Optimizing Resource Utilization by Combining Running Business Process Instances

Christine Natschläger(✉), Andreas Bögl, and Verena Geist

Software Competence Center Hagenberg GmbH,
Hagenberg im Mühlkreis, Austria
{christine.natschlaeger,andreas.boegl,verena.geist}@scch.at

Abstract. Efficient business processes are a critical success factor for organizations in a competitive market environment. One of the key potentials to increase efficiency of business processes is the optimization of resource utilization. The contribution of this paper is a novel approach for combining activities across running process instances to optimize resource utilization; i.e., resources are shared across different process instances. The main benefits of the suggested approach are the identification, disclosure, and application of optimization potentials.

1 Introduction

The execution of business processes is typically managed by some workflow management system which is able to handle multiple instances of a business process. Given such a workflow management system, we subsequently present the *Combined-Instance Approach* that exploits the current state of running process instances to reveal optimization potentials of resources associated with process activities. Typically, organizations pursue two ways to increase efficiency of their business processes. The first way is achieved by specifying an "optimal" sequence of activities to accomplish a given process goal. By contrast, the second way addresses efficiency by increasing productivity and by optimizing the use of resources associated with activities in running process instances. While the first attempt reveals optimization endeavors on the process schema level, the second attempt reflects resource optimization on the process instance level (but only individual instances are considered). The idea of this paper is to bridge the gap between optimization potentials on the schema and on the instance level (and in-between instances). In particular, we suggest sharing resources across running process instances thereby considering restrictions defined in the process schema.

The proposed approach can be applied to almost all types of resources including physical, human, organizational, and financial resources. Concrete examples for processes that can potentially share resources are, e.g., delivery processes, production processes, (business) travel processes, ordering processes, and so on. Considering production processes, the suggested approach can also deal with the problem of small batch sizes and combine activities until a minimum-cost batch size is reached. Hence, with our approach we address the goals of the Industry

F. Toumani et al. (Eds.): ICSOC 2014, LNCS 8954, pp. 120–126, 2015.
DOI: 10.1007/978-3-319-22885-3_11

4.0 project of the German government, which especially emphasizes the demand for adaptable processes and resource efficiency in traditional industries.

2 The Combined-Instance Approach

In this section, we present our approach for combining activities across process instances to optimize resource utilization. An overview of the approach applied to a running example is shown in Fig. 1 and consists of four steps: (1) defining business processes with data objects and constraints, (2) identifying possible combinations, (3) determining the optimization potential, and (4) combining business process instances.

2.1 Business Processes with Data Objects and Constraints

In the first step, the required business process definitions are provided. The *business process schema* $\mathcal{S}$ defines the business process with its data objects and resources and is the basis for all process instances. For our approach, $\mathcal{S}$ is extended with (a) *meta-information* (*mi*) describing, e.g., resources and their capacities, (b) *type-level constraints* (*tc*) specifying general restrictions that

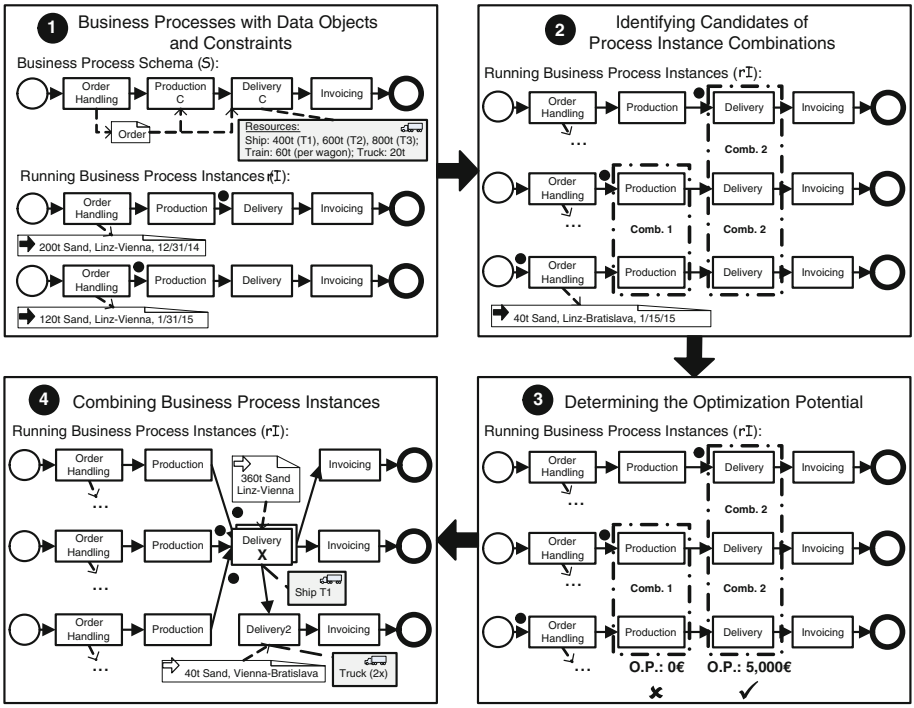

Fig. 1. Combined-instance approach

apply to all process instances either emerging from given data or being specified manually, and (c) *combinable activities* ($\mathcal{C}$) which comprise a *combinable condition* (*cc*) that defines the required matching of two instance activities for a possible combination and an *optimization function* (*of*) that determines the optimization potential of a combination. Every combinable activity is marked with a'C' in the process diagram. Considering type-level constraints, we further distinguish hard and soft constraints, where the latter may not be satisfied depending on the optimization function. All constraints must be formally specified (e.g., based on the *Object Constraint Language* (OCL)) to support business process execution.

In our running example, $\mathcal{S}$ defines an order execution process comprising activities for order handling, production, delivery, and invoicing with corresponding data objects (e.g., order, product, and invoice) and resources (e.g., production and transportation means). Due to space limitations not all of them are shown in Fig. 1. $\mathcal{S}$ is then extended with meta-information like possible transportation means and their capacity (Ship T1: 400t, ...), constraints like the capacity of a ship (*tc* based on *mi*) or that a ship can only be used for cities with a harbor that are connected by a river (manually specified *tc*), and combinable conditions, e.g., for a possible combination of delivery activities the routes must be overlapping. In our example, the production activity and the delivery activity specify a *cc* and an *of*, so these two activities are combinable.

An instantiation of $\mathcal{S}$ is called a business process instance $\mathcal{I}$. During the execution of $\mathcal{I}$, data objects and corresponding constraints defined in $\mathcal{S}$ are instantiated and provide concrete instance-specific data and restrictions. A running process instance $r\mathcal{I}$ further comprises one or more tokens that mark the current position(s) in the process flow (shown by a black-filled circle in Fig. 1).

In our example, the order execution process is instantiated three times. In the first process instance ($\mathcal{I}1$), 200t of sand are ordered and must be sent from Linz to Vienna until 12/31/14. The order of $\mathcal{I}2$ has the same route but with 120t of sand that must be delivered until 1/31/15. Finally, $\mathcal{I}3$ (shown in step 2) comprises an order with 40t of sand that must be sent from Linz to Bratislava until 1/15/15. All three process instances are running but the current positions differ.

Finally, we require some auxiliary functions that return all $r\mathcal{I}$, the current position(s) of $r\mathcal{I}$, the state of $\mathcal{C}$ (open, running, or completed), whether $\mathcal{C}$ is still reachable, and the expected costs and execution time of activities.

2.2 Identifying Candidates of Process Instance Combinations

Inputs to the second step are all definitions of $\mathcal{S}$ and a set of $r\mathcal{I}$. The goal then is to identify candidate pairs of $\mathcal{C}$ that satisfy all constraints. The search for candidate pairs is initiated whenever a token reaches a $\mathcal{C}$ and the triggering $\mathcal{C}$ is compared with the corresponding activity of every other $r\mathcal{I}$. If the other $\mathcal{C}$ is open and reachable (auxiliary functions) and all hard constraints and the *cc* are satisfiable, then the two $\mathcal{C}$ are a candidate pair. Several candidate pairs may be combined to sets of higher cardinality.

In our running example, searches are triggered by the production activity of $\mathcal{I}2$ and the delivery activity of $\mathcal{I}1$. Candidate pairs are the production activities of $\mathcal{I}2$ & $\mathcal{I}3$ (in $\mathcal{I}1$ already completed) and the delivery activities of $\mathcal{I}1$ & $\mathcal{I}2$ and $\mathcal{I}1$ & $\mathcal{I}3$ (which can further be combined to $\mathcal{I}1$ & $\mathcal{I}2$ & $\mathcal{I}3$).

The actual combination of candidates depends on three further conditions:

1. If the "other" (not-triggering) $\mathcal{C}$ will ever receive a token (after a preceding split an alternative path may be taken).
2. If waiting for the "other" (not-triggering) $\mathcal{C}$ will cause a currently satisfiable constraint to be violated (e.g., a deadline).
3. If the combination provides an improvement (optimization potential). This condition is evaluated within the next step.

2.3 Determining the Optimization Potential

Inputs to the third step are sets of combinable candidates. The goal then is to identify whether a possible combination is economically worthwhile by applying the *optimization function* (*of*) defined for every $\mathcal{C}$. Instructions for defining an *of* are provided by the mathematical domain under the terms *multi-objective optimization* and *constrained optimization* (see Sect. 3). In our case, the *of* must identify an optimal approach for the separate solution and the combined solution (e.g., choose fitting production and transportation means) and calculate a value for comparison (e.g., costs, time). Note that there is the possibility that candidates satisfy the *cc* but can, nevertheless, not be fully merged, e.g., due to routes being overlapping but not identical. In this case, we can split an activity so that part of it can be combined and the remaining part is executed individually. Then both sub-activities must be considered in the optimization function.

In our running example, we assume that the optimal transportation means for $\mathcal{I}1$ is a ship of type T1, for $\mathcal{I}2$ two train wagons and for $\mathcal{I}3$ two trucks. For the combined solution, the *of* suggests a ship of type T1 from Linz to Vienna and two trucks from Vienna to Bratislava (activity of $\mathcal{I}3$ is split). Then the loading and transportation costs for both solutions are calculated and compared. We assume that we have an optimization potential of € 5,000 for the combined delivery activities and no optimization potential for the production activities.

2.4 Combining Business Process Instances

Inputs to the fourth step are sets of combinable candidates with optimization potential. The goal then is to combine the activities and update the process instances, thereby providing runtime validation and (manual) authorization of combinations for quality assurance. However, for combining activities of several process instances, we require a new business process element, which we call *Combined-Instance Activity* ($\mathcal{X}$). Syntactically, this element receives the data objects and resources of all merged activities and must satisfy all constraints. It will further be addressed by several incoming and outgoing flows coming from different process instances. The semantics is that $\mathcal{X}$ will consume a token from

every participating process instance, execute the combined activity thereby using the recommended resource(s) and, finally, produce the same amount of tokens and return them to the process instances. For the graphical representation, we recommend two overlapping activities where the front activity is marked with a bold 'X'. A similar element with the required semantics is not available in any other business process modeling language (BPMN, UML, EPC, or YAWL).

For replacing the individual activities with $\mathcal{X}$ we use a deferred approach. The first activity already received a token which triggered the search. We now have to wait for further tokens to reach corresponding candidates in other $r\mathcal{I}$. The waiting is restricted either by a predefined amount of time, by not delaying the activity but considering the time before execution as implicit waiting, or by the deadline of the activity minus the expected execution time. When a second process instance reaches the required position in time, the two activities are replaced by an $\mathcal{X}$ (if necessary an activity is thereby split). When a further candidate receives a token, then the corresponding activity is also integrated in $\mathcal{X}$. The deferred approach is necessary, since some candidates may not be reached at all (preceding split with alternative path) or not reached in time. When all possible activities are integrated, $\mathcal{X}$ is executed and separately written in the log-file of every process instance (together with further split activities).

3 Related Work

Related research is provided by different domains. For example, in the mathematical domain, scheduling problems are studied and algorithms are defined that calculate the optimal solution. Of particular interest are the resource-constrained project scheduling problem, dynamic optimization problems, and constrained optimization problems (see e.g. [3,6]). If several objectives have to be optimized simultaneously, this issue is investigated within multi-objective optimizations [4].

In the business process domain, related research is available concerning optimization of resources (e.g., in the sub-field of business process intelligence [5]), typically based on goals or constraints. In addition, in the areas of service composition and dynamic resource allocation related work exists that deals with similar problems. However, the focus of the suggested approaches is either on the type-level (business process schema) or on individual running process instances (sometimes in combination with previously completed process executions).

A similar approach that also synchronizes running process instances but does not sufficiently address resource combination and optimization potentials is described in [8]. An example in the context of the healthcare domain is presented in [1] and supports instance-level adaption of workflow schemas to prevent repeating or overlapping activities. The paper builds on previous research on flexible workflow management systems (e.g., by [9]) and introduces interesting ideas but restricts the approach of activity crediting to a single workflow instance.

Finally, considering domains like logistics or production, applications and methods have been designed that optimize the utilization of resources in the specific domain (e.g., dynamic logistics process management problems [2,10]).

However, our goal is to dynamically address resource optimization on a higher level of abstraction, i.e., business processes, with the advantage that several resources from different domains can be considered within the same business process (e.g., optimization of production and transportation resources).

So, to the best of our knowledge, there is no other approach that suggested sharing resources across several running process instances.

4 Conclusion

In this paper, we presented a novel approach for resource optimization in business processes. The main idea is to combine activities with similar tasks of several running process instances, thereby sharing resources like transportation or production means. Thus, we address the demand for adaptable processes and resource efficiency identified by the Industry 4.0 project of the German government.

Our future goals are to implement a prototype and to extend the approach by providing exception handling for *Combined-Instance Activities*, by also waiting for future process instances with new combination possibilities (if the deadline is not violated), and by considering similar activities derived from different process schemas (e.g., based on the identification of similarities described in [7]).

Acknowledgements. This publication has been written within the project *Vertical Model Integration* (VMI) 4.0. The VMI 4.0 project is supported within the program "Regionale Wettbewerbsfähigkeit OÖ 2007-2013" by the European Fund for Regional Development as well as the State of Upper Austria. This work was also supported in part by the *AdaBPM* project, which is funded by the *Austrian Research Promotion Agency* (FFG) under the project number 842437.

References

1. Browne, E.D., Schrefl, M., Warren, J.R.: Activity crediting in distributed workflow environments. In: ICEIS (3), pp. 245–253 (2004)
2. Chow, H.K., Choy, K.L., Lee, W.B.: A dynamic logistics process knowledge-based system - an RFID multi-agent approach. Know.-Based Syst. **20**(4), 357–372 (2007)
3. Cruz, C., González, J., Pelta, D.: Optimization in dynamic environments: a survey on problems, methods and measures. Soft Comput. **15**(7), 1427–1448 (2011)
4. Deb, K.: Multi-objective optimization. In: Burke, E.K., Kendall, G. (eds.) Search Methodologies, pp. 403–449. Springer, US (2014)
5. Grigori, D., Casati, F., Castellanos, M., Dayal, U., Sayal, M., Shan, M.C.: Business process intelligence. Comput. Ind. **53**(3), 321–343 (2004)
6. Hartmann, S., Briskorn, D.: A survey of variants and extensions of the resource-constrained project scheduling problem. EJOR **207**(1), 1–14 (2010)
7. Leopold, H., Niepert, M., Weidlich, M., Mendling, J., Dijkman, R., Stuckenschmidt, H.: Probabilistic optimization of semantic process model matching. In: Barros, A., Gal, A., Kindler, E. (eds.) BPM 2012. LNCS, vol. 7481, pp. 319–334. Springer, Heidelberg (2012)

8. Pufahl, L., Weske, M.: Batch activities in process modeling and execution. In: Basu, S., Pautasso, C., Zhang, L., Fu, X. (eds.) ICSOC 2013. LNCS, vol. 8274, pp. 283–297. Springer, Heidelberg (2013)
9. Reichert, M., Rinderle, S., Dadam, P.: ADEPT workflow management system. In: van der Aalst, W.M.P., ter Hofstede, A.H.M., Weske, M. (eds.) BPM 2003. LNCS, vol. 2678, pp. 370–379. Springer, Heidelberg (2003)
10. Wang, Y., Caron, F., Vanthienen, J., Huang, L., Guo, Y.: Acquiring logistics process intelligence: methodology and an application for a chinese bulk port. Expert Syst. Appl. **44**, 195–209 (2014)