MULTIOBJECTIVE MIXED-INTEGER OPTIMIZATION SCHEDULING OF WEB SERVICES

WALCIR FONTANINI; PAULO AUGUSTO VALENTE FERREIRA

Abstract—The article addresses the design of an online, internet business process composed of webservices from different suppliers. We propose a design methodology based on the simultaneous optimization of several quality-of-service criteria: cost, execution time, reliability, availability and reputation. A goal programming formulation for the problem of selecting webservices suppliers results in a mixed-integer linear optimization problem. Preliminary computational tests demonstrate the validity of the approach proposed.

Keywords—E-Commerce, Scheduling, Integer Programming, Multiobjective Programming

1 Introduction

A typical internet business can be thought as composed of several webservices to be supplied by partners with basis in internet technologies. As the number of business partners can be very large, some criteria for selecting them are necessary. We observe that most of the webservices design methodologies do not consider business performance criteria, such as cost or execution time directly, which is the focus of the present paper.

Current approaches to the internet business design problem often adopt a heuristic, trial-and-error strategy, which means that service components are assigned to individual tasks in an one-at-a-time basis. This strategy does not effectively integrate constraints of the business process and preferences of the business designer. As an example, the execution time of the composite webservice may be limited to an given value, or the total cost may not be allowed to exceed a prescribed budget.

In the present paper, a webservice design methodology based on five business performance criteria, namely, execution time, cost, service reputation, reliability and availability, is proposed. The methodology provides an optimized solution for the problem of selecting business partners (individual services). Constraints and preferences are associated with the composite service instead of with individual tasks within the composite service. As in (Zeng et al., 2004) and (Ardagna and Pernici, 2007) the services selection problem is formulated as a mathematical optimization problem.

By introducing a goal programming (multiobjective) formulation ((Gembicki and Haimes, 1975; Steuer, 1989)) for the problem, the designer can express his/her preferences about the criteria in terms of goals to be attained.

The paper is organized as follows. In section 2 a simple scenario is presented. In section 3 we detail the webservices quality model. In section 4 we formulate the webservices scheduling problem as multiobjective mixed-integer optimization problem. In section 5 we obtain and discuss some experimental results. Conclusions are presented in section 6.

2 Example: A Travel Planner

The following representation of a business process is based on the Business Process Management Notation (BPMN) known as (OMG, 2008). Figure 1 presents the symbols used in the example.

![BPMN: Flow Objects](image)

Figure 1: BPMN: Flow Objects.

Our goal here is simply to show how a reasonable scenario can be described by using BPMN.

Consider the composite webservices called Travel Planner, which aggregates several composite webservices, such as flight booking, travel insurance, hotel
room booking, car or biking rental, and route planning. Among these, there are web services that can be executed sequentially or concurrently.

A simplified process diagram specifying the Travel Planner composite webservice is presented in Figure 2. The user must input the destination with departure and return dates into the Travel Planner system, which then uses these informations to determine which flights are available. The next step involves choosing a hotel with available rooms. Lists with tourist sites can be provided to help in a typical travel preferences assessment. These are the most important functionalities of the Travel Planner system.

![Travel Planner System with webservices](image)

Figure 2: Travel Planner System with webservices

### 3 Quality Model for Webservices

A business process can be represented as a task graph, a terminology adopted in (Sinnen, 2007), more convenient for describing optimization problems. Since a business representation may contain IF-THEN-ELSE constructs, a possibility is to generate two separate task graphs with specific execution frequencies. An execution path can include parallel activities, but not activities in THEN and ELSE branches simultaneously.

The conversion rules proposed in this article are as follows. Firstly, if the original BPMN diagram contains gateway decisions, then a pre-processing program – a Java program that converts the BPMN diagram into a task graph – generates two graphs, one for the THEN branch and the other for the ELSE branch. Secondly, the pool and lane elements (or the horizontal parallel bars used for tasks synchronization) are not represented graphically because a depth search can find all the execution paths. Thirdly, the activity, begin and end elements are represented as nodes of the task graph. The graph elements are, in turn, converted into a mathematical optimization model, which considers all the quality (performance) criteria selected. Different optimization models are used in ((Ardagna and Pernici, 2005; Zeng et al., 2003)).

One concept not included in the BPMN standards, but necessary to complete the mathematical model is of execution frequency. When a gateway gives rise to THEN and ELSE elements, it is necessary to generate a real number between 0 and 1 associated with the relative frequency of the execution of that element. In practical implementations, the execution frequencies are obtained from recorded data.

#### 3.1 Quality Criteria of Webservices

In (Zeng et al., 2003; Zeng et al., 2004), five quality criteria are proposed for elementary webservice and then applied to composite webservice. The criteria are (1) execution cost, (2) execution time, (3) reputation, (4) reliability and (5) availability.

1. **Execution Cost.** For a specific webservice $i$ from provider $j$, the execution cost $EC_{ij}$ is the amount that the service client must pay for the execution of a certain operation of the webservice. Either the service providers explicitly publish their execution prices or the prices can be obtained online.

2. **Execution Duration.** For a specific webservice $i$ from provider $j$, the execution duration $ED_{ij}$ measures the expected delay between the execution request and the return of the output. The execution duration is calculated by the expression $ED_{ij} = T_{process} + T_{trans}$, as the execution duration is the processing time $T_{process}$ plus the transmission time $T_{trans}$. Webservice providers publish their processing times or provide means to obtain this information online. The transmission time is estimated from past executions of the webservice operation:

   $$ T_{trans} = \frac{\sum_{k=1}^{n} T_k}{n}, \hspace{1cm} (1) $$

   where $T_k$ is the transmission time of the $k$-th observation of the transmission time, and $n$ is the number of executions considered.

3. **Reliability.** The reliability $R_{ij}$ of a webservice $i$ from provider $j$ is the probability that a request will be met within an expected time interval. This information should also be published online by the provider. Reliability is a measure related to software and hardware structures; it depends on the webservice server and on the quality of its connection with the service client. The reliability is calculated from recorded data by $R_{ij} = N_e/K$, where $N_e$ is the number of correct webservice outputs delivered in
the expected time interval, and $K$ is the number of accesses considered.

4. Availability. The availability $AV_{ij}$ of a webservice $i$ from provider $j$ is a probability that the service will be available. The availability is calculated by $AV_{ij} = T_a/\theta$, where $T_a$ is the total time (in seconds) that the service is available during the last $\theta$ seconds, where $\theta$ is a parameter set by the webservice administrator. The value of $\theta$ depends on the application and varies significantly. In applications in which the service is frequently accessed (e.g., a stock market), small values for $\theta$ are required. If the service is less frequently used (e.g., an online bookstore), a large value for $\theta$ is more appropriate. It is possible to obtain information from the webservice providers about their availabilities.

5. Reputation. The reputation $RP_{ij}$ of a webservice $i$ from provider $j$ is a subjective measure of the client satisfaction. It depends on previous experiences while using the webservice. Different users can have different opinions about the same service. Thus, reputation is defined as the average ranking of the webservice according to its users:

$$RP_{ij} = \frac{\sum_{k=1}^{n} Rank_k}{n},$$

where $Rank_k$ is the ranking given by a user about the service, and $n$ is the number of times that the webservice has been evaluated. It is a common practice that users evaluate web services in a range from 0 to 5, for example.

3.2 Webservice Quality Aggregation

A business process diagram can have many execution paths, depending on the existence of intermediate gateways or parallel processing. In order to aggregate the quality criteria of a business process, the following notation is introduced: an execution path $ep_l$ contains sequences of parallel tasks (including two special tasks, $begin$ and $end$); $L$ is the set of all execution paths; $freq_l$ is the execution frequency of the execution path $ep_l$; a subpath $sp_m$ does not contain any parallel sequences, as it is a subset of an execution path; and an execution plan $ep^l$ for an execution path $ep_l$ is a set of ordered pairs $(i, j)$, where the first element of the pair is the task $i$ that belongs to $ep_l$ and the second element is the webservice provider $j$.

1. Execution Duration, ED. The execution duration of a plan $ep^l$ is the largest sum of all execution durations in each subpath $sp_m$ that belongs to the execution path $ep_l$:

$$ED_l = \max_{sp_m \in ep^l} \sum_{(i, j) \in sp_m, (k, j) \in ep^l} ED_{ij}.$$  

2. Execution Cost, EC. The execution cost of a plan $ep^l$ for an execution path $ep_l$ is the sum of execution costs $EC_{ij}$ of the plan:

$$EC_l = \sum_{(i, j) \in ep^l} EC_{ij}.$$  

3. Reputation, RP. The reputation of a plan $ep^l$ for an execution path $ep_l$ is the weighted average sum of the individual reputations $RP_{ij}$ according to the plan $ep^l$; $N$ is the number of elements in the sample:

$$RP_l = \frac{1}{N} \sum_{(i, j) \in ep^l} RP_{ij}.$$  

4. Reliability, R. The reliability of a plan $ep^l$ for an execution path $ep_l$ is:

$$R_l = \prod_{(i, j) \in ep^l} R_{ij}.$$  

5. Availability, AV. The availability of a plan $ep^l$ for an execution path $ep_l$ is:

$$AV_l = \prod_{(i, j) \in ep^l} AV_{ij}.$$  

4 Webservices Scheduling

The webservice scheduling solution proposed in this paper is based on a policy selection that includes order parameters, agent characteristics, data on past executions, and on the ongoing execution status. A budget constraint assuring that the total operational cost does not exceed a given value is also included.

4.1 A Mixed-Integer Linear Programming Solution

The optimization model below involves elements of integer programming ([Wolsey, 1998]). The variables $y_{ij}$ indicate whether the webservice $i$ from provider $j$ participates in a given execution plan ($y_{ij} = 1$) or not ($y_{ij} = 0$).

1. Duration and execution cost constraints.

Let $A$ be the set of all graph tasks. We assume that there exists a set of web services $WS$ that can be assigned to any task $i \in A$. However, for each task $i$, only one webservice must be assigned. The participation of the webservice providers in a given plan is subjected to the following constraint:

$$\sum_{j \in WS} y_{ij} = 1, \forall i \in A.$$  

Let $ST_d$ represent the earliest starting time of task $i$ in execution path $ep_l$; let $TD_d$ represent the execution duration of task $i$ in execution path $ep_l$. The notation $j \rightarrow k$ indicates that task $k$ directly succeeds task $j$. The parameter $MXT$ is the maximum time to complete the execution of a plan. The following constraints then apply:

$$\sum_{j \in WS} ED_{ij} y_{ij} = TD_d, \forall i \in A, \forall ep_l \in L,$$

$$(TD_d + ST_d) \leq ST_{ij}, \forall j \rightarrow k, j, k \in A, \forall ep_l \in L,$$

$$ED_l \geq \sum_{i \in ep_l} TD_d, \forall ep_l \in L,$$

$$ED_l \leq MXT, \forall ep_l \in L.$$  

DOI: 10.5540/03.2013.001.01.0090 010090-3 © 2013 SBMAC
The constraint (4) imposes that the execution duration for each task \( i \) in an execution path \( ep_1 \) is equal to the sum of execution durations of the webservice providers \( j \in WS \) selected. Constraint (5) captures the fact that, if task \( k \) in execution path \( ep_1 \) directly follows task \( j \) in the same execution path, then task \( k \) does not start before the completion of task \( j \). Constraint (6) indicates that the execution time of each execution path is the largest among the execution times of all sub-paths \( sp_j^k \). Constraint (7) guarantees that the total execution duration of a execution path will never exceed the limit \( MXT \). (It would be incorrect to consider the \( MXT \) as the sum of all execution paths because, actually, the run-time environment will perform only one execution path.) The parameter \( MXT \) is the amount of money available to implement any execution plan. The variable \( EC_i \) is the cost of a execution path \( ep_i \); \( EC_i_j \) is the cost of task \( i \) when performed by the webservice provider \( j \). The following constraints must then hold:

\[
\sum_{j \in ep_i, j \in WS} EC_i_j y_{ij} = EC_i, \quad \forall ep_i \in L, \tag{8}
\]

\[
EC_i \leq MXT, \quad \forall ep_i \in L. \tag{9}
\]

According to (8), the total cost of a execution path is the sum of the costs of the webservice selected to execute task \( i \) in the execution path \( ep_i \). Constraint (9) indicates that the total cost of any execution path never exceeds \( MXT \).

### 2. The reputation constraint.

The parameter \( MNR_j \) is the minimum expected reputation, while the variable \( RP_i \) is the reputation of the execution path \( ep_i \). The parameter \( RP_i_j \) represents the expected reputation of task \( i \) when implemented by the webservice provider \( j \). It follows that:

\[
RP_i = \sum_{j \in ep_i, j \in WS} RP_i_j y_{ij} / |ep_i|, \quad \forall ep_i \in L, \tag{10}
\]

\[
RP_i \geq MNR_j, \quad \forall ep_i \in L, \tag{11}
\]

where \(|ep_i|\) is the number of tasks in \( ep_i \).

### 3. Availability and reliability constraints.

The parameter \( MAN_j \) is the minimum expected availability; the variable \( AV_i \) is the availability of the execution path \( ep_i \). The parameter \( AV_i_j \) represents the expected availability of the task \( i \) when implemented by the webservice provider \( j \). The aggregation functions for availability and reliability are non-linear. Their linearizations are provided by the neperian logarithm (\( \ln \)):

\[
AV_i \geq \ln(MAN_j), \quad \forall ep_i \in L, \tag{12}
\]

\[
AV_i = \sum_{j \in ep_i, j \in WS} \ln(AV_i_j) y_{ij}, \quad \forall ep_i \in L. \tag{13}
\]

\[
R_i \geq \ln(MNR_j), \quad \forall ep_i \in L, \tag{14}
\]

\[
R_i = \sum_{j \in ep_i, j \in WS} \ln(R_i_j) y_{ij}, \quad \forall ep_i \in L. \tag{15}
\]

The following objectives express the selected performance criteria:

\[
\begin{align*}
\min f_1 &= \sum_{ep_i \in L} freq_i EC_i, \tag{16}
\min f_2 &= \sum_{ep_i \in L} freq_i ED_i, \tag{17}
\max f_3 &= \sum_{ep_i \in L} freq_i RP_i, \tag{18}
\max f_4 &= \sum_{ep_i \in L} freq_i AV_i, \tag{19}
\max f_5 &= \sum_{ep_i \in L} freq_i R_i. \tag{20}
\end{align*}
\]

where \( freq_i \) is the (observed) frequency of the execution path \( ep_i \).

### 4.2 The Multiobjective Formulation of the Problem

In this paper, the optimization problem derived from the previous analysis of webservices run-time environments is posed as a multiobjective optimization problem and solved by the well-known method goal attainment \((\text{Gembicki and Haimes, 1975})\). Given a set of objective functions \( f_1, f_2, \ldots, f_m \) defined on a feasible region \( X \), the following optimization is considered:

\[
\begin{align*}
\min \sigma & \quad \text{s.a.} \quad f_i(x) - \sigma w_i \leq f_i, \quad i = 1, \ldots, m \\
& \quad \text{s.t.} \quad x \in X
\end{align*} \tag{21}
\]

where \( f_i \), \( i = 1, \ldots, m \), are goals for the objectives and \( w_i \), \( i = 1, \ldots, m \), are weights associated with the goals; \( \sigma \) is a scalar variable. It is common to assume that \( f_i = f_i, \quad i = 1, \ldots, m \), that is, each goal is equal to the minimum of each individual objective function over the feasible region. Assuming that \( w_i = f_i, \quad i = 1, \ldots, m \), by solving (21) we minimize the largest variation of the objectives with respect to their minima. As in (16)-(20) the objectives \( i = 3, 4, 5 \) must be maximized, problem (21) is reformulated as follows:

\[
\begin{align*}
\min \sigma & \quad \text{s.a.} \quad f_i(x) - \sigma w_i \leq f_i, \quad i = 1, 2 \\
& \quad \text{s.t.} \quad f_i(x) + \sigma w_i \geq f_i, \quad i = 3, 4, 5 \\
& \quad \text{s.t.} \quad x \in X
\end{align*} \tag{22}
\]

where \( f_i = f_i, \quad i = 3, 4, 5 \), that is, each goal is the maximum of the corresponding objective function over the feasible region \( X \). See(Steuer, 1989) , for a review about multiobjective optimization.

### 5 Experimental Results

#### 5.1 Illustrative Example

To numerically test our methodology, a set of random parameters – as, for example, costs – was created by means of a specific Java program that converts the execution graph into specific data structures required by the solver GnuLPK ((Gnu, 2008)). A set of execution graphs for the business process was created. These graphs are not random, as the execution graphs must
be consistent. Each execution subpath that is dependent on IF-THEN-ELSE elements of the graph was executed with equal frequency. As an example, consider the business process illustrated in Figure 3 composed of three execution paths \{ep_0, ep_1, ep_2\}. Table 1 exhibits the parameters used for solving the associated optimization problem.

![Business process in BPMN terminology](image)

Figure 3: An example of business process in BPMN terminology.

<table>
<thead>
<tr>
<th>Expected min (or max)</th>
<th>ep_0</th>
<th>ep_1</th>
<th>ep_2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost</td>
<td>&lt;=266</td>
<td>127</td>
<td>349</td>
</tr>
<tr>
<td>Time</td>
<td>&lt;=23.74</td>
<td>11</td>
<td>29.6</td>
</tr>
<tr>
<td>Reputation</td>
<td>&gt;=0.92</td>
<td>0.95</td>
<td>0.89</td>
</tr>
<tr>
<td>Availability</td>
<td>&gt;=0.82</td>
<td>0.90</td>
<td>0.75</td>
</tr>
<tr>
<td>Reliability</td>
<td>&gt;=0.36</td>
<td>0.54</td>
<td>0.27</td>
</tr>
</tbody>
</table>

Table 1: Execution parameters furnished to the solver for the example.

The time that the user waits in a queue until his/her service starts was added to the execution time of the webservice.

The parameter execution time varied from 0.5 to 8 seconds following a uniform distribution. All tests were carried out on a micro-PC Dual-Core 3GHz with Windows XP.

In Table 2, a report produced by the solver is presented. The solver was initially run for each single objective in (16)-(20), generating ideal values for \(\text{cost} = 266\), \(\text{time} = 23.74\), \(\text{reputation} = 0.9200\), \(\text{availability} = 0.8169\) and \(\text{reliability} = 0.3744\). The solver was run a sixth time to solve the goal programming problem (22) with the ideal goals obtained previously. The goal attainment method returned \(\text{cost} = 356\), \(\text{time} = 30.36\), \(\text{reputation} = 0.8952\), \(\text{availability} = 0.7941\) and \(\text{reliability} = 0.3638\). Unlike the approaches proposed in (Zeng et al., 2003; Ardagna and Pernici, 2005; Ko et al., 2008) based on the minimization of a weighted sum of quality criteria, which imposes some difficulty in selecting the proper weights, the approach proposed in this paper minimizes the maximum percent variation with respect to the ideal performance values.

The sum of the execution times of all six optimization problems was less than a minute. The example had three execution paths \{ep_0, ep_1, ep_2\}, and the optimization report showed that no subpath violated the expected quality limits.

<table>
<thead>
<tr>
<th>Solution</th>
<th>Ideal goal and weight</th>
<th>Percent variation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost</td>
<td>356</td>
<td>266</td>
</tr>
<tr>
<td>Time</td>
<td>30.36</td>
<td>23.74</td>
</tr>
<tr>
<td>Reputation</td>
<td>0.8952</td>
<td>0.9200</td>
</tr>
<tr>
<td>Availability</td>
<td>0.7941</td>
<td>0.8169</td>
</tr>
<tr>
<td>Reliability</td>
<td>0.3638</td>
<td>0.3754</td>
</tr>
</tbody>
</table>

Table 2: Report for 25 tasks and 10 webservice provider scenarios.

5.2 Numerical Analysis

Graphs with 5, 10 and 25 nodes (tasks) were generated using the Java program. The number of webservices providers were respectively 10, 25 and 50. The quality parameters of the webservices were randomly varied. The parameter availability was generated with a uniform distribution between 0.95 and 0.99999. The parameter reputation was similarly generated with values between 0.8 and 0.99. The parameter reliability was generated with a uniform distribution between 0.85 and 0.95.

Table 4 and Table 3 present the percent variations of each objective relative to their ideal values as a function of \(n_{ws}\), the number of webservices, and \(m_{task}\), the number of tasks.

From both Table 4 and Table 3, we observe that for a business process with five tasks, the bottleneck represented by \(\sigma^*\) was the execution time, even though the number of service providers had increased from 10 to 25. For businesses process with 25 tasks, the bottleneck was cost, even though the number of service providers had varied significantly: 10, 25 and 50.

<table>
<thead>
<tr>
<th>Objective</th>
<th>(10_{ws} \times 25_{task})</th>
<th>(25_{ws} \times 25_{task})</th>
<th>(50_{ws} \times 25_{task})</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost</td>
<td>0.9420</td>
<td>1.2410</td>
<td>1.0697</td>
</tr>
<tr>
<td>Time</td>
<td>0.5459</td>
<td>1.1259</td>
<td>0.9172</td>
</tr>
<tr>
<td>Reputation</td>
<td>0.0528</td>
<td>0.0755</td>
<td>0.1038</td>
</tr>
<tr>
<td>Availability</td>
<td>0.0460</td>
<td>0.1220</td>
<td>0.1451</td>
</tr>
<tr>
<td>Reliability</td>
<td>0.0124</td>
<td>0.0286</td>
<td>0.3239</td>
</tr>
<tr>
<td>(\sigma^*)</td>
<td>0.9420</td>
<td>1.2410</td>
<td>1.0697</td>
</tr>
</tbody>
</table>

Table 3: Quality criteria as function of the number of webservices and tasks (Part 1).
<table>
<thead>
<tr>
<th>Objective</th>
<th>10ws × 5\text{task}</th>
<th>25ws × 5\text{task}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost</td>
<td>1.3938</td>
<td>0.9479</td>
</tr>
<tr>
<td>Time</td>
<td>1.5692</td>
<td>1.3860</td>
</tr>
<tr>
<td>Reputation</td>
<td>0.0931</td>
<td>0.0638</td>
</tr>
<tr>
<td>Availability</td>
<td>0.0642</td>
<td>0.0831</td>
</tr>
<tr>
<td>Reliability</td>
<td>0.1479</td>
<td>0.0525</td>
</tr>
<tr>
<td>$\sigma^*$</td>
<td>1.5692</td>
<td>1.3860</td>
</tr>
</tbody>
</table>

Table 4: Quality criteria as function of the number of webservices and tasks (Part 2).

## 6 Conclusions

This paper presented a webservice design methodology based on multiobjective optimization in a context where internet service providers integrate webservices from several partners to deliver an expected quality-of-service to a final customer. The paper focused on the model and solution of scheduling problems associated with business processes in webservice run-time environments. The solution proposed avoids the assignment of weights to the objectives (that is, their aggregation) and, while minimizing their maximum deviation from their ideal goals, provides an useful tool for identifying the critical aspects of the implementation of a given business process in the internet.

The problem of webservice optimization has attracted considerable interest in the literature ((Ko et al., 2008; Huang et al., 2009)). The research opportunities in the field of business process optimization remain quite large. The authors currently develop some refinements of the scheduling model and its extension to interactive decision making contexts.

## References


