Matthias Boehm, Uwe Wloka, Dirk Habich, Wolfgang Lehner

Workload-based optimization of integration processes

Erstveröffentlichung in / First published in:

CIKM08: Conference on Information and Knowledge Management. Napa Valley, 26.-30. Oktober 2008, S. 1479–1480. ACM. ISBN 978-1-59593-991-3.

DOI: https://doi.org/10.1145/1458082.1458343

Final edited form was published in "CIKM08: Conference on Information and Knowledge Management. Napa Valley 2008", S. 1479–1480, ISBN 978-1-59593-991-3
https://doi.org/10.1145/1458082.1458343

Workload-Based Optimization of Integration Processes

Matthias Boehm and Uwe Wloka
Dresden University of Applied Sciences
Database Group
01069 Dresden, Germany
mboehm@informatik.htw-dresden.de
wloka@informatik.htw-dresden.de

Dirk Habich and Wolfgang Lehner
Dresden University of Technology
Database Technology Group
01062 Dresden, Germany
dirk.habich@inf.tu-dresden.de
wolfgang.lehner@inf.tu-dresden.de

ABSTRACT

The efficient execution of integration processes between distributed, heterogeneous data sources and applications is a challenging research area of data management. These integration processes are an abstraction for workflow-based integration tasks, used in EAI servers and WfMS. The major problem are significant workload changes during runtime. The performance of integration processes strongly depends on those dynamic workload characteristics, and hence workload-based optimization is important. However, existing approaches of workflow optimization only address the rule-based optimization and disregard changing workload characteristics. To overcome the problem of inefficient process execution in the presence of workload shifts, here, we present an approach for the workload-based optimization of instance-based integration processes and show that significant execution time reductions are possible.

Categories and Subject Descriptors

H.2.4 [**Systems**]: Query processing; H.2.5 [**Heterogeneous Databases**]: Data translation; H.4.1 [**Office Automation**]: Workflow management

General Terms

Algorithms, Design, Performance

Keywords

Integration Processes, Optimization, Workload Adaptation

1. INTRODUCTION

Due to the trend towards heterogeneous system environments, research on integration processes has gained in importance. Here, integration processes are considered as an abstraction for workflow-based integration tasks, that are executed by a central integration platform. Examples for those platforms are EAI (Enterprise Application Integration) servers, or WfMS (Workflow Management Systems).

Integration processes are pervasive and the overall performance of complete enterprise IT-infrastructures depends on the performance of the central integration platform. A fundamental problem are significant workload changes during runtime. Due to dynamic workload shifts, deployed integration processes may perform inefficiently over time. The majority of existing workflow optimization approaches [2, 3, 4, 5] apply only rule-based optimization techniques and disregard the cost-based optimization based on monitored execution statistics.

With the aim to overcome this profound problem, we present an model-driven approach for the workload-based optimization of instance-based integration processes. The basic idea of our approach is to monitor execution statistics during runtime (workload characteristics), and to use those statistics for periodical re-optimization of the deployed integration processes. The resulting feedback loop opens opportunities to dynamically adapt to changing workload characteristics and hence, overcomes the problem of inefficient process execution in the presence of workload shifts.

Obviously, the optimization of integration processes is related to traditional distributed database optimization. However, there are major differences that necessitated new optimization techniques and algorithms. First, there are additional operators (e.g., Switch, Fork, Iteration, Translation, and Split). Second, the control flow and its optimization must be taken into account. Third, integration processes are deployed once and executed many times. In fact, the optimization of imperative integration processes is much more complex than the optimization of declarative queries.

2. WORKLOAD-BASED OPTIMIZATION

In this section, we give a compact overview of our developed feedback loop for the workload-based optimization of integration processes. The approach is based on our integration process model—the *Message Transformation Model (MTM)* [1] that contains interaction-oriented, control-flow-oriented, and data-flow-oriented operators.

Workload Monitoring and Cost Prediction. Basically, workload characteristics in the means of execution times, data properties (e.g., cardinalities, null-values, ordered criteria) and relative frequencies of alternative process paths are monitored for the single operators of the deployed process plan P. Those statistics are maintained over a sliding time window $W(P)$. Furthermore, we have adapted a cost model from a relational DBMS to the used integration model. Here, we enriched the model with additional semantics for interaction- and control-flow-oriented operators. In

Final edited form was published in "CIKM08: Conference on Information and Knowledge Management. Napa Valley 2008", S. 1479–1480, ISBN 978-1-59593-991-3
https://doi.org/10.1145/1458082.1458343

order to estimate the costs for a rewritten process plan P', we compare the costs C of the single operators weighted with the execution statistics W of the current process plan P. Thus, we estimate missing workload execution statistics with $W(P') = \frac{C(P')}{C(P)} \cdot W(P)$.

Periodical Re-optimization. According to functional aspects of the integration process model, a process plan can be rewritten—based on the cost prediction approach—to a semantical equivalent process plan. There, we must be aware of dependencies between operators. We formally define the problem of periodical re-optimization as follows:

DEFINITION 1. Process Plan Optimization Problem (POP): *A process plan $P = (N, S, F)$ is* optimal *at timestamp T_k with respect to the logged workload $W(P, T_k)$ if no process plan $P' = (N', S, F')$ with $C(P') < C(P)$ exists. Otherwise, the process plan P is* suboptimal. *The* process plan optimization problem *describes the periodical creation of the* optimal *process plan P at timestamp T_k with the period T_{OI} (optimization interval). The workload $W(P)$ is available for a sliding time window of size T_W. An optimization time T_{Opt} with $T_{Opt} = T_{k,1} - T_{k,0}$ and $T_k = T_{k,0}$ is required. We define that $T_W \geq T_{Opt} \wedge T_{OI} \geq T_{Opt}$, even in the worst case. The naive algorithm would comprise three subproblems: (i) the complete creation of alternative process plans, (ii) the cost evaluation of each created process plan, and (iii) the choice of the process plan with minimal costs. Solving the* POP *at timestamp $T_{k,1}$, the result is the optimal process plan correlated to the timestamp $T_{k,0}$.*

THEOREM 1. *The Process Plan Optimization Problem that includes the creation of the* optimal *process plan is NP-hard.*

PROOF. The subproblem of join enumeration is used to proof the theorem. The complexity of join enumeration depends on parameters like the query type (tree, chain, star, cycle, clique), the join tree class, the cost function, and the use of cross products. In general, this problem is NP-hard. A process plan is a directed graph. Hence, all types of join queries are possible. If the used cost model exhibit the ASI property (Adjacent Sequence Interchange), polynomial time algorithms can be found for join enumeration. There, ranks are assigned to base relations, where the sequence of ordered ranks is the optimal join enumeration. However, the cost model does not exhibit the ASI property. In conclusion, the *process plan optimization problem* is also NP-hard. □

The problem of cost-based query optimization in DBMS is also NP-hard. Thus, the proof is not surprising. However, in [4], it was claimed that the optimal Web service query plan can be computed in $O(N^5)$, where n is the number of Web services. The difference is caused by the assumption of negligible local processing costs, made in [4].

Optimization Techniques. In order to reduce the complexity, we use a heuristic algorithm—the *pattern matching optimization* algorithm—that recursively evaluates the operators of process plan P and rewrites that plan to P' using our set of workload-based optimization techniques (shown in Figure 1). Basically, we distinguish between data-flow-oriented and control-flow-oriented optimization techniques. Some of those techniques are similar to distributed query optimization while others (such as the rewriting of Switch-paths) are specific to integration processes. There, we use heuristic join enumeration restrictions that reduces the number of alternative joins orderings to $2(n-1)$.

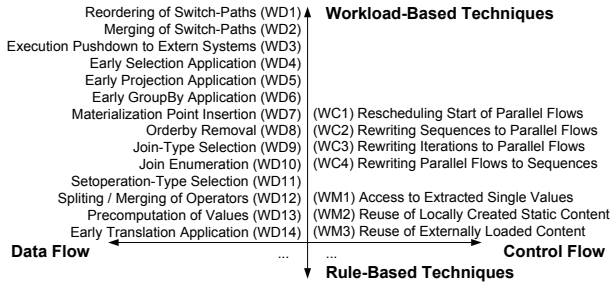

Figure 1: Workload-Based Optimization Techniques

Workload Adaptation. The core optimization algorithm can be influenced by parameters. Here, we explain how a fast and a slow workload adaptation can be realized with the right choice of parameterization. Basically, workload statistics of the current plan P are aggregated over the sliding time window. There, we distinguish the three parameters (i) workload aggregation method, (ii) the sliding window size, and (iii) the optimization interval. The workload aggregation method (i) is the method used to aggregate execution statistics over the sliding window. Here, we use moving-average-based and regression-based methods (with different sensibility properties). The sliding time window size (ii) T_W is the time interval used to aggregate statistics. Hence, its length influences the sensibility of the adaptation (the larger the time window, the less sensible the adaptation). Finally, there is the optimization interval T_{OI} that represents the time interval after which the optimization is triggered and costs are estimated. Clearly, with $T_{OI} \to 0$, we get a continuous workload estimation value and hence, also influence the sensibility (the larger the optimization interval, the less sensible the adaptation).

3. CONCLUSIONS

To summarize, we introduced the workload-based optimization of integration processes to overcome the main problem of inefficient integration processes in the presence of workload shifts. In the area of workflow-based integration tasks, this was considered for the first time. The preconditions of our solution comprise the monitoring of workload and execution statistics as well as the cost prediction. Based on these preconditions, we discussed the NP-hard *Process Plan Optimization Problem (POP)*, including the set of used optimization techniques, approaches for search space reduction as well as fast and slow workload adaptation. The experimental evaluation of this approach has been shown that significant execution time reductions are possible, with low overhead for periodical re-optimization.

4. REFERENCES

[1] M. Böhm, D. Habich, U. Wloka, J. Bittner, and W. Lehner. Towards self-optimization of message transformation processes. In *ADBIS*, 2007.

[2] H. Li and D. Zhan. Workflow timed critical path optimization. *Nature and Science*, 3(2), 2005.

[3] A. Simitsis, P. Vassiliadis, and T. Sellis. Optimizing etl processes in data warehouses. In *ICDE*, 2005.

[4] U. Srivastava, K. Munagala, J. Widom, and R. Motwani. Query optimization over web services. In *VLDB*, 2006.

[5] M. Vrhovnik, H. Schwarz, O. Suhre, B. Mitschang, V. Markl, A. Maier, and T. Kraft. An approach to optimize data processing in business processes. In *VLDB*, 2007.