higher workload conditions an Australian insurance service company may decide to escalate its claim handling by collecting less information than they normally do, e.g. during the storm season when the call volume doubles [18]. Escalation may involve changing the routing of work, the work distribution, or the requirements with respect to available data. Such temporary or periodic redesign measures are necessary to maintain the service quality during a busy season, while other redesigns may be permanent. In a similar vein, a car rental company may decide to make the car wash task before renting out a car optional or replace it by an express wash when the demand is too high or there is a resource shortage. Clearly, this can save on the cost and time of a car wash but it may hurt customer service.

A bank that normally requires two officers to approve a mortgage application may instead have only one officer approve them when the workload is very high. In this situation, it is possible that the quality of the customer service will not suffer but the likelihood of making a bad loan may go up and may impact profitability. Sometimes there are alternative designs for the same process. For example, the bank may choose to

use a faster credit appraisal service that does appraisals in 3 days instead of the normal 6 days but at a higher cost than the normal cost. Thus, there is a tradeoff between the two scenarios that has both cost and time implications. The bank must decide whether to pursue scenario 1, scenario 2 or both scenarios. In a medical context, there may be a tradeoff between a normal test that takes 5 days and costs \$100, and an expedited test that takes 3 days and costs \$300. In such a situation, deciding whether to select the normal test or the expedited test is also an optimization issue.

Our goal in this paper is to develop a way to model such process design scenarios and find the optimal design in view of business constraints. Hence, the ideal design will depend on the actual realities of a dynamic situation. In the literature there have already been efforts to develop best practices and heuristics to improve processes, notably by Mansar and Reijers [11]. Our work is in part inspired by these ideas, and we wish to apply them in the context of a formal model that can help us determine and recommend the best design that satisfies cost and time constraints. We also build upon our previous work on approaches for modeling and optimizing temporal workflows [8] by extending that model to add support for redesign. In particular we show how to add support in our model for 6 different types of process improvements: *optional tasks*; *task replacement*; *task restructuring* (from sequence to parallel); *task combination*; *task splitting*; and *task postponement*.

In qualitative terms we can see that each alternative scenario in the situations described above represents a clear tradeoff with other scenarios. But it is also important to have a mechanism to evaluate and compare them systematically. We will show how such tradeoffs can be evaluated and optimized using metrics like cost, time, quality and flexibility. To the best of our knowledge this paper is a first effort towards a formal approach for evaluating alternative options for redesign.

The main contributions of this paper are as follows: First, we develop a new approach to model process improvement alternatives correctly. Second, we show how these different scenarios can be evaluated in terms of key metrics like cost, quality, time and flexibility. Third, we show how the scenarios and the metrics can be combined into an optimization model. Fourth, we present results of analysis using a realistic case study. Finally, we also describe an implementation approach for our proposal.

This paper is organized as follows. In Sect. 2 we discuss a basic model for describing temporal constraints and show how it can be translated into structural and temporal constraint equations. Then, in Sect. 3, we describe common improvement scenarios based on best practices. Next, Sect. 4 develops an optimal redesign model using evaluation metrics. Section 5 illustrates our approach with a detailed case study. Later, Sect. 6 discusses the main features and limitations of our approach, and Sect. 7 concludes the paper with some thoughts for future work.

2 Preliminaries

To be able to evaluate redesign alternatives the first step is to have a formal method to describe the control flow of a model and the temporal constraints for each task. In this section we describe a simple temporal model and show how structural and temporal constraints are represented.

2.1 A Simple Temporal Model

A temporal model of a process is made by combining two types of constraints: (1) structural constraints, and (2) temporal constraints. The structural constraints capture the control flow of the process to coordinate the proper sequence in which the tasks occur. The temporal flow model considers the permitted durations of each activity and the minimum or maximum gaps between them.

Definition 1. A general temporal process model TP can be represented as:

$$TP = (T, A, X, E, TD, TI)$$

Where

T: set of task nodes, T_1, T_2, \dots

A: set of AND control nodes, A_1, A_2, \dots

X: set of XOR control nodes, X_1, X_2, \dots

E: set of edges among the nodes in $\{T, A, X\}$

TD: set of task duration ranges: $\{(T_i, D_{i_min}, D_{i_max}), \dots\}$, where $D_{i_min}, D_{i_max} \in \mathbb{R}^+$

TI: set of additional inter-task constraints: $\{(T_i, T_j, S|F, S|F, TI_{i_min}, TI_{i_max}), \dots\}$, $TI_{i_min}, TI_{i_max} \in \mathbb{R}^+$

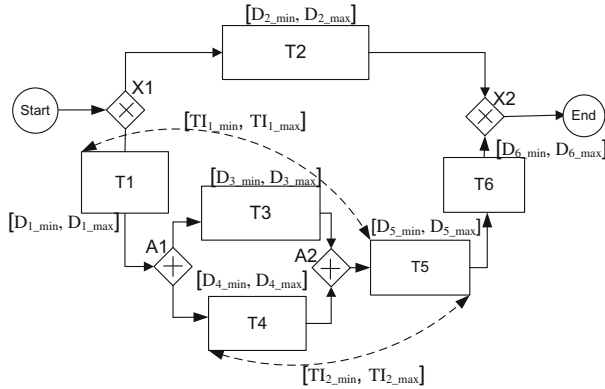


Fig. 1. A basic temporal model with XOR and AND connectors

Figure 1 is an example of a simple temporal model. It shows the control flow, along with $[min, max]$ durations of each task and inter-task constraints. It is expressed as:

$T : \{T_1, T_2, \dots, T_6\}$

$A : \{A_1, A_2\}$

$X : \{X_1, X_2, X_3, X_4\}$

$$\begin{aligned}
E &: \{(\text{start}, X_1), (X_1, T_1), (X_1, T_2), (T_1, A_1), (A_1, T_3), (A_1, T_4), (T_3, A_2), (T_4, A_2), \dots\} \\
TD &: \{(T_1, D_{1_min}, D_{1_max}), (T_2, D_{2_min}, D_{2_max}), (T_3, D_{3_min}, D_{3_max}), \dots\} \\
TI &: \{(T_1, T_5, S, S, TI_{1_min}, TI_{1_max}), (T_4, T_5, S, F, TI_{2_min}, TI_{2_max})\}
\end{aligned}$$

In addition to the time intervals of each task, an inter-task constraint (TI) can also be represented by a dashed line connecting the start or end of a task to the start or end of another task. For example, in Fig. 1, the inter-task constraint TI_2 between T_4 and T_5 requires that the elapsed time from the start of T_4 until the end of T_5 must lie in the $[TI_{2_min}, TI_{2_max}]$ interval. Also note that while we only consider task and inter-task durations, fixed time activities can also be modeled by setting their relative time with respect to the start of a process and converting them into delays with respect to the start activity.

2.2 Structural and Temporal Constraints

Next we show how to map the above model into structural and temporal constraint equations that can be solved using a constraint satisfaction approach. The flow constraints capture the coordination sequence among tasks, while the temporal constraints specify the task and inter-task durations.

Structural Constraints (SC). Structural constraints are represented by structural equations to capture the flow of a process. In doing so, each task, and also the start and end tasks, are treated as binary 0–1 variables (where 1(0) denotes the presence (absence) of a task in a process instance). The structural balance equations for sequence, choice and parallel patterns expressed in terms of their corresponding variable names are shown in Fig. 2. These equations describe the correct behavior of a workflow consisting of various structures. A sequence structure (see row 1 of Fig. 2) requires that two sequential tasks T_1 and T_2 must have the same value, i.e. $T_1 = T_2 = 0$; or $T_1 = T_2 = 1$. Further, at a choice-split node, the balance equation ensures that when a choice-split is activated, only one outgoing branch becomes active but not both (see row 2 of Fig. 2). The behavior at a choice-join node (row 3) forces $X_2 = 1$ only when exactly one but not both of T_1 and T_2 are 1. Note that in row 3, M is a very large number (say, 10000), and Y is an auxiliary variable used to hold a temporary binary value that in turn is used to determine X_2 . Rows 4 and 5 capture the correct behavior at an AND-split or-join node.

Similarly one could describe mandatory, prohibited, co-existing, exclusive and constrained choice patterns. As discussed in [8] a complete structural process model is one that includes: (a) one equation that captures the link of each task T_i (or connector X_i, A_i) to its preceding task(s) and/or connector(s) unless T_i is the first task in the process; and (b) one equation that captures the link of each task T_i (or connector X_i, A_i) to its succeeding task(s) and/or connector(s) unless T_i is the last task in the process. By solving the system of equations simultaneously for a sound process model a solution for an instance of the process is found with values for T_i, X_i , and A_i variables.

Temporal constraints (TC). The temporal constraints express a variety of temporal relationships. Here we consider three types of constraints: flow, task duration and inter-task gap constraints. Temporal Flow (TF) constraints are derived from the edge set E . For every node n_i and successive node pair (n_i, n_j) in E , we add two constraints as:

$$TS_i \leq TF_i$$

$$TF_i \leq TS_j$$

where

TS_i : start time of node i relative to the start time of the workflow instance

TF_i : finish time of node i relative to the start time of the workflow instance


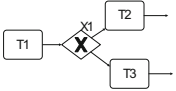
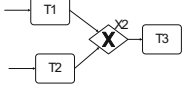
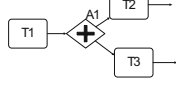
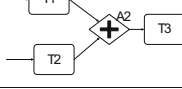
	Structure	Representation	Constraint equation
1.	Sequence		$T2 = T1;$
2.	Choice-Split		$X1 = T2 + T3;$
3.	Choice-Join		$T1 + T2 - 1 \leq M(1 - Y);$ $2 - T1 - T2 \leq M * Y;$ $X2 \leq T1 + T2; X2 \leq Y$ $X2 \geq T1 - T2; X2 \geq T2 - T1;$
4.	Parallel-Split		$T2 = A1; T3 = A1;$
5.	Parallel-Join		$A2 \geq T1 + T2 - 1;$ $A2 \leq T1; A2 \leq T2;$

Fig. 2. Structural balance equations for process modeling structures

The Task duration (TD) constraints ensure that the duration of a task T_i lies between the permitted range $[D_{i_min}, D_{i_max}]$. They are specified as: $D_{i_min} \leq TF_i - TS_i \leq D_{i_max}$. The Inter-task (TI) constraints ensure that the gap or delay between the start (end) of an activity pair (i, j) lies in the permitted $[G_{ij_min}, G_{ij_max}]$ range. They are specified as: $G_{ij_min} \leq TF_j (TS_j) - TS_i (TF_i) \leq G_{ij_max}$. Finally, duration constraints for X and A-split connectors are: $XF_i - XS_i = 0$; and $AF_i - AS_i = 0$, respectively. For A-join connectors, $AF_j = \text{Max}(TF_i), \forall TF_i$ s.t. $(TF_i, AF_j) \in E$. A solution of a (combined structural and temporal) process model is of the form:

$$\forall T_i, (T_i, TS_i, TF_i), T_i = 0 \text{ or } 1, TS_i, TF_i \in \mathbb{R}^+$$

$$\forall X_i, (X_i, XS_i, XF_i), X_i = 0 \text{ or } 1, XS_i, XF_i \in \mathbb{R}^+$$

$$\forall A_i, (A_i, AS_i, AF_i), A_i = 0 \text{ or } 1, AS_i, AF_i \in \mathbb{R}^+$$

if $T_i (X_i \text{ or } A_i) = 0$ then $TS_i (XS_i \text{ or } AS_i)$ and $TF_i (AS_i \text{ or } AF_i)$ are not valid.

It has been shown previously [8] that a process model is consistent if for every valid and complete execution path (from start to end) there exists a solution that satisfies the duration and inter-task constraints.

3 Process Redesign Strategies

In [11] various process improvement strategies to redesign a process are discussed. After analysis, we identified six important strategies that are amenable to our formal approach as shown in Table 1. The first row shows a baseline, existing process and subsequent rows show the effect of applying various strategies to it.

Table 1. Process improvement strategies

Improvement	Representation	T	C	Q
0.Normal process (default or baseline)				
1.Optional task (if $O_2 = 1$, duration of $T_2 = 0$)		+	+	-
2.Task Replacement (if $R_2 = 1$, duration of $T_2 = [R_{2min}, R_{2max}]$)		+	+	-
3.Task Combination (if $C_{12} = 1$, revise durations of T_1, T_2)		+	+	-
4.Task Splitting (if $S_2 = 1$, split T_2 into T_2 and T_2')		-	-	+
5.Parallelism (if $P_{23} = 1$, restructure T_2 and T_3 into a parallel structure)		+	+	-
6.Task postponement (delay T_1 until after T_2 and T_3)		+	+	-

Strategy 1 is to make a task optional so at run time it may be skipped (e.g. skip the car wash task when work load is high). Strategy 2 replaces a normal task with an alternative task (e.g. replace a regular car wash with an express wash). Strategies 3 and 4 combine two (or more) small tasks into one, and split a large task into multiple tasks, respectively. Strategy 5 aims to take two tasks in sequence and run them in parallel to save time. Finally strategy 6 would reorder the tasks in a process such that one task is postponed from its normal position and performed later. Along with the strategy we describe how it can be modeled by modifying the temporal model discussed above. The last three columns show whether the effect of the improvement on time, cost and quality metrics is positive or negative.

Table 2. Modifying constraints to capture redesign alternatives in the model:

Improvement	Benefit	Cost	Modified/New constraints
1. Make T_i optional (Variable $O_i = 1$)	Save time by skipping one approval	Quality may suffer from higher risk of error or poor service	Duration of $T_i = [D_{imin} - O_i * D_{imin}, D_{imax} - O_i * D_{imax}]$
2. Replace T_i with task R_i (Variable $R_i = 1$)	R_i takes Δ_r less time than T_i	R_i may cost more than T_i . It may also not be as reliable	Duration of $T_i = [D_{imin} - R_i * \Delta_r, D_{imax} - R_i * \Delta_r]$
3. Combine T_i, T_j into one task T_{ij} (variable $C_{ij} = 1$)	Finish early and save cost of resource for T_j	Quality may improve if two tasks are closely related. It may also suffer since one point of control is removed	Duration of $T_i = [D_{imin} + C_{ij} * \Delta_c, D_{imax} + C_{ij} * \Delta_c]$; Duration of $T_j = 0$
4. Split T_i into two tasks, T_i and T_i' (variable $S_i = 1$)	Break difficult task into two tasks for better results	Extra cost of handoff between tasks is incurred	Duration of $T_i = [D_{2min} - \Delta_{s1}, D_{2max} - \Delta_{s1}]$. Duration of $T_i' = [D_{2min} - \Delta_{s2}, D_{2max} - \Delta_{s2}]$
5. Change T_i and T_j from sequence to parallel (variable $P_{ij} = 1$)	Finish early since 2 tasks occur in parallel	Possible drop in quality since the two approvals are not in sequence	$TS_j \geq TF_i - M * P_{ij}$ $TS_j \geq TS_{i_pred} - (1 - P_{ij}) * M$ $TS_i \leq TF_{j_succ} - (1 - P_{ij}) * M$
6. Task Postponement (variable $PO_i = 1$)	Do task T_i later out of order. May be able to skip it.	Possible loss of information or accuracy from changing the order of a task	$TS_j \geq TF_i - M * PO_i$ [M is a very large number]

Next we show how these strategies can be captured into the modeling framework developed in the previous section. The modeling approach for each strategy is

described in Table 2. The last three columns show the positive or negative effect of applying each strategy on time (T), cost (C) and quality (Q) on the base model. To make task T_i optional, we introduce another task variable O_i . In addition we make the duration of T_i a function of variable O_i such that it will be 0 when $O_i = 1$. This is equivalent to skipping T_i . Similarly, we introduce a variable R_i to allow replacement of T_i . If $R_i = 1$, then the duration of T_i is adjusted by Δ_r to be the same as the duration of R_i . For task combination, two tasks T_i and T_j are combined into a new task T_i (with a duration $[D_{i\min} + C_{ij} * \Delta_c, D_{i\max} + C_{ij} * \Delta_c]$) and T_j with a duration of 0. In a task split ($S_i = 1$), we replace task T_i with two tasks T_i and T_i' and change their durations. If a variable $P_{ij} = 1$ then tasks T_i and T_j are restructured into a parallel structure. To do so the sequential constraint $TS_j \geq TF_i$ between T_i and T_j is relaxed to $TS_j \geq TF_i - M * P_{ij}$, where M is a very large number (say, 10000). Two more constraints are added to maintain the ordering relationship of T_i and T_j with their preceding (TS_{i_pred}) and succeeding (TS_{i_succ}) tasks. The effect of M in these constraints is to activate them only when $P_{ij} = 1$, and disable them when $P_{ij} = 0$.

For task postponement the ordering of tasks may be changed in a similar way. Thus, $T_1-T_2-T_3$ may be reordered as $T_2-T_3-T_1$ or as $T_1-T_3-T_2$. This requires relaxing the temporal relationship between a postponed task T_i and its successor task T_j . Table 2 summarizes the benefits and costs of each improvement discussed and also shows the modified or additional constraints. We will discuss the correctness of our approach later in the paper.

4 Building and Solving a General Optimal Redesign Model

Above we have described a general framework for incorporating various model improvements into a formal model. In this way a model can be designed by simply setting parameter values for O_i , R_i , C_{ij} , P_{ij} , S_i , PO_i , etc. suitably. However, in general we would like the model to give us an optimal solution that tells us which design to select particularly when several improvements are possible and they cannot be applied at the same time. This means that we need a metric to evaluate each design. In quantitative terms we wish to study the effect of key metrics like cost, time and quality (where a positive effect is good and a negative effect is bad). Thus, we could have an objective function to express the total benefit (or net effect) of a design as the weighted sum of the redesign variables:

$$\max Obj = \sum_i BO_i * O_i + BR_i * R_i + BS_i * S_i + \sum_{i,j} BC_{ij} * C_{ij} + BP_{ij} * P_{ij}$$

Where

BO_i = Benefit coefficient of parameter O_i

BR_i = Benefit coefficient of parameter R_i

BP_{ij} = Benefit coefficient of parameter P_{ij}

BC_{ij} = Benefit coefficient of parameter C_{ij}

BS_i = Benefit coefficient of parameter S_i

BPO_i = Benefit coefficient of parameter PO_i

Each BX_i or BX_{ij} term in the objective function above represents a net benefit of making the design change. Thus, each term captures the main factors of cost and

quality. The time factor is reflected in the model separately as we have seen above. Cost is already in dollar terms and would represent the savings if an optional task i is skipped or the increase/decrease if a task j is replaced by another task and would be a component of coefficient BO_i and BR_j , respectively. Another consideration in these coefficients is the quality factor. Skipping an optional task may hurt the quality of the process and lead to extra cost in repairs or loss because a task was skipped. This would be a second component of the BO_i and BR_j coefficients.

Min **Obj** = $W1 * Cost + W2 * Time + W3 * Quality$
 S.T.
 1) Structural constraints
 2) Temporal constraints
 3) Design improvement constraints
 4) Design integrity constraints
 5) Constraints on metrics, e.g.
 Total time < T_{max} ;
 Total Cost < C_{max} ;
 Quality > Q_{min} ;

Fig. 3. An optimization model for selecting the best redesign

The net effect of these two components would produce a value that may be negative or positive. A negative value of BO_i or BR_j may be compensated by a reduction in the throughput time of the process. For example, consider the effect of skipping task T_2 ('check credit report and appraisal') in the process of Fig. 4. This will clearly have a positive effect on cost (since a resource does not have to work on this task) but negative effect on quality (as it may raise the chances of making a bad loan). Say the imputed effect on cost is a savings of \$50. However, the negative effect on quality has an imputed value of \$100 based on the higher risk of a bad loan. Thus, $BO_2 = 50 - 100 = -50$. Similarly, say, task T_2 can be outsourced to another service that is faster by 5 h but costs more. In this case the quality does not suffer but there is an additional cost of \$25 in using this service. Hence, $BR_2 = -25$.

As we shall illustrate in the next section, the general problem can be framed in different ways by modifying the objective function and the set of constraints. The objective function can be a weighted sum (using weights $W1, W2, \dots$) of cost, time, quality, and flexibility terms. The constraints may be modified to impose various limits on these metrics as well. Then we can frame the problem in terms of a generic mixed integer linear programming (MILP) model as shown in Fig. 3.

5 A Case Study

In this section we discuss a realistic case to illustrate how our redesign optimization approach works. We first describe an example of a mortgage loan approval process, then show how it is modeled using our approach and finally, solve the model to create alternative redesign scenarios.

Consider the example in Fig. 4 of a loan approval process. In this process an application is received (T_1), then a credit report for the applicant and an appraisal report for the property are obtained (T_2) from an external service provider. Depending upon the contents of the reports, some applicants are automatically rejected and in these cases the instance follows the lower path at the XOR node. Along this path, a manager approves the rejection (T_4), an assistant prepares the rejection notice (T_5) and then the applicant is notified (T_{10}). If the reports are ‘OK’, then the instance follows the upper path at the XOR node. Along this path a financial officer makes a recommendation (T_3) and this is followed by two approvals (T_6 and T_7 - by a manager and a VP) in the next two steps. An assistant then prepares the notification (T_8) and it is sent to the applicant (T_{10}).

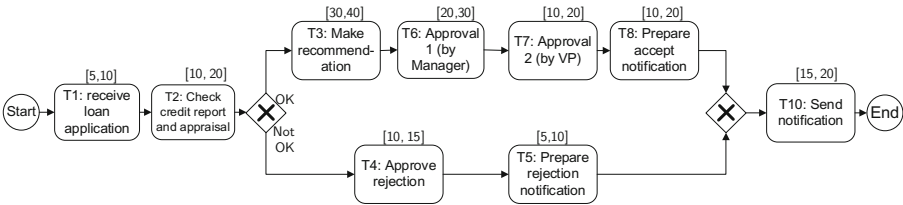


Fig. 4. An example base process model

There are several redesign possibilities in this process model as follows:

- Task T_2 may be replaced by another task R_2 that takes 5 time units less
- Tasks T_6 and T_7 are each optional (but not both together)
- Tasks T_6 and T_7 may be performed in parallel if they are not skipped
- Tasks T_6 and T_7 are combined into a new task T_{67} that takes 5 units more than T_6

Now in this situation, we are interested in creating scenarios that help us to answer the following kinds of questions by solving the optimization model at design/run time:

1. Find the design in which a process instance can finish in the fastest time?
2. Find the design in which a process instance gives the maximum benefit?
3. Find the maximum benefit design subject to an instance finish time limit?

Next we will show how the process of Fig. 4 can be modeled for redesign. Figure 5 (a) shows the original (partial) model and Fig. 5(b) shows the corresponding changes made to the constraint set to ensure that the above design constraints are satisfied. There are three important observations with respect to Fig. 5(b). First we modify the temporal flow constraints to model the effect of a change in the structure of T_i and T_j from sequence to parallel. This is achieved by relaxing the strict $TS_j \geq TF_i$ requirement by adding $-1000 * P_{ij}$ term to the right hand side. Thus, if P_{ij} is 1 then the sequential requirement is relaxed. Moreover, two other constraints are needed to ensure that the sequential relationships of T_i with the successor of T_j , and T_j with the predecessor of T_i , are maintained.

<pre> Minimize Obj. s.t. //Structural constraints (SC) Start. Start = 1; End. End = 1; SF0. T₁ = Start; SF1. T₂ = T₁; SF2. X₁ = T₂; SF3. T₃+T₄ = X₁; SF4. T₆ = T₃; T₇ = T₆; T₈ = T₇; ... //Temporal flow constraints (TF) TF1. TS₇ ≥ TF₆; TF2. TS₈ ≥ TF₇; ... //Temporal duration constraints (TD) TD1. 10 ≤ TF₂ - TS₂ ≤ 20; TD2. 20 ≤ TF₆ - TS₆ ≤ 30; TD3. 10 ≤ TF₇ - TS₇ ≤ 20; </pre>	<pre> Minimize Obj. s.t. //Modified temporal flow constraints (TF) TF1'. TS₇ ≥ TF₆ - 1000*P₆₇; TF2'. TS₇ ≥ TF₃ - 1000*(1 - P₆₇); TF2''. TS₈ ≥ TF₆ - 1000*(1 - P₆₇); ... //Modified duration constraints (TD) TD1'. TF₂ - TS₂ ≤ 20 - 5*R₂ - 20*O₂; TD2'. TF₆ - TS₆ ≤ 30 + 5*C₆₇ - 30*O₆; TD3'. TF₇ - TS₇ ≤ 20 - 20*C₆₇ - 20*O₇; ... // Add design integrity constraints C0. TS_i ≤ TF_i; (for i = 2, 6, 7) C1. P₆₇ = 1 → O₆ + O₇ = 0; C2. C₆₇ = 1 → O₆ + O₇ = 0; C3. P₆₇ + C₆₇ ≤ 1; C4. O₂ + R₂ ≤ 1; C5. C₆₇+O₆ ≤ 1; C₆₇+O₇ ≤ 1; </pre>
(a) Original model	(b) Modified model

Fig. 5. Original model and modifications needed to incorporate improvements

Second, we modify the duration of an optional task T_i based on whether it is skipped ($O_i = 1$), replaced by task R_i ($R_i = 1$), or combined with task T_j ($C_{ij} = 1$). Thus,

$$TF_i - TS_i \geq D_{i_min} - O_i * D_{i_min} - R_i * \Delta_{r1} + C_{ij} * \Delta_{c1}$$

$$TF_i - TS_i \leq D_{i_max} - O_i * D_{i_max} - R_i * \Delta_{r2} + C_{ij} * \Delta_{c2}$$

$$TF_i \geq TS_i$$

Third, additional constraints are needed to ensure the integrity of the design: The duration of a constraint must be non-negative (C_0). Also, if $P_{ij} = 1$ or $C_{ij} = 1$, then T_i and T_j should not be optional tasks (constraints C_1 , C_2). Moreover, P_{ij} and C_{ij} are mutually exclusive (C_3). O_i and R_i are also exclusive as a task cannot be optional and be replaced by another task at the same time (C_4).

It is important to note that our formulation will revert to the original “baseline” model if the O_i , R_i , C_{ij} and P_{ij} variables are all set to 0. Now we discuss the scenarios that were introduced above. In each case we modify the **Obj** function. Also note that in these solutions we focus on the upper path in the process of Fig. 4 that corresponds to the credit report and proposal being ‘OK’ since this is the more interesting case.

Scenario 1: Shortest finish time redesign. Here we set **Obj** = TF_{10} . On solving the model we get $TF_{10} = 70$. The solution for this model using CPLEX [2] is:

- $T_1 = T_2 = T_3 = T_6 = T_7 = T_8 = T_{10} = 1$; $O_2 = O_6 = 1$.

All other variables are 0. The interpretation of this solution is that tasks T_2 and T_6 are skipped since O_2 and O_6 are 1. In doing so we obtain the solution that takes the least time. Note that in the way we construct the model, the values of variables T_2 and T_6 are still 1, but their durations are 0, i.e. they are skipped.

Scenario 2: Max benefit redesign. Now we modify the objective Obj to a benefit function by aggregating the effect of various redesign options as follows:

Obj = Maximize

$$BO_2 * O_2 + BR_2 * R_2 + BO_6 * O_6 + BO_7 * O_7 + BP_{67} * P_{67} + BC_{67} * C_{67} - 0.01 * TF_{10};$$

Where

BO_2 : net benefit from skipping optional task $T_2 = -50$

BR_2 : net benefit from replacing task T_2 with task $R_2 = -25$

BO_6 : net benefit from skipping optional task $T_6 = 25$

BO_7 : net benefit from skipping optional task $T_7 = -50$

BP_{67} : net benefit from doing T_6 and T_7 in parallel = -10

BC_{67} : net benefit from combining T_6 and T_7 into one task = 50

The objective function consists of the sum of the individual benefit from each *design* option. The last term in the objective function includes the finish time of the last task TF_{10} so that among solutions of equal benefit one with the smallest finish time is found. The solution for this model is:

$$T_1 = T_2 = T_3 = T_6 = T_7 = T_8 = T_{10} = 1.$$

$$C_{67} = 1.$$

$$TF_{10} = 95.$$

Obj = 50. (neglecting the effect of $0.01 * TF_{10}$ in the objective function)

In this design tasks T_6 and T_7 are combined. One can see that this design finishes in a time of 95 which is more than for the design in scenario 1.

Scenario 3: Maximum benefit within a time limit. In this scenario, we keep the same objective function as in scenario 2 but add a new constraint: $TF_{10} \leq 75$ to impose a finish time limit. We know from scenario 1 that a solution exists with a finish time of 70. Now we get a solution that is similar to the one in scenario 2 with two changes:

$$O_6 = 1.$$

$$R_2 = 1.$$

$$TF_{10} = 75.$$

$$Obj = -50.$$

This means that a solution does exist within a time limit of 75 by skipping the optional task T_6 and replacing task T_2 with a faster task. However, now the Obj value drops to -50 because a negative benefit of 50 occurs from each adjustment.

Scenario 4: Maximum benefit within a relaxed time limit. Now we modify scenario 2 slightly by relaxing the time limit from 75 to 80. In this case we get a new solution:

$$O_6 = 1.$$

$$TF_{10} = 80.$$

$$Obj = -25.$$

This shows that by accepting an increase of 5 in the time limit, we have a new design with a benefit of -25 , a gain of 25 over scenario 3. In this case task T_6 is optional in the optimal solution.

Table 3 summarizes the four scenarios above and two more scenarios (5,6) by showing the finish time, net benefit and the values of the various design variables. By considering these alternatives a decision can be made on the most suitable design. Row 1 shows that the minimum flow time for an instance is 70 which is an improvement of 30 over the base case, but at a net benefit of -75 . However, optimizing the maximum

benefit (Row 2) shows that the baseline case is dominated by a design ($C_{67} = 1$) where the flow time is 95 and the benefit is 50. The remaining rows (3-6) show tradeoffs between time and benefit by imposing a different time constraint and finding the design with the maximum benefit. It is interesting to note how the design changes in each setting. In fact, no two designs are the same. Most of the design options are selected in some design or another except for the one with the parallel structure. On trying to force a solution by setting $P_{67} = 1$ in the model we get a design with a finish time of 90 and benefit of -10 which is dominated by scenario 6.

Table 3. Understanding tradeoffs among redesign scenarios

Scenario	Finish time	Benefit	O ₂	R ₂	O ₆	O ₇	C ₆₇	P ₆₇
0. Baseline	100	0	0	0	0	0	0	0
1. Least time	70	-75	1	0	1	0	0	0
2. Max benefit	95	50	0	0	0	0	1	0
3. Max benefit with time limit 75	75	-50	0	1	1	0	0	0
4. Time limit is relaxed to 80	80	-25	0	0	1	0	0	0
5. Time limit is relaxed to 85	85	0	1	0	0	0	1	0
6. Time limit is relaxed to 90	90	25	0	1	0	0	1	0

Of course, other scenarios may also be created by a user on demand. The results above suggest that as the various parameters of the model change (e.g. resource cost, service cost, time constraints, etc.) the choice of the best design can change. Hence, it is necessary to revisit the baseline model periodically. Moreover, the ability to change the design of a process instance in response to constraints adds flexibility as illustrated in Table 3. An organization can price its time sensitive services based on the benefit calculations and adjust its process model for each instance accordingly.

6 Discussion and Related Work

We did not model flexibility explicitly. It needs deeper exploration along the lines suggested in [14] based on mix, labor, routing, volume and process flexibility. We also did not consider interactive effects. So, the cost of skipping task T_2 is BO_2 and that of skipping T_6 is BO_6 . However, the cost of skipping both could be a function $f(BO_2, BO_6)$. It is also possible to make the values of benefit coefficients a function of case data. Thus, consider:

If $(\text{Loan_amount} \leq 100 \text{ K})$ then $BO_2 = -50$;

If $(100 \text{ K} < \text{Loan_amount} \leq 200 \text{ K})$ then $BO_2 = -75$;

If $(200 \text{ K} < \text{Loan_amount} \leq 300 \text{ K})$ then $BO_2 = -100$;

Here Loan_amount is a case variable whose value is provided by the user at run time to determine BO_2 . Other parameter values can also be functions of case variables. In practice the values of these parameters have to be determined by the end users based on an understanding of the time required to perform a task (from process logs), wage rate of an employee (from payroll), and other internal records of a company.

The modeling power of our approach is comparable to that of first order logic. To informally argue correctness of our approach we first note that the basic structural and temporal model has been shown to be correct elsewhere [8]. In this paper we extend this formulation by allowing certain tasks to be optional (case 1) or replaceable (case 2). We also allow a pair of tasks to combine into a single task (case 3) or restructure into parallel (case 4). In cases 1–3 there is no structural change in the formulation; only the task duration expressions are changed by introducing new variables like O_i , R_i and C_{ij} respectively as explained in Sect. 5. In case 4, the structure of two tasks, say, T_i and T_j , is changed from sequence to parallel. This requires relaxing the sequence constraint between T_i and T_j by adding a $M * (1 - P_{ij})$ term and modifying the successor and predecessor relationships of T_i and T_j . In all four cases existing relationships are modified to satisfy the alternative designs by introducing new variables. But this does not affect correctness. The split and postponement cases can also be explained with similar reasoning.

Flexibility and the need for managing change, customization and adaptation are important issues in BPM research (see e.g. [15, 17, 18, 20, 21]). One aspect of change is the need for systematic business process redesign. In [4, 11], many different best practices and heuristics for redesign have been proposed. In a related work [6], performance measures like cost, time, quality and flexibility for evaluating a new design are discussed at length. An evolutionary approach for generating redesign alternatives by applying best practices to an existing process is proposed in [13, 14]. While their goals are similar to ours their evaluation method is mostly based on simulation or enumeration. In contrast our approach is novel in that it offers an optimal solution and also the ability to add/modify constraints. Other work on redesign relates to identifying process improvement patterns [22], applying them effectively to processes [10] and detecting weaknesses in models [1]. Approaches based on goal models expressed in KAOS or Tropos/i* notations can offer guidance for process (re)design strategies [12]. Since the same goal model can be converted into multiple process designs it is possible to select the optimal one based on considerations of time, cost, quality and flexibility.

Our work also relates closely with research on configurable processes [3, 5]. Configurable processes are designed for flexibility using constructs like hidden and blocked tasks, and flexible gateways. Some early work on configurable processes was done in the context of EPC diagrams [3] and reference models [16]. A fine survey of business process modeling for variability that covers many configuration approaches appears in [9]. An approach for modeling flexible processes using templates and rules is discussed in [7]. Basically, a configurable model allows for the various kinds of improvements we have discussed here as configuration possibilities. Hence, each configurable node or gateway can be modeled as a redesign option in our framework and optimized with our approach.

7 Conclusions

In this paper we presented a novel approach for optimizing the redesign of process models. It is based on capturing process improvement strategies as constraints in a structural-temporal model. Each improvement strategy is represented by a binary

variable. An objective function that represents a net benefit function of cost and quality is then maximized subject to these constraints to find the best combination of process improvements that can be made to maximize the objective. The strategy variable values in the solution to this MILP formulation show the optimal strategies. We tested this model with a realistic case study and showed that it is possible to generate multiple redesign solutions by modifying the objective function and constraints.

In future work we would like to explore ways to model flexibility and resources in more depth. It will also be useful to extend the current approach to more patterns like knock out, numerical involvement, contact reduction, case types, etc. [11, 13, 19]. There is also a need to study ways to determine the parameters of the objective function more accurately and to analyze the sensitivity of the objective function to them. Further, in our current model a user must identify all valid strategies and include them in the model. However, it would be nice to investigate a recommendation system that will suggest the Top-3 or Top-5 process redesign scenarios to a user by systematically considering the application of all valid improvement strategies. To do so more semantic information about the process model may be needed by the system.

Acknowledgment. This work was initiated while the first author was visiting the BPM group at QUT, Brisbane. He thanks the BPM group, and especially Marcello La Rosa and Chun Ouyang for their inputs and suggestions.

References

1. Bergener, P., Delfmann, P., Weiss, B., Winkelmann, A.: Detecting potential weaknesses in business processes – an exploration of semantic pattern matching in process models. *Bus. Process Manag. J.* **21**(1), 25–54 (2015)
2. CPLEX: Reference manual. IBM corporation (2009)
3. Dreiling, A., Rosemann, M., Alast, W., van der Heuser, L., Schulz, K.: Model-based software configuration: patterns and languages. *Eur. J. Inf. Syst.* **15**(6), 583–600 (2006)
4. Dumas, M., Rosa, L.M., Mendling, J., Reijers, H.A.: *Fundamentals of Business Process Management*. Springer, Heidelberg (2013)
5. Gottschalk, F., van der Aalst, W.M.P., Jansen-Vullers, M.H., la Rosa, M.: Configurable workflow models. *Int. J. Coop. Inf. Syst.* **17**(2), 177–221 (2008)
6. Jansen-Vullers, M.H., Kleingeld, P.A.M., Mariska, N.: Quantifying the performance of workflows. *IS Manage.* **25**(4), 332–343 (2008). <http://dblp.uni-trier.de/db/journals/ism/ism25.html#Jansen-VullersKN08>
7. Kumar, A., Yao, W.: Design and management of flexible process variants using templates and rules. *Comput. Ind.* **63**(2), 112–130 (2012)
8. Kumar, A., Sabbella, S., Barton, R.: Managing controlled violation of temporal process constraints. In: Motahari-Nezhad, H.R., Recker, J., Weidlich, M. (eds.) *BPM 2015*. LNCS, vol. 9253, pp. 280–296. Springer, Switzerland (2015)
9. La Rosa, M., van der Aalst, W.M., Dumas, M., Milani, F.P.: *Business Process Variability Modeling : A Survey*. QUT Reprint (2013)
10. Lohrmann, M., Reichert, M.: Effective application of process improvement patterns to business processes. *Softw. Syst. Model.* **15**(2), 1–23 (2014)

11. Mansar, L., Reijers, H.A.: Best practices in business process redesign: use and impact. *Bus. Process Manage. J.* **13**(2), 193–213 (2007)
12. Nagel, B., Gerth, C., Post, J., Engels, G.: Kaos4SOA - extending KAOS models with temporal and logical dependencies. In: Proceedings of CAiSE Forum, pp. 9–16 (2013)
13. Netjes, M., Mansar, S.L., Reijers, H.A., van der Aalst, W.M.P.: Performing business process redesign with best practices: an evolutionary approach. In: Filipe, J., Cordeiro, J., Cardoso, J. (eds.) ICEIS 2007. LNBIP, vol. 12, pp. 199–211. Springer, Heidelberg (2008)
14. Netjes, M., Mansar, S.L., Reijers, H.A., van der Aalst, W.M.P.: An evolutionary approach for business process redesign - towards an intelligent system. *ICEIS* **3**, 47–54 (2007). <http://dblp.uni-trier.de/db/conf/iceis/iceis2007-3.html#NetjesMRA07>
15. Reichert, M., Weber, B.: Enabling Flexibility in Process-aware Information Systems: Challenges, Methods, Technologies. Springer, Heidelberg (2012)
16. Rosemann, M., van der Aalst, W.M.P.: A configurable reference modelling language. *Inf. Syst.* **32**(1), 1–23 (2007)
17. Sadiq, S.W., Marjanovic, O., Orłowska, M.E.: Managing change and time in dynamic workflow processes. *Int. J. Coop. Inf. Syst.* **9**(1–2), 93–116 (2000)
18. van der Aalst, W.M., Rosemann, M., Dumas, M.: Deadline-based escalation in process-aware information systems. *Decis. Support Syst.* **43**(2), 492–511 (2007)
19. van der Aalst, W.M., Ter Hofstede, A.H., Kiepuszewski, B., Barros, A.P.: Workflow patterns. *Distrib. Parallel Databases* **14**(1), 5–51 (2003)
20. Weber, B., Reichert, M., Mendling, J., Reijers, H.A.: Refactoring large process model repositories. *Comput. Ind.* **62**, 467–486 (2011)
21. Weber, B., Reichert, M., Rinderle-Ma, S.: Change patterns and change support features - enhancing flexibility in process-aware information systems. *Data Knowl. Eng.* **66**(3), 438–466 (2008)
22. Zellner, G.: Towards a framework for identifying business process redesign patterns. *Bus. Process Manage. J.* **19**(4), 600–623 (2013)