一种基于业务流程执行描述语言的分布式 Web 服务发现方法*)

胡建强[1,2] 廖桂平[1]

(湖南农业大学信息科学与技术学院 长沙 410148) (国防科学技术大学计算机学院 长沙 410073)

摘 要 现有的 Web 服务发现方法主要是基于集中式拓扑,并且使用的服务描述仅仅是描述服务接口功能而忽略了业务流程执行相关信息,无法保证 Web 服务组合的相容性,也无法满足组合事务无死锁的要求。同时,这些方法都无法避免高维护代价、单点失效和可扩展性差等问题。本文提出一种基于结构化对等网络的 Web 服务发现方法,引入标识确定性有限状态机 ADFSA(Annotated Deterministic Finite State Automata),将服务业务流程执行语言用于服务匹配,从而避免了潜在的 Web 服务组合不相容和有利于服务组合的自动化。

关键词 服务组合的相容性,标识确定性有限状态机 ADFSA,服务匹配

A Distributed Web Services Discovery Method Based on Business Process Execution Language

HU Jian-Qiang[1,2] LIAO Gui-Ping[1]

(School of Information Science and Technology, Hunan Argricular University, 410148)[2]

(School of Computer Science, National University of Defense Technology, Changsha 410073)[1]

Abstract Current Web service discovery methods are based on centralized approaches where Web services are described with service interface functionality but not business process-related information. It cannot guarantee compatibility of Web service composition, nor can it make Web services easy to complete a deadlock free and bounded process transaction. Furthermore, centralized approaches to service discovery suffer from problems such as high operational and maintenance cost, single point of failure, and bad scalability. Therefore, this paper presents a structured Peer-to-Peer network for Web service discovery in which Web services are located based on service interface functionality and process behavior. It guarantees semantic compatibility of Web service composition, and achieves the automated composition at the level of executable processes.

Keywords Compatibility of Web service composition, Annotated deterministic finite state automata, Service match

1 引言

开放的网络化应用和"软件作为服务"的理念必将导致基于 Internet 环境下软件系统的主要形态、运行方式、生产方式和使用方式发生巨大的变化。未来网络软件开发的一种趋势表现为构造若干 Web 服务动态组合、目标渐趋稳态的软件应用系统[1]。服务组合定义为由各个小粒度的 Web 服务相互通信和协作来实现大粒度的组合服务(Composite Service)。过程模型(如 BPEL4WS,OWL-s)作为目前 Web 服务组合的主流方法之一,要求对过程模型的每一状态,都需要动态发现并绑定合适的(suitable)Web 服务。因此,Web 服务发现作为面向服务体系结构的一个重要组成部分,对能否真正实现跨组织的自动交互、成功组合具有极其重要的意义。

针对 Web 服务方法,目前国内外许多著名的大学和研究机构开展了非常积极有益的探索,主要表现在 Web 服务描述语言和系统拓扑两个方面[2]。

(1)Web 服务描述语言:由单独关注功能或过程描述,向功能和过程兼具的方向发展。

WSDL[3]是工业界广泛采用的标准,充分描述了 Web 服务的接口功能但忽视服务的过程行为信息;BPEL4WS[4]作为 Web 服务的业务执行语言,着重描述 Web 服务的过程行为而忽视服务的内部功能描述;OWL-s[5]是基于本体 OWL 的 Web 服务描述语言,采用明确的、计算机可以理解的语言标识来描述 Web 服务,试图全方位描述服务的功能和过程行为。

(2)系统拓扑:由集中式拓扑,向可扩展、容错性好的结构化或非结构化 Peer-to-Peer 网络发展。

IBM、Microsoft 等公司采用 WSDL 和预定义分类系统实现的 UDDI 系统[6]、卡内基·梅隆大学采用 DAML-s(OWL-s 的前版本)的实现 augment UDDI Registry[7]系统都基于集中式拓扑,虽然简单、易于实现而且数据定位快,但无法避免高维护代价、容易单点失效和可扩展性差等问题。台湾省国立清华大学 Neuron 项目、卡内基·梅隆大学 SpeedR[8]分别基于结构化拓扑 Tapestry 网络和无结构 Gnutella 网络,采用 WSDL 描述和关键字精确匹配 Web 服务。但目前尚未见到采用过程描述,并基于 Peer-to-Peer 网络的 Web 服务发现方法。

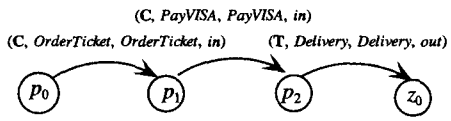
总体来说,现有的 Web 服务发现的方法过于依赖集中式拓扑,同时采用的服务描述只描述服务接口功能而忽略过程相关信息,这无法保证服务组合上下文无死锁和满足服务组合的相容性的要求。针对这一问题,本文提出一种基于结构化 Peer-to-Peer 网络的 Web 服务发现方法,引入标识确定性

*)国家自然科学基金资助项目(No. 90104020)、湖南省自然科学基金(No. 04JJ3021)和湖南高教基金(No. 04C290)。胡建强 博士、副教授,主要研究方向:Web 服务、中间件和网格计算。廖桂平 博士生导师,教授,研究方向:人工智能与农业信息化技术。

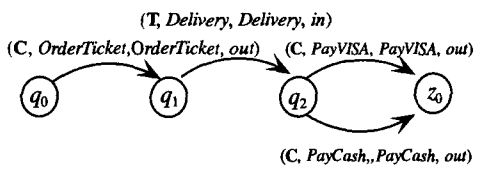
有限状态机 ADFSA(Annotated Deterministic Finite State Automata),将服务过程行为直接用于服务匹配,从而有效地避免了潜在的 Web 服务组合不相容。

2 Web 服务相容性

为便于直观讨论服务组合的相容性,我们采用订票服务的业务过程予以说明服务之间的交互细节(包括 Ticket Service T 和 Customer Service C)[9]。设 Ticket Service 有两个输入接口(OrderTicket、PayVISA)和一个输出接口(Delivery);Customer Service 有两个输出接口(OrderTicket、PayVISA)和一个输入接口(Delivery)。设图中节点表示状态,边表示状态转换和标识(from, porttype, operation, direction)表示消息,其中 from 表示消息发送者,porttype, operation 的语义同 WSDL 文档的 portType, operation 元素,direction 表示消息的方向,取值为 in 或 out;采用"∨"和"∧"分别表示可选消息和强制消息。



(a) Ticket Service 的消息序列



(b) Customer Service 的消息序列带有可选消息

(C, *PayVISA*, *PayVISA*, *out*) ∨ (C, *PayCash*, *PayCash*, *out*)

图 1 Ticket Service 和 Customer Service 交互的消息序列图

图 1(a)给出 Ticket Service 的消息序列:Ticket Service 接收到来自 Customer Service 发出订票请求(C, *OrderTicket*, *OrderTicket*, *in*);Ticket Service 发出送票消息(T, *Delivery*, *Delivery*, *out*)当且仅当收到 Customer Service 的支付消息(C, *PayVISA*, *PayVISA*, *in*)。图 1(b)给出 Customer Service 的消息序列:Customer Service 向 Ticket Service 发出订票请求(C, *OrderTicket*, *OrderTicket*, *out*);Customer Service 收到送票消息(T, *Delivery*, *Delivery*, *in*)才发出的支付消息(C, *PayVISA*, *PayVISA*, *out*)或(C, *PayCash*, *PayCash*, *out*)。

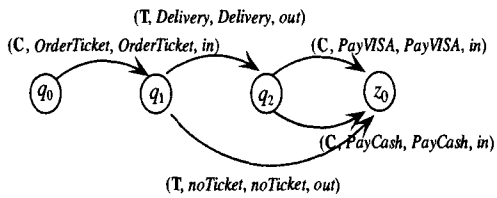
(a)Ticket Service 的消息序列带有强制消息(T, Delivery, Delivery, out) ∧ (T, noTicket, noTicket, out)和可选消息(C, PayCash, PayCash, in) ∨ (C, PayCash, PayCash, in)

(b_0) Customer Service 的消息序列要求可选消息(C, PayVISA, PayVISA, out) ∨ (C, PayCash, PayCash, out)

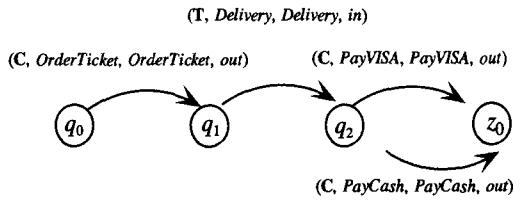
(b_1)Customer Service 的消息序列要求强制消息(T, Delivery, Delivery, in) ∧ (T, noTicket, noTicket, in)

(b_2)Customer Service 的消息序列要求强制消息(T, Delivery, Delivery, in) ∧ (T, noTicket, noTicket, in)

图 2(a)给出 Ticket Service 的消息序列:Ticket Service 收到来自 Customer Service 发出订票请求(C, *OrderTicket*, *OrderTicket*, *in*);Ticket Service 发出送票消息(T, *Delivery*, *Delivery*, *out*)当且仅当收到 Customer Service 的支付消息(C, PayVISA, PayVISA, *in*)或(C, *PayCash*, *PayCash*, *in*)。如果票已售完,Ticket Service 必须使用消息(T, *noTicket*, *noTicket*, *out*)拒绝。图 2(b_0)给出 Customer Service 的消息序列,因为其无法支持强制消息(T, *noTicket*, *noTicket*, *out*),因此 Ticket Service 和 Customer Service 是无法成功交互的。相反地,图 2(b_0)、(b_1)可以处理强制消息(T, *Delivery*, *Delivery*, *in*) ∧ (T, *noTicket*, *noTicket*, *in*),同时支持任一支付消息(T, *Delivery*, *Delivery*, *in*)或(C, *PayCash*, *PayCash*, *out*),因此 Ticket Service 与 Customer Service 成功交互。

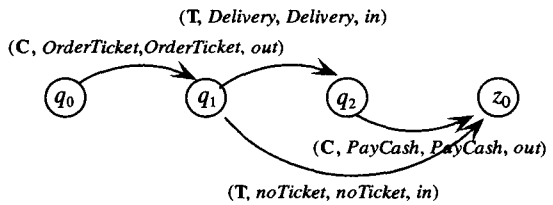


(a) Ticket Service 的消息序列带有强制消息(T, *Delivery*, *Delivery*, *out*) ∧ (T, *noTicket*, *noTicket*, *out*)和可选消息(C, *PayCash*, *PayCash*, *in*) ∨ (C, *PayCash*, *PayCash*, *in*)



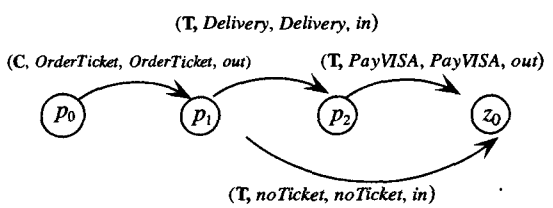
(b_0) Customer Service 的消息序列要求可选消息

(C, *PayVISA*, *PayVISA*, *out*) ∨ (C, *PayCash*, *PayCash*, *out*)



(b_1) Customer Service 的消息序列要求强制消息

(T, *Delivery*, *Delivery*, *in*) ∧ (T, *noTicket*, *noTicket*, *in*)



(b_2) Customer Service 的消息序列要求强制消息

(T, *Delivery*, *Delivery*, *in*) ∧ (T, *noTicket*, *noTicket*, *in*)

图 2 Ticket Service 和 Customer Service 带有强制或可选消息的交互消息序列图

上述例子说明,判断两个服务是否可以成功交互,需要考虑完整的消息序列和强制或可选消息类型。我们称这种成功交互为组合相容。一般地,两个服务的组合相容可以分为语法相容和语义相容。

定义 1(服务组合语法相容) 设两个 Web 服务 X 和 Y 组合语法相容,当且仅当满足:服务 X 的输入接口是服务 Y

的输出接口，服务 X 的输出接口是服务 Y 的输入接口。

设 Ticket Service 有两个输入接口(OrderTicket, PayVISA)和一个输出接口(Delivery)，Customer Service 有两个输出接口(OrderTicket, PayVISA)和一个输入接口(Delivery)。由定义知，Ticket Service 和 Customer Service 是完全语法相容的。图 1 显示这两个服务在个体消息即不考虑消息序列的情况下确实是相容的。事实上，存在顺序完全相反的消息序列因而无法成功交互的情况。Ticket Service 在状态[$p1$]等待收到 *PayVISA* 消息之后发出 Delivery 消息，同时 Customer Service 在[$q1$]等待收到 Delivery 消息才发出 *PayVISA* 消息，两者相互死锁而无法成功组合。因此，仅仅保证两个服务接口功能的语法相容是不够的，需要考虑整个消息序列。

定义 2(服务组合语义相容) 设两个 Web 服务 X 和 Y 组合语义相容，当且仅当满足：

(1)X 的输入接口是 Y 的输出接口，同时 Y 的输入接口是 X 的输出接口；

(2)对于每个服务的任意可到达的消息状态[p]，都可以到达最终消息状态[z]；

(3)若存在可到达消息状态[q]，满足[q]≥[z]都有[q]=[z]。

关于可到达消息状态的定义，文[10]给出更充分的定义。对于图 2 中的(a)和(b_0)是语法相容而不是语义相容，因为 Customer Service 无法处理强制消息。为保证一个成功的业务交互，必须考虑整个消息序列而不是消息个体。对图 2 中的(a)和(b_1)是语义相容，因为共享相同的消息序列包括强制消息在内，避免内在的冲突达到自动组合的目的。

3 标识确定性有限状态机

有限状态机模型可以清晰地描述服务组合双方的状态以及状态间的转换关系。我们采用确定性有限状态机 DFSA (Deterministic Finite State Automata)[11,12]对 Web 服务建模。

定义 3(确定性有限状态机 DFSA) 设 Web 服务 X 是一个元组 $DFSA_x=\langle Q,\Sigma,f,q_0,Z\rangle$，其中

(1)Q 是一个有限状态集合；

(2)$\Sigma\subseteq PL\times PT\times OP\times\{in,out\}$是一个有限消息集，每一元素由集合 PL 中的发送者×portType 集合 $PT\times$operation 集合 $OP\times\{in,out\}$；

(3)$f:Q\times\Sigma\to Q$是一个函数，负责将(state, message)映射到另一个状态 state；

(4)$q_0\in Q$是开始状态；

(5)$Z\subseteq Q$是最终状态集。

DFSA 模型并不能完全解决下面的需求：发送方在某一特定状态发送的强制消息，接收方必须支持。一般情况下，发送方有选择地发送消息，而接收方必须能够处理所有的可选消息。这是因为 DFSA 模型中并没有区分强制消息和可选消息。为避免这一情况，引入标识集是完全必要的。为此构造如下标识确定性有限状态机 ADFSA(Annotated Deterministic Finite State Automata)。

定义 4(标识确定性有限状态机 ADFSA) 设 Web 服务 X 是一个元组 $ADFSA_x=\langle Q,\Sigma,f,q_0,Z,L\rangle$，其中

(1)Q 是一个有限状态集合；

(2)$\Sigma\subseteq PL\times PT\times OP\times\{in,out\}$是一个有限消息集，每一元素由集合 PL 中的发送者×portType 集合 $PT\times$operation 集合 $OP\times\{in,out\}$；

(3)$f:Q\times\Sigma\to Q$是一个函数，负责将(state, message)映射到另一个状态 state；

(4)$q_0\in Q$是开始状态；

(5)$Z\subseteq Q$是最终状态集；

(6)L 是一个标识集，采用逻辑术语$\{\vee,\wedge\}$来表示状态之间的关系。

ADFSA 模型不仅易于表达包括强制消息和可选消息在内的消息序列，并为自动服务组合提供一个合理的基石。例如，ADFSA 模型可以方便地描述图 2(b_1)中 Web 服务的业务抽象流程：

$\langle\{q_0,q_1,q_2,z_0\},\{(C,OrderTicket,OrderTicket,out),(T,Delivery,Delivery,in)\wedge(T,noTicket,noTicket,in),(C,PayCash,PayCash,out)\},f,q_0,z_0,\{\vee,\wedge\}\rangle$，其中 f 满足如下定义：

$f(q_0,(C,OrderTicket,OrderTicket,out))=q_1$；

$f(q_1,(T,Delivery,Delivery,in))=q_2$；

$f(q_1,(T,noTicket,noTicket,in))=z_0$；

$f(q_2,(C,PayCash,PayCash,out))=z_0$。

基于标识确定性有限状态机，可以通过服务匹配来判断一个 Web 服务是否满足请求服务的需要，即保证服务组合相容。对于服务匹配可作如下定义：

定义 5(精确匹配) 设 Web 服务 X 和 Web 服务 X'满足精确匹配当且仅当存在一个从 $ADFSA_x=\langle Q,\Sigma,f,q_0,Z,L\rangle$到 $ADFSA_{x'}=\langle Q',\Sigma',f',q'_0,Z',L'\rangle$的同构映射(同态、双射)$g:Q\mapsto Q'$，即 $g:Q\mapsto Q'$是同态映射，即对于$\forall q\in Q$和$\forall m\in\Sigma$，都有 $g(f(q,m))=f'(g(q),m)$；存在逆映射 $g^{-1}:Q'\mapsto Q$。形式上，记作 Exact($ADFSA_x$, $ADFSA_{x'}$)。

按照定义 5，同构映射要求两个服务具有结构完全相同状态图。状态可能有不同的名字，如果重新标识，两个状态图是完全一致的。显然，精确匹配过于严格而不实用，事实上可替代匹配(PlugIn Match)作为精确匹配的一种特例，完全可以满足服务组合的要求。

定义 6(可替代匹配) 设 Web 服务 X 和 Web 服务 X'满足可替代匹配当且仅当存在一个从 $ADFSA_x=\langle Q,\Sigma,f,q_0,Z,L\rangle$到 $ADFSA_{x'}=\langle Q',\Sigma',f',q'_0,Z',L'\rangle$仿真(Simulation)映射 $g:Q\mapsto Q'$，即对于$\forall q'\in Q'$和$\forall m\in\Sigma$，都有 $f(g(q'),m)=f'(q',m)$。形式上，记作 PlugIn($ADFSA_x$, $ADFSA_{x'}$)。

上述定义用于直接判断两个服务是否满足精确匹配或可替代匹配是非常困难的。一种办法是计算两个标识确定性有限状态机的交集来判断 Web 服务 X 和 Web 服务 X'的相关性。

定义 7(两个 ADFSA 的交集) 设 Web 服务 X 和 Web 服务 X'的交集，$ADFSA_x=\langle Q,\Sigma,f,q_0,Z,L\rangle$，$ADFSA_{x'}=\langle Q',\Sigma',f',q'_0,Z',L'\rangle$的交集 $X\cap X'=\langle Q^*,\Sigma^*,f^*,q_0^*,Z^*,L^*\rangle$，其中 $Q^*{}'=Q\times Q'$，$\Sigma^*=\Sigma\cap\Sigma'$，$f^*((q_{11}^*,q_{21}^*),\alpha)=(q_{12}^*,q_{22}^*)$，$f(q_{11}^*,\alpha)=q_{12}^*\wedge f'(q_{21}^*,\alpha)=q_{22}^*$，$q_0=(q_{10},q_{20})$，$Z^*{}'=Z\times Z'$，

$$L^*=\bigcup_{q_1\in Q,q_2\in Q}\begin{cases}((q_1,q_2),e_1\wedge e_2),\text{若}(q_1,e_1)\in A,(q_2,e_2)\in A', & A\subseteq L,A'\subseteq L'\\ \varnothing, & \text{其它}\end{cases}$$

通过计算两个 ADFSA 的交集比较服务相关性和匹配程度是比较直观的，但计算的实施过于复杂。毕竟对 Web 服务

组合来说，最关注的是双方的消息序列是否能够相容。根据消息序列和标识确定性有限状态机，可以找到一种简单、直接、有效的方法来查找合适的服务，以保证服务组合相容。由标识确定性有限状态机，可以获取Σ集合中的消息序列。

定义 8（可到达路径有限状态机） 一个 Web 服务如果定义为一个交互的消息序列，则一条由初始状态到最终状态的可执行路径称为可到达路径有限状态机 RPFA（Reachable Path Finite Automaton）。

两个服务组合相容的必要条件是至少共享一条可到达路径有限状态机。图 2(a)的 Ticket Service 存在两条可到达路径有限状态机：

$\langle OrderTicket, Delivery \wedge noTicket, PayVISA\rangle$；

$\langle OrderTicket, Delivery \wedge noTicket, PayCash\rangle$

其中“∧”表示强制消息。为保证成功交互，潜在的 Customer Service 应该可替代匹配 Ticket Service。图 2(b_1)存在一条有效的交互消息序列（可到达路径有限状态机）$\langle OrderTicket, Delivery \wedge noTicket, PayCash\rangle$，图 2($b_2$)存在可到达路径有限状态机$\langle OrderTicket, Delivery \wedge noTicket, PayVISA\rangle$，都满足 PlugIn（Customer Service，Ticket Service），即 Ticket Service 与 Customer Service（图 2(b_1)或图 2(b_2)）可共享一条可到达路径有限状态机，因而可以成功组合。

4 Web 服务过程执行语言匹配

上节通过将 Web 服务建模为标识确定性有限状态机，简化了服务匹配的复杂性并且有利于服务发现的自动化。与 Petri 网、工作流网（workflow nets）或状态图相比较，标识确定性有限状态机拥有更低的复杂性。但标识确定性有限状态机的复杂性、抽象层次，还不足以让用户放弃 BPEL4WS、OWL-s ServiceModel 等业务流程描述语言而直接采用标识确定性有限状态机。目前 BPEL4WS 的抽象业务流程描述和标识确定性有限状态机可以解释为消息序列的描述，都强调发送方在某一状态发送的消息，接受方必须支持。若能将 BPEL4WS 映射为标识确定性有限状态机，则基于 Web 服务业务流程语言的直接服务匹配问题就可以简化为可到达路径的匹配问题，从而简化匹配的复杂性。

BPEL4WS 的每个活动被递归转换成分结构和子活动的分结构。每个分结构必须有一个输入状态 q_{in} 进入分结构和一个输出状态 q_{out} 离开分结构。这个分结构可以映射成部分标识确定性有限状态机 PA（Partial Annotated Deterministic Finite State Automata）。

定义 9（部分标识确定性有限状态机） 设 Web 服务 X 是一个元组 $ADFSA_x=\langle Q,\Sigma,f,q_0,Z,L\rangle$，采用消息序列上关联的一个输入状态 $q_{in}\in Q$ 和一个输出状态 $q_{out}\in Q$ 代替由初始状态 q_0 开始的部分标识确定性有限状态机 $PA=\langle Q,\Sigma,f,q_{in},q_{out},Z,L\rangle$。

定义 10（状态重命名） 设状态 q 重新命名成状态 q'，即

$$f(\bar{q},q\to q')=\begin{cases}q', & \bar{q}=q\\ \bar{q}, & \text{其它}\end{cases}$$

定义 11（& 运算） 设 $Q=Q_1\times Q_2$，$\Sigma=\Sigma_1\times\Sigma_2$，$q_{in}=q_{in,1}\times q_{in,2}$，$q_{out}=q_{out,1}\times q_{out,2}$，$F=F_1\times F_2$，$\delta=\{((p,q_1),\alpha,(p,q_2))\in((Q_1\times Q_2)\times\Sigma_2\times(Q_1\times Q_2))\}\cup\{((p,q_1),\alpha,(q_2,p))\in(Q_1\times Q_2)\times\Sigma_1\times(Q_1\times Q_2)\}$，$L=\bigcup_{(q_1,e_1)\in L_1,(q_2,e_2)\in L_2}((q_1,q_2),e_1\wedge e_2)$，则 $PA_1\&PA_2=\{Q,\Sigma,\delta,q_{in},q_{out},Z,L\}$。

BPEL4WS 是基于通信活动结构的，而 ADFSA 是基于服务组合双方的消息交换。幸运的是，由 partnerlink（隐含指定 portType 的角色）、portType 和 operation 元素组成的每个通信活动，很容易映射成 ADFSA 的抽象消息形式（pl,pt,op,in）或（pl,pt,op,out），即每个活动 activity 都可以映射成终止状态为 q_{out}，开始状态为 q_{in} 的部分标识确定有限状态机 PA。

附表 1 给出通信活动（包括 reply，receive 和 invoke）、结构活动（包括 sequence，while，switch 和 flow）具体的映射语法。BPEL4WS 到 ADFSA 的映射是一个由上层容器元素〈process〉到部分标识确定性有限状态机 PA 的递归映射过程。

5 基于结构化网络的服务发现方法

构造的 Web 服务发现仿真系统由一个或多个服务注册节点（注册器的实际控制者）组成，服务注册节点之间的连接拓扑构成 Chord 网络。之所以选择 Chord，是因为它实现简单而且具有非常好的系统特性，可以较好地避免集中式拓扑所固有的高维护代价、单点失效和可扩展性差等问题。

5.1 Chord 网络特性

Chord[13] 网络是一种典型的结构化拓扑（Structured Topology），结构化拓扑的优势在于对象定位机制较小的网络开销和确定性的定位上界。基于 Kargar 的一致性 Hashing，将任何 Key（可到达路径有限状态机）和实际的注册器映射到一个周长为 2^m 的逻辑标识环上，m 的大小使环足以容纳最大可能的参与注册器的数目，同时使得任何两个 Key 或节点映射到同一环上的概率小到可以忽略。在逻辑标识环上引入后继的概念，即环的下一个实际存在的注册节点，将相关的索引放置到其 Key 对应的后继节点中。因此，Chord 保证在节点都知道其后继的情况下，对于任一 Key 沿着后继链，确定性地找到 Key 相应存放的注册节点。

5.2 服务发布

首先将 Web 服务的流程描述语言映射为标识确定性有限状态机 ADFSA，然后提取可到达路径有限状态机 RPFA 并利用哈希函数发布到服务注册节点上。这是因为大多情况下，服务请求方是不知道完全的 ADFSA，而且只关注查找到的服务是否能够保证服务双方组合相容。一个服务如果存在 n 条可到达路径有限状态机，那么它可以被发布 n 次。按照 Chord 拓扑特性，每条可到达路径有限状态机被当被一个 Key，相应的注册器就负责存储和管理该可到达路径有限状态机对应的服务流程执行描述或完整的可到达路径有限状态机集合。

5.3 服务查找

首先将请求服务的流程行为解析为可到达路径有限状态机 RPFA，利用哈希函数得到相应的 Key 值，从而定位到合适的注册节点 RP（Registry Peer），如图 3 所示。每个注册节点 RP 负责维护和查询底层数据库，并由通信引擎（Communication Engine）和本地查询引擎（Local Query Engine）组成。通信引擎负责接收和应答查询消息，同时负责与 Chord 系统的邻居注册节点的协同；查询引擎负责解析请求并与注册器保存的可到达路径有限状态机集合匹配。若满足可替代匹配，说明找到的 Web 服务可与请求服务至少共享一条可到达路径有限状态机，一定程度上可以保证服务双方可以成功交互；否则无法找到与请求服务组合相容的 Web 服务，服务双方可能因死锁而无法成功组合。

为了评价基于流程的Web服务发现方法的有效性，基于LTSA（Labeled Transition System Analyzer）[14]实现从BPEL4WS到ADFSA的映射工具和基于流程的Web服务发现仿真系统[9]。为了保证实验的真实性，随机从Internet搜索Web服务流程描述并映射为ADFSA。因为Chord网络的良好可扩展性，这里只关注系统的延迟（latency）和查准率（precision）。

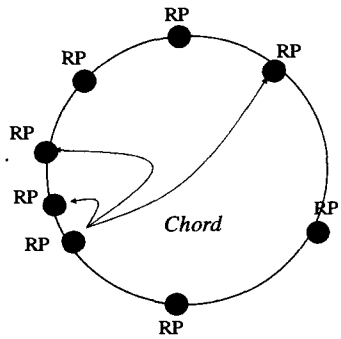


图3 基于Chord网络的注册器连接拓扑示意图

(1)延迟

图4表明了BPEL4WS描述文件数量对系统延迟的关系。当系统的节点设定为2000时，查询路由的开销是O(log(2000))跳步数(hops)。随着BPEL4WS的数量的增长，系统延迟也呈现上升趋势。当BPEL4WS文件数为200时，系统延迟是32.1s；当BPEL4WS的文件数是900时，系统的延迟是50.8s。图像并没有呈现出均匀增长的趋势，这是因为Web服务流程执行描述的复杂性不一样，导致映射BPEL4WS和提取RPFA需要花费的时间存在很大的差异。

(2)查准率

查准率是指能查找与请求服务相容的Web服务的成功率。图5表明了注册器数目对查准率的影响是比较平稳定的(91%～97%)。当注册器数目为2000时，查准率高达96.3%；当注册器数目增加到5000时，查准率为91.2%。这也说明查准率不太受注册器数目的影响，实验表明，这种基于流程行为的服务发现方法是比较高效的，同时查准率也能保证服务组合的需求。

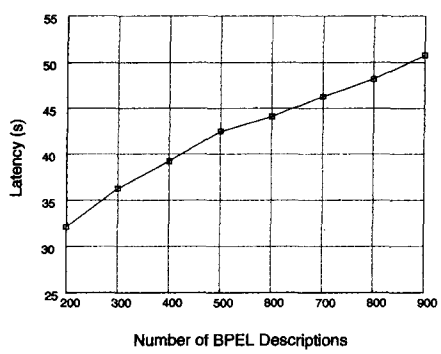


图4 延迟与BPEL4WS描述的文件数量的关系

与基于关键字匹配的UDDI系统SUN JAXR Registry(平均查准率是43%左右和平均延迟在80ms)和基于服务功能描述QWSDL语言的StarWSDS系统(平均查准率在84%左右，平均延迟在324ms)相比，基于服务流程描述的Web服务发现方法在取得更高服务查准率的同时带来了更大的系统延迟。这说明两方面的情况：(1)通过对可到达路径有限状态机RPFA的服务匹配，更能保证服务双方能组合相容和交互成功；(2)系统延迟还不是很理想，尽管服务匹配更直接和简单，但需要BPEL4WS到ADFSA的实时映射以及在提取RPFA的算法上消耗更多的时间。

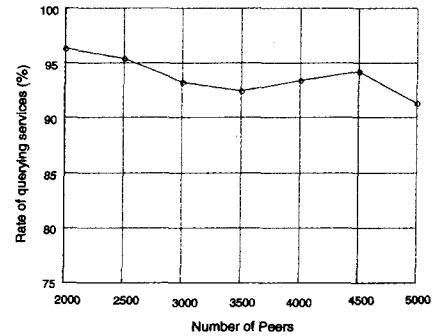


图5 查询成功率与注册器数目的关系

结论 现有的集中式Web服务发现方法，其可扩展性差、容易单点失效，并且过多的依赖于只描述Web服务功能的描述语言，因而无法保证服务组合的相容性。本文引入确定性有限状态机，将基于服务过程语言的服务匹配简化为可到达有限状态机的匹配问题，并且注册器连接拓扑采用基于P2P网络。实验结果表明，采用服务过程描述并基于Chord网络的服务发现方法是可行的、有效的，在很大程度上保证了服务组合的相容性。下一步的工作，包括提高OWL-s，BPEL4WS和Petri网转换成确定性有限状态机的效率和改进可到达路径有限状态机的提取算法等。

参考文献

1 杨芙清. 软件工程技术发展思索. 软件学报，2005,16(1)
2 胡建强，邹鹏，王怀民，等. Web服务描述语言QWSDL和服务匹配模型研究. 计算机学报，2005,28(4)
3 W3C. Web Service Description Language (WSDL) 1.1. http://www.w3.org/TR/wsdl, 2001
4 IBM. Business Process Execution Language for Web Services, Version 1.1. ftp://www6.software.ibm.com/software/developer/library/ws-bpel11.pdf, 2003.5
5 The OWL Services Coalition. Semantic markup for Web Services (OWL-s), Version 1.0, 2004
6 UDDI.org. UDDI Spec TC, Version 3.0.2, 2004. http://www.uddi.org/pubs/uddi_v3.htm
7 Massimo P, Kakahiro K, Payne Terry R, et al. Importing the Semantic Web in UDDI. In: Proc. of Web Services, E-business and Semantic Web Workshop, Tornoto, Canada, 2002. 225～236
8 Sivashanmugam K, Verma K, Mulye R, et al. SpeedR: Semantic P2P Environment for Diverse Web Services. Final Presention, University of Georgia, GA, 2002
9 Hu J, Guo C, Wang H, et al. Web Services Peer-to-peer Discovery Service for Automatic Web Service Composition. In: Proc. of the International Conference on Computer Networks and Mobile Computing (ICCNMC'05), Zhangjiajie, China, 2005
10 Martens A. On usability of Web Service. In: Proceedings of Fourth International Conference on Web Information Systems Engineering Workshops, Roma, Italy, December 2003
11 Wombacher A, Fankhauser P, Neuhold E J. Transforming BPEL into Annotated Deterministic Finite State Automata for Service Discovery. In: the Proceedings of International Conference on Web services, California, USA, 2004. 316～323
12 Wombacher A, Fankhauser P, Mahleko B, et al. Neuhold: Matchmaking for Business Process Based on Choreograhpies. In: Proceedings of International Conference on e-Technology, e-Commerce and e-Service, Taipei, Taiwan, March 2004
13 Stocia I, Morries R, Karger D, et al. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In: Proceedings of ACM SIGCOMM, 2001. 149～160
14 LTSA. http://www.doc.ic.ac.uk/ltsa/bpel4ws/ltsaeclipse031.zip

附表 1　BPEL4WDFSA 的映射语法

BPEL4WS 元素	主要形式	部分标识确定性有限状态机
process	⟨process⟩activity⟨/process⟩	ADFSA=$\langle Q,\Sigma,f,q_{in},Z\cup\{q_{out}\},L\rangle$ $PA=\langle Q,\Sigma f,q_{in},q_{out},Z\cup\{q_{out}\},L\rangle$
reply	⟨reply partnerLink="*pl*" portType="*pt*" operation="*op*"variable="*var*"/⟩ ⟨invoke partnerLink="*pl*"portType="*pt*" operation="*op*"inputVariable="*var*"/⟩	$PA=(\{s_0,s_1\},\{(pl,pt,op,in)\}\{(s_0,(pl,pt,op,in),s_1)\}$ $s_0,s_1,\varnothing,\{(s_0,(pl,pt,op,out)),s_1,true)\})$
receive	⟨reply partnerLink="*pl*"portType="*pt*" operation="*op*"variable="*var*"/⟩	$PA=(\{s_0,s_1\},\{(pl,pt,op,in)\}\{(s_0,(pl,pt,op,in),s_1)\}$ $s_0,s_1,\varnothing,\{(s_0,(pl,pt,op,out)),s_1,true)\})$
invoke	⟨invoke partnerLink="*pl*"portType="*pt*" operation = "*op*" inputVariable = "*var*"$_1$ outputVariable="*var*"$_2$/⟩	$PA=(\{s_0,s_1,s_2\},\{(pl,pt,op,in)\},(pl,pt,op,out)\},\{(s_0,(pl,pt,op,in),s_1)\}(s_1,(pl,pt,op,out),s_2)\},s_0,s_2,\varnothing,\{(s_0,(pl,pt,op,out)),s_1,true),(s_2,true)\})$
while activity	⟨while condition="*cond*"⟩PA_1⟨/while⟩	$PA=f(PA_1,q_{out,1}\rightarrow q_{in,1})$
sequence activity	⟨sequence⟩$PA_1,PA_2,\cdots,PA_n$⟨/squence⟩	$PA=PA_n\cup(\bigcup_{i=1}^{n-1}f(PA_i,q_{out,i}\rightarrow q_{in,i+1}))$
Flow actiity	⟨flow⟩$PA_1,PA_2,\cdots,PA_n$⟨flow⟩	$PA=f(f(\&_{i=1}^{n}PA_i,(q_{in,1},q_{in,2},\cdots,q_{in,n})\rightarrow q_{in}),(q_{out,1},q_{out,2},\cdots,q_{out,n})\rightarrow q_{out})$
switch actiity	⟨switch⟩ ⟨case condition="$cond_1$"⟩PA_1⟨/case⟩ ⟨case condition="$cond_2$"⟩PA_2⟨/case⟩ … ⟨case condition="$cond_n$"⟩PA_n⟨/case⟩ ⟨/switch⟩	$PA=\bigcup_{i=0}^{n}f(f(PA_i,q_{in}\rightarrow q_{in}),q_{out,i}\rightarrow q_{out})$
pick	⟨pick⟩ ⟨onMessage partnerLink="pl_1" portType="pt_1" operation="op_1" variable="var_1"⟩PA_1⟨/onMessage⟩ … ⟨onMessage partnerLink="pl_n" portType="pt_n" operation="op_n" varialbe="var_n"⟩PA_n⟨/onMessage⟩ ⟨onAlarm⟩PA_0⟨/onAlarm⟩ ⟨/pick⟩	$PA=\bigcup_{i=0}^{n}f(f(PA_i,q_{in}\rightarrow q_{in}),q_{out,i}\rightarrow q_{out})$

第二届中国 Agent 理论与应用学术会议(Agent2008)

征文通知

由中国计算机学会人工智能与模式识别专业委员会主办、南京大学承办、苏州大学协办的第二届中国 Agent 理论与应用学术会议定于 2008 年 4 月中下旬在江苏南京举行。本次会议将聚集国内从事 Agent 理论与应用的研究人员和工程技术人员，广泛开展学术交流，研究发展战略，共同促进 Agent 理论与技术的发展和应用。本次会议部分录用的论文将分别在《计算机研究与发展》、《模式识别与人工智能》、《南京大学学报》、《计算机科学》和《广西师范大学学报》正刊上发表。

1　征文范围(包括但不限于)

Agent 和多 Agent 结构；Agent 和多 Agent 系统的形式模型；基于 Agent 的软件工程与方法学；Agent 协商与协调；Agent 拍卖与电子市场 Agent 组织与联盟；Agent 通信和语言；Agent 学习与规划；Agent 系统的计算复杂性；多 Agent 系统环境与性能评价；Agent 仿真；人工社会系统移动；Agent；Agent 与网格计算；Agent 与数据挖掘；Agent 和多 Agent 系统应用；其他 Agent 理论与技术方面的内容。

2　投稿要求

论文未在其他会议或期刊发表过；论文应包括题目、中英文摘要、关键词、正文、参考文献等，参照《计算机研究与发展》的格式；投稿时发送电子邮件至 agent2008@nju.edu.cn，并请注明作者姓名、单位、通信地址、邮政编码、联系电话、电子邮件地址。欢迎省部级以上的基金资助课题组踊跃投稿。

3　重要日期　征文截止日期：2007 年 10 月 31 日　录用通知日期：2007 年 11 月 30 日

4　联系方式

通信地址：江苏省南京市鼓楼区汉口路 22 号 南京大学计算机科学与技术系

邮政编码：210093　联 系 人：王崇骏，商　琳　联系电话：(025)83593163；83686556

会议网址：http://cs.nju.edu.cn/agent2008；电子邮件：agent2008@nju.edu.cn