A GRAPH-BASED APPROACH FOR WEB SERVICES COMPOSITION

Salima Benbernou, Djamal Benslimane, Mohand Said Hacid
LIRIS, Computer Science department, Claude Bernard Lyon 1 University, France

Hamamache Kheddouci
LIESP, Claude Bernard Lyon 1 University, 43 boulevard du 11 Novembre 1918, 69622 Villeurbanne, France

Kamel Tari
omputer Science Department, Bejaia University, Algeria

Keywords: Oriented graph, semantic web services, web services composition, web services discovery.

Abstract: The automatic web services composition plays an important role in the area of semantic web services. In fact, in a real life it is not usual to discover *atomic* web services matching with the user's query. So composing services fulfilling the query is needed. This paper discusses the web services composition model based on an oriented graph. It is based on Bcov algorithm (Benatallah.B et al., 2005a), initially defined for web services discovery and considered as the best cover problem, which it was mapped as the problem of computing the minimal transversals with minimum cost of a weighted hypergraph. We propose two algorithms handling the oriented graph reasoning for web service composition. At first glance, a basic composition version for Bcov algorithm is given. The algorithm composes services over the induced subgraphs of each transversal provided by Becov algorithm, and gives a covering ratio satisfying the query. The second algorithm used a composed services graph maintained by a web services provider. The composition in that case is based on Dijkstra algorithm, where for each two consecutive services in the best minimal transversal provided by Bcov algorithm, a shortest path is computed regarding the composed services graph.

1 INTRODUCTION

Many organizations are putting their business available through the Internet as Web services. One of main challenge issue in the area of semantic web services is the automation of the combination of these services. Therefore, there is a need of tools developing complex services by composing other services with total transparency.

There exists a plethora of different approaches to Web service composition and for that reason a corresponding number of summaries and overviews as well (Hull et al., 2003; Hull, 2003; Rao and Su, 2004).We limit ourselves of some of them.

Composition realized by BPEL4WS rests on a static workflow level without taking any semantic information into account (Yuan.Y et al., 2006). Hereafter, we present recent practical and theoretical approaches to automated composition.

In (Benbernou.S et al., 2004b; Benbernou.S et al., 2004a) an hybrid framework combining DL ontologies with Constraint Satisfaction Problem (CSP) is presented. This approach utilizes a calculus reasoning about a DL terminology and subsequently solving constraints imposed on the given services. The proposition in (Bultan et al., 2003) is expressed by the means of the conversation notion. They introduce a framework for modeling and specifying the global behavior of Web service composition, in which the peers (individual Web service) communicate through asynchronous messages. A global watcher keeps track of messages as they occur. A conversation is in this context a sequence of (classes of) messages observed by the watcher. The peers are represented also by Mealy Finite State Machines (MFSM)(John E. Hopcroft, R. Motwani and Jeffrey D. Ullman, 2001).

(Hamadi and Benatallah, 2003) propose a full algebra of Petri-Nets[1]. Each Web service is represented by

[1]A Finite State Machine (FSM) (John E. Hopcroft, R.

Benbernou S., Benslimane D., Said Hacid M., Kheddouci H. and Tari K. (2007).
A GRAPH-BASED APPROACH FOR WEB SERVICES COMPOSITION.
In *Proceedings of the Third International Conference on Web Information Systems and Technologies - Internet Technology*, pages 290-296
DOI: 10.5220/0001289102900296

such a net including the null net. Operations are for instance sequence, alternative or execution in parallel of two services. However, all existing constraints are implicit within the transitions of the Petri-Nets. In this approach, they only take the behavior into account described by a sequence of actions and messages. In (Berardi.D et al., 2003a; Berardi.D, 2004; Berardi.D et al., 2003b) the authors relate an e-service execution to an internal and an external schema. These schemes correspond respectively to an unfolded internal and external FSM describing the behavior of the service in terms of actions. They do not consider any constraints on inputs/outputs. (Bordeaux.L et al., 2004) constitutes another approach concentrating on composition between two services in a behavioral context. Web service composition can be considered as a form of Artificial Intelligence (AI) (Stuart J. Russell and P. Norvig, 2003) planning task.

In (McIlraith and Son, 2002; Narayanan and McIlraith, 2002) composition is addressed by using the language GOLOG and providing a behavioral semantic of the composition based on Petri Nets (for OWL-S). GOLOG is a logic programming language based on situation calculus(R. Reiter, 2002). The authors treat the Web service composition problem through the provision of high level generic procedures and customizing constraints. GOLOG is adapted for representing and reasoning about this problem. In (Medjahed et al., 2003) the authors present a technique to generate composite services from high level declarative description. The method uses composability rules to determine whether two services can be copmposed. There are syntactical and semantical rules: the syntactical ones for protocol bindings and the semantical rules to manage 1) message composability, 2) operational semantic composability and composition soundness.

In (Doshi.P and Zhao.H, 2006), the authors present a hierarchical approach for composing web processes that may be nested some of the components of the process may be Web processes themselves. They model the composition problem using semi-Markov decision process (SMDP) that generalizes MDPs by allowing actions to be temporally extended. They use these actions to represent the invocation of lower level Web processes whose execution times are different from simple service invocations.

In (Tan.M et al., 2006) is proposed an agent-based method using Fuzzy Distributed Constraint Satisfaction Problem (Fuzzy DisCSP). They provide techniques to solve the problem of QoS composition problem, which is how to compose a service from different subcomponent services so that its overall QoS can satisfy certain requirements. The authors show that by using the composition structures, local constraints can be constructed and used with DisCSP.

In this paper we propose an oriented graph based model for modeling web services composition upon Bcov algorithm. The framework allows to compose services by providing two approaches:

1. by means the services discovered by Bcov algorithm, finds the compositions as paths (or subgraphs) between two services regarding the matching of the outputs with inputs,
2. Dijkstra's algorithm is based on our calculus for the composition to find the shortest Path between two services given a global oriented graph maintained by a web services provider.

In the rest of the paper, we shall not handle the semantic matching between inputs and outputs, we assume it is computed in a black box.

The remainder of the paper is organized as follows: The next section gives two motivating examples for the proposed approaches. In the section 4 we introduce the oriented graph model for web services composition and provide a reasoning approach by looking for a path or a subgraph fulfilling the user query. In Section 5, due to the limitations of the previous algorithm, we provide a second approach using a global *oriented graph* which is managed by a web services provider (for web services composition) and finding the shortest path for the composition of services.

2 MOTIVATING EXAMPLES

In order to show the interactions between web services in terms of inputs and outputs, let us illustrate our approach through examples.

Example 1 *Given a user query looking for a service translator providing the translation text from French to Thai language, the query input is French and the output is Thai. let us assume a web services provider describing a set of following services:*

1. *Translation from French to English* (S_{F-E})
2. *Translation from English to German* (S_{E-G})
3. *Translation from English to Chinese*(S_{E-C})
4. *Translation from Chinese to Thai ...* (S_{C-T})

A web service fulfilling the query constraints is not discovered. However, if we consider all services, we can find a composition of them, answering the query, given by the following ordered sequence $\{(S_{F-E})$,

Motwani and Jeffrey D. Ullman, 2001) allows to verify easily properties of a composition - often using model checking (Merz, 2000) methods for example.

(S_{E-C}), $(S_{C-T})\}$. *This composition is called linear. This composition is made through the matching between outputs and inputs of services.*

Example 2 *Let us consider the client query searching for a travel accommodation. Let us assume a web services provider, giving following web services:*

- *Tour-club, having a client name and credit card as input providing Flight-reservation and Accommodation-reservation.*
- *Leisure-club, having Accommodation-Reservation as inputs providing swimming-leisure .*
- *Renting-car, having Flight-reservation as input providing (Renting-car,Renting-boat).*

The provider could not find an atomic services fulfilling the constraint of the requester, so the answer is a composition of following web services: Tour-club +(Leisure club + renting car), this composition is called non linear.

3 WEB SERVICES COMPOSITION

Informally speaking, each services is defined by its inputs and outputs.

3.1 Basic Definitions

Definition 1 ***(A Web service.)*** *A Web service $\mathcal{S}$ is defined as a pair $(\mathcal{I}, \mathcal{O})$ where $\mathcal{I} = \{\mathcal{I}_i \mid i > 0\}$ a set of service inputs and $\mathcal{O} = \{\mathcal{O}_j \mid j > 0\}$ a set of service outputs.*

Definition 2 ***(Semantic composition.)*** *Two services S_i et S_j can be composed semantically together which we denote by $S_i \triangleright S_j$ iff $\exists \mathcal{O}_k \in \mathcal{O}(S_i), \exists \mathcal{I}_l \in \mathcal{I}(S_j) \mid \mathcal{O}_k \sqsubseteq \mathcal{I}_l$ where $\mathcal{O}(S_i)$ and $\mathcal{I}(S_j)$ are sets of service outputs and inputs respectively, and $\sqsubseteq$ denotes a semantic matching symbol.*

The definition means two services can be composed iff the outputs (not necessarily all) of one match semantically the inputs of the other. We assume the matching is doing through a black box and is done using existing tools as those presented in (Sycara.K et al., 2003; Benatallah.B et al., 2005a).

Definition 3 ***(Atomic web services.)*** *A service S_i is called atomic iff $\nexists (S_i \triangleright S_j)$, where $S_j \neq S_i$.*

Definition 4 ***(A web services oriented-graph.)*** *Let G be an oriented Graph defined by $(S, E)$where, S is a set of web services defined by its inputs and outputs, and E a set of oriented edges, such as an ongoing edge denoted by $(\overrightarrow{S_1, S_2})$ means a service S_1 can be composed with S_2 regarding the inputs and outputs.*

Definition 5 ***(A linear composition.)*** *Let G be an oriented graph defined by (S, E), the composition is called linear iff for each node there is only one outgoing edge, $\forall S_i, S_j \in S, \exists e \in E \mid e = (\overrightarrow{S_i, S_j}), j \geq 1$ and $|e| = 1, where\ |(S_i, Sj)|$ means the cardinality of the edges between S_i, S_j. For instance in Figure 1(a), all services have only one outgoing edge.*

Definition 6 ***(A non linear composition.)*** *Let G be an oriented graph defined by (S, E), the composition is called non linear iff for each node there is more than one outgoing edge, $\forall S_i, S_j \in S, \exists e \in E \mid e = (\overrightarrow{S_i, S_j}), |e| > 1$. For instance in Figure 1 (b), the service S_2 has two outgoing edges.*

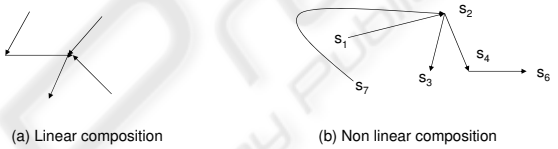

Figure 1: Examples of web services composition graph.

3.2 Overview of Bcov Algorithm

In (Benatallah.B et al., 2005b; Benatallah.B et al., 2005a), an algorithm for web services discovery was provided. It is a hypergraph based approach. The key idea underlying the Becov algorithm is: given a query Q and a set of services S, the problem of discovery consists in finding a subset of S called a "cover" of Q that contains as much as possible of common information with Q and as less possible of extra information with respect to Q. The services and query are described by a conjunction of atoms (in description logic formalism). The services are discovered through the construction of a hypergraph (Σ, Γ) where Σ is a set of vertices corresponding to services description and each conjuncts in the query becomes an edge of Γ in the hypergraph. The algorithm computes the minimal transversals with the best cost for the hypergraph looking for a best cover of services minimising the conjunction of services matching the maximum of atoms in the query (for more details see (Benatallah.B et al., 2005b; Benatallah.B et al., 2005a).
Let us illustrate the algorithm by the following example, let us assume we have a query defined by the following conjunction $Q = A_1 \wedge A_2 \wedge A_3 \wedge A_4 \wedge A_5 \wedge A_6 \wedge$

$A_7 \wedge A8 \wedge A9$ and a set of services $S_1, S_2, S_3, ...S_n$. In Figure 2 is depicted the corresponding hypergraph and the best cover of the query Q, represented by two minimal transversal: $(S_2 \wedge S_4), (S_5 \wedge S_8)$.

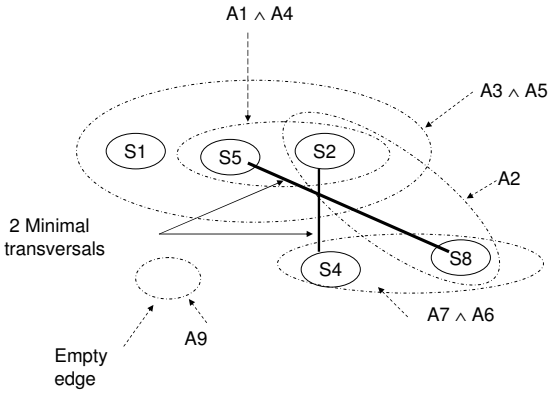

Figure 2: Example of web services hypergraph modeling.

3.3 Oriented Query

Definition 7 ***(oriented query.)*** *A query Q is called oriented and denoted by $\overrightarrow{Q}$ iff the atoms composing Q are ordered.*

Example 3 *Let us consider the following query . "I want to go from Paris to Barcelona on 3rd March, look for an accommodation there for one week (from 3rd to 10th of March), and rent a car". We can find a flight or railway services and hotel service and so on. However, we can not accept what the service hotel offers me before finding flight service . So the discovered services might respect an order to fulfill the query.*

Such reasoning is not provided in Bcov algorithms. So next, we will show how we can improve it by introducing the composition .

3.4 Basic Reasoning with Bcov for the Composition

The idea behind the composition which we propose is given by the following steps:

- for each minimal transversal provided by Bcov algorithm, we build an oriented graph where each node corresponds to a service and the edge are the relationships between inputs and outputs of services (in the transversal).
- after discovering atomic services which are isolated nodes in the graph, we order the services of the minimal transversal regarding the query atoms.
- atomic services are removed from the transversal. And we try to find in the oriented graph, a path (a linear composition) or a subgraph (non linear composition) between first and the last services (of the ordered set) taking into consideration all the intermediary services.
- we compute the cost of the retrieval by considering only the composed services regarding the query. The cost is given by the *covering ratio.*

Definition 8 ***(Covering ratio)*** *It is computed by the ratio of the number of web services induced in the composition on the total of services provided in the best cover set: $\mathcal{CR} = \frac{|composed-web-services|}{|total-services|}$, where $||$ means the cardinality.*

Let us illustrate the approach by the following example:

Example 4 *Let us assume, a query described by the following atoms: $\overrightarrow{Q} = \overrightarrow{A_1}, \overrightarrow{A_2}, \overrightarrow{A_3}, \overrightarrow{A_4}, \overrightarrow{A_5}, \overrightarrow{A_6}, \overrightarrow{A_7}$ and a set of minimal transversal provided by Bcov algorithm:$\{T_1, T_2, T_3\}$ where $T_1 = \{S_1, S_3, S_6, S_7, S_4, S_8\}$, $T_2 = \{S_1, S_2, S_7, S_9, S_{10}, S_{11}\}$, $T_3 = \{S_1, S_2, S_{12}, S_{13}, S_{20}\}$. For T_1, T_2, T_3, we build a corresponding graph where the inputs match the outputs. Figure3 gives an example of the oriented graph associated to T_1. We notice in the graph it may exist a nonlinear composition for S_3.*
Figure4(a)shows the mapping between query atoms and the services provided in T_1. Figure4(b) provides the ordered set after the mapping.
Finally, the aim is to find a path (or a subgraph) starting from S_1 to S_4 including S_6, S_3, S_7 on the oriented graph (S_8 is an atomic service). The only path we can find is a subgraph which is a nonlinear composition (S_3, S_4) and (S_3, S_7) depicted in Figure5. We now compute the covering ratio:$\mathcal{CR} = \frac{(3+1)}{6} = \frac{2}{3}$. The same calculus is done for all minimal transversal and at the end we chose the one with the best ratio.

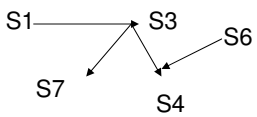

Figure 3: oriented graph for T_1 transversal.

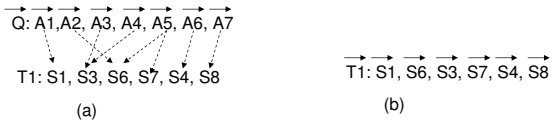

Figure 4: (a)atoms belonging to the best cover service T1, (b) best cover ordered regarding the query atoms.

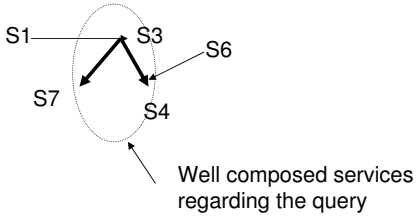

Figure 5: Well composed services in the oriented graph.

The Bcov improvement is given by Bcovcomp algorithm.

3.5 Discussion About Bcovcomp

Statistically speaking, the algorithm can be executed for at most 20 services for each minimal transversal, consequently the composition can be processed by *hand*. Furthermore, the covering ratio is used for many possibilities of the composition. The complexity is reasonable (of course after applying Bcov which is NP-hard) it is linear. However, as we see in the example, we can not find a path from S_1 to S_6. This problem arises due to the fact that the graph is local, where the composition is limited only for services in the transversal. To overcome the problem, in the next section we give an approach based on a global oriented graph maintained by a provider.

4 ORIENTED-GRAPH REPRESENTATION FOR WEB SERVICES COMPOSITION

We assume a web services provider maintains an oriented-graph called *global graph* where each vertex represents a service and an outgoing edge means a possible composition between two services published by the provider (an output of the service with the input of some services). The idea behind the second approach is, considering the transversal minimal with the best cost (not using all minimal transversals as we did in the previous algorithm), finding the shortest path between two consecutive services of the ordered transversal (regarding the oriented query, without considering the atomic services). At the end of the process, we obtain a subgraph which is considered the solution of the query. Let us first recall the shortest path problem.

Algorithm 1 Bcovcomp. (sketch)

Require: A set $\mathcal{BS}$: the minimal transversales of the query Q provided by BCOV.
Ensure: A set $\mathcal{WBS}$: the well composed of the best covers of the query Q.
1: **for all** set of the covers $SC_i \in \mathcal{BS}$ **do**
2: Built an oriented graph $\mathcal{OG}_i$ for SC_i
3: order SC_i regarding the order of query atoms.
4: **for all** $S_j \in SC_i$ **do** { *// Looking for atomic web services*}
5: **if** S_j is atomic **then**
6: the service can be considered in any order in the best cover.
7: **else**
8: It can be composed, keep it.
9: **end if**
10: **end for**
11: Find a path (for linear composition, or a subgraph for non linear composition) stating from the first service to the end one in $\mathcal{OG}$.
12: Compute the covering ratio $\mathcal{CR}_i$
13: **end for**
14: Choose the best cover regarding the best covering ratio.

Definition 9 ***(The shortest path problem.)***. *It is defined as follows: Given a graph $G = (S, E)$, and $e \in E$, we associate to each edge a weight $l(e) \geq 0$ (in our case we assume all edge have a weight equal to 1) we call it the length of the edge u. Find an elementary path μ , starting from an edge S_i to another one S_j, such as the total length $l(\mu) = \sum_{e \in \mu} l(e)$ is as small as possible.*

There are many algorithms handling the shortest path problem, we use the one developed by Dijkstra algorithm (for more details of the algorithm see (E.J.Dijkstra, 1959; S.Skiena, 1990), we denote it by Dijkstra(G,s), which given a graph G finds all shortest paths from s to each other vertex in the graph. And *shortestPath*(G,s,t) uses Dijkstra to find the shortest path from s to t.

Example 5 *Let us illustrate the second approach by considering the previous example: let us assume that the transversal T_1 is the best cover of minimal transversal and the provider maintains a dynamic global oriented graph for the composition of web services depicted in figure6. The aim now is to find a path or a subgraph for each consecutive services of $\overrightarrow{T_1}$:*

shortestPath(G, S_1, S_6), *shortestPath*(G, S_6, S_3), *shortestPath*(G, S_3, S_7), *shortestPath*(G, S_7, S_4). *We consider* S_8 *as atomic service. At the end of the calculus we obtain a global subgraph fulfilling the query by composing services not provided by Bcov. So the subgraph induced between* S_1 and S_6 *is a path defined by* $(S_1, \overrightarrow{S_2}, S_6)$ *(it is a linear composition). We will do the same process for the remainder of the pair of services. The obtained subgraph is depicted in Figure 7. Two services* S_2 *and* S_{10} *was introduced in the composition beside the services provided by Bcov algorithm. A sketch of the algorithm is given by Algorithm 2.*

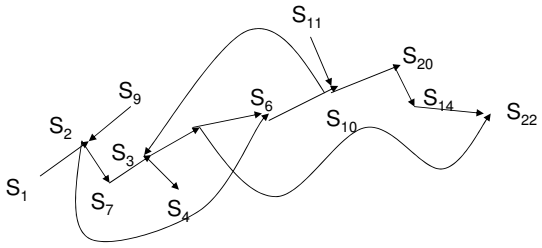

Figure 6: The global oriented graph.

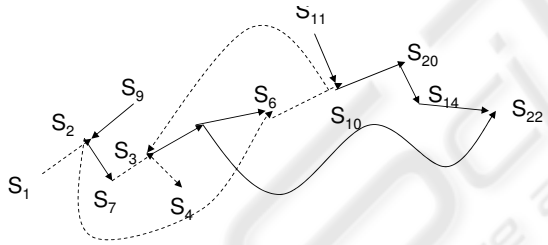

Figure 7: The composition fulfilling the query.

Algorithm 2 Shortest Composition Algorithm $\mathcal{SCA}$**.** (sketch)

Require: The minimal transversal provided by Bcov: T
The global oriented graph $\mathcal{GOG}$
Ensure: A subgraph meaning a possible composition including all services involved in the transversal minimal.
1: **for all** two consecutive services $S_i, S_{i+1} \in T$ **do**
2: Apply $shortestPath(\mathcal{GOG}, S_i, S_{i+1}))$
3: **end for**
4: Provide the global subgraph for all services induced in the composition of services fulfilling the query.

Complexity

The worst-case running time for the Dijkstra algorithm on a graph with n nodes and m edges is $O(n^2)$. Where $O(n^2)$ for node selection and $O(m)$ for distance updates. While is the best possible complexity for dense graphs, but can be significant improved upon for sparse graphs.

5 CONCLUSION

In this paper we have investigate the problem of the web services composition by providing two approaches based on graph theory, built upon Bcov algorithm initially provided for dynamic web services discovery. The first one uses all the minimal transversals provided by Bcov algorithm and tries to find the best one meaning a covering ratio and local oriented graph. However, the second one uses only the minimal transversal with the best cost and tries to find the shortestPath between all ordered services (regarding the query)in the global graph maintained by the web services provider. The result obtained are very interesting but still preliminaries. Taking only input/output into account as features to build a composition relation is not strong and sufficient. That's why our ongoing work is devoted to the extension of the proposed framework to hold the constraints on the inputs/outputs in the graph by means a cost in order to relax the composition.

REFERENCES

Benatallah.B, Hacid.M.S, Leger.A, Rey.C, and Toumani.F (2005a). On automating web services discovery. *VLDB Journal*, 14(1):84–96.

Benatallah.B, Hacid.M.S, Paik.H, Rey.C, and Toumani.F (2005b). Towards semantic -driven, flexible and scalable framework for peering and querying e-catalog communities. *Information Systems International Journal*.

Benbernou.S, Canaud, and Pimont.S (2004a). A semantic web services discovery regarded as constraint satisfaction problem. In *6th Flexible Query Answering Systems FQAS'04*.

Benbernou.S, Canaud.E, Hacid.M.S, and Toumani.F (2004b). Resolution and constraint propagation for semantic web services discovery. In *RIDE*, pages 23–30.

Berardi.D (2004). Automatic composition of finite state e-services. Technical report, American Association for Artificial Intelligence (AAAI).

Berardi.D, Calvanese.D, Giacomo.G, D., Lenzerini.M, and Mecella.M (2003a). Automatic composition of e-

services that export their behavior. In *ICSOC*, pages 43–58.

Berardi.D, Calvanese.D, Giacomo.G, D., Lenzerini.M, and Mecella.M (2003b). A foundational vision of e-services. In *WES*, pages 28–40.

Bordeaux.L, Salaün.G, Berardi.D, and Mecella.M (2004). When are two web services compatible? In *TES*, pages 15–28.

Bultan, T., Fu, X., Hull, R., and Su, J. (2003). Conversation specification: a new approach to design and analysis of e-service composition. In *WWW*, pages 403–410.

Doshi.P and Zhao.H (2006). A hierarchical framework for composing nested web processes. In *4th International Conference on Service Oriented Computing ICSOC'06*.

E.J.Dijkstra (1959). A note on two problems in connection with graphs. *Numerische Math*, 1:269–171.

Hamadi, R. and Benatallah, B. (2003). A petri net-based model for web service composition. In *ADC*, pages 191–200.

Hull, R. (2003). E-service composition: Models and formalisms. In *Description Logics*.

Hull, R., Benedikt, M., Christophides, V., and Su, J. (2003). E-services: a look behind the curtain. In *PODS*, pages 1–14.

John E. Hopcroft, R. Motwani and Jeffrey D. Ullman (2001.). *Introduction to automata theory, languages, and computation.* Addison-Wesley.

McIlraith, S. A. and Son, T. C. (2002). Adapting golog for composition of semantic web services. In *KR*, pages 482–496.

Medjahed, B., Bouguettaya, A., and Elmagarmid, A. K. (2003). Composing web services on the semantic web. *VLDB J.*, 12(4):333–351.

Merz, S. (2000). Model checking: A tutorial overview. In *MOVEP*, pages 3–38.

Narayanan, S. and McIlraith, S. A. (2002). Simulation, verification and automated composition of web services. In *WWW*, pages 77–88.

R. Reiter (2002.). *Knowledge in Action.* The MIT Press.

Rao, J. and Su, X. (2004). A survey of automated web service composition methods. In *SWSWPC*, pages 43–54.

S.Skiena (1990). "dijkstra's algorithm." 6.1.1. *Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*, pages 225–227.

Stuart J. Russell and P. Norvig (2003.). *Artificial intelligence : a modern approach.* Prentice Hall.

Sycara.K, Paolucci.M, Ankolekar.A, and Srinivasan.N (2003). Automated discovery, interaction and composition of semantic web services. *Journal of Web Semantics*, 1(1):27–46.

Tan.M, Nguyen.X, and Kowalczyk (2006). Modelling and solving qos composition problem using fuzzy discsp. In *2006 IEEE International Conference on Web Services ICWS 2006*.

Yuan.Y, Li.Z, and Sun.W (2006). Graph search based approach for BPEL4WS Test Generation. In *IEEE International Conference on Software Engineering Advances ICSEA'06*.