Abstract

Several researches have been proposed to formalize the knowledge preconditions problem; an action or a plan is epistemically feasible. However, since the feasibility is only checked at design-time and is assumed that it also will be carried out at run-time, it is not suitable in the context of Web services composition, where the transaction is important in the distributed environment. In this paper, we address the Interfering Agent Problem, which many of agents interferes the execution of Web services composition and define the transactionally executability of the Web services composition in order to guarantee the atomicity of Web services composition; i.e. an agent predicts the infeasible action before its execution, compose them together with the compensating ones. Besides, we use the \( \mathcal{TCA} \mathcal{L} \mathcal{C} \mathcal{F} \), which has proposed[16] for representing time, actions, and plans so that we can provide decidable, sound, and complete procedures for computing subsumption for Web services composition.

1. Introduction

Web services are a new paradigm for building distributed web applications. As the number of Web services available over the Internet have increased during the last few years, it has become more difficult to find the specific service that can perform a task at hand. It becomes even more difficult when there is no single service capable of performing that task, but there are combinations of existing services. If there are services with semantic descriptions, software agents could automatically perform the tasks without human intervention. In order to realize this vision, Semantic Web has recently emerged[24, 23]. Semantic Web languages, such as OWL, provide the foundations for describing the properties and capabilities of Web services in computer-interpretable form. In 2003, the OWL Services Coalition released the OWL-S 1.0[25], which consists of a set of OWL-based Web service ontologies. OWL-S markup will facilitate the automation of various kinds of tasks, including automated Web service discovery, composition, and execution.

In this paper, we focus on the issues related to the so-called Web services composition. Web services composition has traditionally been conceived as a planning problem in the field of Artificial Intelligence, which has been mainly investigated based on the first-order language and the situation calculus[21, 13, 26]. However, the formal specification of Semantic Web is mainly influenced by Description Logics[22, 11, 17], and the knowledge base of Semantic Web consists of a set of ontologies and individuals, which could be correspond to the terminologies and assertions of Description Logics, respectively. Besides, although Description Logic is a subset of the first-order logic, some variants of Description Logics are more practical than first-order logics because Description Logic could provide high expressivities with desirable computational properties such as decidability, soundness and completeness of deduction procedures[15, 19].

Based on the above advantages of Description Logics, Artale and Franconi have recently proposed a \( \mathcal{TCA} \mathcal{L} \mathcal{C} \mathcal{F} \)[16, 18] language, a class of interval-based temporal language for representing times, actions, and plans based on Description Logics. In this formalism, actions and plans are defined as occurring over time intervals according to the Allen’s proposal[8]. Because a propositionally complete Description Logic with both existential and universal temporal quantification is undecidable, they propose the \( \mathcal{TCA} \mathcal{L} \mathcal{C} \mathcal{F} \) with a limited universal quantification, which is decidable, and supplies sound and complete procedures for computing subsumption.

While Artale and Franconi proposed a class of formalism for uniformly representing time, actions and plans as explained above, they did not discuss the preconditions and effects of actions or plans. They are however important in the services composition as well as STRIPS-like planning systems[2, 4, 9, 14]. In particular, the problem of whether an agent is able to carry out an action or a plan has been addressed as that of whether the action or the plan is physically feasible and epistemically feasible[12].

In order to determine the feasibility, planning systems
check it only when they build and simulate plans, before executing them. However, since Web services are executed with other agents in distributed environments, the feasibility of Web services composition must be checked before as well as during execution, and if there occur partial executions, the transaction management such as compensation is necessary to preserve its atomicity.

In this paper, we address this problem occurring during the execution of Web services composition which is not important in planning, and discuss new definitions of feasibility about actions (Web services) and Web services compositions. For this purpose, we use the $\text{TL-ALCF}$ language as the basic representation of actions and Web services composition.

The remainder of this paper is organized as following. In section 2, we briefly explain the $\text{TL-ALCF}$ language and knowledge preconditions of actions and plans. In section 3, we present the knowledge and the feasibility of actions with respect to Description Logics. In section 4, we discuss the feasibility of Web services composition. In section 5, we compare our work with prior efforts for the related subjects. Finally, in section 6, we summarize this paper.

2. Background

2.1. $\text{TL-ALCF}$

Artale and Franconi propose a class of interval-based temporal Description Logic, $\text{TL-ALCF}$, which is decidable, and supplies sound and complete procedures for computing subsumption. $\text{TL-ALCF}$ is composed by the temporal logic $\text{TL}$, which is able to express temporally quantified terms and the non-temporal Description Logic $\text{ALCF}$[7, 10], which is the propositionally complete Description Logic extending $\text{ALC}$ with features (i.e., functional roles) In this framework, actions are represented through temporal constraints on world states, where each state is a collection of properties of the world holding at a certain time. Plans are built by temporally relating actions and states. Let us introduce the following Simple-Plane-Reservation action executing plane reservations:

Simple-Plane-Reservation $\equiv \circ(x y) (\sharp f x)(\sharp m y)$,

\((\ast \text{SEAT:Vacant})@x \cap (\ast \text{SEAT:Reserved})@y)\)

Fig. 1 shows the temporal dependencies of the intervals in which the concept Simple-Plane-Reservation holds. Simple-Plane-Reservation denotes any action occurring at some interval involving a $\ast \text{SEAT}$ that was once vacant and then reserved. Temporal variables are introduced by the temporal existential quantifier “$\circ$” excluding the special temporal variable $\sharp$, usually called now, and intended as the occurring time of the action type being defined. The temporal constraints $(\sharp f x)(\sharp m y)$ state that the interval denoted by $x$ should finish with the interval denoted by $y$, and that $\sharp$ should meet $y$. The parametric feature $\ast \text{SEAT}$ plays a role of the formal parameter of the action, mapping any individual action of type Plane-Reservation, independently from time. In order to evaluate a concept at an interval, the evaluation of concept at the interval $(\ast \text{SEAT:Vacant})@x$ and $(\ast \text{SEAT:Reserved})@y$ state that $\ast \text{SEAT:Vacant}$ is qualified at $x$ and $\ast \text{SEAT:Reserved}$ is qualified at $y$.

2.2. Knowledge Preconditions

The problem of whether an agent is able to carry out an action or a plan can be considered as two issues: physically feasible, which is physically possible to carry out the action or the plan, and epistemically feasible, which is whether an agent knows enough to perform the action or the plan. In particular, that of characterizing the epistemically feasibility of actions or plans called the knowledge preconditions problem, and many researchers have studied a way to formalize this problem as follows.

Hintikka[1] formalized the knowledge, agent A knows what something is, and he represented it by using a modal theory as follows: $\exists x(\text{know}(A, X = B))$. Subsequently, Moore[3] proposed a formalism of the knowledge and action by adopting a possible worlds theory based on Hintikka’s proposal, and Davis extended Moore’s theory for considering the knowledge preconditions problem of the determinate and indeterminate plans. Morgenstern[6] also modified Moore’s theory by using a syntactic theory and extended it to apply to plans involving multiple agents and the communications between them. In the following definitions, one is Moore’s definition for an epistemically feasible action and other is Davis’s definition for an executable plan

- An action E is epistemically feasible for agent A at time T if and only if A knows at T a specific behavior that constitutes that action in T

1) Davis use the notion of the “executability” instead of the epistemically feasibility in order to formalize the determinate and indeterminate plans
A plan \( P \) is executable for agent A at time T if and only if,

1. \( P \) terminates when executed starting in T; and
2. After any beginning of the execution of \( P \) starting in T,
   (a) A will know whether \( P \) has successfully finished;
   (b) A will know of every action whether or not it is a next step of \( P \); and
   (c) All the next steps of \( P \) are feasible

3. Actions

In this section, we use the \( TL^* - \text{ALCF} \) to represent the actions with a temporal theory and our formalisms are defined and extended by adopting Moore’s solution.

3.1. A Description Logic Representation of Knowledge

Hintikka represented the knowledge of an agent, A knows what B is, as the notion of \( \exists x (\text{know}(A, X = B)) \) with propositional attitude so that it can be easily transformed into Description Logic \( \text{ALC} \), the subset of \( \text{ALCF} \).

Definition 1 Using Description Logic \( \text{ALC} \), the knowledge precondition of “A knows what B is” is expressed as follows:

\[
\text{Precondition} = \forall \text{know}.A \cap \exists \text{what}.B
\]

Proof. A sentence of “A knows what B is” could be divided into two sentences; “All A knows something” and “B is something(what)”. They are expressed as following first-order logic:

\[
\text{Precondition}(x) = \forall z. (\text{know}(x, z) \rightarrow A(x)) \cap \exists y. (\text{what}(x, y) \cap B(y)), \text{and this formula is equivalent to that of Definition 1}.
\]

3.2. Feasibility of Actions

While Artale and Franconi proposed a formalism for representing times, actions, and plans, they did not clearly state the precondition, what one knows at the beginning, and the effect, how the world state and the knowledge of the agent change as a result of the execution, of actions and plans.

However, they are more important in the context of Web services composition than the general planning system, since actions constituting a Web services composition are not executed as an atomic unit respectively, but the execution of the actions are influenced by their precondition and effect which related to each other; the effect of an action previously executed should be the prerequisite of the following action.

In fact, even if Artale and Franconi did not clearly state the precondition and effect of an action, since the action is defined as occurring over time intervals, its precondition and effect could also be represented as the set of world states before and after the action is executed.

Prior to stating the Web services composition, we need to clarify the definitions for the precondition and the effect of an action as follows. In particular, suppose that an action is executed over finite time interval \( \sharp = [T_1, T_2] \) as an atomic unit during its execution.

Definition 2 The precondition of an action \( E \) which an agent executes over a time interval \( [T_1, T_2] \) is represented as the world state of which an agent knows at time \( T_1 \).

Definition 3 The effect of an action \( E \) which an agent executes over a time interval \( [T_1, T_2] \) is represented as the world state of which are changed as the result of the execution of the action at time \( T_2 \).

Fig. 2 shows the concept Plane-Reservation which extends the Simple-Plane-Reservation as shown in Sect. 2.1. The concept denotes any action adding the states of \( \star \text{CARD} \) that was once valid and then charged to the Simple-Plane-Reservation. Thus, in this example, the precondition of the action is states of which there is a vacant seat and the credit card is valid, and the effect is that of which a seat is reserved and the credit card is charged after the execution.

**Figure 2. Temporal dependencies of the intervals in which the concept Plane-Reservation holds**

The following is the definition for an epistemically feasible action. This is similar to Moore’s definition for actions as shown in Sect. 2.2, but there are two differences with ours specialized in the context of Web services compositions;
First, an action is executed within a finite interval. Second, the components constituting an action are rephrased as the precondition and effect of an action.

**Definition 4** An action $E$ executing over a time interval $[T_1, T_2]$ is epistemically feasible for agent $A$ at time $T_1$ if and only if $A$ knows the precondition and effect of $WS$ at time $T_1$.

While many researchers have studied the knowledge preconditions problem for the epistemically feasibility, they did not consider the physical feasibility – briefly introduced in Sect. 2.2 – since it is a relatively simple problem deciding whether an action could be executed physically. However, in this paper, it is necessary to clarify the physical feasibility in order to represent the completion of actions regardless of their success or failure as the following:

**Definition 5** An action is called “success” if the effect of an action $\sqsubseteq$ the actual result of executing an action, and “failure” \footnote{The failure is not a logically impossible action (always feasible action), but an error state as a result of executing an infeasible action.} if the action is not succeeded.

**Definition 6** An action $E$ executing over time interval $[T_1, T_2]$ is physically feasible for agent $A$ at time $T_1$ if and only if $A$ can execute the action physically regardless of its success or failure over time interval $[T_1, T_2]$.

### 4. Web services compositions

#### 4.1. Interfering Agent Problem

While there exist several common features between planning and Web services composition – their semantics are based on Description Logics and they try to find how to compose and execute the actions efficiently – there also exists a difference according to their domains as follows:

Planning systems only check the epistemically and physical feasibility at design-time when plans are built and simulated, and assume that the designed plans will be executed at run-time successfully. In fact, this assumption could be very natural in general planning systems. However, Web services are mainly available in a distributed environment on the Internet, and many agents share limited resources to perform the services. For this reason, even if compositions of Web services are feasible at design-time, they might not be feasible at run-time, since other agents could change the preconditions with respect to each web service during their executions. Therefore, an agent should predict actions which might not be feasible before the execution, and compose them together with their compensating actions, respectively. In particular, this problem is very difficult to resolve perfectly because agents have to predict the future. Therefore, in this paper, we propose a way to preserve the atomicity of Web services compositions as a solution to this problem.

#### 4.2. Feasibility of Web services compositions

Davis’s definition for feasible plans did not consider the run-time environments which world states could be changed by other agents. Thus, we propose a definition for preserving atomicity of Web services composition, extending Davis’s definition of the executable plan as shown Sect. 2.2.

**Definition 7** A Composition of Web services $C$ executing over a time interval $[T_1, T_2]$ is finite, possible empty sequence $a_1, \ldots, a_n$, of atomic actions, denoting the composition of the partial functions, where each actions are related to each other.

**Definition 8** A Composition of Web services $C$ is transactionally executable for agent $A$ at time $T_1$ if and only if when $A$ executes $C$ over $[T_1, T_2]$, $A$ knows at time $T_1$ that,

1. $C$ terminates at time $T_2$ after executed starting at time $T_1$; and
2. After any beginning of the execution of $C$ starting at time $T_1$,
   
   (a) $A$ will know whether $C$ has successfully finished;
   
   (b) $A$ will know of every action whether or not it is a next step of $C$; and
   
   (c) All the next steps of $C$ are physically and epistemically feasible;
   
   (d) $A$ will know of the compensating actions which is associated with each completed action when an action fails; and
   
   (e) All the compensating actions are physically and epistemically feasible

In order to execute the composition of Web services as an atomic unit, an agent should know of the compensating actions with respect to each action before the execution and if an action is not feasible during its execution, the agent should execute its compensating action instead. While this extension is to preserve the atomic property of Web services composition in the distributed environments, if compositions of Web services could be executed alone without any interference of other agents, the transactionally executable Web services composition is equivalent to the executable plans.
4.3. An Example of Transactionally Feasible Web services composition

In this section, we illustrate the transactional feasibility of Web services composition through an “Airline Reservation” example. In particular, we adopt the backward/forward recovery mechanism of SAGAS\[5\] for long-lived transaction.

Let us introduce the Reservation-Plane-Seats WSC which an agent reserves a number of reservations for flights and C-Plane-Reservation\(^3\), a compensating action, which can reserve the seat of another plane when the Plane-Reservation action fails as shown below. Suppose that the agent can reserve a seat at once to assign the individual position of seats.

\[
\text{Reservation-Plane-Seats} = \Diamond (\text{iji'}) (\text{ij} \text{b} \text{i'} \text{j'}) (\text{ij} \text{b} \text{ji'}) (\text{i'} \text{b} \text{j}). \tag{1}
\]

\[
\text{Plane-Reservation}(\text{i}) \sqcap \text{C-Plane-Reservation}(\text{j}) \sqcap \text{Plane-Reservation}(\text{i'}) \sqcap \text{C-Plane-Reservation}(\text{j'})
\]

Figure 3. Temporal dependencies of the intervals in which the concept Reservation-Plane-Seats holds

\[
\text{C-Plane-Reservation} = \Diamond (\text{lmnpq}) (\text{lm} \text{b} \text{♯}) (\text{♯} \text{f} \text{m}) (\text{♯} \text{m} \text{n}) (\text{♯} \text{f} \text{p}) (\text{♯} \text{m} \text{q}) \tag{2}
\]

\[
\text{Plane-Reservation}(\text{i}) \sqcap \text{Vacant}(\text{i'}) \sqcap \text{Reserved}(\text{i'}) \sqcap \text{Valid}(\text{p}) \sqcap \text{Charged}(\text{q})
\]

Fig. 4 shows the temporal dependencies of the intervals in which the concept C-Plane-Reservation holds

Fig. 3\(^4\) shows the temporal dependencies of the intervals in which the concept Reservation-Plane-Seats holds. This concept consists of the sequence of Plane-Reservation as shown Sect. 2.1, and C-Plane-Reservation. If the effect of Plane-Reservation is “Failed”, i.e. not physically feasible, then C-Plane-Reservation becomes feasible and it will be executed. Conversely, if not “Failed”, i.e. physically feasible, then it is not feasible and will not be executed.

Algorithm : Reserve-Plane-Seats (N reservations)

if Reserve-Plane-Seats is feasible then
while there remains to be reserved do
/* 1 ~ N reservations */
if Plane-Reservation(i-th SEAT, CARD) is feasible then /* 0 <= i < N */
Execute the Plane-Reservation(i-th SEAT, CARD);
else if Plane-Reservation(i-th SEAT, CARD) is not feasible then
C-Plane-Reservation(i-th SEAT, CARD)
i = i+1;
end if
end if
end if

SubAlgorithm : C-Plane-Reservation (i-th reservation)

Rollback the Plane-Reservation(i-th SEAT, CARD);
j = i;
while (there does not exist a save-point || there remains completed reservations) do
/* 1 ~ (i-1)-th reservations */
if not (there exists a save-point || first reservation) then
j = j-1;
Cancel Plane-Reservation(j-th SEAT, CARD);
end if
end while
j < i do
j = j+1; Plane-Reservation(j-th SEAT', CARD);
end
return;

The above algorithm states how to reserve reservations using the action of the concept Reservation-Plane-Seats. If an agent reserves the N-th seat and cannot reserve the (N+1)-th seat since other agents have reserved all of the vacant seats, the agent can not finish the Web services composition. Instead, the agent can reserve a seat on another plane by performing the C-Plane-Reservation, the compensating action of the Plane-Reservation. Fig. 5 shows the temporal dependencies of these two concepts.

In the backward/forward recovery, if an action fails, the agent has to rollback the failed action and perform compensating actions which is associated with each action going back to the save-point. However, in this example, we consider a case where the agent takes a save-point whenever an action is executed; the rollback operation is performed over the time interval of the “Failed” and the agent directly start the forward recovery without backward recovery.

5. Related Work

Recently, several approaches for automatic Web services composition have been proposed. They are closely similar
to traditional planning system and based on logical formalism in AI literature.

McIlraith and Son[21] proposed an approach on adapting Golog, which is a logical programming language built on top of situation calculus. They also defined the notion of knowledge and physically self-sufficient programs, which is very similar to Davis’s work. However, their formalism is not based on Description Logics, and the atomicity of Web services composition was not considered.

Wu et al.[26] proposed the SHOP2 planning system which translates the DAML-S process into the HTN planning task. While they can directly design and execute Web services composition, the procedures for computing subsumption for Web services composition are undecidable.

The work of Morgenstern has many similarities to ours. Morgenstern modified the theory of Moore and Davis by using a syntactic theory and extended it to apply to plans involving multiple agents and the communications between them. However, while he enables to finish the actions which an agent cannot execute alone by delegating to other agents, our work is different from his work in that we propose a way to preserve the atomicity of Web services composition although it fails.

6. Conclusions and Future Work

In this paper, we addressed the Interfering Agent Problem which could occur during the execution of Web services composition, and defined the transactionally executable of Web services composition. This makes us guarantee that Web services composition is executed as an atomic unit. We also showed the algorithm for the transaction of Web services composition adapting the backward/forward algorithm of SAGAS. As the basic representation language of actions and Web services composition, we bring the $TL-ALCF$, which is decidable, and supplies sound and complete procedures for computing subsumption. Consequently, we could take the advantages of $TL-ALCF$ for Web services composition.

However, we think that some more researches are needed and we leave some considerations as future work.

1. Since the formalism of our proposal is not sufficient, we will complement it according to the logic-based approach.

2. We will consider a variety of cases where Web services composition is executed and generalize our idea based on these cases. For instance, in some scenarios of Web service transaction[27], it is possible that Web services composition is executed a non-atomic unit. While it is not clearly feasible, omniscient agents have to be able to distinguish them.

Acknowledgement

This work was supported in part by the Ministry of Information & Communications, Korea, under the Information Technology Research Center (ITRC) Support Program

References


of the Association for Computing Machinery Special
Interest Group on Management of Data (ACM SIG-

York University (1987)

for Concept Languages. Technical Research Report
RR-90-04, DFKI, Germany (1990)

F. Allen, H. A. Kautz, R. N. Pelavin and J. D. Tennen-
berg, editors, Reasoning about Plans, Chap. 1, pp.2–
68, Morgan Kaufmann (1991)

with Constraint Networks and an Application to Plan
Recognition. In Proc. of the 3rd International Confer-
ence on Principles and Knowledge Representation and

(1992)

ily. In F. W. Lehmann, editor, Semantic Networks in
Artificial Intelligence, pp.133–178, Pergamon Press
(1992)


Situation Calculus. Annals of Mathematics and Artifi-
cial Intelligence, 14(2-4):251–268 (1995)

[14] P. T. Devanbu and D. J. Litman. Taxonomic Plan Rea-

Reasoning in description logics. In G. Brewka, editor,
Principles of Knowledge Representation, Studies in
Logic, Language and Information, pp.193–238, CSLI
Publications (1996)

Logic for Reasoning about Actions and Plans, Journal

of the 6th International Conference on Logic for
Programming and Automated Reasoning (LPAR’99),

[18] A. Artale and E. Franconi. A Survey of Temporal Ex-
tensions of Description Logics. Annals of Mathematics
and Artificial Intelligence (AMAI), 30(1-4):171–

ini. Reasoning in Expressive Description Logics. In A.
Robinson and A. Voronkov, editors, Handbook of
Automated Reasoning, pp.1581–1634, Elsevier Sci-
ence Publishers (2001)

[20] F. Baader, I. Horrocks, and U. Sattler. Description log-
ics for the semantic web. KI - Künstliche Intelligenz,

[21] S. McIlraith and T. C. Son. Adapting Golog for Com-
position of Semantic Web Services. In Proc. of the 8th
International Conference on Principles and Knowl-
edge Representation and Reasoning (KR’02), pp.482–
496 (2002)

[22] D. Nardi, R. J. Brachman. An Introduction to De-
scription Logics. In the Description Logic Handbook,
edited by F. Baader, D. Calvanese, D. L. McGuinness,
D. Nardi, P. F. Patel-Schneider, pp.5–44, Cambridge
University Press (2002)

[23] F. Baader, I. Horrocks, and U. Sattler. Description log-
ics as ontology languages for the semantic web. In D.
Hutter and W. Stephan, editors, Festschrift in honor
of Jörg Siekmann, Lecture Notes in Artificial Intelli-

Web Services, IEEE Intelligent Systems, 18(1):90–93
(2003)

[25] The OWL Services Coalition. OWL-S: Semantic
services/ (2003)

Automatic Web Services Composition using SHOP2.
In Workshop on Planning for Web Services (2003)

[27] OASIS Technical Committee Draft. Web Ser-
vice Composite Application Framework. In