语义 Web 服务自动组合定义、方法及验证调查*

王向辉[1,2]，冯志勇[2]
(1. 山东建筑大学 计算机学院，济南 250101；2. 天津大学 计算机学院，天津 300072)

摘　要：针对近几年在语义 Web 服务自动组合上的研究成果，深入归纳和总结了语义 Web 服务组合的定义、组合方法以及正确性验证技术，对语义 Web 服务组合问题进行分类和形式化定义，并归纳出一个语义 Web 服务自动组合框架。按照不同问题分析和比较相关的组合方法，同时总结了组合方案正确性验证技术。最后，根据前面的分析和总结，提出了语义 Web 服务自动组合当前面临的问题和挑战。

关键词：语义 Web 服务；形式化定义；组合框架；组合方法；正确性检验
中图分类号：TP301　　**文献标志码**：A　　**文章编号**：1001-3695(2014)05-1292-10
doi:10.3969/j.issn.1001-3695.2014.05.003

Survey on definition, methods and verification of semantic Web service automatic composition

WANG Xiang-hui[1,2], FENG Zhi-yong[2]
(1. *School of Computer Science & Technology*, *Shandong Jianzhu University*, *Jinan* 250101, *China*; 2. *School of Computer Science & Technology*, *Tianjin University*, *Tianjin* 300072, *China*)

Abstract: There are many research findings in semantic Web service composition area recently. This paper summarized and concluded definitions, composition methods and verification technologies in this area. It classified composition problems and gave their formal definitions respectively. And it summarized an automated semantic Web service composition framework. Then, it analyzed and compared composition methods according to different problems, and summarized verification technologies for one composition solution. At last, on the basis of analysis above, this paper proposed current problems and challenges in front of automated semantic Web service composition researcher.

Key words: semantic Web service; formalization; composition framework; composition methods; verification technology

0　引言

Web 服务是由服务提供商发布到互联网上的可编程模块，由于 Web 服务的接口描述(WSDL[1])同服务的实现细节完全解耦，并通过标准通信协议(SOAP，HTTP)进行存取，因此跨语言、跨平台实现的 Web 服务间进行互操作是很方便的。Web 服务也因此成为实现 SOA[2] 架构的首选，广泛应用于企业应用集成、B2B 集成和服务计算中[3]。

目前，互联网上有相当数量的可用 Web 服务，这些服务来自不同的服务提供商，涉及若干不同的领域。从数量庞大的 Web 服务中找到满足用户需要的服务，一直是实现 Web 服务真正价值的关键问题[3]。如果无法找到单个满足需要的服务，就需要找到一组相关服务，并将它们组合起来形成一个满足用户需要的复合服务，这样的过程就是 Web 服务组合(Web service composition，WSC)。人们希望在组合过程中尽量减少人工的干涉，因此，关于 WSC 方面的研究主要集中于如何自动地进行 WSC。语义 Web 的出现为 WSC 的自动化提供了基础[4]，目前出现了一些优秀的语义 Web 服务本体，如 OWL-S[5]、WSMO[6] 和 SAWSDL[7] 等，它们为基于语义的 WSC 实现提供了丰富的语义信息。这些语义信息对于 Web 服务间可组合性判断起着重要的作用，而这种判断又是实现 WSC 的基础。学术界通常将基于语义信息的 WSC 称为语义 WSC。

当前关于语义 WSC 的研究有很多，出现了一些优秀的研究综述[8~12]，它们从不同角度对语义 WSC 的研究进行了总结。但这些综述中有的发表时间较早没有包括最近的研究成果[9~11]，有的只是语义 WSC 研究的一个侧面[9,12]。本文在这些已有综述的基础上，融入该领域的最新研究技术，尝试从一个全新的视角展现语义 WSC 的研究现状。根据对 Web 服务的不同描述，本文将当前的研究方法分为四类，即基于 I/O、基于 IOPE、QoS 感知和其他的方法。其中，前三类方法受到的关注较多，也是本文描述的重点；同时本文也将对基于功能语义、用户约束等的方法进行简单的描述。

为了明确研究问题，本文在总结相关研究的基础上给出了各类问题的形式化定义；同时提出语义 WSC 系统的一般框架，以帮助研究者了解系统的组成，能够从整体上把握每个研究点在整个系统中的作用和位置。对每一类问题，主要按照使用技术的不同进行分类和比较，同时介绍可用的测试集及其特点。组合方案正确性的验证对于其执行有着很重要的作用，因此，本文介绍了当前所采用的语义 Web 服务组合验证方法和技术。此外，在语义 WSC 中的研究中主要用到了两类技术：工作

收稿日期：2013-09-26；**修回日期**：2013-11-08　**基金项目**：山东省高等学校优秀青年教师国内访问学者资助项目(2013)；国家自然科学基金资助项目(61173155，61070202)；山东省软件工程重点实验室开放项目(2011SE001)

作者简介：王向辉(1979-)，女，山东昌邑人，讲师，博士研究生，主要研究方向为服务计算、工作流技术(wxh_225@163.com)；冯志勇(1965-)，男，博导，教授，主要研究方向为软件体系结构、知识工程、语义网、多 agent 系统.

流技术和 AI 技术[8,9],本文只介绍使用 AI 技术的相关方法,尤其是 AI 规划技术。

1　语义 WSC 问题定义

WSC 问题本质上是 Web 服务发现问题,即在进行服务发现时,如果在已知的 Web 服务集合中无法找到满足用户需求的单个服务,如何找到多个相关的 Web 服务,并将它们有效地集成在一起形成满足用户需求的组合服务的过程。这里,已知的 Web 服务集合也被称为 WSC 的问题域。实际中,一个 Web 服务由若干操作组成,每个操作描述了 Web 服务所实现的一个功能,而每个操作主要由输入参数集(inputs)和输出参数集(outputs)来描述[1]。在学术界,为了研究上的方便,通常假设问题域中的每个 Web 服务都是只有一个操作的服务,相应操作的输入和输出参数也称为 Web 服务的输入和输出参数(简称 I/O)。这样,Web 服务的组合问题本质上是操作之间的组合问题。在后面的描述中若无特别说明,均遵守这个假设。

I/O 参数仅描述了 Web 服务功能的一个方面,完整的功能描述还包括 Web 服务执行的前提条件(preconditions,P)和执行后的效果(effects,E),其中 P 和 E 通常使用基于 I/O 参数的逻辑表达语言表示(如在 OWL-S 中使用 SWRL 表示)[5]。实际中,根据所关注的描述元素不同,Web 服务可大致分为信息提供类和状态改变类两类。前者只有 I/O 参数描述,如天气预报服务,其中 I 为城市和日期,O 为天气的相关信息;后者则不仅包括 I/O 参数,还包括前提条件和效果,这四部分描述元素简称为 IOPE,如付款服务,其中 I 为信用卡号、金额,O 为付款状态(成功或失败),P 为信用卡号对应的信用卡是有效的,E 为信用卡金额发生变化,即世界状态发生了改变。当然从调用 Web 服务的角度看,除了知道其功能属性 IOPE 外,其他一些非功能属性也是很重要的,如服务的吞吐量、响应时间、可用性以及执行代价等。这些非功能属性决定着 Web 服务的质量,简称为 Web 服务的 QoS。传统的 WSC 问题中,两个 Web 服务(仅有 I/O)是否是可组合的主要取决于前面服务的输出参数;是否与后面服务的输入参数在语法上匹配,若匹配,则可组合,否则不可组合。而实际中参与组合的两个服务可能来自不同的供应商,他们对于同一个 I/O 概念可能会用不同概念表达,这种情况用传统的 WSC 方法就无法进行组合,显然这种限于语法上的匹配不利于解决 WSC 问题。语义 Web 的出现克服了这个缺陷,通过对 Web 服务中的每个概念使用同一本体标注语义的方式,可以对两个概念进行语义级的匹配,然后在本体推理和 AI 技术的帮助下实现高度自动化的 WSC[4]。学术界通常将使用了语义匹配技术的 WSC 称为语义 WSC。

近年来,在语义 WSC 的研究领域出现了很多较成熟的解决方案。按照解决方案对问题域中 Web 服务的不同描述将 WSC 问题分为三类:基于 I/O 的语义 WSC、基于 IOPE 的语义 WSC 以及 QoS 感知的语义 WSC。在这三类语义 WSC 定义中,都会用到概念间的语义匹配问题,因此,在分别描述这三类语义 WSC 的定义之前,先给出语义匹配的相关概念的形式化定义。

定义 1　语义匹配关系。令 C 为概念集合,$x,y\in C$,RC 为语义匹配函数,若 $y=RC(x)$,则说明在函数 RC 的作用下,概念 y 的实例可用概念 x 的实例代替,即 x 在语义上匹配 y,或 y 在语义上被 x 匹配,表示为序偶 $\langle x,y\rangle\in RC$。

在后面的描述中,通常将 RC 看做是语义匹配关系的集合。这里 RC 对应于某种概念匹配的方法,如利用本体描述中的层次关系计算两个概念之间的语义距离,然后根据设定的阈值判断两个概念在语义上是否是匹配的[13]。

定义 2　语义满足关系 $\subseteq_{rc}$。令 C 为概念集合,RC 为 C 中概念间的语义匹配关系集合,R 和 S 是 C 的子集,则 $R\subseteq_{rc} S=(\forall x)(x\in R\rightarrow(\exists y)y\in S\wedge\langle x,y\rangle\in RC)$。

1.1　基于 I/O 的语义 WSC 问题

I/O 参数是描述 Web 服务功能的核心元素,不同 Web 服务间的 I/O 参数匹配对于 WSC 起着关键作用。在这类问题的研究中,仅有 I/O 参数描述的若干 Web 服务组成了 WSC 的问题域。实际中,这些参数是从当前可用的 WSDL 或 OWL-S 的 Profile 描述中提取出来的[14~19]。在该问题域下,用户的服务查询请求也相对简单,仅由提供的输入参数集合和期望的输出参数集合组成[17]。整个 WSC 问题的解决方案是一个有限的可顺序执行的 Web 服务集合序列 $\{W_1,W_2,\cdots,W_n\}$[16,17],其中 $W_i(1\leqslant i\leqslant n)$ 是若干 Web 服务的集合,满足三个条件:a)用户的输入参数满足 W_1 中 Web 服务的执行;b)W_i 的输入被用户的输入参数以及 W_1 到 W_{i-1} 的输出参数的并集所满足;c)用户的输入参数以及 W_1 到 W_n 的输出参数的并集满足用户期望的输出参数。

需要注意的是,同一个 WSC 问题可能有多个解决方案,将结果序列中 Web 服务集合的个数最少的解决方案称为最理想的,即路径最短解决方案。但在问题域中 Web 服务规模很大的情况下,寻找路径最短解决方案的问题是一个 NP-hard 问题[16,17],因此,研究者通常都是设计算法给出接近于最短解决方案的近似方案。关于解决方案的路径长度,也成为评价算法优劣的指标之一[16]。

为了更清楚地描述基于 I/O 的语义 WSC 问题,下面综合前面文献中的定义,归纳出该问题相关的一系列形式化定义。

定义 3　基于 I/O 的 Web 服务。令 C 为概念集合,一个 Web 服务是一个三元组 $ws\langle Is,Os\rangle$,其中,ws 为服务名,Is 表示 Web 服务的输入参数,Os 表示 Web 服务的输出参数,满足 Is、$Os\subseteq C$。在后面的描述中,分别使用 $ws.Is$、$ws.Os$ 表示 ws 所接收的输入参数集合和能够提供的输出参数集合。

定义 4　基于 I/O 的服务请求。令 C 为概念集合,一个服务请求 R 是一个二元组 $\langle Is,Os\rangle$,其中,Is 表示用户提供的输入参数集合,Os 表示用户期望的输出参数集合,且满足 Is、$Os\subseteq C$。

定义 5　基于 I/O 的语义 WSC 问题。令 C 为概念集合,RC 为 C 中概念间的语义匹配关系集合,WS 为 Web 服务集合,Rq 为用户的服务请求。基于 I/O 的语义 WSC 问题定义为从 WS 中自动寻找一个满足 Rq 的带标记的有向无环图 $WG\langle WV,WE,WL\rangle$ 的问题。其中:

a)WG 为该图对应的复合 Web 服务名称。

b)WV 为节点集合,每个节点对应 WS 中的一个 Web 服务,即 $WV\subseteq WS$。

c)WE 为边集,表示两个 Web 服务间的匹配关系。对于任意节点 ws_i、$ws_j\in WV$,且 $ws_i\neq ws_j$,若 $\exists\,\mathrm{out}_i\in ws_i.Os$,$\exists\,\mathrm{in}_j\in ws_j.Is$,满足 $\langle\mathrm{out}_i,\mathrm{in}_j\rangle\in RC$,则 $\langle ws_i,ws_j\rangle\in WE$,即存在一条由 ws_i 指向 ws_j 的边。

d)WL 为边上的标注集合,表示边两端的 Web 服务输入输出概念间的语义匹配关系集合。若 $\langle ws_i,ws_j\rangle\in WE$,则该边上的标注为 WL 的一个子集 $WL_{ij}=\{\langle x,y\rangle\in RC\mid x\in ws_i.Os\wedge y\in ws_j.Is\}$。

e)令 $\forall ws_i\in WV$,则对 ws_i 的限制如下:若 ws_i 的入度为 0,满足 $ws_i.Is\subseteq_{rc} Rq.Is$;若 ws_i 的入度大于 0,则令 $WS_{in}=\{ws_1,$

$ws_2, \cdots, ws_n\}$ 为所有指向 ws_i 的节点集合,则 WS_{in} 中的 Web 服务能满足 ws_i 的输入子集为 $M = \{y \mid (x \in C$ 使得 $\langle x, y\rangle \in \cup_{1 \leqslant k \leqslant n} WL_{ki}\}$,若 $|M| < |ws_i.Is|$,满足 $ws_i.Is - M \subseteq_{rc} Rq.Is$;令 $WG.Os = \cup_{ws \in WV} ws.Os$,即图中所有 Web 服务的输出参数集合,则满足 $Rq.Os \subseteq_{rc} WG.Os$。

这里,以带标记的有向无环图的形式来表示 WSC 的组合方案,能更加直观地展现解决方案的控制流和数据流结构。同时,组合方案的 Web 服务集合的顺序序列形式也可以很容易地根据图中 Web 服务间的依赖关系得到。至于解决方案的路径长度,即文献[17]中提到的时间步概念,能够通过计算图中输入参数和输出参数间的最大路径长度得到。若限制 WG 的路径长度是所有解决方案中最短的,则 WG 为该问题的最佳解决方案。

1.2 基于 IOPE 的语义 WSC 问题

状态改变类 Web 服务在现实世界中是大量存在的,如订票服务、购物服务等,这些服务的功能除了 I/O 参数外,还需要前提条件 P 和执行效果 E 来共同描述。而在有状态改变类 Web 服务参与的 WSC 中,如果仅考虑其 I/O 参数信息显然会有损组合方案的准确性。在这种情况下,对于问题域中的 Web 服务来说,若有 PE 信息,则在 WSC 中需综合考虑。文献[13,20~25]基于从 OWL-S 中抽取的 IOPE 信息进行语义 WSC。在判断问题域中不同的 Web 服务是否可组合时,不仅对 I/O 参数进行语义匹配,还需要利用语义匹配技术扩展前件 P 或执行效果 E[13,21]。用户的服务查询请求也比基于 I/O 的语义 WSC 复杂,不但可以表达提供的输入参数和期望的输出参数,而且还可以使用已有谓词说明初始条件和目标状态。其解决方案也需要加上前件 P 和效果 E 上的限制。为了使基于 IOPE 的语义 WSC 问题更加清晰,下面综合相关文献中的定义,归纳出该问题的一系列形式化定义。

定义 6 基于 IOPE 的 Web 服务。令 C 为概念集合,Pre 为以 C 中元素为变元的谓词集合,一个 Web 服务是一个五元组 $ws\langle Is, Os, Ps, Es\rangle$,其中,$ws$ 为服务名;Is 表示 Web 服务的输入参数,Os 表示 Web 服务的输出参数,且满足 Is、$Os \subseteq C$;Ps 为前提条件,Es 为执行效果,且满足 Ps、$Es \subseteq$ Pre。同样地,使用 $ws.Is$、$ws.Os$、$ws.Ps$、$ws.Es$ 分别表示 ws 的输入参数集合、输出参数集合、前提条件集合以及执行效果集合。

定义 7 基于 IOPE 的服务请求。令 C 为概念集合,Pre 为以 C 中元素为变元的谓词集合,一个服务请求 R 是一个四元组$\langle Is, Os, \text{Inits}, \text{Goals}\rangle$,其中,$Is$ 表示用户提供的输入参数集合,Os 表示用户期望的输出参数集合,且满足 Is、$Os \subseteq C$;Inits 为用户提供的初始条件,Goals 为用户期望的目标状态,且满足 Inits, Goals $\subseteq$ Pre。

定义 8 基于 IOPE 的语义 WSC 问题。令 C 为概念集合,Pre 为以 C 中元素为变元的谓词集合,RC 为 C 中概念间的语义匹配关系集合,WS 为 Web 服务集合,Rq 为用户的服务请求,基于 IOPE 的语义 WSC 问题定义为从 WS 中自动寻找一个满足 Rq 的带标记的有向无环图 $WG\langle WV, WE, WL\rangle$的问题。关于 WG 的解释如下:

a) WG 为该图对应的复合 Web 服务名称。

b) WV 为节点集合,每个节点对应 WS 中的一个 Web 服务,即 $WV \subseteq WS$。

c) WE 为边集,表示两个 Web 服务间的匹配关系,对于任意节点 ws_i、$ws_j \in WV$,且 $ws_i \neq ws_j$,若$\langle ws_i, ws_j\rangle \in WE$,即存在一条由 ws_i 指向 ws_j 的边,则需满足下面的条件之一:

(a) $\exists \text{out}_i \in ws_i.Os$, $\exists \text{in}_j \in ws_j.Is$,满足$\langle \text{out}_i, \text{in}_j\rangle \in RC$,且对于 $\forall e \in ws_i.Es$,满足$\neg(\exists p(p \in ws_j.Ps) \wedge (e \wedge p = \text{false})$;

(b) $\forall \text{out}_i \in ws_i.Os$,满足$\neg(\exists \text{in}_j(\text{in}_j \in ws_j.Is) \wedge \langle \text{out}_i, \text{in}_j\rangle \in C)$,且 $\exists e \in ws_i.Es$, $\exists p \in ws_j.Ps$,满足 evident(e, p),其中,evident(e, p)表示 e 是使 p 成立的证据之一。

d) WL 为边上的标注集合,表示边两端的 Web 服务输入输出概念间的语义匹配关系集合。若$\langle ws_i, ws_j\rangle \in WE$,则该边上的标注为 WL 的一个子集 $WL_{ij} = \{\langle x, y\rangle \in RC \mid x \in ws_i.Os \wedge y \in ws_j.Is\}$。

e) 令 $\forall ws_i \in WV$,则对 ws_i 的限制如下:若 ws_i 的入度为 0,满足 $ws_i.Is \subseteq_{rc} Rq.Is$,且 $Rq.\text{Inits} \Rightarrow ws_i.Ps$;若 ws_i 的入度大于 0,令 $WS_{in} = \{ws_1, ws_2, \cdots, ws_n\}$ 为所有指向 ws_i 的节点集合,则 WS_{in} 中的 Web 服务能满足 ws_i 的输入参数子集为 $M = \{y \mid \exists x \in C$ 使得$\langle x, y\rangle \in \cup_{1 \leqslant k \leqslant n} WL_{ki}\}$,同时 WS_{in} 所满足 ws_i 的前提条件的子集为 $P = \{p \mid p \in ws_i.Ps$,且$\wedge(\cup_{1 \leqslant k \leqslant n} ws_k.Es) \Rightarrow p\}$,则满足 $ws_i.Is - M \subseteq_{rc} Rq.Is$,且 $Rq.\text{Inits} \Rightarrow ws_i.Ps - P$;令 $WG.Os = \cup_{ws \in WV} ws.Os$, $WG.Es = \cup_{ws \in WV} ws.Es$,即图中所有 Web 服务的输出参数集合,则满足 $Rq.Os \subseteq_{rc} WG.Os$,且$\wedge WG.Es \Rightarrow Rq.\text{Goals}$。

同前面基于 I/O 的语义 WSC 问题定义类似,这里也是使用有向无环图的形式来表达组合方案。有些文献中也对基于 IOPE 的组合问题使用同样的形式来定义[26],但其对边以及节点的描述比较模糊,不能清晰地表达组合方案的控制流和数据流结构,这里对其中的定义作了合理的修改和扩展。

需要注意的是,基于 IOPE 的 WSC 问题定义与基于 I/O 的最大不同在于前者在 Web 服务、服务请求以及最终的组合方案中都加入了对 PE 的描述、限制和推理,其最佳组合方案已不是简单的最短路径就可以了,而是在满足 PE 情况下的最短路径。这使得前者描述的问题比后者更加复杂,显然解决问题的难度也相应增加。

1.3 QoS 感知的语义 WSC 问题

前面提到的两类 WSC 问题的定义是基于问题域中 Web 服务的功能描述,因此,它们的最佳组合方案考虑的是路径最短。但在实际中,用户可能会提出基于非功能属性的请求,最常见的是关于服务质量 QoS 的请求,如希望得到的组合方案的吞吐量最大、响应时间最短等,同时路径较长的组合方案可能提供最优的 QoS,这些需求促使了 QoS 感知的 WSC 问题的产生,并受到学术界的关注[27]。不可否认,基于 QoS 组合方案的存在离不开 Web 服务的功能描述,而语义信息是自动化 WSC 的关键,因此,前面介绍的两种语义 WSC 问题是这里 QoS 感知的语义 WSC 问题的基础[23,26,28,29]。下面根据前面文献中的定义归纳出基于 QoS 的语义 WSC 问题的形式化定义。

定义 9 QoS 感知的 Web 服务。令 F 为功能集合,其中的元素要么都是 I/O(定义 3),要么都是由 IOPE 描述(定义 6),Q 为服务质量准则集合。一个 Web 服务是一个三元组 $ws\langle Fs, Qs\rangle$,其中,ws 为服务名;Fs 表示 Web 服务的功能属性,满足 $Fs \in F$;$Qs = \{\langle \text{criteria}, \text{value}\rangle \mid \text{criteria} \in Q$, value 为具体的数值$\}$,表示 Web 服务的具体服务质量,如$\langle \text{responseTime}, 500\rangle$。

定义 10 QoS 感知的服务请求。令 Q 为服务质量准则集合,一个 QoS 感知的服务请求 QRq 是一个二元组$\langle Rq, Qr\rangle$,其中,Rq 为基于 I/O 或 IOPE 的语义 WSC 服务请求,表达的是功能请求;Qr 为用户期望的服务质量要求,是对准则的各种限制集合,如最小响应时间或最大吞吐量、最小代价等。

定义 11　QoS 感知的语义 WSC 问题。令 Q 为服务质量准则集合,FQ 为基于 Q 的服务质量计算函数,QRq 为 QoS 感知的服务请求,WS 为 QoS 感知的 Web 服务集合,QoS 感知的语义 WSC 问题定位为自动寻找一个满足 QRq 的带标记的有向无环图 WG 的问题。其中,WG 的描述与定义 5 或 8 中的基本相同,但需要利用 FQ 计算 WG 对应的组合方案的整体服务质量,限制其服务质量满足 QRq 中期望的服务质量要求 Qr。

QoS 感知的语义 WSC 问题可以被简化为全局优化问题,因此其计算复杂度也是 NP-hard[18]。这说明在问题域规模很大时,找到完美的组合方案是很耗时的工作,因此,很多研究者致力于用最快的时间找到最接近完美的解决方案[18,28,30]。

1.4　其他语义 WSC 问题

前面介绍的语义 WSC 问题中,Web 服务的功能使用 I/O 或 IOPE 描述,其组合方案也是依据 I/O 或 IOPE 间的匹配关系。但是实际中,可能会有两个或多个 Web 服务具有完全相同的 I/O 或 IOPE,但提供的功能却不相同,例如两个服务 findNestestAirport(I(currentLocate),O(address)) 和 getNestestStation(I(currentLocate),O(address)),它们的输入参数都是当前所在的位置 currentLocate,输出参数是地址信息 address,但这两个服务的功能是不同的,前者提供最近的机场地址,后者则是最近的车站地址。如果在组合时仅考虑 I/O 匹配就会产生违背用户初衷的结果。其解决方法是在 WSC 的过程中,除考虑 IOPE 间的匹配外,还需考虑功能语义间的匹配[31]。在这种情况下,一个 Web 服务的功能除了 I/O 或 IOPE 外,还包括操作的名字 action 以及操作的对象 object,如 findNestestAirport 服务的 action 为“find”,对象为“airport”,而 getNestestStation 服务的 action 为“get”,对象为“station”,这样每个 I/O 或 IOPE 相同的 Web 服务通过 action 和 object 描述的功能语义区分开了。当然用户在表达服务请求时,通过增加对 object 的限制,使最终的组合方案符合自己期望的应用领域。通常将在 IOPE 匹配基础上再考虑功能语义匹配的 WSC 问题称为基于功能语义的 WSC 问题。

此外,前面提及的服务请求是基于功能和 QoS 的,表明用户期望的服务组合方案首先能够实现用户的功能需求,然后在此基础上满足 QoS 需求。用户对 QoS 的要求只是简单地符合某个 QoS 准则的最优方案[28,30],但实际中,每个 Web 服务的 QoS 有很多准则,它们反映了 Web 服务对外宣称的服务标准和基本信息,如响应时间、价格、等级以及供应商名等。基于这些准则,用户可能会对组合方案提出更多的约束条件,例如在旅游请求中,要求酒店的等级必须是 4 星级的,且 4 星级的价格不超过 500 美元,5 星级的不超过 750 美元;酒店和租车服务应由同一个服务供应商提供等[32]。这些约束条件有的是时间上的,有的是逻辑上的,它们是影响用户需求的质量条件,也给组合方案的生成带来了困难。在进行语义 WSC 时不仅要考虑 IOPE 的匹配,还需考虑满足这些约束条件,通常将这类组合问题称为用户约束下的语义 WSC 问题。

与前三部份提到的基于 I/O 或 IOPE 以及 QoS 感知的语义 WSC 问题相比,这里的两类语义 WSC 问题受到的关注较少。限于篇幅,不再给出它们的形式化定义。

2　自动的语义 WSC 系统框架

从理论上讲,解决 WSC 问题的过程本质上是构建一个 WSC 软件系统的过程,该软件系统提供的功能支持 WSC 生命周期中的主要活动。许多文献也提出了 WSC 过程框架,文献[8]中将 WSC 过程分为五步,即服务表示、翻译、流程生成、评估和执行,每个步骤都需要不同的语言、平台和方法。该文献只是综述了组合 Web 服务流程生成的方法。

为了给 WSC 领域的研究者提供一个交流的平台,从 2005 年开始,每年都举办 Web 服务竞赛(简称 WS-challenge)[26,27]。从 2008 年开始进行基于语义的 WSC 竞赛,并在规则中提出了 WSC 竞赛程序,该程序与 WSC 软件系统的构建密切相关。这个程序包括服务器端和客户端两部分。服务器端是进行 WSC 的核心,包括导入阶段和组合系统查询阶段,其中导入阶段主要是进行服务文件、语义本体文件以及 QoS 文件的解析,组合系统查询阶段包括计算服务组合和解决方案生成功能;客户端用来接收服务请求和评估解决方案。

文献[12,33]将整个服务组合过程分为用户和组合系统两部分。用户可以是一个人也可以是一个软件系统,它通过描述目标来请求一个组合,也为这个组合提供一些输入数据;组合系统主要用来进行工作流的设计和执行,包括生成可选的组合方案、选择最优组合方案以及执行最优组合方案。

当然,在多数语义 WSC 研究中,Web 服务间的可组合性是基于语义得到的,并没有考虑它们在语法上的兼容性,也没有考虑组合流程是否真正符合用户需求,而这两方面对于组合流程的正确执行起着关键的作用。因此,在 WSC 系统中对组合流程的校验也是非常重要的功能。综上所述,自动的语义 WSC 组合过程可以大致分为三个阶段,即用户交互阶段、组合阶段、执行阶段。其中,用户交互阶段主要是表达服务请求;组合阶段的任务是预处理 Web 服务集和服务请求,以及生成组合流程;执行阶段的任务是验证和执行组合流程。一个自动 WSC 系统的一般框架如图 1 所示。该系统的唯一用户只是基于统一的域本体向系统提出服务请求,在该请求无法用单个 Web 服务满足的情况下,系统自动根据域本体和可用的 Web 服务生成组合流程,并验证初步组合流程的有效性和正确性,进一步生成可执行的流程,最后交由相应的流程引擎来执行。

在自动 WSC 系统组成的三个阶段中,最核心的是组合阶段,因此当前许多关于 WSC 的研究也主要集中在这个阶段。但是在实际的研究中,这三个阶段并不相互独立,它们有时需要相互调用共同完成 WSC 过程,比如在用户表达服务请求不完整的情况下,服务组合过程有可能还需要用户再提供其他信息。

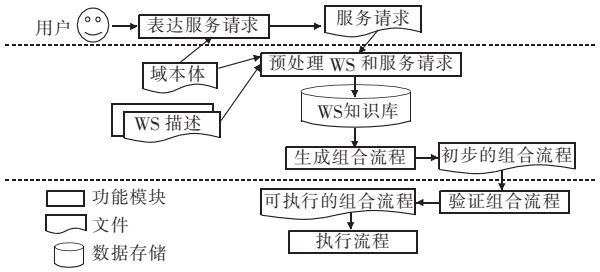

图 1　语义 WSC 系统一般框架

3　自动的语义 WSC 方法

AI 领域的启发式图搜索技术和规划技术被广泛用于进行自动的语义 WSC。前者主要采用依赖图建模 WSC 的问题域,然后在该图上运行启发式的图搜索算法找到最终的组合方案;后者主要是将 WSC 的问题域和服务请求分别转换成 AI 规划

的问题域、初始条件和目标,然后利用现有的或改进的 AI 规划算法得到规划路径,再将规划路径转换成相应的组合方案。下面按照第二部分的问题分类,分别描述它们的建模方法、采用的算法或 AI 规划方法以及组合方案的特点。

3.1 基于 I/O 的语义 WSC 方法

基于 I/O 的语义 WSC 是所有语义 WSC 中处理起来最简单的一种情况。其中,I/O 参数概念间的语义匹配关系和依赖关系能够很好地反映不同 Web 服务间的可组合性,因此,利用这种匹配关系构建的图成为获得组合方案的基础[14,15]。

文献[14]利用接口自动机技术,将每个 Web 服务建模成一个由输入参数集、输出参数集和它们间的依赖集集合组成的接口自动机。参数间依赖关系的确定同 Web 服务的类型相关,若 Web 服务为简单服务,则 I/O 参数间直接存在依赖关系,否则,若 Web 服务为复合服务,则根据具体的流程结构确定 I/O 参数间的依赖关系。根据这些信息为每个 Web 服务构建了一个以 I/O 参数为节点、以它们间依赖关系为有向边的依赖图,并在边上标记能够产生这种依赖关系的所有 Web 服务集合。最后根据参数间的相等匹配关系,将所有的 Web 服务对应的子图整合成一个完整的依赖图,并将查询请求也转换成一个有向子图。这样,WSC 问题转换成了在完整的依赖图上寻找能够覆盖查询请求子图的 Web 服务子图集合问题。该文献中的算法分两步:利用深度优先搜索算法找到满足请求的 Web 服务集合;将查询路径上的 Web 服务按照 OWL-S 中规定的结构组合起来,形成一个符合 OWL-S 流程模型的新 Web 服务。在文献[14]中,所建立的依赖图中只能体现单个 I/O 参数间的依赖,若多个参数间存在依赖则无法体现。

文献[15]使用本体来描述参数、操作、服务间的关系,形成了域本体、Web 服务本体和流程本体。这些本体分别建模了问题域中 Web 服务的 I/O 参数概念间的层次关系、Web 服务间的相似关系以及 Web 服务间的前驱/后继关系。其中,后者定义了 Web 服务间的接口匹配关系,它是基于前者输出参数和后者输入参数的语义相似度确定的;而相似度由本体距离或它们的特征相似度确定。在流程本体的基础上,该文献提出了考虑 QoS 和接口参数语义匹配度的最优服务组合算法,该算法的基础是一个最短路径算法。

Web 服务间的接口匹配图为进行 WSC 提供了充分的信息,但在文献[15]提供的方法中,匹配图是在算法运行的过程中构建的,这使得相应的算法在问题域规模很大时需要花费很长时间才能找到组合方案。因此,预先构建接口匹配图,对于提高组合算法的效率有很重要的意义。文献[19]根据 Web 服务 I/O 参数间的语义匹配关系,预先存储了 Web 服务间的连接关系,其本质上也是构建了接口匹配图。基于存储的连接关系,首先从请求的输入参数出发,使用基于广度优先的前向链搜索算法找到有用的 Web 服务,然后利用请求的输出参数使用基于深度优先的后向搜索算法找出若干组合方案,并采用 token 措施确保得到的是非冗余的组合方案。所谓非冗余的组合是指去掉其中任何一个 Web 服务将不再满足服务请求的组合方案。该文献使用大数量的测试集验证了相应的算法在大规模问题域下仍然保持良好的执行效率。

同样,为了解决组合效率问题,文献[34]提出了一种逐步扩展 Web 服务关系图的方法。在该方法中,服务关系图开始只包含查询请求的 I/O 参数,逐步将与这些参数在语义上匹配的所有 Web 服务引入到关系图中;然后利用深度优先搜索算法查找解决方案,若没有,则调整服务请求,继续将与请求语义相关的 Web 服务引入到关系图中,继续上述过程;最后用户根据 QoS 对所有组合候选进行排序和过滤。这种方法可在一定程度上减少搜索空间,但也有可能带来搜索不完全的情况。

AI 规划技术[35]经常被用来解决 WSC 问题。在规划上下文中,将 I/O 参数看做是状态中的元素,而问题域中的每个 Web 服务是消费和产生这些状态元素的动作,所有动作组成一个知识库。如果服务请求的输入参数集是初始状态,输出参数集是目标状态,则 WSC 问题被转换成从知识库中查找满足目标的一系列动作规划的问题。AI 规划中,规划图含有丰富的知识信息,通常被用做进行规划的基础,它描述了状态的变迁过程,其变迁是由动作的发生引起的,因此本质上是一个状态转移图。一些关于 WSC 研究就是基于这样的规划图进行的[16,17],它们将 WSC 问题转换成状态空间上的搜索问题[35]。

文献[16]中,可用 Web 服务集合被看做一个状态节点网络,该网络是一个有向图 $Gs(Vs,Es)$,其中 $s_i \in Vs$ 表示一个状态,Es 是有向边(或弧)(s_i,s_j) 的集合,每条弧的权重是调用代价 $c(w)$,这里 $w \in \Omega(s_i)$ 且 $s_j = f(w,s_i)$。其中,$\Omega(s)$ 是 Web 服务 $w \in W$ 的集合,满足 $w^1 \subseteq s$,即在状态 s,w 可被调用或是可用的。转移函数 $f(w,s) = s'$,它将一个状态 s 映射到一个状态 s',使得对于 $w \in \Omega(s)$,满足 $s' = s \cup w^o$。这里,w^1 和 w^o 分别表示服务 w 的输入参数集合和输出参数集合。服务请求的输入参数映射为初始状态 s_0,输出参数映射为目标状态 s_n,WSC 问题被转变成寻找从 s_0 到 s_n 的最短路径问题。该文献借鉴图规划算法,采用两步搜索的方法获得组合方案。传统的图规划算法目的是最小化完成规划的时间步(每个时间步可能完成多个动作),而文献[16]采用关于参数调用代价的启发,目的是最小化组合方案中的 Web 服务数量。第一步的前向搜索算法找到参与组合的 Web 服务集合,并计算从请求输入参数到输出参数路径上的所有参数的调用代价和产生这些参数的 Web 服务集合;第二步的衰退搜索是基于贪婪算法的后向搜索,在执行过程中利用第一步得到的启发信息,最终得到一个 Web 服务的执行序列。文献[16]在各种不同特征的数据集上验证了该算法的有效性和可量测性。

图规划思想对于解决 WSC 问题很有效,文献[17]也是采用类似的方法,并提出了服务规划图的概念。它是包含参数节点和服务节点的有向无环分层图,其中参数节点和服务节点交替出现,且服务节点能被前面一层的参数节点满足,其第一层参数节点为服务请求的输入参数,最后一层为参数节点层,包含服务请求所期望的输出参数。文中通过三个步骤实现 WSC:a)根据服务网络[36,37]的抽象服务将部分或完全匹配的 Web 服务分组,借助于分组将每个输出参数概念与产生它的 Web 服务建立关联,并存储起来;b)利用前向搜索算法构建服务规划图;c)利用启发式的后向衰退算法获得组合方案。其启发方法是优先选择输出数量较多的服务;在输出数量相等的情况下,优先选择输入数量较少的服务;优先选择已经选择过的服务。其启发的目的是尽可能减少组合方案中的服务数量。

除了图规划外,其他 AI 技术也受到 WSC 研究者的注意。文献[18]提出利用 SAT 技术找出最短组合方案的方法。类似于文献[16]中的建模方法,它首先将 WSC 问题转换成 STS(状态转移系统)上的可达性问题。在 STS 中,其中的状态是布尔变量的集合,每个布尔变量表示是否相应的类型有对应的实例,WS 被看做是状态间的转移,若一个 WS 在状态 q 被调用,

则新的状态 q' 在 q 的基础上增加 WS 的输出实例。然后设计算法利用现有的 SAT 解决器确定是否存在从请求的初始状态到目标状态的路径。若存在则抽取最短路径,形成 Web 服务的顺序序列。文献[26]利用约束逻辑编程实现 WSC。首先构建 Prolog 事实库,包括服务列表、I/O 参数以及这些参数间的语义关系;然后利用 Prolog 实现的多步变窄算法获得组合方案。该算法首先从请求的输入参数开始,查找能满足的 Web 服务,过滤掉对组合无用的服务,然后从用户期望的输出参数开始删除掉冗余的服务,最后生成 Web 服务的组合序列。

3.2　基于 IOPE 的语义 WSC 方法

对于包含状态改变类 Web 服务的问题域来说,如果仅使用基于 I/O 的语义 WSC 方法进行组合,那么组合方案的正确性会降低。因此在 WSC 过程中,综合考虑 IOPE 的信息是很有必要的,也受到学术界的广泛关注,出现了一些切实可行的解决方案[13,20~25]。这些解决方案主要采用 AI 规划技术实现,如基于逻辑编程的规划、HTN、图规划等。此外,鉴于状态改变类 Web 服务的复杂性,为了保证方法的正确性,解决方案通常遵循 IRP(调用和合理的持久化)假设[20,24],其含义包括两个方面:a)在初始状态下,信息提供类服务应该是可执行的;b)从这些服务收集到的信息不能被外部或后续的动作所改变。下面将按照所采用的规划技术的不同描述典型的解决方案。

3.2.1　基于逻辑编程的方法

在 AI 规划领域,可以使用逻辑编程技术解决规划问题。其中问题域中的知识使用一阶谓词逻辑来表示,主要有两种形式化的方法,即情景演算(situation calculus,SC)和事件演算(event calculus,EC)[35]。

SC 基于情景表达动作和它们的效果,一个情景(situation)对应于动作的一个序列或历史。在 WSC 的过程中,语义 Web 服务中的原子流程被转换成动作,这些动作由 SC 定义,形成一系列 SC 公理集合,该集合称为一个域理论(domain theory)[24]。语义 Web 服务中的复合流程将使用逻辑编程语言来表示,如 Golog[38]。它是一种高级逻辑编程语言,构建在情景演算之上,提供了将若干原子动作装成复杂动作的超逻辑结构,包括顺序、测试、动作的非确定选择、参数的非确定选择、非确定迭代、条件以及 while 循环。文献[24]对 Golog 作了适当的修改,可以使用 Golog 构建可重用的高层通用程序,这些程序被存储在共享的通用程序本体中,以便用户选择和执行。Golog 程序的执行依赖于一个 MG 解释器,即在线执行信息收集动作(信息收集服务),离线模拟世界改变动作(世界改变服务)的执行。最终,用户得到的是一个满足服务请求的状态改变类 Web 服务的序列。直观地说,文献[24]中的通用 Golog 程序相当于预先手工定义的工作流模板,用户只是选择一个模板,然后利用 MG 解释器生成一个具体的动作计划,所以该方法的自动化程度不高。

SC 不能表示连续和并发动作,而这些情况在 WSC 中经常出现,因此有研究者使用基于 EC 的方法来解决这个问题[23,25]。EC 是基于时间点表达动作及其效果,它的基本本体包括动作或事件、流以及时间点。它能用已有的事件公理表示并发的动作,并用复合事件表达复杂目标。在使用 EC 进行规划时,OWL-S 描述的原子流程和复合流程都用 EC 逻辑公式表示,文献[25]中给出了 OWL-S 到 EC 的转换规则。转换过来的所有 EC 公式就组成了域知识库,基于该知识库和 EC 表达的目标状态,运行诱导原理证明器[39]产生规划。文献[23]基于 EC 提出了一个整体的组合和执行框架(WSCE)。它将 WSC 的组合和执行阶段严格分开,首先利用 Web 服务参数的类型信息,基于 EC 公理执行诱导原理证明器生成规划,即生成若干 WSC 的组合模板;然后根据用户的约束及 Web 服务的 QoS 选出符合需求的组合模板,并实例化成具体的组合流程;最后,基于用户输入的参数值执行组合流程。

3.2.2　基于 HTN 的方法

由于 HTN 规划中的任务分解概念类似于 Web 服务组合中的流程分解,现有 HTN 规划技术能够用来解决 WSC 问题。文献[20]使用 SHOP2 HTN 规划系统来自动组合 OWL-S 描述的 Web 服务。其中,OWL-S 中的原子流程和复合流程分别被转换成操作和方法,所有的操作和方法就构成了 SHOP2 的规划域。服务请求是一个抽象的组合流程,它由若干抽象的子任务组成。整个规划的过程就是根据规划域中提供的若干方法,找出每个子任务对应的具体操作序列。在规划过程中,如果某个子任务没有具体的方法可用,则需要请求用户给出该子任务的分解方法。最后,将所有具体操作序列转换成 OWL-S 对应的 Web 服务序列,形成一个 Web 服务组合方案。在规划期间,为了确定当前状态,该方法直接执行了信息提供类 Web 服务,模拟了状态改变类 Web 服务的执行,所以最终的组合方案中都是状态改变类 Web 服务。借助于知识库中丰富的分解方法,HTN 规划器能够快速地找到组合方案。但是其知识库的构建需要更加专业的知识,这限制了 HTN 方法在一般 WSC 中的应用。

3.2.3　基于图规划的方法

图规划是进行 AI 规划的常用方法,当前有很多优秀的图规划器,如 JGraph[40]、FF[41]、Simplanner[42]等。这些规划器都支持标准的 AI 规划语言 PDDL,因此一些研究者关注于如何将语义 WSC 问题转换成使用 PDDL 表达的标准 AI 规划问题。

OWLS2PDDL[21]就是这样的转换工具,它将 OWL 描述的服务请求和 OWL-S 描述的服务集合转换成 PDDL 表示的标准规划问题。其转换规则是:OWL 中出现的类型和属性需要转换成 PDDL 谓词,OWL-S 服务需要映射成相应的动作。由于每个服务可由 IOPE 描述,在转换时,直接将 P 转换成动作的前件谓词,E 转换成效果谓词;对于任意一个输入参数增加同名的 PDDL 动作,且增加其前件谓词 agentHasKnowledgeAbout(input param);输出参数直接转换为效果谓词,表示为 agentHasKnowledgeAbout(output param)。文献[21]利用该工具和一个 Xplan 规划器构建了一个语义 WSC 框架 OWLS-Xplan,Xplan 在 RPG(放松的规划图)规划器 FF 的基础上增加了局部搜索和 HTN 规划,所以这两个规划技术的优点在 OWLS-Xplan 系统中得以体现。

文献[13]也提出了类似于 OWLS-Xplan 的语义 WSC 框架 PORSCE。不同于 OWLS2PDDL 的转换规则,PORSCE 将服务的输入参数和前提条件转换成动作(对应于每个 OWL-S)的前件谓词,并根据用户提出的语义近似值,使用 OOM 工具抽取概念间的语义距离,找出复合需求的输入—输出匹配关系,然后利用这些关系扩展初始条件中的事实和每个动作的执行效果,这样可以辅助创建近似的输入—输出匹配规划。PORSCE 中规划的执行依赖于外部支持 PDDL 的标准规划器,如文中使用的 JGraph 规划器。

在可用 Web 服务集合和服务请求不变的情况下,文献[13,21]中的方法能够正确运行。但现实世界是一个动态的

环境,会出现 Web 服务的增加和丢失,用户的请求也经常会发生变化,因此需要采取一定的措施来应对这些变化。文献[43]在 OWLS-Xplan 框架中增加了重新规划组件,用于处理操作的新增和丢失以及目标的修改;文献[13]提供了可视化的组合方案修改接口,通过它用户可以选择替换或删除某个 Web 服务,然后通过重新规划技术获得新的组合方案。

此外,文献[13,21,43]在规划过程中只是模拟了所有 Web 服务的执行,并没有真正执行,因此最终得到的是完整的组合方案。但在现实世界中,Web 服务的可用性只有在真正调用时才能获得,因此有必要通过执行 Web 服务的方式来获得准确的组合方案。文献[22]提出一个基于 Simplanner 规划器和 WS-事务的自动 WSC 框架。Simplanner 是一个任一规划器,能够进行动态以及部分可观察环境下的规划,它主要基于 RPG 进行目标分解和启发式搜索。文中的框架根据获得的组合方案执行其中的 Web 服务,若执行失败,则撤销该服务的影响,然后修改当前的状态并利用 Simplanner 进行重新规划,无须从头开始。这种方法能很好地适应现实世界动态变化的特点,但为了能撤销执行失败的 Web 服务,要求每个 Web 服务都遵循 WS-事务规范。为了适应真正执行的需要,框架对 OWLS2PDDL 工具作了适当扩展,将 OWL-S 对应的动作称为逻辑动作,OWL-S grounding 对应的 WSDL 中的操作定义为物理动作。

3.2.4 基于 Petri 网的方法

除了前面介绍的逻辑编程和规划技术外,Petri 网的结构化分析技术也被用于解决 WSC 问题。文献[44]使用 Horn 子句规则建模 Web 服务间的接口依赖和行为依赖关系,最终将 WSC 问题转换成 Horn 子句的逻辑推理问题。文中通过两步完成 WSC,首先使用一个线性时间算法确定组合方案是否存在,同时返回组合需要的规则集;然后使用 Petri 网建模这些规则集,并利用它的结构分析技术来获得组合服务。其中,在构建 Web 服务的依赖规则时,不仅考虑服务 I/O 参数间的接口匹配依赖,还需要关注相互间行为约束的兼容性,前者保证了服务间的可操作性,而后者保证其能够相互协作。

3.3 QoS 感知的语义 WSC 方法

在前面介绍的基于 I/O 或 IOPE 的语义 WSC 方法中,可能会获得多个组合方案,许多研究基于 Web 服务的 QoS 对组合方案进行排序和过滤[23,26,34]。其中,QoS 的计算主要基于响应时间和吞吐量,文献[27]给出了不同结构复合 Web 服务的 QoS 计算方法。不同于先组合后筛选的方法,文献[15,18,28~30,45]在组合过程中就考虑 QoS 因素,最终直接获得 QoS 最优的组合方案。这里主要介绍这种一体化的解决方法。

考虑 QoS 的最短路径算法经常被用来解决这样的问题[15,30]。文献[15]在选择路径中匹配的 Web 服务时,不仅考虑非功能质量因素 QoS,而且还关注功能质量,即接口参数语义匹配度,并通过用户提供的参数增强或弱化其中一方对组合方案的影响。文献[30]提供的方法类似于文献[16],本质上是两步的搜索算法:a)使用广度优先查找可用的 Web 服务,并保存产生的每个输出参数以及每个被满足的 Web 服务的最优 QoS;b)使用深度优先反向查找具体的执行路径。该算法通过了大规模 Web 服务集上的测试,能够在较快的时间内找到较优 QoS 的组合方案。

文献[45]定义了功能质量模型,它包括匹配质量和共同描述度两个维度。前者测量概念间的匹配关系,其中 exact 关系表示为 1,plugin 为 3/4,subsume 为 1/2,intersection 为 1/4;后者根据匹配参数在共同概念表达中的紧密程度描述它们间的相似度。功能质量和非功能质量 QoS 中的响应时间和吞吐量两个维度共同决定候选组合方案的质量;在这些质量因素的基础上,利用遗传算法选择出最优的组合方案。

此外,由于 QoS 最优方案的获取是 NP-hard 的,一些研究者采用任一时间算法来进行 WSC[18,28],这种算法可以很快得到一个组合方案,并且时间越长,结果越好。文献[18]基于近似搜索算法 beam stack 和启发策略设计了任一时间 QoS 感知的 WSC 算法,并验证了其可以更快地找到质量较高的组合方案。文献[28]根据 QoS 值扩展规划图,将其值标记成边的权重,然后在该图基础上运用 Dijkstra 算法查找最优方案。由于 Dijkstra 算法会遍历到图中的每个顶点,随着时间的推移,它总能找到无冗余的 QoS 最优方案。文中将 I/O 参数的语义匹配关系进行了扁平化处理,不用每次在匹配时都运行 OWL 推理机,这样节省了组合的时间。

文献[29,30]针对特定的测试集,给出了求 QoS 最优解的算法,该算法改进了传统的两步搜索算法,它在正向搜索时以迭代的方式计算了每个输出参数的最优 QoS 值,然后在该 QoS 值的基础上执行后向搜索算法,每次选取能够提供 QoS 最优值的 Web 服务,最终构成整个 QoS 值最优的组合方案。文献[29]还给出了 QoS 值最优的全解构造方法。

3.4 其他的语义 WSC 方法

在多个 Web 服务具有相同的 I/O 参数的情况,服务的功能语义对正确组合方案的获得有着重要的作用。文献[31]使用 Web 服务的 I/O 参数以及动作、对象来描述 Web 服务的功能,其中动作和对象描述了服务对哪个对象执行什么动作,即功能语义。Web 服务的 I/O 参数和对象使用域数据本体描述,同时引入域动作本体描述动作之间的语义关系。根据这些域本体,文中构建了一个服务关系图 SRG,展现了所有 Web 服务的功能元素以及相互之间的关系,它包括数据依赖图(表示服务间的依赖关系)、动作图(表示域动作间的关系)以及这两个图间的映射。通过对 SRG 的搜索找出符合用户功能需求的组合方案。实际中,用户使用各种约束能够更好地表达服务请求,但这也给 WSC 的实现带来难度。文献[32]提出了一种基于服务模板的语义 WSC 方法,最后获得的具体组合方案必须满足用户提出的各种约束。它将约束下的 WSC 问题转换成了约束满足问题,然后在约束解决器的帮助下找到组合方案。文中将服务组合上下文上的约束分为时序和服务选择约束两类,并提供建模语言表达这些约束。此外,在确定 Web 服务是否满足约束时,充分考虑其语义模型。

3.5 语义 WSC 测试

语义 Web 服务测试集是进行 WSC 研究必不可少的元素,尤其是作为标准的基准测试集,目前在语义 WSC 领域还没有这样的测试集。研究中经常用到的是国际 WSC 竞赛中的测试集[18,26,28,30],从 2008 年开始,其测试集才包含语义信息[27]。文献[16]自行设计了测试集生成工具 WSBen,它可以生成各种不同类型的测试集,但是没有语义信息。这两种测试集都是虚拟的,其中的 Web 服务没有真实的语义,因此不能模拟真实 WSC 场景。于是,有的研究者自行从网络上爬取若干 Web 服务,然后手工修改形成测试集。由于网络上的语义 Web 服务并不多,通常是爬取 WSDL,然后利用本体自行标注语义[17]。与前两种相比,这种方法获得的数据集语义丰富,但 Web 服务

数量偏少,不适合大规模环境下的适应性测试。还有一些研究者自行设计了少量的 Web 服务,仅是为了验证框架的功能性[21~23,31]。此外,在语义 WSC 中,测试指标为执行时间、QoS 和组合方案的语义质量三类。不同类型的 WSC 方法侧重点不同,限于篇幅,这里不再一一指出。

3.6　分析与总结

基于 I/O 的语义 WSC 是四类中最简单、也是最基础的,其 WSC 方法在对问题的建模方法、采用的组合算法以及对组合方案的要求是不同的,其相关分类如表 1 所示。其广泛采用的方法是使用状态转移系统进行建模,然后改进图规划的两步搜索算法,最终获得最短路径非冗余的服务序列。

表 1　基于 I/O 的语义 WSC 方法分类

分类方法	具体类别	相关文献
Web 服务建模方法	参数依赖图	[14]
	Web 服务接口匹配图	[15,19,34]
	状态转移系统	[16~18]
	约束逻辑规则	[26]
组合算法	普通搜索算法	[14,15,34]
	两步搜索算法	[16~19]
组合方案	最短服务序列	[14,15,18]
	最短服务序列且最少服务数	[16,17]
	非冗余的组合序列	[19]
	普通的组合序列	[26,34]

相对于基于 I/O 的 WSC 来说,基于 IOPE 的 WSC 对问题域的建模更加复杂,也需要更加复杂的推理技术。前面已经按照所采用的技术对基于 IOPE 的语义 WSC 方法进行了分类,但除了使用的技术不同外,这些方法在规划执行期间有的没有执行任何 Web 服务[13,21],有的只执行了信息提供类 Web 服务[20],还有的执行了所有的 Web 服务[22]。根据在规划期间是否执行某些 Web 服务,文献[11]将 WSC 方法分为动态和静态两类方法。前者规划的生成和执行是交叉进行的;而后者规划的生成和执行是严格分开的,即在规划阶段不会执行任何 Web 服务。表 2 详细列出了这部分出现的 WSC 方法的相关分类。

表 2　基于 IOPE 的语义 WSC 方法分类

类别		相关文献	所采用技术
静态	方法	[21,13,43]	图规划
		[44]	基于 Petri 网的逻辑推理
		[23]	EC 演算
动态方法	只执行信息提供类服务	[20]	HTN 规划
		[24,25]	SC 或 EC 演算
	执行全部服务	[22]	图规划

QoS 感知的语义 WSC 是在前两种组合的基础上增加了 QoS 约束。研究者试图设计优质的算法快速找到 QoS 最优的组合方案,但由于全局优化问题大多数是 NP-hard 问题,遗传算法、最短路径算法、任一时间算法等近似算法被广泛用于解决这个问题。当然,针对特定的测试集和单个 QoS 值,是能够快速求出其最优组合方案的。单个 QoS 属性并不能全面评价组合方案的好坏,因此,多个 QoS 属性加上其他功能质量方面属性,它们混合在一起才能形成对组合方案的全面评价。这方面的研究比较少,会是将来的一个研究热点。此外,功能语义和各种行为约束会影响组合方案的正确性,这方面也有一些研究,但还不成熟。

4　组合方案验证技术

前面介绍的大多数组合方法中,最终产生的是 Web 服务的组合序列,这些组合序列主要是以能够被流程引擎直接执行的组合代码文档(如 BPEL 等)形式呈现。当然,也有其他抽象的呈现形式,如 UML 活动图或状态图[46]、WS-CDL[47]、BPMN[48]等,它们能够描述 Web 服务间的相互调用关系和其中的消息传递,但很少涉及具体的执行细节。目前有很多现成的转换工具将这些抽象形式转换成 BPEL 代码。然而,在从设计组合序列到生成可执行组合代码的整个过程中,并没有验证组合流程的正确性,即组合流程是否与用户需求一致,是否满足可达性、安全性、有界性、可执行性等这些保证程序正确执行的特点。若在设计阶段不进行这样的验证,则会将错误转移到流程执行阶段,到时处理错误的代价将会更大。因此,在设计阶段对组合流程进行正确性的验证是很有必要的。

可以将前面的验证特点分为两类,即一致性验证和特性验证。所谓一致性验证指的是验证两个从不同抽象级别、不同侧面描述的组合流程是否整体一致,如 WS-CDL 或 BPMN 描述的编排与参与编排的 Web 服务的具体实现模型(WS-BPEL)进行一致性的校验[49]。同时为了方便编排的设计,用户可以使用 UML 可视化工具来构建编排,然后转换成 WS-CDL 或 MSC(消息顺序图)再进行后面的验证[49,50]。此外,有研究者将组合流程建模为控制行为和操作行为两部分,其中前者与具体的应用逻辑无关,而后者描述流程底层的业务逻辑。然而,在一个流程中,控制行为指导操作行为的正确执行,因此控制行为和操作行为的一致性也需要得到保证[51,52]。特性验证指的是判断组合流程是否满足可达性、安全性、可执行性等一系列行为特点[53~56]。

4.1　验证方法与技术

由于进程代数能够描述服务组合的主要结构,如顺序、并行及条件执行,并且具有很强的表达能力,因此,它被广泛用于组合流程的形式化建模中。文献[49]将用 WS-CDL 或 BPMN 描述的设计规范表达成 CSP(通信顺序进程),同时也将参与编排的 Web 服务的实现 WS-BPEL 转成 CSP,最后使用支持 CSP 的模型验证工具 FDR 来验证两者间的一致性。文献[51]使用 FSP(有限状态进程)来建模抽象的 UML 消息顺序图和具体的 BPEL4WS,使用 LTSA 工具的安全检验功能对两者进行一致性验证。文献[55]使用 UML-AD 描述流程设计规范,并使用 LOTOS 语言建模该规范中的各种工作流模式,从而建模整个设计流程;然后使用支持 LOTOS 语言的模型检验工具 CADP 进行各种行为特性的验证,这些行为特性可以使用 mu 演算或时序逻辑 CTL * (包含线性时序逻辑 LTL 和分支时序逻辑 CTL)描述;最后提供了将通过验证的 UML-AD 转换成 BPEL 代码的工具。

组合流程通常也会用 STS(状态转移系统)进行建模。在文献[51,52]中,使用 UML 状态图描述组合流程的控制和操作行为及它们之间的关联,操作行为被建模为一个 STS,需要验证的特性从控制行为结构中抽取出来,并使用 CTL * 来表达这些特性。模型检验工具 NuSMV 使用 CTL * 特性和操作行为对应的 STS 作为输入,最终提供两种行为间的一致性验证。时序自动机也是一种 STS,它可以方便描述组合流程中的时间约束特性。文献[56,57]使用时序自动机分别建模了由 WS-CDL 和 WS-BA 描述的流程设计逻辑,需验证的安全性、活性等行为特性由 CTL * 表达。模型检验工具 UPPAAL 将转换来的时序自动机和 CTL * 表达式作为输入,最终确定流程设计是否满足某些行为特性。文献[56]还证明了 WS-CDL 的子集转换为时序自动机方法的完备性,并不是所有的基于 STS 的方法中都使

用现有的验证工具。文献[54]提出了在组合流程的验证中充分考虑其数据流特点的方法,它采用抽象技术建模与组合正确性相关的数据来隐藏不相关的数据。使用 ground 模型表示系统中的数据、数据上的操作以及数据流,同时在 ground 模型的基础上将 BPEL 描述的组合流程定义成一个 STS,然后从基于知识的 K-model 和基于分支的 B-model 对 STS 进行不同的抽象。在 K-model 和 B-model 共同的作用下完成对 STS 行为特性的验证。它自行设计和开发了验证所需的算法和开发框架。

同前两种形式化工具类似,Petri 网也具有很强的表达能力和有效的分析技术,因此经常被用于组合流程的建模和验证研究中。文献[53]使用有色 Petri 网建模 OWL-S 表达的语义 Web 服务组合流程,并设计了基于有色 Petri 网的可达性、有界性以及组合服务语义一致性的判断算法。文献[58]将两个使用 BPEL 描述的抽象流程建模成有色 Petri 网,从而分析它们的兼容性。文献[17]中的语义 Web 服务流程一致性验证部分,也是用 Petri 网技术建模组合流程,并利用 Petri 网的结构化分析技术计算每个 Web 服务的可执行性。

4.2 分析与总结

Web 服务的组合流程本质上是一个由若干 Web 服务组成的软件系统,因此,模型检验技术能够成功用于 Web 服务组合流程的正确性验证。近年来,使用该技术进行 Web 服务组合和验证的研究也出现了不少,其思路大体一致,即首先在验证初期需要利用形式化的工具(如状态转移系统、进程代数、Petri 网等)对参与组合的每个 Web 服务、组合流程以及需要验证的模型和特性进行形式化描述,然后利用现有的模型检验工具(CADP、UPPAAL、NuSMV 等),或使用 Petri 网结构化分析方法进行各种正确性的验证。在整个过程中,形式化工具对于最终模型检验工具的选择和执行效率起着决定性作用。

5 结束语

本文从一个全新的视角展现了近些年来语义 WSC 的研究成果。为了使研究者能够明确语义 WSC 的边界,在总结和归纳现有研究的基础上,根据对问题的不同描述将语义 WSC 问题分为四类:基于 I/O、基于 IOPE、QoS 感知及其他的语义 WSC 问题,并以统一的方式给出了它们的形式化定义。根据现有研究中采用的组合框架,归纳出一个语义 WSC 系统的一般框架,在框架中展示了解决 WSC 问题需要进行的主要工作,使研究者对解决该问题有一个整体的认识;并且按照前面的分类对每个语义 WSC 问题的组合方法进行了分类描述和总结,为了方便研究,分析和比较了当前流行的语义 WSC 测试集;最后,将组合方案的正确性验证工作归纳为一致性验证和特性验证两类,并分别介绍了这两类的现有研究成果。

从第 1 章的问题分类和定义中,注意到语义 WSC 问题包含三个要素,即问题域(Web 服务集合)、服务请求和组合方案。现实世界对这三个要素的不同要求决定了解决问题所采用的方法和难度。综合前面介绍的现有研究成果,下面按照这三个要素总结语义 WSC 当前面临的问题和挑战。

1)问题域　在大多数的语义 WSC 研究中,都默认问题域是静态的,即不会存在 Web 服务的增加、删除和修改,其对外宣称的可用性等不会发生改变[14~17],同时,默认其中 Web 服务的输出和执行结果是确定的。在这种确定的静态域下,查找路径最短且服务数最少的组合方案问题是 NP-hard 的,因此对获得最优解的高效 WSC 方法研究一直是一个开放的研究。

对于静态的问题域来说,其中的 Web 服务可能存在不确定的输出和执行效果,也可能出现其内部状态是部分可观察的,即无法观察到 Web 服务内部的流程结构。这种不确定和部分可观察性增加了语义 WSC 的难度,需要利用相关技术来解决,如动态 AI 规划技术[35]。但目前这方面的研究较少。

静态的问题域确实方便了对语义 WSC 的研究,但这只是理想状态。实际中 Web 服务大多来自互联网,它们组成的问题域是动态的,即不断会有新服务出现,也随时会有某些旧服务变得不可用。面对这样随时可能发生变化的问题域进行语义 WSC 是一个有挑战的工作。文献[13,43]通过增加重新规划技术来解决这种动态性,但高效、实时、正确的重新规划技术值得进一步研究。

2)服务请求　其复杂程度会对 WSC 问题的难度产生影响,尤其是带用户约束的请求,这些约束不仅包含对 Web 服务基本信息的约束,如某几个服务需同一个供应商,也包含 QoS 指标间的相互约束,同时包含对最终组合序列的时序约束。目前还缺少全面的约束建模方法,文献[32]进行了这方面的尝试,不过它是基于在服务模板上的 WSC,其自动化程度不高。如何将抽象技术和基于 I/O 或 IOPE 的语义 WSC 技术相结合,形成一体化且高效的语义 WSC 解决方案是一个值得研究的问题。在前面的研究中,总是假定用户提供了完整的服务请求,但实际中用户往往不清楚自己应该为 WSC 提供哪些信息,并期望能与 WSC 过程进行必要的交互。因此,对用户如何参与 WSC 的研究是有必要的。

3)组合方案　描述了 Web 服务组合序列内部的控制流和数据流结构。当前大多数的自动语义 WSC 方法中,得到的组合方案只支持顺序(sequence)、分离(split)和分离—合并(split-joint)结构,对于条件、迭代和并发支持较少。这与所采用的组合方法有很大关系。因此,设计一个语义 WSC 方法,使它能获得支持条件、迭代或并发控制结构也是有意义的工作。

组合方案的功能质量,即 Web 服务间的语义匹配质量,对方案的正确性有着重要的影响。正确建模组合方案的功能质量,并与其 QoS 合理整合形成综合质量模型是有意义的[15,45]。综合质量模型在语义 WSC 中的使用方式和对组合方案正确性的影响还需进一步探讨。此外,在前面提到的多数研究中,组合方案是基于接口参数概念间的语义匹配构建的,可能由于匹配的概念采用异构的数据结构,造成其语法层次无法匹配,阻碍服务间的互操作。再有,第 4 章描述了验证语义 Web 服务组合的相关方法和技术,但它们大都与组合过程相对独立。如何将组合过程、验证过程、语法校验整合在一起,使得它们可以共同作用,一次性生成一个可实现的组合方案也是一个值得关注的方向。

参考文献:

[1] CHRISTENSEN E, CURBERA F, MERDITH G, *et al*. Web services description language(WSDL) 1.1[EB/OL].(2001-03-15)[2013-10-01]. http://www.w3.org/wsdl.

[2] HIGH J R, KINDER S, GRAHAM S. IBM's SOA foundation: an architecture introduction and overview[EB/OL].(2005-12-15)[2013-10-01]. http://download.boulder.ibm.com/ibmdl/pub/software/dw/webservices/ws-soa-whitepaper.pdf.

[3] FERRIS C, FARRELL J. What are Web services? [J]. Communication of the ACM, 2003, 46(6): 31.

[4] GRIGORIS A, Van FRANK H. A semantic Web primer[M]. 2nd ed. Cambridge: MIT Press, 2008.

[5] MARTIN D, BURSTEIN M, HOBBS J, *et al*. OWL-S: semantic markup for Web services[EB/OL]. (2004-11-22)[2013-10-01]. http://www.w3.org/Submission/OWL-S/.

[6] ROMAN D, KELLER U, LAUSEN H, *et al*. Web service modeling ontology[J]. Applied Ontology Journal, 2005, 1(1): 77-106.

[7] FARRELL J, LAUSEN H. Semantic annotations for WSDL and XML schema[EB/OL]. (2007-08-28)[2013-10-01]. http://www.w3.org/TR/sawsdl/.

[8] RAO Jing-hai, SU Xiao-meng. A survey of automated Web service composition methods[C]//Proc of the 1st International Conference on Semantic Web Services and Web Process Composition. Berlin: Springer, 2005: 43-54.

[9] SYU Y, MA Shang-pin, KUO J Y, *et al*. A survey on automated service composition methods and related techniques[C]//Proc of the 9th IEEE International Conference on Services Computing. 2012: 290-297.

[10] CHARIF Y, SABOURET N. An overview of semantic Web services composition approaches[J]. Electronic Notes in Theoretical Computer Science, 2006, 146(1): 33-41.

[11] KLUSCH M. Semantic Web service coordination[M]//CASCOM: Intelligent Service Coordination in the Semantic Web. Basel: Springer, 2008: 59-104.

[12] BARTALOS P, BIELIKOVÁ M. Automatic dynamic Web service composition: a survey and problem formalization[J]. Computing and Informatics, 2012, 30(4): 793-827.

[13] HATZI O, VRAKAS D, BASSILIADES N, *et al*. The PORSCE II framework: using AI planning for automated semantic Web service composition[J]. The Knowledge Engineering Revien, 2013, 28(2): 137-156.

[14] HASHEMIAN S V, MAVADDAT F. A graph-based approach to Web services composition[C]//Proc of Symposium on Applications and the Internet. Washington DC: IEEE Computer Society, 2005: 183-189.

[15] ARPINAR I B, ZHANG Ruo-yan, ALEMAN-MEZA B, *et al*. Ontology-driven Web services composition platform[J]. Information Systems and E-Business Management, 2005, 3(2): 175-199.

[16] OH S C, LEE D, KUMARA S R. Effective Web service composition in diverse and large-scale service networks[J]. IEEE Trans on Services Computing, 2008, 1(1): 15-32.

[17] 胡佳. 语义 Web 服务自动组合及验证的研究[D]. 天津: 天津大学计算机科学与技术学院, 2010.

[18] KIL H. Efficient Web service composition: from signature-level to behavioral description-level[D]. University Park: The Pennsylvania State University, 2011.

[19] KWON J, LEE D. Non-redundant Web services composition based on a two-phase algorithm[J]. Data & Knowledge Engineering, 2012, 71(1): 69-91.

[20] SIRIN E, PARSIA B, WU Dan, *et al*. HTN planning for Web service composition using SHOP2[J]. Web Semantics: Science, Services and Agents on the World Wide Web, 2004, 1(4): 377-396.

[21] KLUSCH M, GERBER A, SCHMIDT M. Semantic Web service composition planning with OWLS-Xplan[C]//Proc of AAAI Fall Symposium on Semantic Web and Agents. [S. l.]: AAAI Press, 2005.

[22] KUZU M, CICEKLI N K. Dynamic planning approach to automated Web service composition[J]. Applied Intelligence, 2012, 36(1): 1-28.

[23] OKUTAN C, CICEKLI N K. A monolithic approach to automated composition of semantic Web services with the event calculus[J]. Knowledge-based Systems, 2010, 23(5): 440-454.

[24] NARAYANAN S, SHEILA A. Adapting GOLOG for composition of semantic Web services[C]//Proc of the 11th International Conference on World Wide Web. 2002: 77-88.

[25] AYDIN O, CICEKLI N K, CICEKLI I. Automated Web services composition with the event calculus[C]//Proc of the 8th International Conference on Engineering Societies in the Agents World Ⅷ. Berlin: Springer, 2008: 142-157.

[26] BANSAL S K, BANSAL A, GUPTA G. Semantics-based Web service composition engine[C]//Semantic Web Services. Berlin: Springer, 2012: 329-343.

[27] Web services challenge 2010[EB/OL]. (2010-11-10)[2013-10-01]. http://ws-challenge.georgetown.edu/wsc10/.

[28] YAN Yu-hong, CHEN Min, YANG Yu-bin. Anytime QoS optimization over the PlanGraph for Web service composition[C]//Proc of the 27th Annual ACM Symposium on Applied Computing. New York: ACM Press, 2012: 1968-1975.

[29] 邓水光, 黄龙涛, 吴斌, 等. 一种 QoS 最优的语义 Web 服务自动组合方案[J]. 计算机学报, 2013, 36(5): 1015-1029.

[30] XU Bin, LUO Sen, YAN Yi-xin, *et al*. Towards efficiency of QoS-driven semantic Web service composition for large-scale service-oriented systems[J]. Service Oriented Computing and Applications, 2012, 6(1): 1-13.

[31] SHIN D H, LEE K H, SUDA T. Automated generation of composite Web services based on functional semantics[J]. Web Semantics: Science, Services and Agents on the World Wide Web, 2009, 7(4): 332-343.

[32] KARAKOC E, SENKUL P. Composing semantic Web services under constraints[J]. Expert Systems with Applications, 2009, 36(8): 11021-11029.

[33] YANG Jian, PAPAZOGLOU M P. Service components for managing the life-cycle of service compositions[J]. Information Systems, 2004, 29(2): 97-125.

[34] SHIAA M M, FLADMARK J O, THIELL B. An incremental graph-based approach to automatic service composition[C]//Proc of IEEE International Conference on Services Computing. [S. l.]: IEEE Press, 2008: 397-404.

[35] RUSSELL S J, NORVIG P. Artificial intelligence: a modern approach[M]. 3rd ed. [S. l.]: Pearson Education, 2011.

[36] 陈世展. 服务网络: 基于语义和社会化关系的 Web 服务计算基础设施[D]. 天津: 天津大学, 2009.

[37] 冯志勇, 陈世展, 王辉. 服务网络[J]. 中国计算机学会通讯, 2010, 6(9): 26-31.

[38] LEVESQUE H J, REITER R, LESPERANCE Y, *et al*. GOLOG: a logic programming language for dynamic domains[J]. Journal of Logic Programming, 1997, 31(1): 59-83.

[39] SHANAHAN M. Abductive event calculus planners[J]. Journal of Logic Programming, 2000, 44(1-3): 207-240.

[40] JPlan: Java GraphPlan implementation[CP/OL]. (2004-03-30)[2013-10-01]. http://sourceforge.net/projects/jplan/.

[41] HOFFMANN J, NEBEL B. The FF planning system: fast plan generation through heuristic search[J]. Journal of Artificial Intelligence Research, 2001, 14(1): 253-302.

[42] SAPENA O, ONAINDIA E. Planning in highly dynamic environments: an anytime approach for planning under time constraints[J]. Applied Intelligence, 2008, 29(1): 90-109.

[43] KLUSCH M, RENNER K U. Fast dynamic re-planning of composite OWL-S services[C]//Proc of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology. Washington DC: IEEE Computer Society, 2006: 134-137.

[44] TANG Xian-fei, JIANG Chang-jun, ZHOU Meng-chu. Automatic Web service composition based on Horn clauses and Petri nets[J]. Expert Systems with Applications, 2011, 38(10): 13024-13031.

(下转第 1311 页)

效性和优越性,因此,本实验主要进行 MMCM、AMMCM 算法与 LICM、ECM、MCM 算法之间的比较分析。推荐质量的评价标准同样采用平均绝对偏差(MAE)[7]来衡量,即 MAE 值越小,预测越准确,推荐质量越高。实验计算结果如图 4 所示。

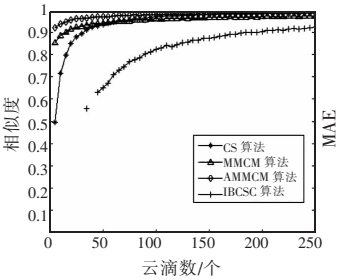

图 3　云滴数对各算法所计算出云模型间相似度的精度影响

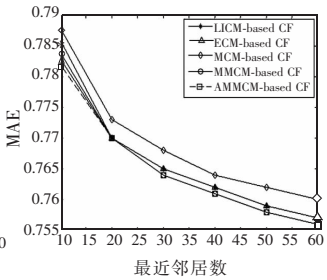

图 4　最近邻居数对基于各相似度算法的 CF 的 MAE 值的影响

从图 4 中易发现五种算法的 MAE 值随最近邻居数变化的趋势大体一致,MMCM 与 AMMCM 算法的 MAE 值基本上小于其他三种算法的 MAE 值,说明基于本文所提算法的协同推荐质量总体优于基于 LICM、ECM、MCM 算法的效果。MMCM、AMMCM 算法除在邻居数为 10 时 MAE 值有所差异,其他位置完全一致。由以上实验效果验证了 MMCM 与 AMMCM 算法在面对实际数据集时的可行性和有效性。

4　结束语

基于云模型的相似度量方法的改进不仅从理论上丰富了云模型体系,在应用上更有其实际意义。本文所提的 MMCM 和 AMMCM 算法是在分析和联系模糊数学理论与云理论的基础上提出的一种基于云间贴近度的相似度量分析方法。两种算法均具有形式简单、精度高的优良特性,且能在云滴数极少的情况下依然保持算法的高精确性。算法不仅适用于传统正态云模型,而且适用于其他任何形式云如半云、组合云等。最后通过实验验证了 MMCM 和 AMMCM 算法在对正态云模型描述方面的可行性和有效性,取得了良好效果。

云模型是一种处理定性与定量之间转换的良好模型。如何应用相似云及其相似度量分析方法将是未来的一个发展方向,改进现有相似度量分析方法亦是今后的工作要点。

参考文献:

[1] 苗夺谦,李德毅,姚一豫,等. 不确定性与粒计算[M]. 北京:科学出版社,2011:1-25.

[2] 刘常昱,李德毅,杜鹢,等. 正态云模型的统计分析[J]. 信息与控制, 2005,34(4):236-239.

[3] 张勇,赵东宁,李德毅. 相似云及其度量分析方法[J]. 信息与控制,2004,33(2):129-132.

[4] 蔡绍滨,方伟,赵靖,等. 基于区间的云相似度比较算法的研究[J]. 小型微型计算机系统,2011,32(12):2456-2460.

[5] 李海林,郭崇慧,邱望仁. 正态云模型相似度计算方法[J]. 电子学报, 2011,39(11):2561-2567.

[6] 李德毅,孟海军,史雪梅. 隶属云和隶属云发生器[J]. 计算机研究与发展,1995,32(6):15-20.

[7] 张光卫,李德毅,李鹏,等. 基于云模型的协同过滤推荐算法[J]. 软件学报,2007,18(10):2403-2411.

[8] 李德毅,杜鹢. 不确定性人工智能[M]. 北京:国防工业出版社,2005:144-150.

[9] 胡宝清. 模糊理论基础[M]. 武汉:武汉大学出版社,2010:187-221.

[10] 刘常昱,冯芒,戴晓军,等. 基于云 X 信息的逆向云新算法[J]. 系统仿真学报,2004,16(11):2417-2421.

[11] LI De-yi,HAN Jia-wei,SHI Xue-mei,*et al*. Knowledge representation and discovery based on linguistic atoms[J]. Knowledge-based Systems,1998,10(7):431-440.

[12] LI De-yi. Knowledge representation in KDD based on linguistic atoms [J]. Journal of Computer Science and Technology, 1997, 12 (6):481-491.

(上接第 1301 页)

[45] LÉCUÉ F. Optimizing QoS-aware semantic Web service composition [C]//Proc of the 8th International Semantic Web Conference. Berlin:Springer,2009:375-391.

[46] OMG. Unified modeling language(UML)[EB/OL]. (2007-11-04) [2013-10-01]. http://www.omg.org/spec/UML/2.1.2/Infrastructure/PDF.

[47] KAVANTZAS N, BURDETT D, FLETCHER T, *et al*. Web services choreography description language[EB/OL]. (2005-11-09) [2013-10-01]. http://www.w3.org/TR/ws-cdl-10/.

[48] CHINOSI M,TROMBETTA A. BPMN:an introduction to the standard [J]. Computer Standards & Interfaces,2012,34(1):124-134.

[49] YEUNG W L. A formal and visual modeling approach to choreography based Web services composition and conformance verification[J]. Expert Systems with Applications,2011,38(10):12772-12785.

[50] FOSTER H,UCHITEL S,MAGEE J,*et al*. Model-based verification of Web service compositions[C]//Proc of the 18th IEEE International Conference on Automated Software Engineering. 2003:152-161.

[51] SHENG Q Z,MAAMAR Z,YAO Li-na,*et al*. Behavior modeling and automated verification of Web services[J]. Information Sciences, 2014,258(2):416-433.

[52] BENTAHAR J,YAHYAOUI H,KOVA M,*et al*. Symbolic model checking composite Web services using operational and control behaviors [J]. Expert Systems with Applications,2013,40(2):508-522.

[53] NI Yue,FAN Yu-shun. Model transformation and formal verification for semantic Web services composition[J]. Advances in Engineering Software, 2010,41(6):879-885.

[54] KAZHAMIAKIN R,PISTORE M. Static verification of control and data in Web service compositions[C]//Proc of International Conference on Web Services. 2006:83-90.

[55] DUMEZ C,BAKHOUYA M,GABER J,*et al*. Model-driven approach supporting formal verification for Web service composition protocols [J]. Journal of Network and Computer Applications,2013,36 (4):1102-1115.

[56] CAMBRONERO M E,DIAZ G,VALERO V,*et al*. Validation and verification of Web services choreographies by using timed automata [J]. Journal of Logic and Algebraic Programming,2011,80(1): 25-49.

[57] RAVN A P,SRBA J,VIGHIO S. Modelling and verification of Web services business activity protocol[C]//Proc of the 17th International Conference on Tools and Algorithms for the Construction and Analysis of Systems. Berlin:Springer,2011:357-371.

[58] TAN Wei,FAN Yu-shun,ZHOU Meng-chu. A Petri net-based method for compatibility analysis and composition of Web services in business process execution language[J]. IEEE Trans on Automation Science and Engineering,2009,6(1):94-106.