Discrete Applied Mathematics 196 (2015) 100–114

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 15 November 2014

*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

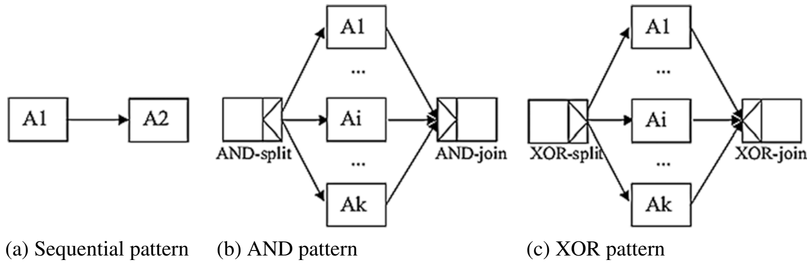


**Fig. 1.** Considered workflow patterns.

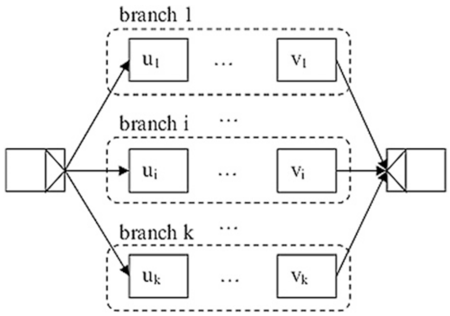


**Fig. 2.** Branches of XOR pattern.

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 $A1, \dots, Ak$ 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.