

Web 服务请求输入闭包与可满足性分析

叶恒舟, 罗晓娟, 牛秦洲

(桂林理工大学 信息科学与工程学院, 广西 桂林 541004)

摘 要: 大规模服务库的快速服务组合方法是当前研究的难点, 研究快速的服务请求可满足性判定方法与服务集优化方法是解决该类问题的有益探索. 通过引入 Web 服务请求输入闭包概念, 论证了一种服务请求可满足性判定方法和 Web 服务集优化方法, 并通过实验分析了领域属性总数与服务总数的比值对用户请求可满足性及服务集优化效果的影响. 提出了最小用户代价输入集的概念, 给出了一种近似求解最小代价输入集的贪心算法.

关键词: Web 服务; 服务组合; 服务库优化; 输入闭包

中图分类号: TP311

文献标识码: A

文章编号: 1000-7180(2011)07-0168-03

Web Service Request Input Closure and Satisfiability Analysis

YE Heng-zhou, LUO Xiao-juan, NIU Qin-zhou

(College of Information Science and Engineering, Guilin University of Technology, Guilin 541004, China)

Abstract: The research on rapid composition approach for large scale services is still a critical challenge. Exploring service composition satisfiability and service deposit optimization is a useful attempt for such issues. By introducing the concept of Web service request input closure, the paper demonstrates a method of determining service request satisfiability and optimizing Web service deposit, and analyses the effect about the ratio of the total number of field properties and Web services to user request satisfiability and Web service deposit optimization by experiments. The concept of minimal user input cost is put forward and a greedy algorithm to calculate the approximate minimum cost input set is proposed.

Key words: web service; service composition; service deposit optimization; input closure

1 引言

随着 Web 服务技术的发展, Web 网络正在成为 Web 服务的提供者, 而 Web 服务作为面向对象体系(service-oriented architecture, SOA)的基本构建被广泛重视, 研究的重点则是 Web 服务的发现与组合. 服务组合的方法可以是静态的、自动的或者半自动的, 这些方法大都受制于参加组合的原子服务的个数. 随着网络平台中原子服务个数的增加, 寻找局限于特定原子服务集的服务组合方法显得尤为重要. 如文献[1]提出的基于 I/O 依赖矩阵的 Web 服务自动组合方法, 只有当该依赖矩阵仅仅与有关

的少数原子服务关联时, 才比较实用; 文献[2-4]也存在类似的问题. 另一方面, 一些采用图搜索算法的服务组合方法在用户请求不可满足时往往需要搜索整个解空间, 时间复杂度很高. 如文献[5]采用基于回溯树的搜索算法, 在用户请求不可满足时达到指数级时间复杂度; 文献[6]采用的半自动组合方法, 由于不可预知服务请求是否可满足, 用户将很难抉择是否要继续进行组合请求试探. 文中引入了 Web 服务请求输入闭包与关联服务集的概念, 在不考虑 Web 语义的前提下, 探讨了用户请求的可满足性问题、服务集优化问题以及用户请求输入参数不明确时的解决方案.

收稿日期: 2010-10-18; 修回日期: 2010-11-15

基金项目: 广西科学研究与技术开发计划项目(桂科攻 0537020-40); 广西教育厅科研项目(200911LX135)

2 Web 服务库和服务请求

若仅考虑 Web 服务的功能,Web 服务可以抽象为一个二元组 $s(I, O)$, 其中 s 为服务名, I 为输入集合, O 为输出集合. 领域内的所有 Web 服务构成服务库, 即 $S = \{s_1, s_2, \dots, s_n\}$. 如果参数之间存在语义冲突, 可以利用领域内语义标注方法^[7] 解决.

一个服务请求可表示为 $r(I, O)$, 其中 I 为用户给定的输入集, O 为用户期望的输出集, 规定 $r. I \cap r. O = \emptyset$.

定义 1 (服务请求可满足) 对于给定的服务库 S 和服务请求 r , 若 $\forall o \in r. O$, 存在服务序列 $s_1, s_2, \dots, s_t \in S (t \geq 1)$, 满足 $s_1. I \subseteq r. I, o \in s_t. O, s_i. I \subseteq \bigcup_{k < i} (s_k. O) \cup r. I$, 则称 r 是可满足的.

3 服务请求输入闭包与关联集

定义 2 (服务请求输入闭包) 对于给定的服务库 S , 服务请求 r 的输入闭包 $\text{Closure}(S, r)$ 为:

- (1) $r. I \subseteq \text{Closure}(S, r)$;
- (2) 若 $\exists s \in S, s. I \subseteq \text{Closure}(S, r)$, 则 $s. O \subseteq \text{Closure}(S, r)$.

定理 1 对于给定的服务库 S 和服务请求 r , r 是可满足的, 当且仅当 $r. O \subseteq \text{Closure}(S, r)$.

证明:(充分性) 若 r 是可满足的, 由定义 1 知, $\forall o \in r. O$, 存在 $s_1, s_2, \dots, s_t \in S (t \geq 1)$, 满足 $s_1. I \subseteq r. I, o \in s_t. O, s_i. I \subseteq \bigcup_{k < i} (s_k. O) \cup r. I$.

先对 t 应用数学归纳法证明结论: $s_t. I \subseteq \text{Closure}(S, r)$.

(1) 当 $t = 1$ 时, $s_1. I \subseteq r. I \subseteq \text{Closure}(S, r)$, 结论成立.

(2) 假设当 $t < k$ 时结论成立, 即 $s_t. I \subseteq \text{Closure}(S, r) (t < k)$, 从而 $s_t. O \subseteq \text{Closure}(S, r)$. 而 $r. I \subseteq \text{Closure}(S, r)$, 当 $t = k$ 时, $s_t. I \subseteq \bigcup_{k < t} (s_k. O) \cup r. I \subseteq \text{Closure}(S, r)$, 故结论成立.

综上知 $s_t. I \subseteq \text{Closure}(S, r)$, 从而 $o \in s_t. O \subseteq \text{Closure}(S, r)$. 故 $r. O \subseteq \text{Closure}(S, r)$.

(必要性) 设在 $\text{Closure}(S, r)$ 的计算过程中依次使用了服务 $s_1, s_2, \dots, s_t \in S$, 由 $\text{Closure}(S, r)$ 的定义有: $s_1. I \subseteq r. I, s_i. I \subseteq \bigcup_{k < i} (s_k. O) \cup r. I$,

$$\text{Closure}(S, r) = \bigcup_{k=1}^t (s_k. O) \cup r. I.$$

若 $r. O \subseteq \text{Closure}(S, r)$, 而 $r. I \cap r. O = \emptyset$, 故 $r. O \subseteq \bigcup_{k=1}^t (s_k. O)$. $\forall o \in r. O$, 在 $s_1, s_2, \dots, s_t$ 中一定

存在 s_n 满足 $o \in s_n. O$, 则服务序列 $s_1, s_2, \dots, s_n$, 满足: $s_1. I \subseteq r. I, o \in s_n. O, s_i. I \subseteq \bigcup_{k < i} (s_k. O) \cup r. I$, 故 r 是可满足的.

定义 3 (服务请求关联集) 对于给定的服务库 S , 服务请求 r 的关联集 $RS(S, r) = \{s \mid s \in S \wedge s. I \subseteq \text{Closure}(S, r)\}$.

定理 2 对于给定的服务库 S 和服务请求 r , $\text{Closure}(S, r) = \text{Closure}(RS(S, r), r)$.

证明: 显然, $RS(S, r) \subseteq S$, 由定义 2 易知: $\text{Closure}(RS(S, r), r) \subseteq \text{Closure}(S, r)$.

设在 $\text{Closure}(S, r)$ 计算过程中依次使用了服务 $s_1, s_2, \dots, s_t \in S$, 则 $s_1. I \subseteq r. I, s_i. I \subseteq \bigcup_{k < i} (s_k. O) \cup r. I \subseteq \text{Closure}(S, r)$, $\text{Closure}(S, r) = \bigcup_{k=1}^t (s_k. O) \cup r. I$. 故 $s_1, s_2, \dots, s_t \in RS(S, r)$, 且可依次验证 $s_1, s_2, \dots, s_t$ 在 $\text{Closure}(RS(S, r), r)$ 的计算过程中也可以依次使用, 从而 $\text{Closure}(RS(S, r), r) \supseteq \bigcup_{k=1}^t (s_k. O) \cup r. I = \text{Closure}(S, r)$. 故 $\text{Closure}(S, r) = \text{Closure}(RS(S, r), r)$.

4 服务请求输入闭包计算与应用

4.1 Web 服务请求输入闭包求解算法

根据 Web 服务请求输入闭包的定义, 很容易给出其求解算法 calRClosure . 该算法遍历每个服务 s , 如果 s 的输入集包含于输入闭包, 则将 s 的输出集加入输入闭包中; 重复该操作, 直到输入闭包不再发生变化. 对于每个服务 s , 判断 $s. I$ 是否属于 closure 的操作, 最多执行 n 次 (n 为服务个数), 增加 $s. O$ 到 closure 的操作最多执行 1 次, 总的时间复杂度为 $O(n^2)$.

算法 1 calRClosure

Input: $S = \{s_1, s_2, \dots, s_n\}$ and $r(I, O)$

Output: a set of io

Set closure as a set of io;

Add $r. I$ to closure;

hasChanged = true;

while(hasChanged){

 hasChanged = false;

 for each s in S {

 if ($s. I \subseteq \text{closure}$) {

 add $s. O$ to closure;

 hasChanged = true;}}

return closure

4.2 Web 服务请求可满足性分析

定理 1 给出了在不考虑 Web 语义时 Web 服务请求可满足性的判定方法. 文中用仿真实验分析了对于给定的服务集, Web 服务请求的可满足性, 如图 1 所示.

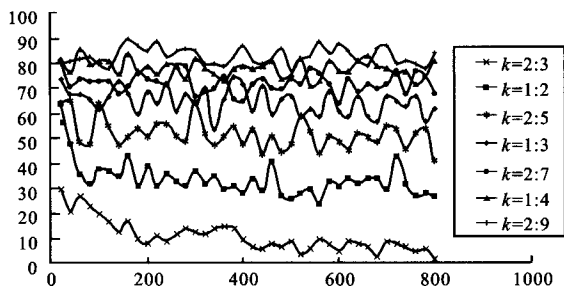


图 1 不同 k 值的服务请求可满足率

其中 k 为所涉及到的 IO 总数与服务集中服务总数的比值, 反映出服务集中服务的关联程度, k 值越小, 关联程度越大; x 轴为服务总数, y 轴为服务请求可满足的百分比. 实验时, 对于每个 Web 服务库, 设定服务总数 n (记为 $S = \{s_i \mid i = 1, 2, \dots, n\}$), IO 总数为 $k * n$, 每个 s_i 在 IO 中随机取出 1 ~ 5 个作为输入集, 再从剩下的 IO 中随机取出 1 ~ 5 个作为输出集; Web 服务请求的输入集与输出集也采用类似方法产生. 为减少特例影响, 每种规模的数据样本为 100, 结果取平均值.

实验表明, 当 k 值越大, 也就是服务之间的关联程度越小时, Web 服务请求的可满足性越低.

4.3 Web 服务库优化

定理 1 和 2 表明, 对于给定的服务库 S , 服务请求 r 的可满足性问题与组合问题只需在其关联集 $RS(S, r)$ 上进行讨论, 从而缩小求解空间, 减少搜索时间. 采用文献[8]所用的服务组合算法, 图 2 演示了 Web 服务库的优化效果. 其中 x 轴为服务总数, y 轴采用优化算法后的平均求解时间与未采用优化算法的平均求解时间的比值; 虚线为对应 k 值的三阶多项式拟合曲线. 实验分析表明, k 值越小, 优化效果越明显; 当 k 值较小时, 通过优化操作, 可快速剔除无关服务, 大大缩小搜索空间, 缩短搜索时间.

4.4 最小代价用户输入集

现在的 Web 服务组合研究, 大多假设 Web 服务请求是明确的, 即用户已经给定具体的输入参数集以及期望的输出结果. 事实上, 很多时候, 用户可能只是明确期望的输出结果, 而不知道该提供哪些输入参数; 或者, 可以提供一些输入参数, 但其中某些参数的获取, 可能需要付出比较大的代价, 因而希望明确这些参数是否真的必须提供. 为此, 提出了最小

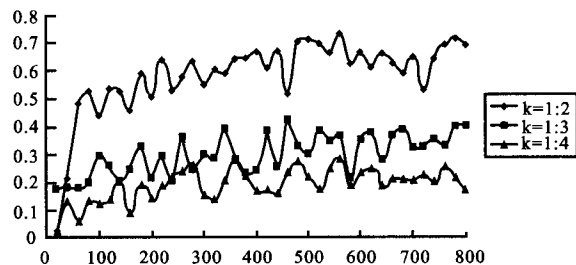


图 2 不同 k 值的优化效果示例

代价用户输入集的定义.

定义 4 (最小代价用户输入集) 对于给定的服务库 S , 和服务请求 $r(I, O)$, 其中 $I = \{i_1, i_2, \dots, i_t\}$, 用户为获取参数 i_t 需要付出的代价为 c_i , $C_I = c_1 + c_2 + \dots + c_t$ 为用户输入集的代价, 称满足如下条件的集合 $\text{Min}I$ 为最小代价用户输入集.

- (1) $\text{Min}I \subseteq I$;
- (2) 若 $r(I, O)$ 是可满足的, 则 $r(\text{Min}I, O)$ 也是可满足的;
- (3) 若存在集体 $\text{Min}I'$ 满足条件(1)、(2), 则 $C_{\text{Min}I'} \geq C_{\text{Min}I}$ 成立.

算法 calMinI 给出了一种求解近似最小代价用户输入集的方法. 算法 calMinI 借用了贪心算法思想, 按照代价由高到低的顺序依次试探用户输入集的参数, 若忽略该参数后, 用户请求仍是可满足的, 则删除该参数. 算法 calMinI 的时间复杂度为 $O(n)$, n 为 I 中元素的个数.

算法 2 calMinI

```

Input:  $S = \{s_1, s_2, \dots, s_n\}$  and  $r(I, O)$ 
Output: a subset of  $I$ 
Sort  $i_1, i_2, \dots, i_t$  as cost desc
Set tempI =  $I$ 
For  $k = 1$  to  $t$  {
  Delete  $i_k$  from tempI;
  Closure = calClosure( $S, r(\text{tempI}, O)$ );
  if ! ( $r, \text{OClosure}$ ) add  $i_k$  to tempI}
return tempI

```

5 结束语

文中引入了 Web 服务请求输入闭包与关联集的概念, 提出并证明了两条性质, 给出了一种快速有效的 Web 服务请求可满足性判断方法与 Web 服务库的优化分析, 并进行了实验分析. 提出并初步探讨了最小代价用户输入集的问题. 今后的研究工作主要有: (1) 语义环境下的 Web 服务请求输入闭包概

(下转第 174 页)

设计并实现了一块多接口网卡,有 8 个千兆位以太网接口,采用本文提出共享缓冲区方案设计数据通道,使用自主编写的 FPGA 硬件程序实现通信管理. 在 Xilinx Integrated Software Enviroment 10.1 上查看综合报告(Synthesis Report),使用 88 块 Block RAM,其中 16 块为数据通道占用,剩余 72 块为网卡其它模块以及连接冗余. 如使用非共享缓冲区方案,则至少需要 200 块 Block RAM. Block RAM 的需求增加了 127%. XC5VLX110T 的已不能满足要求,需要更换为 XC5VLX330T^[5]. 由此可见,共享缓冲区方案明显节省了硬件资源,降低了实现复杂度.

6 结束语

文中提出了一种基于共享缓冲区技术的多接口网卡驱动实现方案. 多个物理接口被虚拟为一个逻辑接口,与协议栈进行数据交换,通过自定义报头实现物理接口的识别. 该方案简化了硬件结构,降低了设计难度,减小了硬件 FPGA 资源占用,降低了实现成本.

参考文献:

- [1] Jonathan Corbet, Alessandro Rubini & Greg Kroah-Hartman. Linux 设备驱动程序[M]. 魏永明,译. 北京: 中国电力出版社, 2006.
- [2] 郑魁,杨志义,曾小芸. 基于 Linux 的光纤通道网卡驱动程序开发. 微电子学与计算机[J]. 2007(1)
- [3] 宋宝华, Linux 设备驱动开发详解[M]. 北京: 人民邮电出版社. 2008.
- [4] 赵利娜,贾豫东,欧攀. 嵌入式 Linux 系统中的多层次驱动程序[J]. 计算机工程, 2009, 35(9): 69-71.
- [5] Xilinx Support Web Site. Virtex5 series parameter data sheet [EB/OL]. [2010-12-01]. <http://www.xilinx.com/support/documentation/virtex-5.html>

作者简介:

赵博 男, (1981-), 博士研究生, 研究方向为业务流分类.

刘勤让 男, (1975-), 博士, 副教授, 研究方向为片上系统, 网络安全.

李高鹏 男, (1985-), 助理工程师, 研究方向为嵌入式系统驱动程序开发.

(上接第 170 页)

念及其性质研究;(2)最小代价用户输入集的高效精确求解算法研究.

参考文献:

- [1] Abrehet Mohammed Omer, Alexander Schill. Web services composition using input/output dependency matrix [C]// Proceedings of the 3rd workshop on agent-oriented software engineering challenges for ubiquitous and pervasive computing. United Kingdom, London: IEEE, 2009: 21-26.
- [2] Ivo Calado, Heitor Barros, Ig Ibert Bittencourt. An approach for semantic web services automatic discovery and composition with similarity metrics[C]// Proceedings of the 2009 ACM symposium on Applied Computing. Honolulu, Hawaii: ACM, 2009: 694-695.
- [3] 李增智,董少然. 基于 Petri 网和 WS-BPEL 的自动服务组合研究[J]. 微电子学与计算机, 27(9): 100-102.
- [4] 李琦,夏阳,束长军,等. 一种基于与或关系的 Web 服务

组合算法研究[J]. 微电子学与计算机, 2010, 27(8): 222-225.

[5] 邓水光,吴健,李莹,等. 基于回溯树的 Web 服务自动组合[J]. 软件学报, 2007, 18(8): 1896-1910.

[6] Michael DiBernardoa, Rachel Pottingera, Mark Wilkinson. Semi-automatic web service composition for the life sciences using the BioMoby semantic web framework [J]. Journal of Biomedical Informatics, 2008, 41(5): 837-847.

[7] 许斌,李涓子,王克宏. Web 服务语义标注方法[J]. 清华大学学报: 自然科学版, 2006, 46(10): 1784-1787.

[8] 叶恒舟,罗晓娟,牛秦洲. 基于与或图的 Web 服务自动组合[J]. 计算机工程与设计, 2010, 31(11): 2645-2647.

作者简介:

叶恒舟 男, (1980-), 硕士, 讲师. 研究方向为 Web 服务、网络构架等.