

doi:10.3969/j.issn.1673-4785.2009.06.004

## 语义 Web 服务组合中的服务建模及规划算法

万长林<sup>1,2</sup>, 陈立民<sup>1,2</sup>, 王竹晓<sup>1,2</sup>, 王文杰<sup>2</sup>, 史忠植<sup>1</sup>

(1. 中国科学院计算技术研究所 智能信息处理重点实验室, 北京 100190; 2. 中国科学院 研究生院, 北京 100049)

**摘要:** Web 服务的语义建模是指对 Web 服务的属性、功能和结构等进行语义描述, 使用户能够自动地定位、选择、使用和组合。本文以动作理论和描述逻辑为基础对服务的逻辑层信息(控制流和数据流)进行语义建模, 用带前提和结果的原子动作描述简单服务, 复杂动作描述组合服务的控制流, 服务间的依赖关系描述数据流, 并提出一个基本的 Web 服务自动组合框架。在该框架中, Web 服务自动组合被划分为逻辑层和实现层 2 部分, 于是服务的自动组合问题在逻辑上归结为一个动作规划问题, 在实现上归结为一个根据动作选择具体服务的 service 选择问题。最后给出一种新的自动组合规划算法, 该算法充分考虑了语义推理和智能规划的结合, 在对问题的描述能力和运行效率上做了较好的平衡。

**关键词:** 语义 Web; 动态描述逻辑; 自动服务组合

**中图分类号:** TP393 **文献标识码:** A **文章编号:** 1673-4785(2009)06-0490-07

## Semantic model and planning algorithm for Web service composition

WAN Chang-lin<sup>1,2</sup>, CHEN Li-min<sup>1,2</sup>, WANG Zhu-xiao<sup>1,2</sup>, WANG Wen-jie<sup>2</sup>, SHI Zhong-zhi<sup>1</sup>

(1. Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, CAS, Beijing 100190, China; 2. Graduate University of Chinese Academy of Sciences, Beijing 100049, China)

**Abstract:** This paper proposed a semantic model for Web service based on dynamic description logic (DDL), mainly concern about the logic/functional facets. In this model, a simple service was modelled as an atomic action with some preconditions and effects. The control flows of composite service were modelled by using complicated action, and the data flows were modelled as dependent relationship between atomic actions. A basic framework for automated Web service composition was introduced. In this framework, service composition was divided into two stages: logic planning stage and grounding stage. Therefore, a service composition problem was reduced to an action planning problem at planning stage and a service selection problem at grounding stage. At last, a new planning algorithm for automated composition was proposed. The algorithm achieves good balance between DDL reasoning and AI planning.

**Keywords:** semantic Web; dynamic description logic; onogy; automated service composition

万维网(Web)正从最初的一个单纯提供文本和图像的容器向提供服务发展<sup>[1]</sup>。Web 服务被定义为能够在万维网上被发布、定位和调用的自包含、自描述的模块化应用程序。Web 面向服务计算体系结构(SOA)是一种设计软件系统的合理方法,它在网络中发布可被发现的接口,通过这些接口同时为最终

用户和其他应用程序提供服务<sup>[2]</sup>。服务组合是 SOA 的重要特征,即将多个基本服务组合成一个新的复杂服务来提供单个基本服务所不具备的聚合的功能。早期的服务组合多为手工方式,服务提供者在设计服务的商务逻辑的同时选择具体的组件服务。但随着服务提供商(WSP)和服务数量持续增长,以及服务运行环境的动态性,服务组合的自动化变得极为关键。当前对自动服务组合的研究大概可归于两大类。一类基于工作流组合,代表性工作是 BEA 系统公司、IBM 公司和微软公司等提出的 BPEL4WS<sup>[3]</sup>。BPEL4WS 是一种 Web 组合服务的工

收稿日期:--

基金项目:国家自然科学基金资助项目(60775035, 60970088); 国家“863”计划资助项目(2007AA01Z132); 国家“973”计划资助项目(2003CB317004, 2007CB311004); 国家科技支撑计划资助项目(2006BAC08B06); 中国科学院研究生院院长基金资助项目(085101JM03)。

通信作者: 万长林. E-mail: wanc1@ics.ict.ac.cn.

业标准,它提供了一种对业务流程和交互协议正式规范的语言.另一类基于智能规划,代表性工作是斯坦福大学、马里兰大学和卡内基-梅隆大学等机构提出的 OWL-S<sup>[4]</sup>. OWL-S 采用基于描述逻辑的 OWL 语言对 Web 服务的属性、功能和结构进行形式化描述使其可被计算机理解.通过将 Web 服务描述与 PDDL<sup>[5]</sup>之间的转换来将 Web 服务组合问题转化为智能规划问题,于是就可以利用现有的多种规划器来实现 Web 服务组合.比如, Schuster<sup>[6]</sup>等人提出多态过程模型 PPM 将服务模式化为状态机来描述服务的可能状态及状态间的转换,就可以通过状态机的推理来实现. Keller 等<sup>[7]</sup>将 Web 服务的可实现性以及包含关系归结为谓词演算的定理证明问题. Narayanan 等<sup>[8]</sup>用 Petri 网模拟原子组件的复合过程,将服务组合归结为 Petri 网的可达性问题. Mcilraith 等<sup>[9]</sup>利用 Golog 刻画复合过程,应用现有 Golog 的规划算法和工具解决服务的组合问题.但以上这些方法都没有考虑领域本体在规划中的作用. Siren<sup>[10]</sup>认为层次任务网 HTN 规划中的任务分解和 OWL-S 中的过程分解十分相似,将 OWL-S 与 HTN 结合来解决服务组合问题,该方法在先验知识足够的条件下效率很高,但在实际应用中往往缺乏必要的先验知识. Gu 等<sup>[11]</sup>结合情景演算和描述逻辑来解决服务组合问题,具有很强的描述能力且避免了对先验知识的依赖,但该方法的计算复杂度较高.为提高效率, Hoffman 等<sup>[12]</sup>用简化的本体形式化语言刻画服务及状态,然后用前向规划方法解决服务组合问题.

本文首先从服务描述的层面,提出了基于动态描述逻辑 DDL 的 Web 服务模型,动态描述逻辑 DDL<sup>[13-14]</sup>将动作理论和描述逻辑有机地结合在一起,即能描述静态知识,也能描述动态知识,是描述逻辑的一种动态扩展.基于该 Web 服务模型研究了基于动态描述逻辑的语义 Web 服务自动组合,重点是基于 DDL 推理算法的 Web 服务组合规划算法.

## 1 语义 Web 服务建模

### 1.1 原子 Web 服务建模

在 OWL-S 中 Web 服务的功能、输入输出等信息的语义描述是通过在 OWL-S 本体的 ServiceProfile 概念中 hasPrecondition、hasResult、hasInput 和 hasOutput 等属性来实现的,在对 Web 服务语义建模时每个具体的服务都需要为其指定包括前提、结果、输入和输出等的语义描述.因为互联网上通常有多

个 Web 服务提供商提供具有相似功能的 Web 服务,不同的 Web 服务提供商提供的逻辑上功能相同的服务可能输入/输出接口不兼容或者服务质量不同.本文对 Web 服务的建模将逻辑功能和具体实现区分开来,即用动态描述逻辑中的一个动作来描述具有相同逻辑功能的一类 Web 服务,而不是对每个 Web 服务都描述其前提、结果、输入和输出等信息.对于任一原子 Web 服务,通过为其关联一个原子动作来进行语义建模.动态描述逻辑 DDL 中的原子动作可定义如下:

$$a(x_1, \dots, x_n) \equiv (P, E). \quad (1)$$

式中:  $a$  为原子动作名;  $P$  和  $E$  分别为动作的前提条件和执行结果,它们都用 DDL 中的公式 (Formula) 表示;  $x_1, \dots, x_n$  为在  $P$  和  $E$  中出现的所有个体名 (也可称之为动作定义式中的变量). DDL 中的公式由如下产生式定义:

$$\begin{aligned} \varphi, \psi ::= & C(u) \mid R(u, v) \mid = v \mid T(u, e) \mid \neg \varphi \mid \\ & \varphi \vee \psi \mid < \pi > \varphi. \end{aligned} \quad (2)$$

式中:  $\varphi, \psi$  为公式名,  $u, v$  为个体名,  $e$  为有型域的个体名,  $C$  为概念名,  $R$  为抽象角色,  $T$  为有型角色,  $\pi$  为动作. 且形如  $R(u, v)$ ,  $\neg R(u, v)$ ,  $T(u, v)$ ,  $\neg T(u, v)$ ,  $C(u)$ ,  $\neg C(u)$  的公式被称为简单公式 (primitive formula).

下面以 OWL-S 联盟基于 OWL-S 1.2 的图书购买原子服务 ExpressCongoBuy 为例说明上述的原子服务语义建模方法.对 ExpressCongoBuy 服务,将其关联到一个原子动作 ExpressBuyBook,该原子动作可简单描述如下:

```
define atomic action ExpressBuyBook
variables: ECB_BookISBN, ECB_SignInInfo,
          ECB_CreditCardNumber, ECB_AcctID, ECB_CreditCard, ECB_Output
preconditions: hasAcctID( ECB_SignInInfo, ECB_AcctID) & validity( ECB_CreditCard, Valid ) & creditNumber( ECB_CreditCard, ECB_CreditCardNumber ) & InStockBook( ECB_Book ) & hasISBN( ECB_Book, ECB_BookISBN )
effects: type( ECB_Output, OrderShippedAcknowledgement ) & shipment( ECB_Shipment ) & shippedTo( ECB_Shipment, ECB_AcctID ) & shippedBook( ECB_Shipment, ECB_Book )
```

## 1.2 组合 Web 服务建模

组合 Web 服务是多个 Web 服务的聚集,如将其类比于工作流,组合服务中成员服务触发的控制逻辑即为组合服务的控制流,成员服务之间的数据(消息)传递即为组合服务的数据流。

### 1.2.1 控制流语义建模

BPEL4WS 和 OWL-S 采用特定的结构化的构造子(活动)来描述一组基本服务执行的顺序,这些构造子说明了组合服务是如何通过将基本服务组合在一起而创建得来的. 用动态描述逻辑中的复杂动作对服务进行语义建模,通过描述逻辑中的动作构造子,由基本服务逐步构造成复杂服务,即通过 DDL 中的动作构造符“?”、“;”、“ $\cup$ ”以及“ $*$ ”由原子动作逐步构造成复杂动作. DDL 中的动作由如下产生式生成:

$$\begin{aligned} \pi, \pi' &\rightarrow a(x_1, \dots, x_n) \mid \\ \varphi? \mid \pi \cup \pi' \mid \pi; \pi' \mid \pi^*. \end{aligned} \quad (3)$$

式中: $a(x_1, \dots, x_n)$  是原子动作,  $\varphi$  是一个公式. 形如  $\varphi?$ ,  $\pi \cup \pi'$ ,  $\pi; \pi'$  以及  $\pi^*$  的动作分别为测试(test)、选择(choice)、顺序(sequence)及迭代(iteration)动作。

表 1 列出了 BPEL4WS、OWL-S 和 DDL 的控制流,并显示了它们之间的对应关系(其中  $\pi; \pi'$  为服务定义式,  $\varphi$  为条件式)。

表 1 BPEL4WS、OWL-S 和 DDL 控制流的简单比较

Table 1 A brief comparison of control-flows in BPEL4WS, OWL-S and DDL

BPEL4WS (Structured Activity)	OWL-S (Control Construct)	DDL (Action Construct)
Sequence	Sequence	$\pi; \pi'$
	Any-Order	$(\pi; \pi') \cup (\pi; \pi')$
Switch	If-Then-Else	$(\varphi?; \pi) \cup ((\neg \varphi)?; \pi')$
While	Repeat-While	$(\varphi?; \pi) *; (\neg \varphi)?$
	Repeat-Until	$\pi; ((\neg \varphi?; \pi) *; (\varphi)?$
Pick	Choice	$\pi \cup \pi'$
Flow	Split	
	Split + Join	

由表 1 可以看出 BPEL4WS 和 OWL-S 中的控制流除了 Flow、Split 和 Split + Join 都可以在 DDL 中找到等价描述. 相对于 BPEL4WS 和 OWL-S 侧重对组合服务中基本服务执行顺序的描述, DDL 侧重于对基本服务的动态内涵刻画, 以及组合服务内部组合的逻辑关系的描述. 对组合服务控制流的语义建模综合表 1 所述, 控制流做适当取舍, 在 DDL 的基

础上引入了 6 个控制流 (Sequence、Switch、Repeat-While、Repeat-Until、Flow 和 Pick) 的宏定义:

$$\begin{aligned} \text{Sequence}(\{s_1, \dots, s_n\}) &::= s_1; \dots; s_n, \\ \text{Switch}(\{\varphi_1, \dots, \varphi_n\}, \{s_1, \dots, s_n\}) &::= \\ &(\varphi_1?; s_1) \cup \dots \cup (\varphi_n?; s_n), \\ \text{Repeat-While}(\varphi, s) &::= (\varphi? s) *; (\neg \varphi)?, \\ \text{Repeat-Until}(\varphi, s) &::= s; ((\neg \varphi)?; s) *; \varphi?, (4) \\ \text{Pick}(\{s_1, \dots, s_n\}) &::= s_1 \cup \dots \cup s_n, \\ \text{Flow}(\{s_1, \dots, s_n\}) &::= s_1 \cap \dots \cap s_n. \end{aligned}$$

式中: $s, s_1, \dots, s_n$  为构成复杂动作的成员动作,  $\varphi, \varphi_1, \dots, \varphi_n$  为条件式, Flow 的构造符“ $\cap$ ”为本文在 DDL(SHOIN(D)) 中引入的动作构造子, 表示  $n$  个动作  $s_1, \dots, s_n$  可以同时发生, 它等同 OWL-S 中的 Split + Join, 而忽略不检查并发结果导致不确定逻辑结果的 Split 控制流. 本文不在 DDL 中为 Any-Order 引入宏定义是因为 Any-Order 的语义是它的成员服务. 可以以任意顺序执行, 也就是说, 如果一个 Any-Order 成立, 那么任意指定的一个有同样成员服务的 Sequence 控制流也成立, 且具有相同的逻辑结果. 最后, If-Then-Else 控制流是 Switch 控制流的一种特例, 它们可以简单地等价替换. 基于上述 DDL 复杂动作宏定义的服务可以被定义如下:

$$\begin{aligned} s \rightarrow t \mid \text{Sequence}(S) \mid \text{Switch}(\Psi, S) \mid \text{Pick}(S) \mid \text{Flow}(S) \mid, \\ \text{Repeat-While}(\varphi, s) \mid \text{Repeat-Until}(\varphi, s). \end{aligned} \quad (5)$$

式中: $s$  是服务的名称,  $t$  表示基本服务, Sequence、Repeat-While、Repeat-Until、Switch、Pick 和 Flow 是控制流的名称,  $S$  表示一组服务, 称为  $s$  的“成员服务”,  $\Psi$  表示一组条件式。

### 1.2.2 数据流语义建模

除了控制结构, OWL-S 使用绑定类来描述进程之间的数据流. 绑定被描述为使用其他进程数据的过程, 并使这种过程及数据来源的具体参数明确化. BPEL4WS 通过对活动之间的交互状态建模来处理数据流. 交互状态由接收和发送的消息及其他的相关数据组成. 但是, OWL-S 和 BPEL4WS 都不对复杂的数据流进行显式声明, 而复杂数据流是包含复杂控制流的组合服务必然将会导致的结果. 以图 1 所示的一个简单场景为例。

如图 1 所示, 任务 PayRental 的输入是任务 CarRental 或 BikeRental 的输出. 尽管 BPEL4WS 和 OWL-S 可以显式声明 CarRental 和 PayRental 的数据流以及 BikeRental 和 PayRental 之间的数据流, 但它们没有给出声明 2 个数据流之间的可选关系的方法. 为解决该问题, 在数据流的声明引入 AND, OR 2 个结

构进行扩展. 考虑一个包含  $n$  个基本服务任务的组合服务  $s$ , 可以定义服务  $s$  的数据流如下所示:

$$f \rightarrow \langle t_i, t_j \rangle \mid \text{AND}(F) \mid \text{OR}(F). \quad (6)$$

式中:  $1 \leq i, j \leq n$ ,  $f$  是数据流的名称,  $t_i$  和  $t_j$  是基本服务任务的名称,  $F$  表示一组数据流. 一个元组  $\langle t_i, t_j \rangle$ , 或简记为  $f_{ij}$ , 表示一个从起点服务任务  $t_i$  到目标服务任务  $t_j$  的简单数据流. AND、OR 是 2 个用来定义复杂数据流的结构. 例如,  $\text{AND}(\{f_{ik}, f_{jk}\})$  表示服务任务  $t_k$  应具有从  $t_i$  和  $t_j$  2 个服务任务的输入数据流;  $\text{OR}(\{f_{ik}, f_{jk}\})$  表示服务  $t_k$  应具有从服务任务  $t_i$  或  $t_j$  的输入数据流.

### 1.3 Web 服务动作建模本体

根据以上服务建模的定义, 提出了一个基于动态描述逻辑的语义 Web 服务动作建模本体 WSAMO (Web service action modeling ontology). 该本体包括 Web 服务的控制流定义、数据流定义和服务参与者所

需的概念和概念之间的关联, 如图 2 所示.

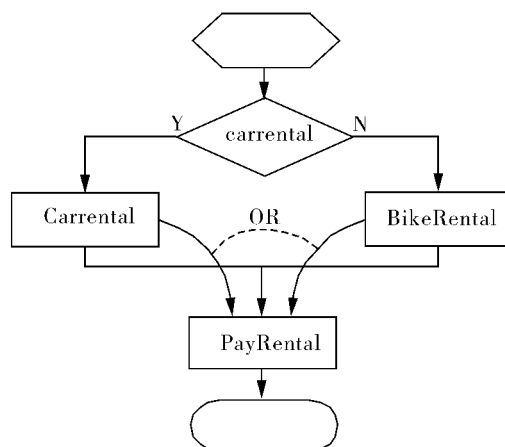


图 1 一个含有可选数据流依赖关系的组合服务

Fig. 1 A composite service including optional data-flow dependence relationship

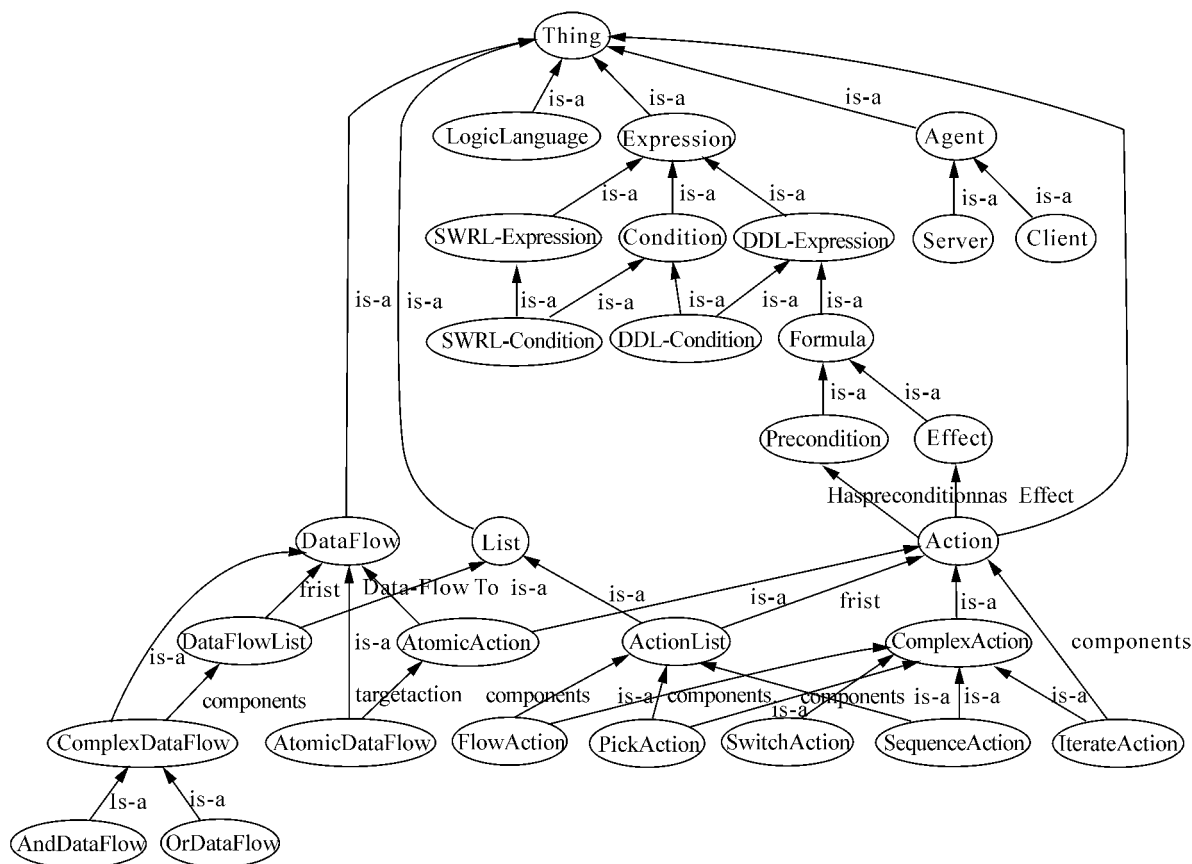


图 2 Web 服务建模本体概念和关联

Fig. 2 The concepts and roles of Web service action modeling ontology (WSAMO)

## 2 自动服务组合框架

通过给基本的 Web 服务添加前提和结果等语义信息, 可以将 Web 服务在逻辑上等价于一个原子动作, 而由基本服务自动组合成复杂服务的过程在

逻辑上就等价于构造并判断若干原子动作组合成的复杂动作在给定状态下是否成立. 一个基本的自动服务组合框架由规划器 (Planner)、匹配器 (Matcher)、选择器 (Selector) 和部署器 (Deployer) 等部分构成, 如图 3 所示.

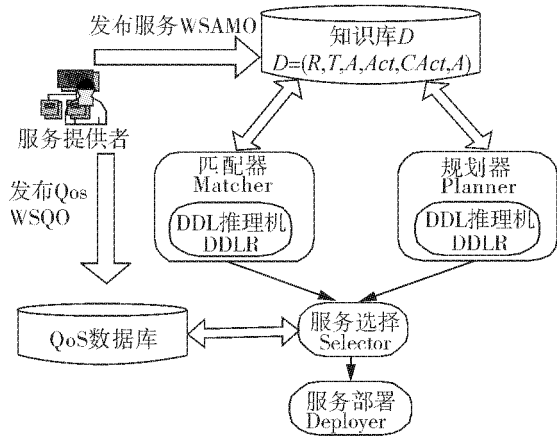


图3 基于DDL的自动服务组合框架

Fig. 3 An automated service composition framework based on dynamic description logic (DDL)

图3中的Web服务用WSAMO本体描述其功能性信息,并发布到DDL知识库中;匹配器负责在知识库中查找匹配用户需求的动作;规划器负责由基本动作构造复杂动作,并由DDL推理机判断复杂动作是否满足任务要求.如果复杂动作不可满足,则规划器继续尝试构造新的复杂动作,直至发现一个匹配用户需求的复杂动作.如果匹配器发现了匹配的动作,则由服务选择根据QoS数据集中的信息为该动作指定具体服务,并供客户调用;如果规划器发现了匹配的复杂动作,则由服务选择根据QoS数据集中的信息为复杂动作中的原子动作指定具体服务,然后将复杂动作在工作流引擎上部署为工作流服务供客户调用.

### 3 自动服务组合规划算法

本文的自动服务组合方法类似文献[12],主要区别在于本文的服务建模具有更严格的逻辑基础和更强的描述能力,且更多地结合了智能规划中的优化技术以提高算法效率,包括:1)通过将自动服务组合划分为逻辑和实现2层,避免了规划过程中对类似服务重复考虑的问题;2)在规划过程中定义了启发函数提高搜索效率.本文的自动服务组合方法首先将用户提交的需求和已有知识转化成一个智能规划问题域,通过解决这个逻辑上的规划问题得到一个满足用户需求的动作序列,然后从各个动作关联的服务中选取合适的服务,组成一个实际可执行的组合服务<sup>[15-16]</sup>.比自动组合规划算法就是解决在规划域中发现满足要求的动作序列的问题.该算法的核心是一个启发式搜索算法,启发规则为候选动作序列目标.已知一个DDL知识库 $D = (R, T, A_{Act}, C_{Act}, A)$ ,其中 $R$ 和 $T$ 通过角色构造符、概念构造符

等符号对静态知识进行了结构化的描述; $A_{Act}$ 和 $C_{Act}$ 表示的关于动作的知识,其中 $A_{Act}$ 为原子动作集合, $C_{Act}$ 为根据前述控制流宏定义的复杂动作的集合; $A$ 刻画的关于具体状态的知识,类似于情景演算中对情景的刻画.用户提交的初始输入信息为 $I = (R', T', A')$ 表达用户具有的静态知识和具体状态,用户的目标状态表达为一个DDL公式 $goal$ .根据知识库 $D$ 和用户输入 $I$ 及目标公式 $goal$ 可建立规划域 $P = (S_0, Acts, goal)$ ,其中初始状态集 $S_0 = (R \oplus R', T \oplus T', A \oplus A')$ 为2个静态知识库 $(R, T, A)$ 和 $(R', T', A')$ 的融合,动作集 $Acts = A_{Act} \cup C_{Act}$ .基于以上定义,自动服务组合规划算法可描述如图4.

#### Algorithm 1 ActionPlan( $S, Acts, goal, plan$ )

**Input:** the state  $S$ , the action list,  $Acts$ , the planning goal, and a plan that is a sequence action.

**Output:** a plan that meets the goal, or an empty plan.

```

1: create an empty plan list: plan s;
2: for each action  $a \in Acts$  do
3:   create a new plan: new  $\leftarrow plan + a$ ;
4:   if IsPlanExecutable ( new p ) then
5:     evaluate how close new p to the goal by using function
       Evaluate Plan ( new p, S, goal )  $\rightarrow d$ ;
6:     if  $d = 1$ . ie. new p meets the goal then
7:       return new p;
8:     else
9:       insert new p into the plan is decrease order of evaluation
         value d;
10:    end if
11:  end if
12: end for
13: if plan s is empty then
14:   return NULL;
15: else
16:   for each new p  $\in plans$  do
17:     return Actionplan ( S, Acts, goal, new p )
18:   end for
19: end if

```

#### (a)算法1:自动服务组合规划算法 ActionPlan

#### Algorithm 2 EvaluatePlan( $S, Acts, goal, plan$ )

**Input:** the state  $S$ , the planning goal  $goal$ , and a plan that is a sequence action

**Output:** a numeric value between 0 and 1.

```

1: max  $\leftarrow 1$ 
2: rewrite formula goal to a disjunctive normal form formula
    $G = \bigvee_{i=1}^k ( \bigwedge_{j=0}^m \varphi_j )$ ;
3: count  $\leftarrow 0$ ;
4: for each clause  $C = \bigwedge_{j=1}^m \varphi_j$  do
5:   for each primitive formula  $\varphi \in C$  do
6:     formula  $\psi \leftarrow \text{Conj}(S) \wedge \neg \langle plan \rangle \neg \varphi$ ;
7:     if formula  $\psi$  is satisfiable then
8:       count  $\leftarrow count + 1$ ;
9:     end if
10:  end for
11: if max < count/m then
12:   max  $\leftarrow count/m$ ;
13: end if
14: if max = 1 then
15:   return 1;
16: end if
17: end for
18: return max;

```

#### (b)算法2:组合评估算法 EvaluatePlan

#### 图4 自动服务组合规划算法

Fig. 4 The planning algorithm for DDL based automated service composition

算法 1 是自动服务组合规划算法,其中的 IsPlanExecutable 函数作用是判断一个动作序列是否可执行.假设给定一个动作序列  $newp = a_1; a_2; \dots; a_n$ ,  $newp$  在状态  $S$  下可执行的充分必要条件是如 DDL 公式(7):

$$[(a_1 \cup \dots \cup a_n) * ] \prod \wedge \text{Conj}(S) \wedge \neg \langle newp \rangle true \quad (7)$$

可满足,其中,  $\prod = \bigwedge_{i=1}^k (\text{Conj}(P_i) \rightarrow \langle a_i \rangle true)$ ,  $P_i$  是动作  $a_i$  的前提,  $\text{Conj}(S)$  表示由  $S$  中的各个公式作为合取项构成的合取式.算法 2 是组合评估算法 EvaluatePlan,其作用是评估一个动作序列使目标公式达成的程度.假设给定一个动作序列  $newp$  和一个目标公式  $g$ ,  $newp$  在状态  $S$  下执行后公式成立的充分必要条件是如 DDL 公式(8):

$$\text{Conj}(S) \wedge \langle newp \rangle \neg g \quad (8)$$

可满足. EvaluatePlan 算法的基本思想是把目标公式  $g$  规范化为析取范式  $G = \bigvee_{i=1}^k (\bigwedge_{j=1}^m \varphi_j)$ , 其中  $\varphi_j$  为简单公式.对每一个析取项  $\bigwedge_{j=1}^m \varphi_j$ , 如果有  $n$  个  $\varphi_j$  可满足,则该析取项的达成程度为  $n/m$ , 目标公式  $g$  的达成程度为所有析取项的最大达成程度.

#### 4 结束语

本文将动态描述逻辑 DDL 用于对 Web 服务的控制流和数据流建模.将原子服务描述为 DDL 中的原子动作,并将 Web 服务组合中的基本控制流描述为 DDL 的动作宏,这样的模型相对 BPEL4WS 和 OWL-S 更为简洁且同时又能满足逻辑层推理和规划的描述需要.另外,Web 服务中的数据流也被定义为 DDL 中包含与、或(AND, OR)算子的复杂公式,所以对数据流也具备更强的建模能力.本文还给出一个基本的自动服务组合框架,将组合过程划分为逻辑层和实现层 2 个层面.自动服务组合问题首先在逻辑层上做为一个智能规划问题来求解,得出满足用户需求的动作规划,避免了在规划过程中过多考虑服务细节的问题.然后根据动作规划在实现层选择合适的 Web 服务组合成新的服务.在 Web 服务的语义建模基础上,给出一种新的自动组合规划算法,该算法充分考虑了语义推理和智能规划的结合,在对问题的描述能力和运行效率上做了较好的平衡.

#### 参考文献:

[1] MCILRAITH S, ZENG T C H. Semantic Web services[J]. IEEE Internet Systems, 2001, 16(2): 46-53.

[2] PAPAOGLOU M. Service-oriented computing: concepts, Characteristics and directions[C]//Fourth International Conference on Web Information Systems Engineering, WISE2003. Roma, Italy, 2003:3-12.

[3] ANDREWS T, CURBERA F, DHOLAKIA H, et al. Business process execution language for Web services version 1.1[EB/OL]. [2007-02-12]. <http://www.ibm.com/developerworks/library/specification/ws-bpel/>.

[4] MARTIN D. OWL-S: semantic markup for web services [EB/OL]. [2004-08-26]. <http://www.daml.org/services/owl-s/1.1/overview/>.

[5] MCDERMOTT D. A temporal logic for reasoning about processes and plans[J]. Cognitive Science, 1982(6): 101-155.

[6] SCHUSTER H, GEORGAKOPOULOS D, CICHOCKI A, et al. Modeling and composing service-based and reference process-based multi-enterprise processes[J]. Advanced Information Systems Engineering, 2000, 1789:247-263.

[7] KELLER U, LAUSEN H, STOLLBERG M. On the semantics of functional descriptions of web services[J]. The Semantic Web: Research and Applications, 2006, 4011:605-619.

[8] NARAYANAN S, MCILRAITH S A. Simulation, verification and automated composition of web services[C]//The 11th International Conference on World Wide Web. 2002. Honolulu, Hawaii, USA, 2002: 77-88.

[9] MCILRAITH S, SON T C. Adapting golog for composition of semantic web services[C]//The 8th International Conference on Principles of Knowledge Representation and Reasoning (KR2002). Toulouse, France, 2002:482-493.

[10] SIRIN E. Combining description logic reasoning with AI planning for composition of web services[D]. College Park: University of Maryland, 2006.

[11] GU Y, SOUTCHANSKI M. Decidable reasoning in a modified situation calculus[C]//Proceedings of the International Joint Conference on Artificial Intelligence. Hyderabad, India, 2007:1891-1897.

[12] HOFFMANN J O, BERTOLI P, PISTORE M. Web service composition as planning, revisited: in between background theories and initial state uncertainty[C]//Proceedings of the National Conference on Artificial Intelligence. Vancouver, Canada: AAAI Press, 2007: 1013-1018.

[13] 史忠植, 董明楷, 蒋运承, 等. 语义 Web 的逻辑基础[J]. 中国科学:E 辑, 2004, 34(10):1123-1138.

SHI Zhongzhi, DONG Mingkai, JIANG Yuncheng, et al. The logical foundation for semantic Web[J]. Science in China Series E: Information Sciences, 2004, 34(10):

1123-1138.

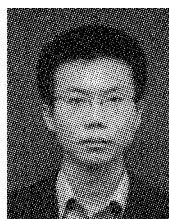
- [14] 史忠植, 常 亮. 基于动态描述逻辑的语义 Web 服务推理[J]. 计算机学报, 2008, 31(9): 1599-1611.

SHI Zhongzhi, CHANG Liang. Reasoning about semantic Web services with an approach based on dynamic description logics[J]. Chinese Journal of Computers, 2008, 31(9): 1599-1611.

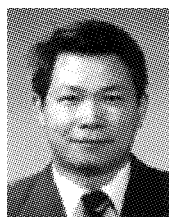
- [15] ZENG L, BENATALLAH B, DUMAS M, et al. Quality driven web services composition[C]//Proceedings of the 12th International Conference on World Wide Web. Budapest, Hungary: ACM. 2003:411-421.

- [16] WAN C, ULLRICH C, CHEN L, et al. On solving QoS-aware service selection problem with service composition[C]// The Seventh International Conference on Grid and Cooperative Computing Shenzhen, China, 2008:467-474.

#### 作者简介:



万长林,男,博士研究生。主要研究方向为人工智能,语义 Web 和组合优化。



史忠植,男,1941 年生,研究员,博士生导师,主要研究方向为智能科学、人工智能、多主体系统、数据挖掘、机器学习、知识工程等。1979 年、1998 年、2001 年均获中国科学院科技进步二等奖,1994 年获中国科学院科技进步特等奖,2002 年获国家

科技进步二等奖。发表学术论文 400 余篇,出版专著 5 部。

## 第 2 届集体智能计算国际会议

### The 2nd International Conference on Computational Collective Intelligence

Computational Collective Intelligence (CCI) is most often understood as an AI sub-field dealing with soft computing methods which enable making group decisions or processing knowledge among autonomous units acting in distributed environments. Web-based systems, social networks and multi-agent systems very often need these tools for working out consistent knowledge states, resolving conflicts and making decisions. The aim of this conference is to provide an internationally respected forum for scientific research in the computer-based methods of collective intelligence and their applications in (but not limited to) such fields as semantic web, social networks and multi-agent systems.

SEMANTIC WEB: semantic annotation of web data resources; Web Services (service description, discovery, composition); ontology management (mediation and reconciliation, creation, evaluation, merging, alignment, evolution, linking); automatic metadata generation; (semi-) automatic ontology creation; semantic web inference schemes; reasoning in the semantic web; knowledge portals; information discovery and retrieval in semantic web; etc.

SOCIAL NETWORKS: computational technologies in social networks creation and support; advanced groupware and social networks; models for social network emergence and growth; ontology development in social networks; advanced analysis for social networks dynamics; social networks and semantic communication.

MULTIAGENT SYSTEMS: cooperative distributed problem solving. Task and resource allocation. Mechanism design, auctions, and game theory. Modeling other agents and self. Multi-agent planning. Negotiation protocols. Multi-agent learning. Conflict resolution. Trust and reputation management. Privacy, safety and security. Scalability, robustness and dependability. Social and organizational structures. Verification and validation. Novel computing paradigms (autonomic, grid, P2P, ubiquitous computing). Brokering and matchmaking. Agent-oriented software engineering, including implementation languages and frameworks. Mobile agents. Performance, scalability, robustness, and dependability. Verification and validation. E-business agents. Pervasive computing. Privacy, safety, and security.