ARTICLE IN PRESS

Discrete Applied Mathematics ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Web services composition: Complexity and models☆

V. Gabrel *, M. Manouvrier, C. Murat

PSL, Université Paris-Dauphine, 75775 Paris Cedex 16, France
CNRS, LAMSADE UMR 7243, France

ARTICLE INFO

Article history:
Received 1 October 2013
Received in revised form 16 October 2014
Accepted 27 October 2014
Available online xxxx

Keywords:
Web service composition
QoS
Workflow
Complexity
Series–parallel directed graph
Mixed integer linear program

ABSTRACT

A web service is a modular and self-described application callable with standard web technologies. A workflow describes how to combine the functionalities of different web services in order to create a new value added functionality resulting in composite web service. QoS-aware web service composition means to select a composite web service that maximizes a QoS objective function while satisfying several QoS constraints (e.g. price or duration). The workflow-based QoS-aware web service composition problem has received a lot of interest, mainly in web service community. This general problem is NP-hard since it is equivalent to the multidimensional multiple choice knapsack problem (MMKP). In this article, the theoretical complexity is analysed more precisely in regard to the property of the workflow structuring the composition. For some classes of workflows and some QoS models, the composition problem can be solved in polynomial time (since the workflow is a series–parallel directed graph). Otherwise, when there exist one or several QoS constraints to verify, the composition problem becomes NP-hard. In this case, we propose a new mixed integer linear program to represent the problem with a polynomial number of variables and constraints. Then, using CPLEX, we present some experimental results showing that our proposed model is able to solve big size instances.

1. Introduction and background

A Web Service (WS) is a modular and self-described application that uses standard web technologies to interact with other services [6]. WS are grouped into repositories (e.g. ProgrammableWeb[1] contains about 10.000 WS of different categories and BioCatalogue[2] provides more than 2.000 WS devoted to life science). When WS is limited to relatively simple functionalities, it is necessary to combine a set of individual WS to obtain a more complex one, namely a composite WS [5]. The WS composition problem aims at selecting a set of existing WSs such that the composition of those WS can satisfy the user's functional and non-functional requirements [8]. To differentiate several WS having the same functionality, QoS criteria (e.g. price, duration, etc.) can be used to select the "best" WS satisfying the users' requirements [12,15].

The QoS-aware service composition is the subject of numerous studies. As mentioned in [13], two approaches must be distinguished. In the first one, a predefined workflow is supposed to be known. This workflow describes a set of "abstract" tasks to be performed. Moreover, associated to each task, a set of WSs with similar functionalities (but different QoS) is

☆ This work is supported by the Action de Recherche Fondamentale en Recherche Opérationnelle (ARFRO) of the French Operational Research Group (GDR RO).

* Corresponding author at: PSL, Université Paris-Dauphine, 75775 Paris Cedex 16, France.
E-mail addresses: gabrel@lamsade.dauphine.fr (V. Gabrel), manouvrier@lamsade.dauphine.fr (M. Manouvrier), murat@lamsade.dauphine.fr (C. Murat).

1 http://www.programmableweb.com.

2 https://www.biocatalogue.org.

http://dx.doi.org/10.1016/j.dam.2014.10.020
0166-218X/

Please cite this article in press as: V. Gabrel, et al., Web services composition: Complexity and models, Discrete Applied Mathematics (2014), http://dx.doi.org/10.1016/j.dam.2014.10.020

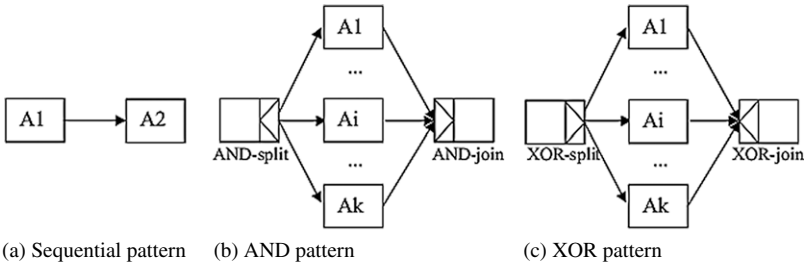


Fig. 1. Considered workflow patterns.

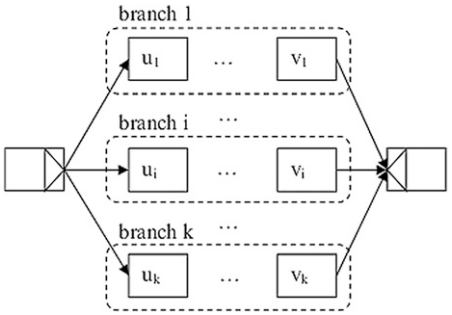


Fig. 2. Branches of XOR pattern.

also known. The composition problem is then to select one WS per task in order to respect QoS objective and constraints. In the second approach (see for example [14]), the existence of a predefined workflow is not assumed. Discovery and connection between WS are automatically performed by syntactic and/or semantic matching. Several methodologies have been proposed: AI planning (with AND/OR graph and A* algorithm, Satisfiability algorithm), 0–1 linear programming (solving with a branch and bound), Petri-Net, etc.—see a systematic review in [2]. In this article we focus on the first approach. WS discovery and semantic reconciliation are out of the scope of this paper.

The following first two sub-sections describe our context: the workflow structure of the composite WS and the QoS criteria. The third subsection analyses related work. The last one presents the outline of the paper.

1.1. Process model described by workflow

A workflow describes how to combine the functionalities of different WS in order to satisfy the user [16]. In a workflow, an activity represents a set of WSs sharing the same functionality, and a pattern represents temporal dependency between different activities. In this article, we consider the three more frequent patterns: sequence, parallel (AND) and exclusive choice (XOR).

Fig. 1 represents these workflow patterns, based on the YAWL model [18], where *Ai* is an activity. The sequential pattern, see Fig. 1(a), indicates that $A1$ must be executed before $A2$. The XOR and AND patterns start with a split and finish with a join. In AND pattern, see Fig. 1(b), all activities $A1, \ldots, Ak$ have to be executed, possibly in parallel. For XOR pattern, see Fig. 1(c), only one activity among $A1$ to Ak has to be executed.

In the following, we consider general complex workflows in which these patterns can be recursively concatenated and interlaced. XOR or AND patterns can be decomposed on several branches (each branch being a workflow containing several activities linked by patterns), where the first vertex u_i of branch i can be an activity or a split and where the last vertex v_i can be an activity or a join. For XOR pattern (see Fig. 2), one and only one branch must be selected. For AND pattern, all branches must be performed.

In a complex workflow, an end-to-end route is a path going from the first vertex of the workflow to the last one containing all branches of each AND pattern belonging to the path and, containing exactly one branch of each XOR pattern belonging to the path.

Example 1. An example of such complex workflow is given in Fig. 3, representing a planning travel process. During the first activity ($A1$), the user wants to book a flight ticket. Then the process is divided, by a XOR pattern, into three possible sub-processes: first solution, the user rents a car ($A2$) and then books a hotel ($A3$) with a parking ($A4$) (with an AND pattern), either the user requests a travel agency to organize all the travel ($A5$), nor he books a taxi ($A6$) and a hotel ($A7$).

Therefore, there exist three possible end-to-end routes for this planning travel process: $\{A1, A2, A3, A4\}$, $\{A1, A5\}$, $\{A1, A6, A7\}$. In the end-to-end route $\{A1, A2, A3, A4\}$, $A3$ and $A4$ can be performed in parallel.

Please cite this article in press as: V. Gabrel, et al., Web services composition: Complexity and models, Discrete Applied Mathematics (2014), http://dx.doi.org/10.1016/j.dam.2014.10.020

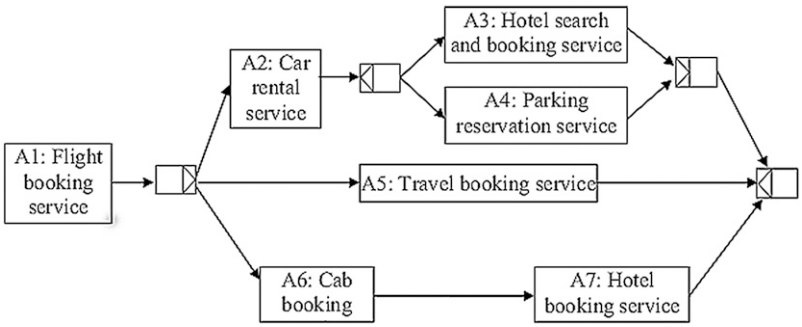


Fig. 3. Example of a complex workflow.

Several WSs, with similar functionality, permit to perform the same activity (e.g. the ProgrammableWeb repository contains 7 flight ticket booking WS, 12 hotel booking ones and 45 travel booking WS). In the following, we denote W_i the set of WSs available for performing the activity i. These WSs may have different Quality of Service (QoS) levels (see for example the QWS DataSet[3] [3]). These QoS criteria represent the non-functional properties of web services.

In the workflow-based context, the composition problem is simultaneously the problem of:

- selecting an end-to-end route,
- and choosing a WS for performing each activity belonging to the selected end-to-end route.

1.2. QoS criteria

In a large majority of studies (e.g. [4,21]), the QoS criteria are: execution price, execution duration/response time, reputation, reliability (the probability that the WS execution performs successfully) and availability. The way to compute the score of a composite WS depends on the type of the QoS criterion and on the pattern. For some criteria, the score of a composite web service is the sum of the web services' values belonging to the composition (e.g. price), for others, the score is based on a max operator (e.g. time for WS executed in parallel), otherwise, the score is the multiplication of the web services' values belonging to the composition (e.g. reliability).

In the following, we consider that each WS j performing activity Ai is evaluated on three criteria:

- the first criterion is the cost of WS j denoted c_{ij},
- the second criterion is the duration of WS j denoted d_{ij},
- the third is the reliability of WS j, denoted p_{ij}.

We choose these three criteria because they represent different ways of computing composite WS performances: the first criterion is a sum-type measure to be minimized, the second is a min/max-type measure to be minimized and the third one is a product-type measure to be maximized.

1.3. Related work

The workflow-based QoS-aware service composition problem has received a lot of interest (see e.g. [12,15] two recent surveys).

Different models are proposed to take into account the QoS: in one hand, the QoS criteria are used to define an objective function to optimize and, in other hand, some QoS constraints are introduced (e.g. the duration has to be less than 5 s). Concerning the workflow structure, two classes of workflow are usually considered:

- in the first one, the workflow induces only one end-to-end route (this is the case when the workflow contains SEQ and AND patterns only) then the problem is to select one WS for performing each activity of this end-to-end route;
- in the second one, there exist several possible end-to-end routes (this is the case when the workflow contains XOR patterns) then the composition problem simultaneously induces the selection of one end-to-end route and, the selection of one WS for each activity belonging to the selected end-to-end route.

In the large majority of the studies considering the first class of workflow, the QoS-aware service composition problem is modelled as a Multidimensional Multi-choice Knapsack Problem (MMKP) (see e.g. [19,20]). The statement of this classical combinatorial problem, known to be NP-hard, is the following: given m classes of items, each class i containing n_i items, each item being evaluated on q criteria, MMKP is the problem of selecting exactly one item of each class in order to maximize the value of the solution on one particular criterion while satisfying $(q-1)$ constraints (the constraint j states that the value of the solution on the criterion j must be lower or equal to an upper bound value). Given an end-to-end route, the QoS-aware service composition problem is equivalent to MMKP: an activity can be seen as a class of items, an item as a WS evaluated

[3] http://www.uoguelph.ca/~qmahmoud/qws/.

Please cite this article in press as: V. Gabrel, et al., Web services composition: Complexity and models, Discrete Applied Mathematics (2014), http://dx.doi.org/10.1016/j.dam.2014.10.020

on q QoS-criteria, the problem is to select one WS per activity in order to maximize a QoS-criterion and to verify $(q-1)$ QoS constraints. In [7], a heuristic is proposed to solve this problem.

Considering the second class of workflow, Zeng et al. in [20] propose a model that contains as many constraints as end-to-end routes in the workflow. This model is used to solve some instances but, as the number of end-to-end routes exponentially grows with the workflow size, some large-size instances may not be solved and even less formulated. Thus, authors propose to find near-optimal solutions in polynomial time with an approximate algorithm. In [21], authors also claim that all end-to-end routes of the workflow must be generated in order to find the best composite WS. Therefore, they propose to decompose the problem into two steps: (step 1) an end-to-end route is chosen and, (step 2) they propose a MIP formulation to select one WS per activity belonging to the chosen route in order to maximize QoS criteria. With such a decomposition, the obtained solution is not necessarily optimal.

The real challenge is then to include into the same optimization problem the selection of activities and the selection of WS for performing each activity structured by a complex workflow. In [10], we have proposed a 0–1 linear model for representing the composition problem based on a complex workflow satisfying transactional properties (i.e. properties allowing to guarantee reliable composite WS execution), but without QoS constraints. Our model is the first one with a polynomial number of variables and constraints.

The theoretical complexity of the QoS-aware service composition problem has not been extensively studied. Only the connection between this problem and the MMKP, which theoretical complexity is well known, have been shown. More recently in [11], Goldman and Ngoko propose a polynomial-time algorithm to determine the composite WS with minimal average execution time. Their algorithm is based on a recursive reduction of the workflow.

1.4. Outline of the paper

The theoretical complexity can be analysed more precisely in regard to the structural property of the considered workflow. For the workflows considered, and some QoS models, the composition problem may be easy to solve. In Section 2, we determine the classes of the QoS-aware composition problem that can be solved in polynomial time. We set that the considered workflow is a series–parallel directed graph. Then we show that the QoS-aware WS composition problem can be solved in polynomial time when the QoS requirements are only expressed in terms of the criterion to optimize. Otherwise, when there exist additional QoS constraints to verify, the composition problems become NP-hard. Section 3 is devoted to the resolution of such an NP-hard case. We propose a new mixed integer linear program for exactly solving the following NP-hard WS composition problem: to minimize the cost criterion and to satisfy two QoS constraints concerning execution time and reliability. Our model is very interesting since it avoids the enumeration of all end-to-end routes and the number of variables and constraints is a linear function of the workflow size. Numerical experiments (with CPLEX), presented in Section 4, show reasonable execution times although we note an increase of computation time when going from 1 to 2 constraints. Finally, Section 5 concludes the paper.

2. Polynomial cases and algorithms

2.1. Workflow property

A workflow (with interlaced patterns SEQ, AND and XOR) is a directed graph $G = (V, E)$ with $|E| = m$. G has a unique source representing the beginning of the process and a unique sink representing the end of the process. Since the arcs of G represent temporal precedence constraints, G does not contain any circuit. The vertices of G represent the activities and the AND and XOR patterns of the workflow. Then, the set of vertices V can be partitioned into 5 subsets: V_A ($|V_A| = n$) is the set of vertices representing the activities, V^S_{XOR} and V^S_{AND} (with $V^S = V^S_{XOR} \cup V^S_{AND}$) the sets of vertices representing the split of XOR and AND patterns respectively, and V^J_{XOR} and V^J_{AND} (with $V^J = V^J_{XOR} \cup V^J_{AND}$) the sets of vertices representing the join of XOR and AND patterns respectively.

Example 2. Fig. 4 contains the graph associated with the workflow presented in Fig. 3: the number of each vertex is surrounded by a dotted line below the vertex.

The class of Series–Parallel (SP) directed multigraphs (introduced by [9]) is recursively defined as follows:

- A directed edge (r, s) is a SP graph with r the source and s the sink.
- Let G' and G'' be two SP graphs. Their series and parallel compositions are also a SP graph:
  - the series composition of G' and G'' is obtained by merging the sink of G' with the source of G'',
  - the parallel composition of G' and G'' is obtained by merging the sources of G' and G'' and the sinks of G' and G''.

Clearly, a workflow (with interlaced patterns SEQ, AND and XOR) is a SP graph. Indeed, the SEQ pattern exactly corresponds to the series composition. The XOR and AND patterns exactly correspond to the parallel composition. Let us consider a XOR or AND pattern, the source is the split, denoted r, and the sink is the join, denoted s. Such a pattern is obtained by parallel composition of k workflows as illustrated in Fig. 5.

The fact that the workflow is an SP graph will be useful in the following section to establish the computational complexity of the WS composition problem.

Please cite this article in press as: V. Gabrel, et al., Web services composition: Complexity and models, Discrete Applied Mathematics (2014), http://dx.doi.org/10.1016/j.dam.2014.10.020

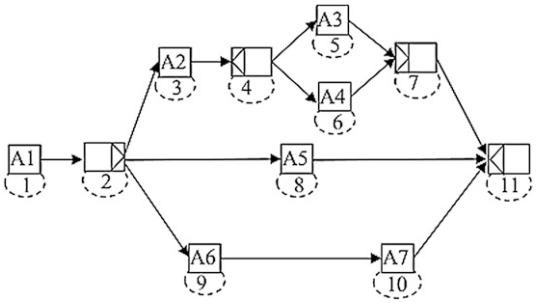


Fig. 4. Numbered graph associated with a workflow.

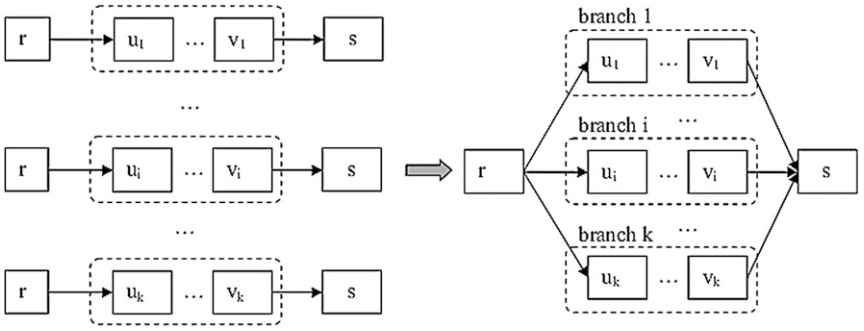


Fig. 5. Parallel composition.

In the following section, we show that the composition problem can be solved in polynomial time when the QoS requirements are expressed in terms of criterion to optimize. This criterion can be a sum-type criterion (like the cost or a score defined by a weighted-sum utility function), a product-type criterion (like the availability) or a max/min-type criterion (like the execution time).

2.2. Computational complexity of the WS composition problem with QoS criterion to optimize

In this section, the composition problem amounts to select an end-to-end route and to choose a WS for performing each activity belonging to the selected route in order to optimize a single criterion (which can be a sum-type, a product-type or a max/min-type criterion) without considering QoS constraint.

Let us first underline that, in this context, WS can be independently chosen for each activity. Before selecting an end-to-end route, it is sufficient to select for each activity *Ai* the WS in W_i with the best value on the retained QoS criterion (the best can be the WS with the minimal cost, with the minimal execution duration, or with the highest availability value). Then, each activity i of the workflow being valued by the value v_i of its (best) selected WS, the composition problem is only to choose an end-to-end route with optimal activity value.

Example 3. In the example of Fig. 4, we define, for performing each activity, three WSs with the values on the three criteria given in Table 1.

For each activity, we select the best web service on each criterion and we obtain the values reporting in Table 2:

If we consider the end-to-end route $\{A1, A2, A3, A4\}$, the scores of the best composite WS are:

- on cost criterion: $13 + 10 + 15 + 9 = 47$ (WS 3 for A_1, WS 2 for A_2, WS 1 for A_3 and WS 3 for A_4),
- on execution time criterion: $3 + 2 + \max\{1, 1\} = 6$ (WS 2 for A_1, WS 3 for A_2, WS 2 for A_3 and WS 1 or 2 for A_4),
- on reliability criterion: $1 \times 1 \times 1 \times 1 = 1$ (WS 1 for A_1, WS 3 for A_2, WS 3 for A_3 and WS 2 for A_4).

In the following sub-sections, we prove that the QoS-aware WS composition problem with a criterion to optimize can be solved in polynomial time for all types of criterion (sum-type, product-type or min/max type).

2.2.1. Sum-type and product-type criteria cases

Theorem 1. *The WS composition problem can be solved in polynomial time when optimizing a sum-type QoS criterion on a workflow (with interlaced patterns SEQ, AND and XOR).*

Proof. The considered composition problem can be formulated as a shortest path problem between the source and the sink of the workflow including the following transformation: for each AND pattern, the branches are linked by SEQ patterns as illustrated in Fig. 6.

Please cite this article in press as: V. Gabrel, et al., Web services composition: Complexity and models, Discrete Applied Mathematics (2014), http://dx.doi.org/10.1016/j.dam.2014.10.020

Table 1
WS QoS values for the example of Fig. 4.

Vertex	Activity	(c_{i1}, d_{i1}, p_{i1})	(c_{i2}, d_{i2}, p_{i2})	(c_{i3}, d_{i3}, p_{i3})
1	*A*1	(17, 4, 1)	(16, 3, 0.9)	(13, 5, 0.95)
3	*A*2	(16, 3, 0.99)	(10, 4, 0.94)	(18, 2, 1)
5	*A*3	(15, 4, 0.95)	(20, 1, 0.97)	(17, 5, 1)
6	*A*4	(17, 1, 0.99)	(18, 1, 1)	(9, 2, 0.97)
8	*A*5	(10, 5, 0.99)	(15, 4, 0.95)	(14, 4, 0.97)
9	*A*6	(16, 2, 0.97)	(18, 1, 0.95)	(15, 3, 0.98)
10	*A*7	(13, 3, 1)	(10, 4, 0.98)	(16, 1, 1)

Table 2
Best activities' value on QoS criteria.

Activities	A1	A2	A3	A4	A5	A6	A7
Best cost	13	10	15	9	10	15	10
Best execution time	3	2	1	1	4	1	1
Best availability	1	1	1	1	0.99	0.98	1

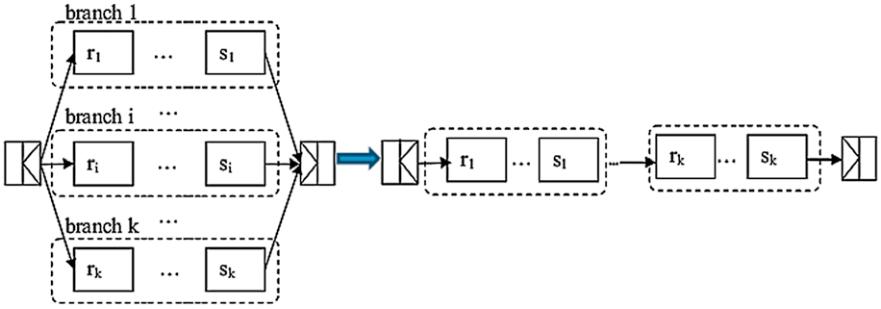


Fig. 6. AND pattern and its transformed workflow H.

In the transformed workflow denoted H, there are only SEQ and XOR patterns. In such a workflow, any path between the source r and the sink s exactly corresponds to an end-to-end route in the initial workflow G. The value on the sum-type criterion of an end-to-end route is the sum of the value of activities belonging to this route. Considering the following values on each directed edge (i, j) in H

$$l(i,j) = \begin{cases} v_i & \text{if } i \in V_A \\ 0 & \text{otherwise} \end{cases}$$

the value of a path from r to s, denoted $\mu[r, s]$, is $l(\mu[r, s]) = \sum_{(i,j)\in\mu[r,s]} l(i, j)$ and exactly corresponds to the value of the associated end-to-end route.

Consequently, considering a sum-type criterion, the minimal value end-to-end route is represented by the shortest path from r to s in the edge-valued transformed workflow H.

Since H does not contain directed cycle, the complexity of the shortest path algorithm that can be used is in $O(m_H)$ (m_H being the number of arcs of H—cf. e.g. [1]).

Example 4. In the workflow presented in Fig. 4, the activities are valued on the sum-type cost criterion (see values of selected WS reported in Table 2). The optimal solution is described by the shortest path in the edge-valued graph presented in Fig. 7. The value of the optimal composite WS $\{A1, A5\}$ is equal to 23.

Let us remark that the problem of maximizing a sum-type criterion can also be solved in polynomial time since the transformed workflow does not contain directed cycle.

In the product-type criterion case, we set the following values on each directed edge (i, j) of H:

$$l(i,j) = \begin{cases} v_i & \text{if } i \in V_A \\ 1 & \text{otherwise.} \end{cases}$$

The value of a path is $l(\mu[r, s]) = \prod_{(i,j)\in\mu[r,s]} l(i, j)$. Thus $l(\mu[r, s])$ exactly corresponds to the value of the associated end-to-end route on the product-type criterion. The problem remains a shortest path problem from the source to the sink in the transformed workflow. The $O(m_H)$ shortest path algorithm can be modified to determine the shortest path (with the same computational complexity) in case of product-type criterion (see also [1]). Consequently, we have:

Theorem 2. *The WS composition problem can be solved in polynomial time when optimizing a product-type QoS criterion on a workflow (with interlaced patterns SEQ, AND and XOR).*

Please cite this article in press as: V. Gabrel, et al., Web services composition: Complexity and models, Discrete Applied Mathematics (2014), http://dx.doi.org/10.1016/j.dam.2014.10.020

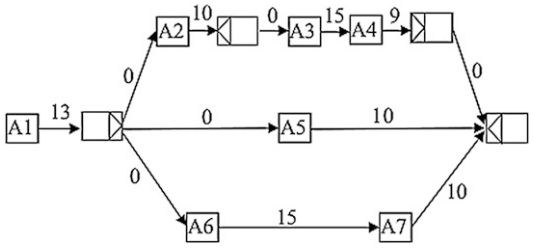


Fig. 7. Acyclic edge-valued graph corresponding to the example of Fig. 4.

2.2.2. Min/max-type criterion case

For the time criterion, the evaluation of a solution is different. The execution time of a composite web service is the sum of the WS performed in sequence (including the activities belonging to the selected branch of XOR patterns) plus the execution time of each selected AND pattern. The execution time of an AND pattern is equal to the maximal execution time between the branches execution time of the pattern. Consequently, considering the transformation presented in Section 2.2.1 with value of edges equal to the execution time, the value of a path is not equal to the execution time of the corresponding composite web service.

In order to optimize the time criterion, we use the binary decomposition tree associated with SP graph. In [17], Valdes et al. propose a $O(m)$ algorithm for defining the binary decomposition tree $G = (V, E)$. This algorithm consists to apply two kinds of reduction operation namely:

- The series reduction: given u, v and w, three vertices of V such that $e = (u, v)$ is the unique arc with terminal extremity v, and $f = (v, w)$ the unique arc with initial extremity v. The series reduction operation $\mathcal{S}$ reduces the arcs e and f by a new one (u, w). This reduction deletes the vertex v and their two incident arcs.
- The parallel reduction: given u and v two vertices of V relied by two arcs e_1 and e_2, the parallel reduction operation $\mathcal{P}$ reduces e_1 and e_2 by a new one $e = (u, v)$.

In this binary decomposition tree denoted $\mathcal{T}$, each leaf l_e is associated with an arc $e = (u, v)$ in G and each internal node of $\mathcal{T}$ is either $\mathcal{S}$-node or $\mathcal{P}$-node. In our case, we distinguish the XOR and AND patterns by introducing respectively $\mathcal{P}_{\mathcal{O}}$-node and $\mathcal{P}_{\mathcal{A}}$-node.

Example 5. The leaves-valued binary decomposition tree associated with the workflow described in Fig. 4 and the execution time reported in Table 2 is presented in Fig. 8. The binary tree contains 13 leaves, representing the 13 edges of the workflow. Its root node $\mathcal{T}$ represents the sequence between activity $A1$ (node numbered 1 in the graph) and the XOR pattern represented by split node 2 and join node 11 in the workflow. Its left leaf represents the edge between activity $A1$ and the XOR-split operator 2. Its right child represents the XOR pattern of the workflow, whose left sub-tree represents the two first branches of the pattern and, whose right sub-tree represents the last branch of the pattern. This right sub-tree contains three leaves, representing from left to right, the edges between split node 2 and activity $A6$ (node 9 in the workflow), activities $A6$ and $A7$ (node 10) and, activity $A7$ and join node 11.

Theorem 3. *The WS composition problem can be solved in $O(m)$ when optimizing a min/max-type QoS criterion on a general workflow (with interlaced patterns SEQ, AND and XOR).*

The following algorithm computes for each node x of $\mathcal{T}$ the label $time(x)$ which is the minimal execution time of the workflow described by the subtree rooted by x. In this algorithm, $root(\mathcal{T})$ returns the root of the binary tree $\mathcal{T}$ and, $select(x)$ returns a not-labelled node such that its two children x_l and x_r are labelled.

```
forall leaves l_uv of T do
  if u ∈ V_A then time(l_uv) ← d_u
  else time(l_uv) ← 0
  l_uv is labelled
while root(T) is not labelled do
  select(x)
  if x is a S-node then time(x) ← time(x_l) + time(x_r)
  if x is a P_A-node then time(x) ← max{time(x_l); time(x_r)}
  if x is a P_O-node then time(x) ← min{time(x_l); time(x_r)}
  x is labelled
```

In this algorithm, we label each vertex one and only one time. The number of leaves being equal to m, the number of nodes in the decomposition tree is in $O(m)$ inducing the complexity of the proposed algorithm.

Please cite this article in press as: V. Gabrel, et al., Web services composition: Complexity and models, Discrete Applied Mathematics (2014), http://dx.doi.org/10.1016/j.dam.2014.10.020

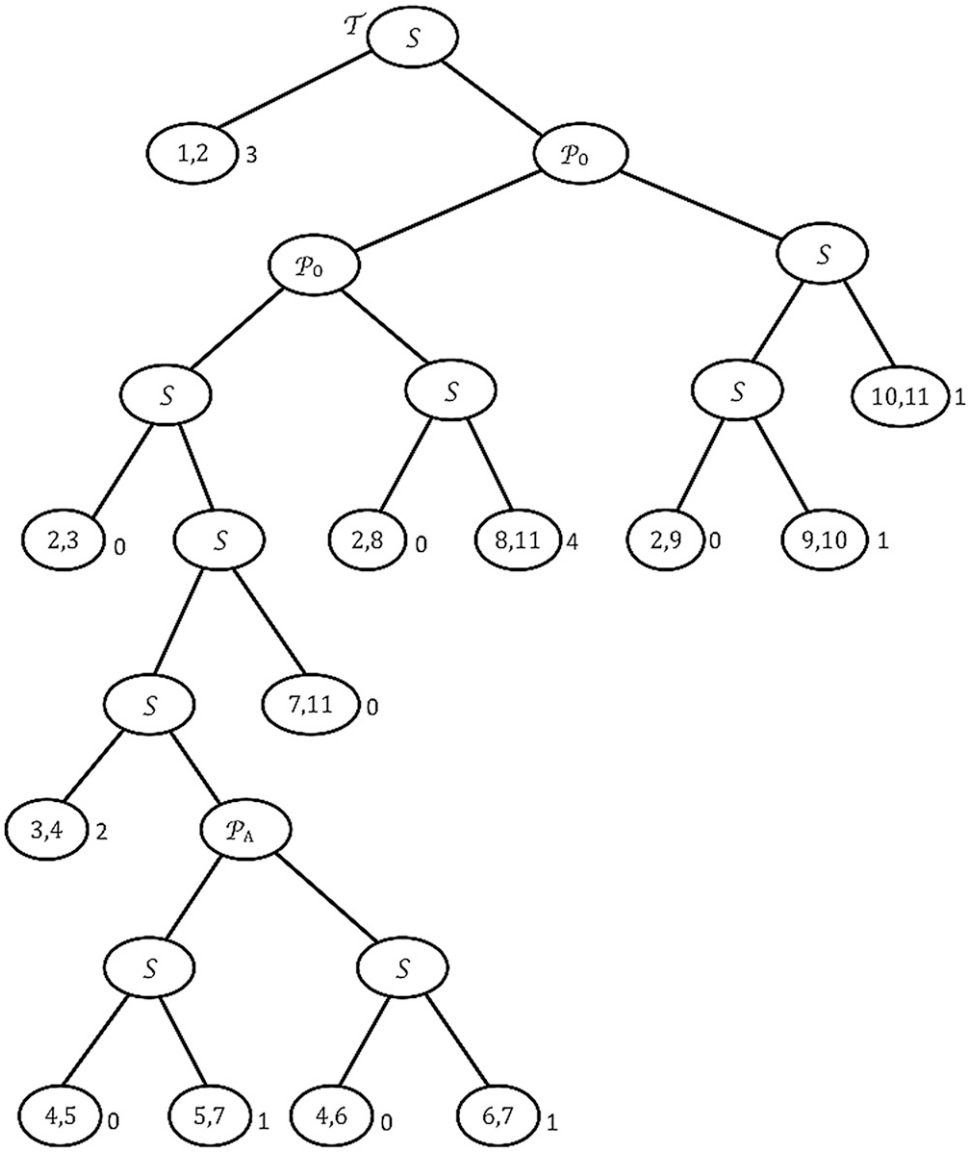


Fig. 8. Binary decomposition tree corresponding to the workflow of Fig. 4.

Example 6. In Fig. 9, we present the labelling of the nodes after applying the previously presented algorithm. The value of the optimal composite WS on the time criterion is equal to 5.

The optimal solution is described by bold lines in Fig. 9. This solution is obtained by recursively selecting both children when the root node is $\mathcal{P}_{\mathcal{A}}$ or $\mathcal{S}$ node, and by selecting the child with the minimum label when the root node is $\mathcal{P}_{\mathcal{O}}$.

Remark 1. If the workflow only contains SEQ and AND patterns (without exclusive choice of activities) the problem of determining the composite WS with minimal execution time is simplified and can be formulated as a longest path problem between the source and the sink.

Thus, considering only QoS criterion to optimize, the workflow-based WS composition problem is easy to solve (even for more complex criterion like execution time). The computational difficulty comes from QoS constraints. Indeed, let us recall that, even for workflow with SEQ patterns only, Yu and Lin show in [19] that the composition problem with one QoS constraint is exactly the Multiple-choice Knapsack Problem which is known to be NP-hard.

3. A new mixed integer linear model for a NP-hard WS composition problem

In this section, we focus on an NP-hard version of the WS composition problem: determining a composite WS, structured by a complex workflow, with minimal cost while satisfying execution time and reliability constraints. We propose a mixed

Please cite this article in press as: V. Gabrel, et al., Web services composition: Complexity and models, Discrete Applied Mathematics (2014), http://dx.doi.org/10.1016/j.dam.2014.10.020

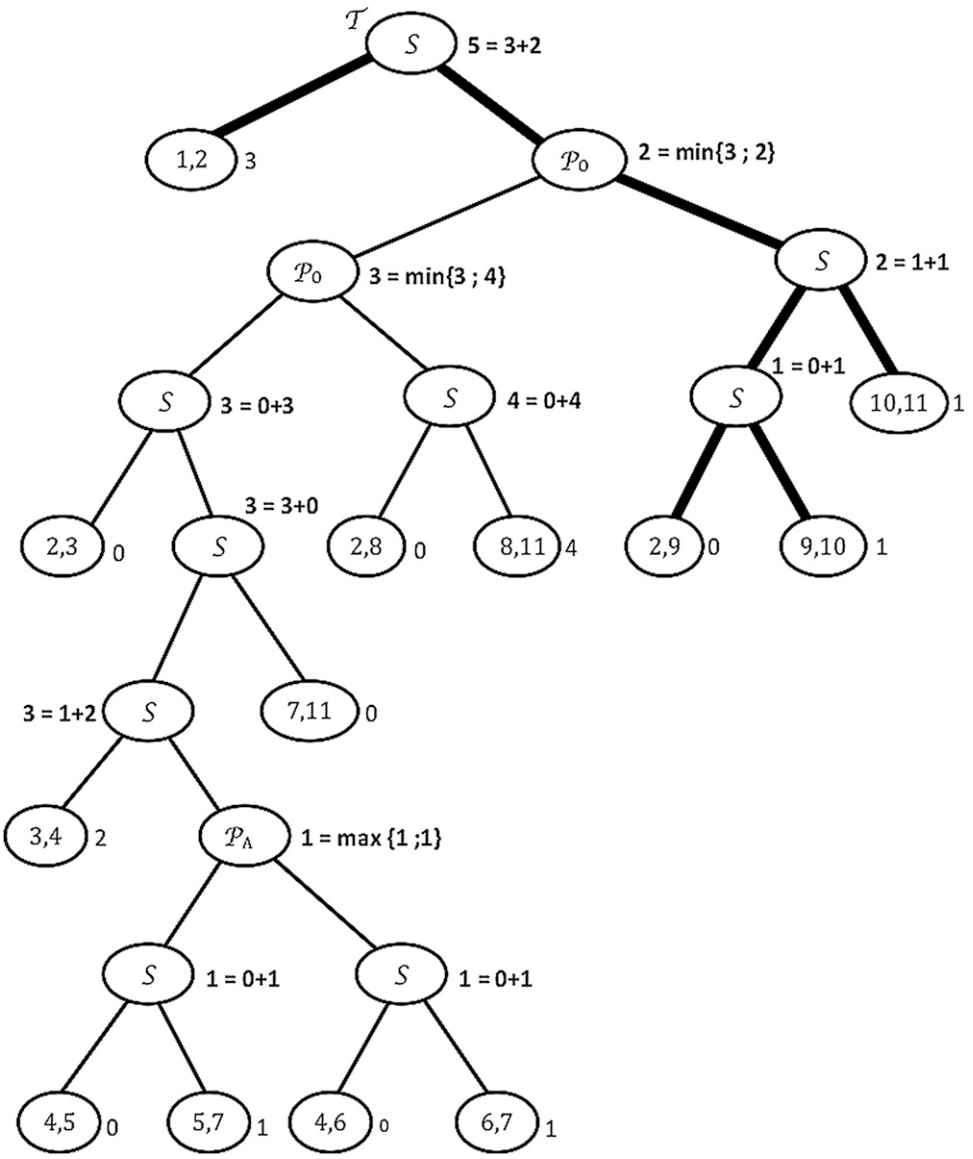


Fig. 9. Optimal solution and *time*(*x*) computed from tree of Fig. 8.

integer linear programming-based model for representing and exactly solving this problem. First, let us remark that when some global QoS constraints have to be satisfied, WS cannot be independently chosen for each activity. Thus, it is no more possible to decompose the problem in two stages: in the first stage, select the best WS for each activity and, in the second stage, select an end-to-end route. The choice of activities and WS is linked and we introduce decision variables for both activities and WS selections.

3.1. Data

The inputs of our model are:

1. a workflow which is a graph $G = (V, E)$, describing the execution order of a set A of n activities and the patterns (SEQ, AND and XOR) combining them,
2. a list W_i of $|W_i|$ web services which can perform each activity Ai (we denote by $W = \cup_{i=1}^{n} W_i$ the set of all WSs with total cardinality $|W|$),
3. a set of vectors (c_{ij}, d_{ij}, p_{ij}), each vector representing the QoS values of WS j performing Ai.

As usual, $\forall v \in V$, $\Gamma^+(v)$ represents the set of direct successors of vertex v and $\Gamma^-(v)$ the set of direct predecessors of vertex v.

Please cite this article in press as: V. Gabrel, et al., Web services composition: Complexity and models, Discrete Applied Mathematics (2014), http://dx.doi.org/10.1016/j.dam.2014.10.020

As seen in Section 2.1, each AND (resp. XOR) pattern is represented by two vertices u and v, where u represents the split of the pattern and v represents the join. For each vertex v which represents a join of AND or XOR pattern, let s be a function which returns the vertex number of the corresponding split. For example, in Fig. 4, $s(11) = 2$. Moreover, let $a(v)$ be the indice of the activity in the workflow associated with vertex v. For example in Fig. 4, vertex 3 is associated with activity $A2$ then $a(3) = 2$. By convention, $a(v) = 0$ if v is not associated with an activity and we define an empty set of WS for this fictitious activity ($W_0 = \emptyset$).

The set E of arcs can be partitioned into 3 subsets: E_{SEQ}, E^S and E^J.

- E_{SEQ} is the set of arcs (u, v) corresponding to sequence patterns, where $u \in V_A \cup V^J_{AND} \cup V^J_{XOR}$ and $v \in V_A \cup V^S_{AND} \cup V^S_{XOR}$,
- $E^S = E^S_{AND} \cup E^S_{XOR}$ is the set of arcs (describing the split operator) whose initial extremity belongs to $V^S_{AND} \cup V^S_{XOR}$,
- $E^J = E^J_{AND} \cup E^J_{XOR}$ is the set of arcs (describing the join operator) whose final extremity belongs to $V^J_{AND} \cup V^J_{XOR}$.

3.2. Decision variables for selecting activities and WS

As previously explained, our WS composition problem is to select simultaneously the activities to be performed and one WS for each chosen activity. For the selection of activities, we introduce 3 types of variables:

1. $\forall v \in V_A$, x_v is a binary variable equal to 1 if the activity with index $a(v)$ is performed and 0 otherwise,
2. $\forall v \in V^J_{AND} \cup V^S_{AND}$, x_v is a binary variable equal to 1 if the corresponding join or split operator of the AND pattern represented by v is successful (meaning that all activities inside the AND pattern are performed) and 0 otherwise,
3. $\forall v \in V^J_{XOR} \cup V^S_{XOR}$, x_v is a binary variable equal to 1 if the corresponding join or split operator of the XOR pattern represented by v is successful (meaning that only one branch inside the XOR pattern is performed) and 0 otherwise.

Any end-to-end route is represented by a 0–1 vector $x = (x_v)_{v=1,\ldots,|V|}$. For the selection of WS, we introduce the following variables: $\forall Ai \in A$ and $\forall j \in W_i$, w_{ij} is a binary variable equal to 1 if activity Ai is performed by WS j and 0 otherwise.

3.3. Constraints for selection

We have to modelize the fact that each selected activity must be realized by exactly one WS and that no WS have to be retained for non-selected activities.

(C1) *WS selection*. For each vertex v belonging to V_A, we have to select at most one WS for activity $a(v)$: $x_v = \sum_{j \in W_{a(v)}} w_{a(v)j}$.

Moreover, the WS composition is completed if the last vertex of the workflow is selected.

(C2) *Ending activity or operator*. For the last vertex of indice $|V|$, we have: $x_{|V|} = 1$. This constraint can be seen as an objective to achieve.

The constraints induced by the workflow, presented in the following section, describe the patterns (and the corresponding activities) that must be selected in order to satisfy constraint (C2).

3.4. Constraints induced by the workflow

We can represent the workflow constraints in a linear form.

(C3) *Sequential pattern*. For each arc $(u, v) \in E_{SEQ}$, if the pattern is selected then $x_u = x_v = 1$. If it is not the case then $x_u = x_v = 0$. So we have: $x_u = x_v$.

(C4) *AND pattern*. For each vertex $v \in V^J_{AND}$, the corresponding AND join pattern v is performed if and only if the last vertex u of each branch is selected: $x_v = x_u \ \forall u \in \Gamma^-(v)$.

(C5) *XOR pattern*. For each vertex $v \in V^J_{XOR}$, if $x_v = 1$ then only one branch has to be selected and, if $x_v = 0$, then no branch is selected. Considering the last vertex u of each branch, these two cases are represented by:

$$x_v = \sum_{u \in \Gamma^-(v)} x_u.$$

(C6) *SPLIT–JOIN*. For each AND (resp. XOR) pattern associated with vertices u and v (with $u = s(v)$), we have: $x_v = x_u$.

Example 7. Applied to the example of Fig. 4, we have the following data: $V_A = \{1, 3, 5, 6, 8, 9, 10\}$, $V^S_{AND} = \{4\}$, $V^J_{AND} = \{7\}$, $V^S_{XOR} = \{2\}$ and $V^J_{XOR} = \{11\}$. Values of functions a and s are represented in Table 3.

Then, we have the following constraints induced by the workflow:

$$\begin{aligned}
&(C2) : x_{11} = 1\\
&(C3) : x_1 = x_2 \quad x_3 = x_4 \quad x_9 = x_{10}\\
&(C4) : x_5 = x_7 \quad x_6 = x_7\\
&(C5) : x_{11} = x_7 + x_8 + x_{10}\\
&(C6) : x_2 = x_{11} \quad x_4 = x_7.
\end{aligned}$$

Please cite this article in press as: V. Gabrel, et al., Web services composition: Complexity and models, Discrete Applied Mathematics (2014), http://dx.doi.org/10.1016/j.dam.2014.10.020

3.5. Qos measures

Concerning the value of a composite WS w represented by the w_{ij} variables, each criterion induces a specific analysis. For the first criterion which is a sum-type criterion, we have: $c(w) = \sum_{i=1}^{n} \sum_{j \in W_i} c_{ij} w_{ij}$. For the second criterion, which is a max-type criterion, the analysis is much complex since some WSs can be executed in parallel inside the same AND pattern. Indeed, in order to compute the execution time of a given AND pattern, we have to compute the execution time of each branch of the pattern (which is the sum of the execution time of the WS chosen for each activity belonging to the branch). Then, the execution time of the considering AND pattern is equal to the maximal branch execution time.

In our model, we introduce the following additional variables: t_v represents the starting time of the vertex v, $\forall v \in V$. The ending time of vertex v is obtained by adding to t_v its execution time depending on the type of v:

- the execution time of a vertex representing an activity $a(v)$ is computed as follows: $\sum_{j \in W_{a(v)}} d_{a(v)j} w_{a(v)j}$ (when the activity $a(v)$ is not performed, all variables $w_{a(v)j}$ are equal to 0 and the execution time is 0),
- the execution time of a vertex representing a pattern is equal to 0.

We have to introduce a set of constraints in order to compute the starting time variables. To begin, we define the starting time of the first vertex of the workflow:

(T0) $t_1 = 0$.

To force $t_v = 0$ when the vertex v is not selected, we introduce the following constraint:

(T1) $\forall v \in V \setminus \{1\}$, $t_v \leq M x_v$, with M a big constant which does not limit the t values. We propose to take M equal to $M = \sum_{i \in A} \max_{j \in W_i} d_{ij}$.

For the sequential patterns, we introduce the following constraint:

(T2) $\forall (u, v) \in E_{SEQ}$ $t_v \geq t_u + \sum_{j \in W_{a(u)}} d_{a(u)j} w_{a(u)j}$ with the convention that $a(u) = 0$ and $W_0 = \emptyset$ when u is a pattern. Thus, when u represents a pattern, the inequality becomes: $t_v \geq t_u$.

Example 8. For the example of Fig. 4 with WS values presented in Table 1, we have:

- for arc (2, 3), constraint (T2) is: $t_3 \geq t_2$
- for arc (9, 10), constraint (T2) becomes: $t_{10} \geq t_9 + 2w_{a(9)1} + w_{a(9)2} + 3w_{a(9)3}$.

For the AND patterns, we introduce the following constraints:

(T3) the first vertex v of each branch can begin when the split represented by u is performed, $\forall u \in V^S_{AND}$, $\forall v \in \Gamma^+(u)$, $t_v \geq t_u$

(T4) the join pattern v can be performed when the last activity of each branch is ended, $\forall v \in V^J_{AND}$, $\forall u \in \Gamma^-(v)$, $t_v \geq t_u + \sum_{j \in W_{a(u)}} d_{a(u)j} w_{a(u)j}$.

Example 9. In our example, only $u = 4$ is a split AND pattern with two branches, then (T3) induces two constraints: $t_5 \geq t_4 t_6 \geq t_4$.

The vertex 7 is the only join AND pattern, then (T4) induces the constraints:

$$t_7 \geq t_5 + 4w_{a(5)1} + w_{a(5)2} + 5w_{a(5)3}$$
$$t_7 \geq t_6 + w_{a(6)1} + w_{a(6)2} + 2w_{a(6)3}.$$

For a split XOR pattern u, only one branch is executed. Then, for the first vertex v of each branch, we have to satisfy the non-linear constraint:

$$\forall v \in \Gamma^+(u), \quad t_v x_v \geq t_u.$$

From (C5), only one branch is selected. Let us denote v' the first vertex of the selected branch. We have: $\sum_{v \in \Gamma^+(u)} t_v x_v = t_{v'} x_{v'}$. Constraint (T1) implies that $t_v = 0$, $\forall v \neq v'$ thus the previous expression can be linearized as follows: $\sum_{v \in \Gamma^+(u)} t_v x_v = \sum_{v \in \Gamma^+(u)} t_v$.

For the XOR patterns, we introduce the following constraints:

(T5) the starting time of the first vertex of the selected branch can be obtained as follows, $\forall u \in V^S_{XOR}$ $\sum_{v \in \Gamma^+(u)} t_v \geq t_u$

(T6) for the ending time of the patterns, $\forall v \in V^J_{XOR}$ $t_v \geq \sum_{u \in \Gamma^-(v)} (t_u + \sum_{j \in W_{a(u)}} d_{a(u)j} w_{a(u)j})$.

Example 10. In the example in Fig. 4, let us consider the vertex $u = 2$ which is a split XOR pattern. Constraint (T5) gives: $t_3 + t_8 + t_9 \geq t_2$.

The join vertex associated being $v = 11$, constraint (T6) gives:

$$t_{11} \geq t_7 + t_8 + 5w_{a(8)1} + 4w_{a(8)2} + 4w_{a(8)3} + t_{10} + 3w_{a(10)1} + 4w_{a(10)2} + w_{a(10)3}.$$

Finally, we have to introduce a global constraint to limit the total duration time to T, a limit fixed by the user. Since we set $t_1 = 0$, the total execution time of the composite WS is given by: $t_{|V|} + \sum_{j \in W_{a(|V|)}} d_{a(|V|)j} w_{a(|V|)j}$.

Please cite this article in press as: V. Gabrel, et al., Web services composition: Complexity and models, Discrete Applied Mathematics (2014), http://dx.doi.org/10.1016/j.dam.2014.10.020

Consequently, the global constraint on time is:

$$\text{(Q1)}\ t_{|V|} + \sum_{j \in W_{a(|V|)}} d_{a(|V|)j} w_{a(|V|)j} \leq T.$$

For the third criterion, which is a multiplicative-type criterion, we apply the same approach as in [21]. The global reliability of a composite WS w, denoted $r(w)$, is given by:

$$r(w) = \prod_{(i=1,\ldots,n,\ j \in W_i): w_{ij}=1} p_{ij}.$$

If we want to guarantee a reliability of probability P (fixed by the user) for the selected composition, then we have to add: $r(w) \geq P$. The linearization of this expression is natural by considering the log function:

$$\log(r(w)) = \sum_{(i=1,\ldots,n,\ j \in W_i): w_{ij}=1} \log(p_{ij}) \geq \log(P).$$

Since $\sum_{(i=1,\ldots,n,\ j \in W_i): w_{ij}=1} \log(p_{ij}) = \sum_{i=1}^{n} \sum_{j \in W_i} \log(p_{ij}) w_{ij}$, the global constraint on reliability is:

$$\text{(Q2)}\ \sum_{i=1}^{n} \sum_{j \in W_i} \log(p_{ij}) w_{ij} \geq \log(P).$$

Constraint (Q1) represents the fact that the execution time is limited by T. Constraint (Q2) represents the fact that the reliability must be at least equal to P. The following section presents the final model.

3.6. Model

Our purpose is to minimize the cost criterion and to satisfy two QoS constraints concerning the execution time and the reliability. Our completed model is:

$$\begin{cases}
\min \sum_{i=1}^{n} \sum_{j \in W_i} c_{ij} w_{ij} & \\
x_v = \sum_{j \in W_{a(v)}} w_{a(v)j} & \forall v \in V_A \ \text{(C1)} \\
x_{|V|} = 1 & \text{(C2)} \\
x_u = x_v & \forall (u,v) \in E_{SEQ} \ \text{(C3)} \\
x_u = x_v & \forall (u,v) \in E^J_{AND} \ \text{(C4)} \\
\sum_{u \in \Gamma^-(v)} x_u = x_v & \forall v \in V^J_{XOR} \ \text{(C5)} \\
x_{s(v)} = x_v & \forall v \in V^J_{AND} \cup V^J_{XOR} \ \text{(C6)} \\
t_1 = 0 & (T0) \\
t_u \leq M x_u & \forall u \in V \setminus \{1\} \ \text{(T1)} \\
t_v \geq t_u + \sum_{j \in W_{a(u)}} d_{a(u)j} w_{a(u)j} & \forall (u,v) \in E_{SEQ} \ \text{(T2)} \\
t_v \geq t_u & \forall (u,v) \in E^S_{AND} \ \text{(T3)} \\
t_v \geq t_u + \sum_{j \in W_{a(u)}} d_{a(u)j} w_{a(u)j} & \forall (u,v) \in E^J_{AND} \ \text{(T4)} \\
\sum_{v \in \Gamma^+(u)} t_v \geq t_u & \forall u \in V^S_{XOR} \ \text{(T5)} \\
t_v \geq \sum_{u \in \Gamma^-(v)} \left(t_u + \sum_{j \in W_{a(u)}} d_{a(u)j} w_{a(u)j} \right) & \forall v \in V^J_{XOR} \ \text{(T6)} \\
t_{|V|} + \sum_{j \in W_{a(|V|)}} d_{a(|V|)j} w_{a(|V|)j} \leq T & \text{(Q1)} \\
\sum_{i=1}^{n} \sum_{j \in W_i} \log(p_{ij}) w_{ij} \geq \log(P) & \text{(Q2)} \\
w_{ij} \in \{0,1\} & \forall i = 1,\ldots,n\ \forall j = 1,\ldots,|W_i| \\
x_v \in \{0,1\} & \forall v = 1,\ldots,|V| \\
t_v \geq 0 & \forall v = 1,\ldots,|V|.
\end{cases}$$

Please cite this article in press as: V. Gabrel, et al., Web services composition: Complexity and models, Discrete Applied Mathematics (2014), http://dx.doi.org/10.1016/j.dam.2014.10.020

Table 3
Functions a and v applied to the example of Fig. 4.

v	1	3	5	6	7	8	9	10	11
$a(v)$	1	2	3	4		5	6	7	
$s(v)$					4				2

Table 4
Linear program size of our experimentation.

Number of activities	Number of constraints	Number of variables for			
		10 WS	50 WS	100 WS	200 WS
10	55	131	531	1 031	2 031
50	269	648	2648	5 148	10 148
100	514	1285	5285	10 285	20 285

Example 11. Let us consider the example of Fig. 4 and Table 1. In order to determine the less expensive composite WS, we have to solve our linear program by setting $M = T = \sum_{i=1}^{7} \max_{j\in\{1,2,3\}} d_{ij} = 28$ and $P = 10^{-6}$.

An optimal solution is given by the following end-to-end route $\{A1, A5\}$ with the third WS for $A1$ and the first WS for $A5$. The cost of the optimal solution is 23, its duration is 10 and its reliability is 0.94.

If the duration time has to be less than 6 (setting $T = 6$ in Q1) then the minimal cost solution is given by the end-to-end route $\{A1, A6, A7\}$ with the second WS for $A1$, the first WS for $A6$ and the third for $A7$. The solution value is equal to 48, the duration time of this composite WS is equal to 6 and its reliability is 0.87.

The number of binary variables is equal to $|V|+|W|$, plus $|V|$ real and non negative variables which gives $O(|W|)$ variables. The number of constraints is equal to $2n+3+2|E_{SEQ}|+3|E^J_{AND}|+3|V^S_{XOR}|+|V^J|$ which gives $O(|E|)$ constraints. Indeed, we have n constraints for (C1), $|E_{SEQ}|$ constraints for (C3) and (T2), $|E^J_{AND}| = |E^S_{AND}|$ constraints for (C4), (T3) and (T4), $|V^J_{XOR}| = |V^S_{XOR}|$ for (C5), (T5) and (T6) and $|V^J|$ for (C6).

Let us underline that our model can represent several variants of the composition problem: minimize the execution time subject to a budget constraint, maximize an aggregate score subject to several QoS constraints .... In practice, it is of interest to know the best value we can obtain on each criterion. For that, by changing the objective function of our model, we can minimize total execution time (without any QoS constraint) or minimize total cost (without time constraint).

Example 12. For the previous example, by minimizing t_{11} and removing (Q1) and (Q2), we obtain 5 which is the minimal duration of a composite WS (with a total cost equal to 50). This composite WS is described by: $w_{12} = w_{62} = w_{73} = 1$.

4. Experimentations

The model was solved using CPLEX solver, and the experiments were carried out on a Dell PC with Intel (R) Core TM i7-2760, with 2.4 GHz processor and 8 Go RAM, under Windows 7 and Java 7. For testing purpose, we randomly generated workflows with interlaced XOR and AND patterns, by varying the number of activities (from 10 to 200), and by randomly varying the number of AND and XOR patterns (representing at most 30% of the number of activities in the workflow) as well as their position in the workflow. The WS registry was generated by randomly assigning QoS values to each candidate WS (a value between 10 and 50 for the price and a value between 50 and 200 ms for the duration). For a given number of activities and a given number of WS per activity, 10 instances are randomly generated in order to compute average value.

In order to analyse the impact of one QoS constraint on computation time and on optimal solution value, we choose in a first step to relax the reliability constraint Q2: the reliability of each WS is equal to 1 and $P = 10^{-6}$.

Table 4 reports the average size of considered linear programs. The number of constraints only depends on the workflow.

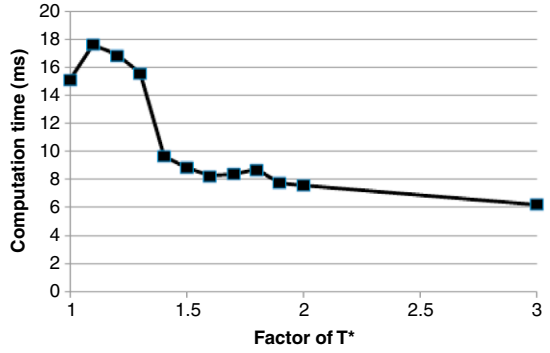
In the following experiments, for a given workflow and WS registry, we begin by determining the composite WS with minimal duration time, denoted T^*.

In order to analyse the impact of the constraint Q1 on computation time, we first consider 50-activities workflows with 100 WS per activity. In Fig. 10(a), we report the computation time when T increases from T^* to $3T^*$. This figure clearly shows that "easy" cases (with computation times less than 9 ms) appear for high values of T (greater than $1.5T^*$). The computation times are higher (around 17 ms) when T varies from $1.1T^*$ to $1.3T^*$.

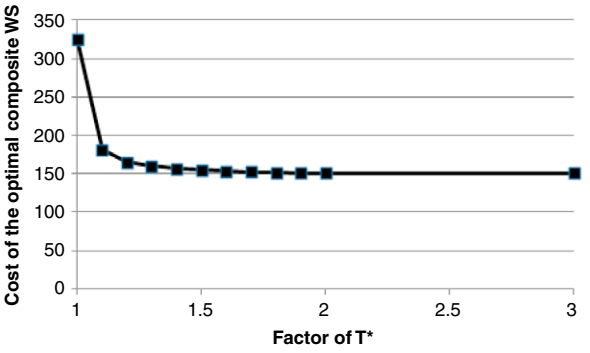
In order to analyse the influence of the number of WS on the computation time, we first consider 50-activities workflows with an increasing number of WS (from 50 to 500 per activity). Then, we compute minimal cost composite WS in two cases: in the first case, the total duration must not exceed $1.1T^*$ and, in the second case, the duration must not exceed $1.5T^*$ (the QoS constraint Q1 is less restrictive). The results are presented in Fig. 11(a).

As previously noted, for a given number of WS, the optimal solution is found more rapidly when the constraint on time is "relaxed" (equal to $1.5T^*$). For realistic instances, around 50 WS per activity, the average computation time (obtained on 10 workflows) is less than 7 ms when T equals $1.5T^*$, and less than 13 ms when T equals $1.1T^*$. These experimental results are very promising. Moreover, it appears that the execution time linearly increases with the number of WS.

Please cite this article in press as: V. Gabrel, et al., Web services composition: Complexity and models, Discrete Applied Mathematics (2014), http://dx.doi.org/10.1016/j.dam.2014.10.020

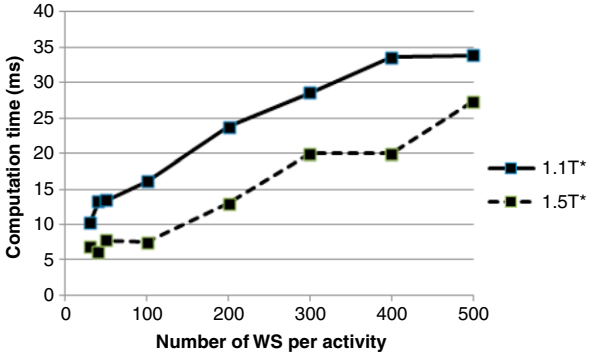


(a) Computation time in function of the QoS time constraint.

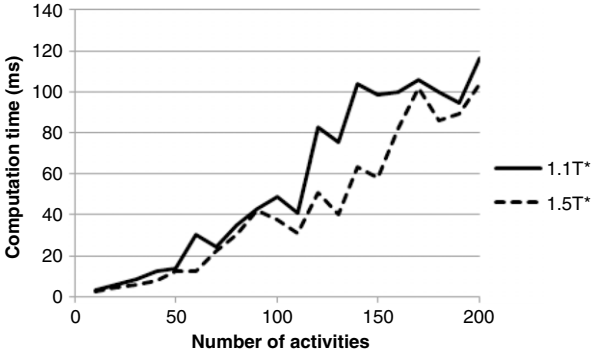


(b) Cost of the optimal composite WS in function of T.

Fig. 10. Computation time and cost for 50-activities workflows with 100 WS per activity.



(a) For 50-activities workflow.



(b) For 100-activities workflow.

Fig. 11. Computation time by varying the number of WS and the number of activities.

Fig. 10(b) shows the link between the cost of the optimal composite WS and the time constraint for 50-activities workflows with 100 WS per activity. It clearly appears that the cost of the optimal composite WS rapidly decreases when the QoS time constraint is relaxed. More precisely, when T increases from T^* to $1.2T^*$, the cost of the optimal solution is divided by 2. When T equal to $1.5T^*$, the minimal cost of the optimal composite WS is reached in a very large majority of cases.

Afterwards, we consider different workflows with an increasing number of activities (with 100 WS per activity). Fig. 11(b) presents the evolution of computation times for activities' number varying from 10 to 200. We note that the computation time linearly increases with the number of activities. Moreover, the computation time is of the same order for the two considered values of T.

In a second round of experiments, we re-introduce the reliability constraint Q2 to our model. The score of each WS on the reliability criterion is randomly chosen in the range [0, 1] and we set $P = 0.5$. Our experiments, reported in Fig. 12, show that the linear program is more difficult to solve with two QoS constraints. Indeed, the computation time exponentially increases with the number of WS. However, it is still reasonable for realistic instances (an average of 100 ms for a 50-activities workflow with 50 WS per activity).

These computational experiments are very promising. With the proposed model, the WS composition problem with a sum-type criterion to optimize and a single QoS constraint is tractable even for large size instances. However, the same problem with an additional QoS constraint is more difficult to solve for large size instances with a standard solver like CPLEX.

5. Conclusion

This article is devoted to the workflow-based WS composition problem, once the WS discovery and the semantic reconciliation are done. We present a theoretical complexity analysis of QoS-aware workflow-based WS composition, pointing out polynomial cases. Our theoretical complexity analysis underlines that the computational difficulty of the WS composition problem comes from global QoS constraints. When the QoS requirements are criteria to optimize, the problem becomes easy to solve even for nonlinear criterion.

When QoS constraints have to be considered, we propose a mixed integer linear program for determining a composite WS minimizing a sum-type criterion. This model allows to simultaneously optimize the activities and the chosen WS per

Please cite this article in press as: V. Gabrel, et al., Web services composition: Complexity and models, Discrete Applied Mathematics (2014), http://dx.doi.org/10.1016/j.dam.2014.10.020

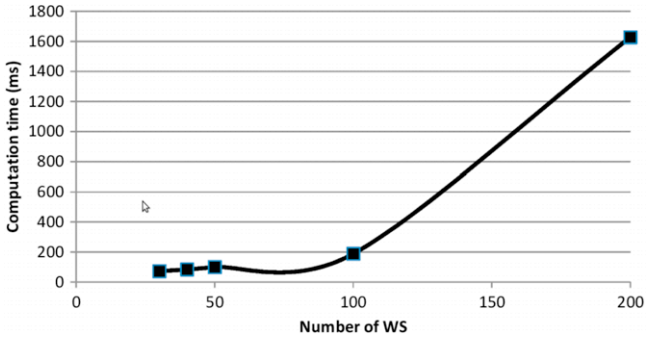


Fig. 12. Computation time for 50-activities workflow with $T = 1.5T^*$ and $P = 0.5$.

activity in a workflow including interlaced XOR and AND patterns. Compared with graph-based approaches [20], we do not have to enumerate all the potential end-to-end routes in the workflow. Our model is the first one with polynomial numbers of variables and constraints (polynomial in function of the number of WS and patterns). Extensive computational experiments on random workflows and WS registries show that our approach is very promising for exactly solving the QoS-aware composition problem.

However, the computation time is more important when several QoS constraints are considered. It takes several seconds to find the optimal solution for big size instances and this may be too long for real time computation. Therefore, a study on graph reduction has to be specifically conducted (since the graph is series–parallel) in order to simplify the mixed integer linear formulation. On the other hand, it would be useful to propose approximate algorithms to find solutions very rapidly. With our linear programming approach, we are able to determine the optimal solution, to compute the distance between the optimal solution and the approximate one and thus to obtain an experimental approximation ratio. This is left for future research.

References

[1] R.K. Ahuja, T.L. Magnanti, J.B. Orlin, Network Flows: Theory, Algorithms, and Applications, Prentice Hall, 1993.
[2] A. Aleti, B. Buhnova, L. Grunske, A. Koziolek, I. Meedeniya, Software architecture optimization methods: A systematic literature review, IEEE Trans. Softw. Eng. 39 (5) (2013) 658–683.
[3] E. Al-Masri, Q.H. Mahmoud, Investigating Web services on the World Wide Web, in: Proc. of the 17th Int. Conf on World Wide Web, WWW'08, 2008, pp. 795–804.
[4] M. Alrifai, D. Skoutas, T. Risse, Selecting skyline services for QoS-based Web service composition, in: Proc. of the 19th Int. Conf. on World Wide Web, WWW'10, 2010, pp. 11–20.
[5] G. Baryannis, O. Danylevych, D. Karastoyanova, K. Kritikos, P. Leitner, F. Rosenberg, B. Wetzstein, Service composition, in: Service Research Challenges and Solutions for the Future Internet, in: LNCS, vol. 6500, Springer-Verlag, 2010 pp. 55–84. (Chapter).
[6] B. Benatallah, M. Dumas, Z. Maamar, Definition and execution of composite Web services: The SELF-SERV project, IEEE Data Eng. Bull. 25 (4) (2002) 47–52.
[7] R. Berbner, M. Spahn, N. Repp, O. Heckmann, R. Steinmetz, Heuristics for QoS-aware Web service composition, in: Proc. of the IEEE Int. Conf. on Web Services, ICWS'06, IEEE Computer Society, Washington, DC, USA, 2006, pp. 72–82. http://dx.doi.org/10.1109/ICWS.2006.69.
[8] L. Cui, S. Kumara, D. Lee, Scenario analysis of Web service composition based on multi-criteria mathematical goal programming, Serv. Sci. 3 (4) (2011) 280–303.
[9] R. Duffin, Topology of series–parallel networks, J. Math. Anal. Appl. 10 (1965) 303–313.
[10] V. Gabrel, M. Manouvrier, I. Megdiche, C. Murat, A new 0–1 linear program for QoS and transactional-aware Web service composition, in: Proceedings IEEE Symposium on Computers and Communications, ISCC, 2012, pp. 845–850.
[11] A. Goldman, Y. Ngoko, On graph reduction for QoS prediction of very large Web service compositions, in: Proc. of the 2012 IEEE Int. Conf. on Services Computing, SCC, 2012, pp. 258–265.
[12] J.E. Haddad, Optimization techniques for QoS-aware workflow realization in Web services context, in: Z. Lacroix, M. Vidal (Eds.), Proc. of the Third Int. Conf. on Resource Discovery, RED'10, in: LNCS, vol. 6799, 2012, pp. 134–149.
[13] S.-C. Oh, D. Lee, S.R.T. Kumara, Web Service Planner (WSPR): An effective and scalable Web service composition algorithm, Int. J. Web Serv. Res. 4 (1) (2007) 1–22.
[14] S.-C. Oh, D. Lee, S.R.T. Kumara, Effective Web service composition in diverse and large-scale service networks, IEEE Trans. Serv. Comput. 1 (1) (2008) 15–32.
[15] A. Strunk, QoS-aware service composition: A survey, in: Proc. of the 2010 Eighth IEEE European Conf. on Web Services, ECOWS'10, IEEE Computer Society, Washington, DC, USA, 2010, pp. 67–74. http://dx.doi.org/10.1109/ECOWS.2010.16.
[16] W. Tan, M. Zhou, Business and Scientific Workflows: a Web Service-Oriented Approach, in: IEEE Press Series on Systems Science and Engineering, Wiley-IEEE Press, 2013.
[17] J. Valdes, R. Tarjan, E. Lawler, The recognition of series parallel digraphs, SIAM J. Comput. 11 (2) (1982) 298–313.
[18] YAWL: Yet Another Workflow Language, retrieved Septemper 24, 2013, from http://www.yawlfoundation.org/.
[19] T. Yu, K.-J. Lin, Service selection algorithms for Web services with end-to-end QoS constraints, Inf. Syst. e-Bus. Manage. 3 (2005) 103–126.
[20] T. Yu, Y. Zhang, K.-J. Lin, Efficient algorithms for Web services selection with end-to-end QoS constraints, ACM Trans. Web 1 (2007) 1–26.
[21] L. Zeng, B. Benatallah, A. Ngu, M. Dumas, J. Kalagnanam, H. Chang, QoS-aware middleware for Web services composition, IEEE Trans. Softw. Eng. 30 (5) (2004) 311–327.

Please cite this article in press as: V. Gabrel, et al., Web services composition: Complexity and models, Discrete Applied Mathematics (2014), http://dx.doi.org/10.1016/j.dam.2014.10.020