软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2014,25(6):1180−1195 [doi: 10.13328/j.cnki.jos.004440]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

支持大规模个性化功能需求的服务网络构建*

王忠杰, 徐 飞, 徐晓飞

(哈尔滨工业大学 计算机科学与技术学院,黑龙江 哈尔滨 150001)

通讯作者: 王忠杰, E-mail: rainy@hit.edu.cn

摘 要: 提出一种面向大规模功能个性化需求的服务组合方法,支持服务的大规模个性化定制.现实服务场景通常面临多客户的大量并发的请求,且不同客户对服务的功能需求存在差异.传统方法在应对该场景时需要针对每个需求分别构建服务组合方案,成本代价高.该方法同时考虑多需求,利用潜在收益为需求排序,利用多需求在功能上的相似性、采用渐进迭加策略逐步形成可定制服务方案(称为服务网络),采用需求分组策略降低算法时间复杂性.通过实验,将该方法与其他组合策略进行对比分析,证明了方法的有效性.该方法把服务网络作为持久化存在的基础设施,将个性化和标准化结合起来,以最少的服务来满足尽可能多的需求,达到优化的成本有效性和客户满意度,可应用于各类现代服务领域.

关键词: 服务组合;个性化需求;大规模需求;服务网络;渐进迭加

中图法分类号: TP311

中文引用格式: 王忠杰,徐飞,徐晓飞.支持大规模个性化功能需求的服务网络构建.软件学报,2014,25(6):1180−1195. http://www.jos.org.cn/1000-9825/4440.htm

英文引用格式: Wang ZJ, Xu F, Xu XF. Service network planning method for mass personalized functional requirements. Ruan Jian Xue Bao/Journal of Software, 2014,25(6):1180−1195 (in Chinese). http://www.jos.org.cn/1000-9825/4440.htm

Service Network Planning Method for Mass Personalized Functional Requirements

WANG Zhong-Jie, XU Fei, XU Xiao-Fei

(School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China)

Corresponding author: WANG Zhong-Jie, E-mail: rainy@hit.edu.cn

Abstract: A massive personalized functional requirement oriented service composition method is presented to support service mass customization. In the realm of real-world service domains, there are usually massive customers who concurrently raise their personalized requirements. For such kind of scenarios, traditional approaches have to deal with each requirement one by one, consequentially leading to high cost. The proposed method considers massive requirements together. First, massive requirements are sorted based on the potential benefit. Then an iterative enhancement strategy is adopted to gradually enhance an existing composition solution based on the similarities between requirements. And finally, a service solution (called service network) with high customization degree is formed. Requirement consolidation is adopted to reduce the algorithm complexity. Comparisons made between the new method and two traditional composition approaches by experiments show the superiority of the new method. The method uses service network as a persistent infrastructure being a combination of personalization and standardization of services, and its objective is to select the minimal amount of candidate services to meet maximal amount of functional requirements, so as to achieve the best cost-effectiveness and high degree of customer satisfaction. The method can be applied to various modern service industries.

Key words: service composition; personalized requirements; massive requirement; service network; iterative enhancement

在实际服务场景中,服务提供者通常面临着客户的大量并发请求,且不同客户的需求存在差异,因此仅靠一套标准化的组合服务方案通常难以满足所有需求.目前,服务计算领域的研究中存在两种策略来应对该问题:

* 基金项目: 国家自然科学基金(61033005, 61272187)
收稿时间: 2012-11-23; 修改时间: 2013-01-07; 定稿时间: 2013-06-21

第 1 种策略是采用传统的服务组合方法[1]:利用互联网上的大量可用服务,根据客户需求,动态为其组织出一个组合方案并提供给该客户使用.如图 1 所示,在面对大规模个性化需求时,该策略分别为每个需求单独构造服务方案,其优点在于可以为每个需求实现完全的个性化,客户满意度很高,但其成本代价非常高,需要构造并维护多套组合方案,而一旦需求被满足之后,组合方案就废弃不用.因此,这并非“成本有效”的途径.

服务方案的构造成本高,不仅体现在组合算法从海量服务中选择出恰当的服务并将其组织在一起的时间和成本,在实际应用中还包括与选定服务的提供者进行协商[2]、构建可执行的服务方案(例如 BPEL)、将服务方案部署在物理环境下的时间和成本.目前的 Web 服务组合较少考虑这部分成本,通常假定所有候选服务都是直接可用的.这与现实存在较大的偏差.

何谓“成本有效”[3,4]?一个服务方案所带来的收益应大于它的构造和运营成本,否则对服务提供者来说,构造这个方案就失去了意义.服务方案被使用的次数越多、满足客户需求的数量和频度越大,那么它就越有价值,就可以为提供者带来越多的收益.

软件即服务(SaaS)是解决大规模个性化的服务定制问题的第 2 种有效策略,它以集中式的策略聚集大量的软件服务,通过配置与负载平衡等手段,使一个软件服务实例同时满足多租户的个性化需求[5].这种策略的优点在于:规避了来自不同服务提供者的软件服务在描述规约、开发技术、运行环境方面的差异性,提高了软件服务方案的成本有效性;但不足在于:SaaS 中包含的服务功能主要来自于同一个 SaaS 提供商的开发、运营与维护,较难利用 Internet 上丰富的服务资源满足需求,故可定制能力与范围受限.

用于解决上述挑战的一种有效策略是“大规模定制”,它在以下 4 个方面区别于传统的服务组合与 SaaS:

(1) 服务方案不应只满足单个需求,为了做到成本有效,它应具备同时满足多客户个性化需求的能力;

(2) 服务方案不应是临时性的结果,而应长期存在、可以持续的完善和扩展;

(3) 根据某一特定顾客的需求,对服务方案进行定制(而非重新构造)来满足其个性化需求;

(4) 服务方案的构造成本、维护成本、定制成本的总和应低于它适应大规模客户的需求所带来的收益.

将这种用以满足不同顾客的大规模个性化需求的服务方案称为服务网络(service network).

本文主要解决面向大规模个性化需求的服务网络构建问题,其输入是顾客提出的大量个性化功能需求,算法从候选服务中进行选择和组合,形成一个服务网络,在成本有效和客户满意度方面做到综合最优,如图 2 所示.

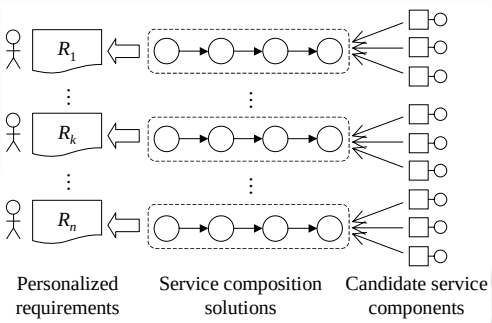


Fig.1 Traditional service composition strategy

图 1 传统的服务组合策略

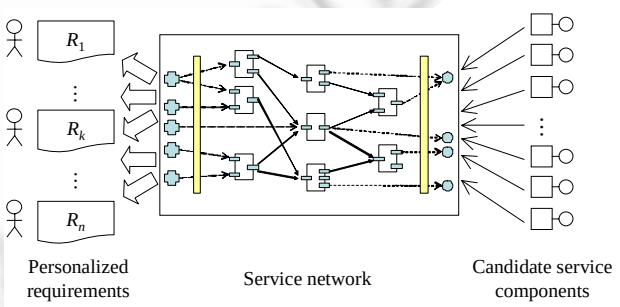


Fig.2 Mass customization oriented service composition strategy

图 2 面向大规模个性化的服务组合策略

虽然目前服务计算领域针对服务组合问题已有了较多方法,但很难将其直接应用于大规模个性化需求的场景,主要面临两个挑战:

(1) 在候选服务规模很大的情况下,服务组合问题是一个 NP-Complete 问题[6],在大规模需求环境下,其搜索空间更大(通常是面向单一需求的服务组合算法的 n 倍,n 为需求数量),因此很难找到多项式时间复杂度的算法.如何降低时间复杂性、寻求合理的近似算法,是面临的挑战之一;

(2) 当需求参数规模和各需求差异性较大时,构造出的服务网络中包含的服务节点数量和服务之间连接关

系密度均巨大,由此导致的服务网络构造成本、维护成本和使用成本也巨大.如何降低服务网络的规模和复杂度,在满足群体顾客的大规模通用性和满足个体顾客的个性化需求之间取得优化的折中,达到成本有效性,是面临的挑战之二.

服务“个性化”的含义很广泛,包括功能上的差异、QoS 上的差异等,均表现为偏好(preferences)[7,8].针对 QoS 需求的大规模个性化满足问题,我们在之前的研究中给出了初步的求解方法,具体见文献[9].本文主要针对功能方面个性化需求,体现在客户提供的输入及期望输出(即 IOPE)上的差异性.针对单一功能需求的服务组合研究,目前的工作主要是基于人工智能规划(AI planning)的方法[3,10−15],使用各类形式语言(例如情景演算、事件演算、模态逻辑、PDDL)描述客户需求的初始状态、目标状态和各候选服务可实现的状态转换进行描述,算法是一个具有推理能力的规划器,采用前溯或后溯策略在候选服务集合中寻求一条组合路径,从而实现从初始状态到目标状态的演变.典型方法包括基于状态空间的规划(例如 Strips 方法、启发式搜索规划器等)、基于图的规划(例如 GRAPHPLAN,STAN,SGP)、组合优化方法(例如整数规划)等,文献[16−18]给出了对该问题域研究结果的综述.但是,这些研究结果定位在单一需求的可满足性规划,缺乏对多需求场景的考虑.针对多需求的服务组合,某些研究者在传统服务组合方法进行了改进,尝试着对需求分组,将多个相似的需求聚合在一起统一进行组合,降低了最终组合方案的数量[6,19].但是这种改进的目标是为了提升组合的效率,并未将“成本有效”作为其优化目标,未能充分考虑满足客户需求的成本.

为克服上述不足,通过估算需求的潜在收益为多需求进行排序,采用渐进迭加策略按次序处理各个需求,在为下一需求构造服务方案时,利用之前已经形成的服务方案,对其进行扩展而逐步形成可定制和成本有效的服务网络;在求解过程中,利用需求之间的覆盖关系对其进行分组,以降低算法的时间复杂度.该方法扩展了传统的服务组合研究,为服务的大规模个性化定制研究提供了初步基础.

本文第 1 节给出基本定义,建立问题的数学模型.第 2 节对求解策略进行分析,提出基于不同策略的 4 种算法 RCA,SCA,IEA 和 CIA.第 3 节给出实验与对比分析.最后是结论与分析.

1 服务网络及其构建问题

1.1 服务网络

服务网络(service network)是将分散在 Internet 和现实中的各类服务(软件服务、人工服务、信息、资源)等按特定方式连接形成的网络,彼此之间通过协同与互操作协议进行交互,完成客户需求.服务网络不是为了满足单个客户需求的,任何客户可以向它提出个性化需求,服务网络通过定制得到一个子网络来满足个体需求,这个子网络等价于传统的面向单一客户需求的服务组合方案.因此,服务网络可以看作是领域相关的、面向大规模客户的、持久存在的服务基础设施.表 1 简要对比了服务网络与传统 SaaS、服务组合之间的异同.

Table 1 Comparisons between SaaS, traditional service composition and service network

表 1 对 SaaS、传统服务组合方法、本文服务网络方法的对比

方法	SaaS	服务组合	服务网络
持久性	永久存在,面向多用户的多次需求	临时存在,面向单次需求,需求满足之后即释放	永久存在,面向多个客户的多次需求
构建和维护方式	由 SaaS 开发商/运营商负责 SaaS 中大多数服务的开发、运营和升级	由算法自动或半自动生成,用户使用之后即释放,下次需求时重新组合	来源于 Internet 上已存在的大量服务,利用这些服务搭建起网络雏形,并逐步持续完善
使用方式	最初是标准化服务,不同租户分别手工配置,配置信息永久保存而不释放	用户提出需求,算法自动或半自动生成个性化的服务方案	用户提出需求,服务网络为其定制具体方案,使用后释放该临时方案

服务网络分为 3 部分:输出参数集、服务集、输出参数集,每个服务包含若干个输入/输出参数,3 个部分之间通过参数传递连接起来形成网络结构.图 3 给出了由 9 个输入参数、10 个输出参数、8 个服务节点构成的服务网络,图 4 给出了对图 3 进行定制而得到的两个定制方案.

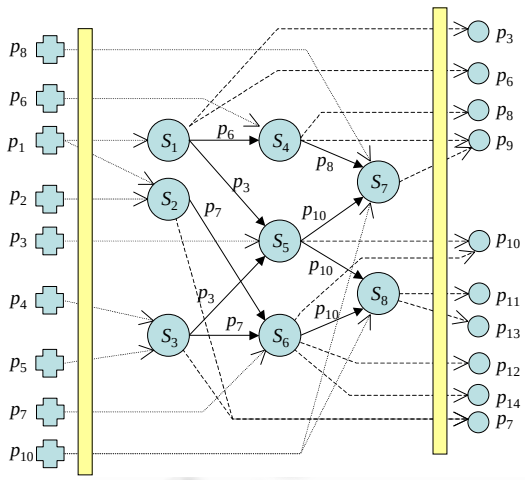


Fig.3 An example of service network

图 3 服务网络的示例

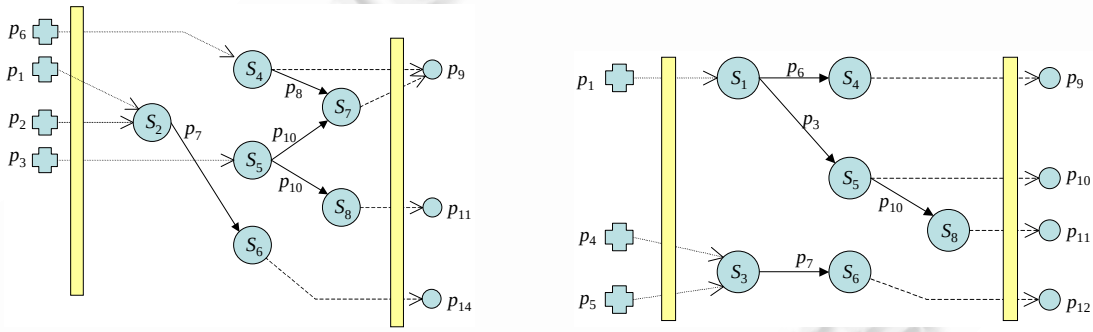


Fig.4 Two customized solutions of the service network in Fig.3

图 4 图 3 中 SN 的两个定制方案示例

形式化地,服务网络可定义为 $SN=(I,O,SV,F_{IS},F_{SS},F_{SO})$,其中,$I=\{p_i\}$,$O=\{p_j\}$分别表示 SN 的输入参数集和输出参数集,每个参数 p_i/p_j 表示为语义本体的形式;$SV=\{S_i\}$为服务集合,每个服务 $S_i=\{I_i,O_i,Price\}$由其输入参数集、输出参数集、价格构成;$F_{IS}=\{is_{ij}\}$表示由输入参数集到服务集之间的有向边,每条边 $is_{ij}=p_i\rightarrow S_j(p_i\in I,S_j\in SV)$表示输入参数 p_i 被传递到服务 S_j;类似的,$F_{SO}=\{so_{ij}\}$表示由服务集到输出参数集之间的有向边,$so_{ij}=S_i\rightarrow p_j(S_i\in SV, p_j\in O)$表示服务 S_i 将其输出参数 p_j 作为 SN 的输出参数之一;$F_{SS}=\{ss_{ikj}\}$, $ss_{ikj}=S_i\xrightarrow{p_k}S_j$,$S_i,S_j\in SV$ 表示两个服务 S_i 和 S_j 之间传递参数 p_k.通过 SV 中各服务的互连,逐步完成从 I 中的输入参数向 O 中的输入参数的转换.

SN 满足以下约束条件:

(1) $I(SN)=\bigcup_{S_i\in SV(SN)} I(S_i)$:服务网络的输入参数集等价于全部服务节点的输入参数集合的并集;

(2) $O(SN)=\bigcup_{S_i\in SV(SN)} O(S_i)$:任何服务节点的输出,都可以直接作为服务网络的输出;

(3) 对$\forall S_j\in SV$,若 $p_k\in I(S_j)$,集合 $F_{IN}(S_j,p_k)=\{is_{kj}|is_{kj}=p_k\rightarrow S_j\}\cup\{ss_{ikj}|\ ss_{ikj}=S_i\xrightarrow{p_k}S_j\ \}$表示 SN 中从输入参数集或其他服务节点到 S_j 且传递参数 p_k 的有向边的集合.那么:

- 对$\forall k$,$F_{IN}(S_j,p_k)$中所有边之间存在“or”关系.这意味着 SN 在进行定制时,只要 $F_{IN}(S_j,p_k)$中有一条边存在,S_j 的输入参数 p_k 就可以被满足.例如图 3 中的 S_5 节点有 3 条入边均传递 p_3,它们之间是 or 关系,在图 4 的两个定制方案中只要保证有一条边存在即可;

- 对$\forall k,l(k\neq l)$,集合$F_{IN}(S_j,p_k)$和$F_{IN}(S_j,p_l)$之间是 and 关系,即最终的定制方案中需包含每个集合中至少一个边,S_j的输入参数才能被满足.例如图 3 中的S_2有两条入边,但其传递的参数分别为p_1和p_2,故它们必须同时存在才能保证S_2被执行;

(4) 同理,对$\forall p_j\in O(SN)$,集合$F_{OUT}(p_j)=\{so_{ij}|so_{ij}=S_i\rightarrow p_j\}$中各条边之间是 or 关系,例如图 3 中$p_9$和$p_{10}$.

or 关系在 *SN* 中的存在,意味着 *SN* 可能有多种方式来满足某个特定需求,这是 *SN* 定制能力的具体体现.在针对具体需求的定制方案中,所有 or 关系都将被消除,以一种确定的方式向用户提供服务.记$K_{jk}^{\mathrm{OR}}=|F_{IN}(S_j,p_k)|$为$S_j$的输入参数$p_k$的入边数目,记$K_j^{\mathrm{OR}}=|F_{OUT}(p_j)|$为输出参数$p_j$的入边数目.

1.2 个性化功能需求

服务网络面向大规模个性化服务需求,因此它的构建和使用均依赖于客户需求的特征.本文定位于功能性需求,将功能需求简要定义为 $R=(I,O,WTP)$,$I=\{p_i\}$,$O=\{p_j\}$分别表示客户所提供的输入参数集合和期望获得的输出参数集合,*WTP* 表示当 *O* 中的全部参数均可产生时客户愿意支付的价格.

1.3 候选服务

SN 的主要构成要素是服务节点,这些候选服务来自于分布在互联网上的、由不同组织所提供的服务.从技术角度,可采用 WSDL 描述服务的功能和调用信息,使用 OWL-S 等描述其语义信息,采用 WSLA 等刻画其 QoS 信息.由于本文侧重于服务网络的功能层面构建,忽略 QoS 而将服务简化表示为 $S=\{I,O,Price\}$,分别表示候选服务的输入参数集、输出参数集、价格.由于不考虑 QoS 因素,可假定任何两个服务的输入/输出参数集合必然存在差异,否则即视为同一个服务.

1.4 服务网络构建问题

服务网络构建问题的输入是:大规模候选服务集 $CS=\{S\}$、大规模个性化需求集 $PR=\{R_i\}$,$|PR|=n$ 为需求的个数.*PR* 中的所有需求均请求同一领域的服务功能,具有领域相似性,它们可以是已通过其他方式被满足的历史需求,也可以是尚未被满足的新需求,代表着该领域典型需求分布.利用这组需求来构造服务网络 *SN*,若服务网络能以较低的代价满足这组需求,就代表着同领域内未来出现的其他需求也可以被该网络较好满足.

该问题的输出分为两部分:

(1) 服务网络 *SN*,且$I(SN)\subseteq\bigcup_{i=1}^{n}I(R_i),O(SN)\supseteq\bigcup_{i=1}^{n}O(R_i)$;

(2) 对$\forall R_i\in PR$,产生一个C_i,它为 *SN* 的一个子网络,是用以满足R_i的具体服务方案.

该问题是一个优化问题,其目标是:以最小的代价构造 *SN* 以满足需求,并可为服务提供方带来最大收益.将该目标称为“成本有效性”:

$$\max B-C-M.$$

其中,

(1) $B=\sum_{R_i\in PR}WTP(R_i)$为被满足的需求所带来的总收益;

(2) $C=\sum_{S_j\in SV}\left(Price(S_j)\times\left|CR(S_j)\right|\right)$表示 *SN* 为满足 *PR* 中的需求而需向各服务节点提供者支付的服务使用价格,其中,$CR(S_j)=\{R_i|S_j\in SV(C_i)\}\subseteq PR$表示服务$S_j$所参与满足的需求集合;

(3) 服务网络维护成本 $M=\mu\times|SV|$,μ为服务节点的单位协商和运营成本.*SN* 包含的服务越多,其维护代价越高.

2 服务网络构建算法

2.1 问题分析与求解策略

与传统服务组合方法服务方案数目与需求数目比为 1:1 不同,该问题是多需求共享一个方案(*n*:1),因此,求解策略在于如何将 *n*:1 组合问题转化为 1 个或若干个 1:1 组合问题.这里给出了 3 种转换策略:

- 策略 1:采用传统组合方法为每个需求分别构造方案,再对这 *n* 个方案进行合并;
- 策略 2:将 *n* 个需求合并得到 *m*(1≤*m*≤*n*)个虚拟需求,再采用策略 1 构造方案;
- 策略 3:从并行为 *n* 个需求分别构造方案(策略 1)变为串行构造方案,始终围绕 1 个方案进行迭代扩展.

这 3 种策略的基本过程如图 5~图 7 所示,图中 ORSC 意指传统的面向单一需求的服务组合算法(one-requirement-oriented service composition).本节后续部分将分别针对这 3 种策略给出具体的求解算法,并对它们的性能和效果进行对比分析.

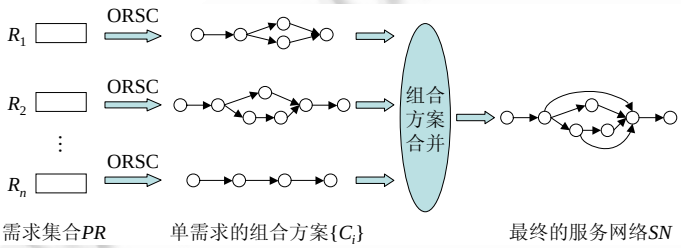


Fig.5 Strategy 1: Composition solution merging based service network planning

图 5 策略 1:基于方案合并策略的服务网络构建

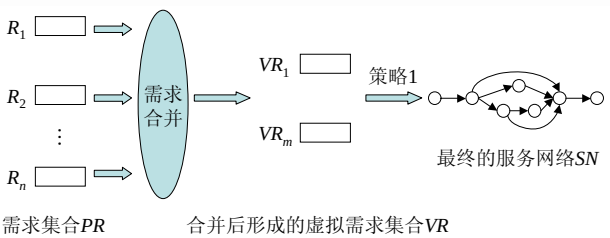


Fig.6 Strategy 2: Requirement consolidation based service network planning

图 6 策略 2:基于需求合并策略的服务网络构建

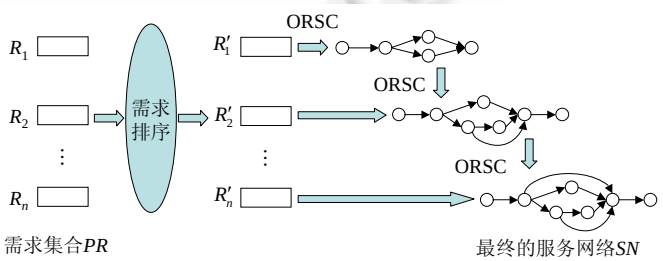


Fig.7 Strategy 3: Iterative enhancement based service network planning

图 7 策略 3:基于渐进迭加策略的服务网络构建

2.2 策略1:基于方案合并策略的构建算法(SCA)

该策略是传统 1:1 组合策略在多需求场景下的直接应用:为每个需求构建一个方案,然后将它们合并.

算法 1. Solution-Consolidation-Based algorithm (SCA).

Input: *CS*, *PR*;

Output: *SN*, $\{C_i\}$.

(1) for each $R_i \in PR$

(2) $SN_i \leftarrow ORSC(R_i, CS)$ //为每个需求分别构造组合方案

(3) if $SN_i \neq \varnothing$

(4) $SR \leftarrow SR \cup R_i$, $C_i \leftarrow SN_i$

(5) $SN \leftarrow Merge(SN_1, SN_2, \ldots, SN_n)$ //将 *n* 个组合方案进行合并

(6) return *SN*, $\{C_i\}$

该算法的时间复杂度取决于需求的规模 *n*,等价于 *n* 倍 ORSC 算法复杂度.除了时间复杂度高,步骤(3)为每个需求单独构建方案也未能充分考虑各方案之间服务的复用,这使得最终的 *SN* 中包含服务数量过大,成本有效性较低.本文仅将其作为传统组合方法在本问题上的应用,以与本文所提方法作对比.

2.3 策略2:基于需求合并策略的构建算法(RCA)

策略 2 在策略 1 的基础上考虑多需求之间的相似性,根据各需求输入/输出参数集合之间的覆盖关系对多需求进行归并分组,每个组内的所有需求可通过同一个组合方案加以满足,故可用一个具有最大输出参数集合和最小输入参数集合的虚拟服务需求作为代表,从而压缩了需求的数量,提高策略 1 的效率.

定义(需求之间的覆盖). 对两个需求 R_1 和 R_2,若满足 $I(R_1) \subseteq I(R_2)$,$O(R_1) \supseteq O(R_2)$,则表示凡是可以满足 R_1 的组合方案一定可以满足 R_2,此时称 R_1 覆盖了 R_2,表示为 $Cover(R_1, R_2)$.

算法 2. Requirement-Consolidation-Based algorithm (RCA).

Input: *CS*, *PR*;

Output: *SN*, $\{C_i\}$.

(1) Partition *PR* into *m* subsets $\{VPR_1, \ldots, VPR_m\}$ w.r.t. the Cover relation

(2) for each VPR_i(i=1,…,m)

(3) $VR_i \leftarrow (VI_i, VO_i, VWTP_i)$, $VI_i \leftarrow \bigcap_{R_{ij} \in VPR_i} I(R_{ij})$, $VO_i \leftarrow \bigcup_{R_{ij} \in VPR_i} O(R_{ij})$, $VWTP \leftarrow \sum_{R_{ij} \in VPR_i} I(R_{ij})$

(4) $VR \leftarrow \{VR_1, VR_2, \ldots, VR_m\}$

(5) return $SCA(CS, VR)$

该算法步骤(1)根据需求之间的覆盖关系将原始需求集合划分为 *m* 个不相交的子集,各子集内的多个需求形成基于 *Cover* 关系的偏序结构.步骤(2)、步骤(3)为每个子集构造一个虚拟服务需求,包含了该子集中最小的输入参数集合和最大的输出参数集合.最后,步骤(5)调用策略 1 的 SCA 算法构造得到最终的服务网络方案.

与策略 1 相比,该策略对需求空间进行了压缩,将 *n*:1 组合问题简化为一个 *m*:1 组合问题,故时间复杂度降低了 *n*/*m* 倍,在最差情况下与策略 1 的时间复杂性等价(*m*=*n*,意即需求集合中完全不存在覆盖关系),在最好情况下等价于传统单需求组合方法(*m*=1).

该策略只考虑了需求覆盖这种严格关系,但对以下情况则无能为力:假设 R_1 希望通过提供{*a*,*b*,*c*}得到{*y*,*z*};R_2 希望通过提供{*b*,*d*}得到{*x*,*z*}.在策略 2 下,这两个需求不能相互覆盖,需要分别调用单需求组合算法;但实际上它们之间仍存在某些相似性,例如均提供输入参数 *b*,均需得到输出 *z*.为了进一步提升最终服务方案的复用度,给出策略 3.

2.4 策略3:基于渐进迭加策略的构建算法(IEA)

该策略采用渐进迭加的策略处理多个需求.假设针对前 *i* 个需求构造的服务网络是 $SN^{(i)}$,那么在处理第 *i*+1 个需求 R_{i+1} 时,不是为其重新构造方案,而是先检验 $SN^{(i)}$ 是否可满足它,针对 R_{i+1} 期望的但无法由 $SN^{(i)}$ 生成的那部分输出参数,调用 ORSC 算法构造新方案,再将结果与 $SN^{(i)}$ 合并起来.

算法 3. Iterative-Enhancement-Based algorithm (IEA).

Input: *CS*, *PR*;

Output: *SN*, $\{C_i\}$.

(1) $PR \leftarrow Sort(PR)$ //对需求进行排序

(2) if n=1

(3) $SN^{(n)} \leftarrow ORSC(R_n, CS), C_n \leftarrow SN^{(n)}$

(4) return $SN^{(n)}, C_n$

(5) $SN^{(n-1)} \leftarrow IEA(PR \backslash \{R_{n-1}\}, CS)$ //递归调用 IEA,为前 n–1 个需求构造方案

(6) $\overline{SN^{(n-1)}} \leftarrow Prune(SN^{n-1}, I(SN^{(n-1)}) \setminus I(R_n))$ //剪枝,去掉与 R_n 无关的输入参数、服务和边

(7) $O_N(R_n) = O(R_n) \setminus O(\overline{SN^{(n-1)}})$ //找到 R_n 中无法被 $SN^{(n-1)}$实现的输出参数集合

(8) if $O_N(R_n) \neq \varnothing$

(9) $VR_n \leftarrow (O(\overline{SN^{(n-1)}}) \cup I(R_n), O_N(R_n), WTP(R_n))$ //构造虚拟需求

(10) $SN_N^{(n)} \leftarrow ORSC(VR_n, CS)$ //为其单独构造一个组合方案

(11) $SN^{(n)} \leftarrow Merge(SN^{(n-1)}, SN_N^{(n)})$ //将新方案与原方案合并

(12) $C_n \leftarrow Prune(SN^{(n)}, O(SN^{(n)}) \backslash O(R_n), O(SN^{(n)}) \backslash O(R_n))$

(13) return $SN^{(n)}, \{C_i\}$

该算法的步骤(1)调用 *Sort* 对全体需求进行排序,排序标准为 $\nu(R_i)=PB(R_i)+\theta_1\times|O(R_i)|-\theta_2\times|I(R_i)|$,$\theta_1$=0.01,$\theta_2$=0.001.它分为 3 部分:需求 R_i 若被满足后可能带来的潜在收益 $PB(R_i)$、R_i 的输出参数个数、R_i 的输入参数个数.在潜在收益相同的情况下,用最少的输入参数产生最多输出参数的需求将被优先处理.

算法的第(2)步~第(4)步处理需求数量 n=1 的情况,直接调用 ORSC 算法得到相应的组合方案即可.若 n>1,第(5)步递归调用 IEA 算法,得到前 n–1 个需求的组合方案 $SN^{(n-1)}$.第(6)步调用 Prune 算法对 $SN^{(n-1)}$进行剪枝,去掉其中不属于 R_n 的输入参数 $I(SN^{(n-1)})\backslash I(R_n)$和相关的服务节点/输出参数,进而在第(7)步、第(8)步中进行分析,判断剪枝后的服务网络 $\overline{SN^{(n-1)}}$ 是否可以生成 R_n 所需的全部输出参数,$O_N(R_n)$表示无法被 $\overline{SN^{(n-1)}}$ 生成的输出参数.若 $O_N(R_n)$不为空,则在第(9)步中构造一个虚拟需求 VR_n,其输入参数是 R_n 的全部输入参数集 $I(R_n)$以及 $\overline{SN^{(n-1)}}$ 的全部输出参数,其期望的输出参数是 $\overline{SN^{(n-1)}}$ 中缺失的 $O_N(R_n)$.这意味着,除了可以使用 $I(R_n)$, $\overline{SN^{(n-1)}}$ 的输出参数集也可以被充分利用以生成 $O_N(R_n)$.第(10)步调用 ORSC 算法为 VR_n 生成相应的组合方案 $SN_N^{(n)}$,进而在第(11)步中使用方案合并算法 Merge 将其与原方案 $SN^{(n-1)}$合并,最终形成满足 $SN^{(n)}$.第(12)步为 R_n 生成相应的组合方案.

在上述过程中,存在两种极端情况:

(a) $O_N(R_n)=O(R_n)$:意味着原方案无法生成 R_n 期望的任何输出参数,算法第(8)步~第(11)步相当于单独调用 ORSC 为 R_n 构造全新方案;

(b) $O_N(R_n)=\varnothing$:意味着 R_n 所期望的全部参数均可由其输入参数集从 $SN^{(n-1)}$中生成出来,因此无须构造增补方案.

IEA 算法的时间复杂性取决于调用 ORSC 算法的次数.上述情况(a)为最差的情形,需要调用 n 次;情况(b)为最好的情况,仅需调用 1 次,故 IEA 的复杂性介于二者之间.

下面给出了一个例子.假设 $SN^{(n-1)}$如图 3 所示,$I(R_n)=\{p_1,p_3,p_{15}\}$,$O(R_n)=\{p_9,p_{10},p_{16}\}$.算法 IEA 第(6)步对 $SN^{(n-1)}$进行剪枝的过程如图 8 所示,结果如图 9 所示.

由图 9 可知,IEA 第(7)步所得到的 $O_N(R_n)=\{p_{16}\}$,而另外两个参数$\{p_9,p_{10}\}$可以由原网络生成.为此,构造虚拟服务需求 VR_n,其输入参数集为$\{p_1,p_3,p_{15},p_6,p_8,p_9,p_{10},p_{11},p_{13}\}$,其输出参数集为$\{p_{16}\}$,第(10)步调用 ORSC 算法后得到的补充方案如图 10 中多边形虚线框所示.与原方案 $SN^{(n-1)}$合并之后得到的结果 $SN^{(n)}$如图 11 所示.

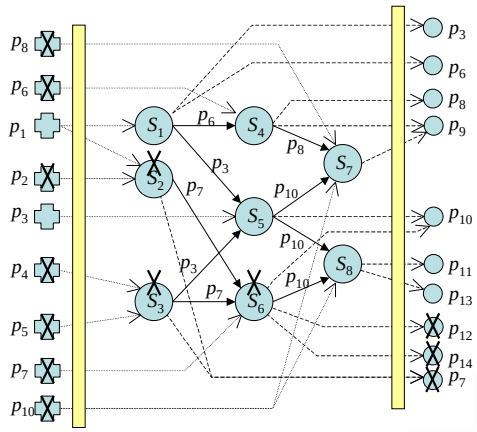


Fig.8 An example of service network pruning
图 8 服务网络剪枝过程示例

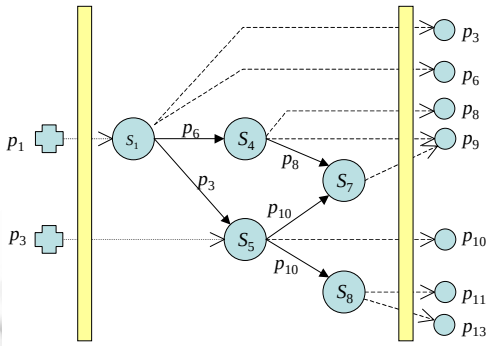


Fig.9 Result of pruning
图 9 剪枝结果示例

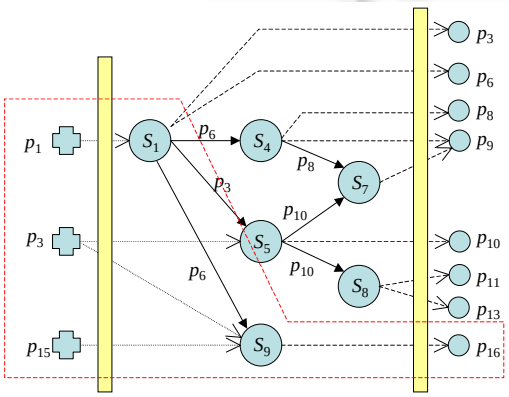


Fig.10 Enhancement on the original service network
图 10 对原服务网络的增强

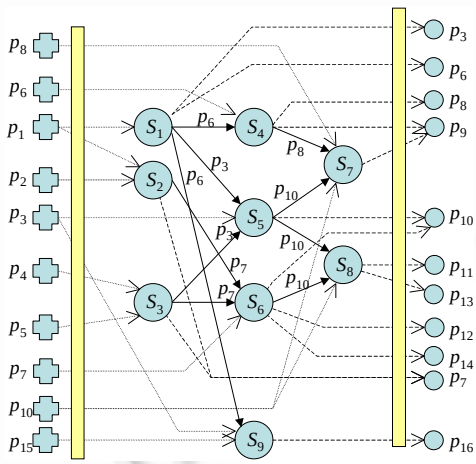


Fig.11 Merge with the original service network
图 11 对原服务网络的合并

从以上分析可以看出,策略 2 通过分组提高了传统算法的效率,策略 3 通过渐进迭加进一步提升了策略 2 生成服务方案的复用度.将二者复合起来,形成基于需求合并+渐进迭加策略的构建算法(CIA).该算法相当于将 RCA 的步骤(1)~步骤(4)加入到 IEA 的起始处,分组之后再按渐进迭加策略进行网络构建.该策略的主要优势是降低了 IEA 的时间复杂度.

2.5 基础算法

2.5.1 需求的潜在收益计算 $PB(R_i)$

算法 IEA 第(1)步需要计算单个需求的潜在收益 $PB(R_i)$以进行需求排序.$PB(R_i)$度量了“需求 R_i 若被满足,它的方案能够从其自身以及后续尚未被满足的需求集中带来的潜在收益”.这里利用两个需求之间的距离 $D(R_i,R_j)$来近似估算,这里的距离是指两个需求的输入参数之间、输出参数之间的相似度,相似度越大,表示 R_i 的方案就越有可能满足 R_j 并从 R_j 中获得收益.具体地,令

$$\xi_1(R_i,R_j)=\frac{|O(R_i)\cap O(R_j)|}{|O(R_j)|},\xi_2(R_i,R_j)=\frac{|I(R_i)\cap I(R_j)|}{|I(R_i)|}.$$

ξ_1越大,表示 R_i的方案中包含了越高比例的 R_j所需要的输出参数;ξ_2越大,表示 R_j的输入参数集可满足 R_i的方案所需的输入参数的比例越高.这两个值越大,意味着 R_i的方案就越能满足 R_j,从而

$$D(R_i,R_j)=\xi_1(R_i,R_j)\times\xi_2(R_i,R_j).$$

因此,R_i的潜在收益为

$$PB(R_i)=WTP(R_i)+\sum_{R_j\in PR,R_j\neq R_i}(D(R_i,R_j)\times WTP(R_j)).$$

2.5.2 面向单一需求的组合算法 ORSC

该算法在 IEA,SCA,RCA 中均有用到,是传统的服务组合算法,输入是一个需求 R_i和候选服务集 CS,输出是一个满足需求 R_i的服务组合方案 C_i.由于该问题的研究已经很充分,这里不再给出具体的算法,详细可见文献[20]等.该算法在最差情况下的时间复杂度为 $O(mk^2)$,最好情况下时间复杂度为 $O(k\log m)$,其中:m 为各候选服务所拥有的参数数目的平均值,一般不大于 20;k 为 CS 中候选服务的数量.

需要注意的是,本文假设所有候选服务的输入参数都是原子参数,但在实际应用中经常出现复合参数的情形,例如 S_1有一个结构体参数 $A(a,b,c,d)$,S_2有一个 $B(a,b,c)$参数.为了充分利用这种复合参数之间的相似性,可将其拆分为多个原子参数.为了保证此类参数生成的一致性,在使用 ORSC 构造组合方案时需增加额外的判断:包含在同一复合参数内的各原子参数必须来源于同一个其他服务的输出,而不能由多个服务分别提供.

2.5.3 服务网络剪枝算法 Prune

该算法在 IEA 步骤(6)和步骤(12)用到,其作用是对服务网络 SN 进行修剪,从中删除特定的输入参数集、输出参数集以及相关的服务节点和边.

算法 4. Pruning an SN (Prune).

Input:当前服务网络 SN,需剪掉的输入参数集合 $\overline{P_{IN}}$,需剪掉的输出参数集合 $\overline{P_{OUT}}$;

Output:服务网络 SN'.

(1) $SN'\leftarrow SN$

(2) $I(SN')\leftarrow I(SN)\setminus\overline{P_{IN}}$

(3) $O(SN')\leftarrow O(SN)\setminus\overline{P_{OUT}}$

(4) for each $is_{ij}\in F_{IS}(SN)$

(5) if $p_i\in\overline{P_{IN}}$ then delete is_{ij} from SN'

(6) for each $S_j\in SV(SN')$ from left to right side of SN'

(7) for each input parameter p_k

(8) if $K_{jk}^{\mathrm{OR}}=0$ then delete S_j and all its related edges from SN'

(9) for each $p_j\in O(SN')$

(10) if $K_j^{\mathrm{OR}}=0$ then delete p_j from SN'

(11) return SN'

算法循环检查 SN 的每个输入参数、输出参数、各服务节点的输入参数,时间复杂性等于这些参数数目之和.算法示例已在图 8 和图 9 中给出.

2.5.4 服务网络合并 Merge

该算法在 SCA 算法的步骤(5)和 IEA 算法的步骤(11)被调用,其作用是将 m 个服务网络$\{SN_i\}$合并起来形成一个新服务网络 SN. $I(SN)=\bigcup_{i=1}^{m}I(SN_i)$, $O(SN)=\bigcup_{i=1}^{m}O(SN_i)$, $SV(SN)=\bigcup_{i=1}^{m}SV(SN_i)$ .3 种有向边$\{is\}$,$\{ss\}$,$\{so\}$若有重复,在 SN 中仅保留 1 次即可.

2.6 小 结

本节给出的 4 种算法 SCA,RCA,IEA,CIA 围绕着降低时间复杂性和降低服务网络规模和复杂度两个挑战

逐步展开.SCA 是传统服务组合方法在多需求场景下的应用,时间复杂性高;RCA 通过需求之间的相似性对其进行分组,缩小了搜索空间,降低了时间复杂性;IEA 考虑需求之间的相似性,采用渐进迭加策略来降低服务网络的规模与复杂度;CIA 则将 SCA 和 RCA 两种策略复合在一起,在应对两个挑战方面取得综合优化的效果.该结论可通过第 3 节的实验结果加以证实.

3 实验与对比分析

3.1 实验设置

实验将针对 4 种算法:方案合并策略(SCA)、需求合并策略(RCA)、渐进迭加策略(IEA)、需求合并+迭加策略(CIA),对比它们在不同设置下的算法效率、最终服务网络的成本有效性、服务复用度等方面的差异.

编程环境:Eclipse 3.7+JDK1.6;实验环境:Intel Core i3 双核 3.10GHz,2.93G 内存,32 位 Windows 7 系统.

实验数据采用模拟生成的方式获得.针对候选服务的模拟生成:设定 1 000 个参数,每 50 个分为一个小组,从 1 000 个参数中随机选取 1~20 个参数作为输入参数,从后 900 个参数中随机选取 5 个~10 个参数作为输出参数(最多来自 3 个不同的小组),形成一个候选服务.针对客户个性化需求的模拟生成:从前 100 个参数中随机选取 1 个~10 个作为需求的输入,从后 100 个参数中随机选取 5 个~10 个作为需求的输出.

3.2 实验结果与对比分析

在 100 个需求、2 000 个候选服务的情况下,分别采用方案合并策略和渐进迭加策略进行服务网络的生成,得到如图 12 和图 13 所示的结果.前者包含 114 个节点、902 条边,后者 81 个节点、372 条边.后者比前者少了 28%的节点数和 58.8%的边数,规模和复杂度都得到了较大程度的改善.由此可见,渐进迭加策略相对于传统服务组合方法的优势是很明显的.

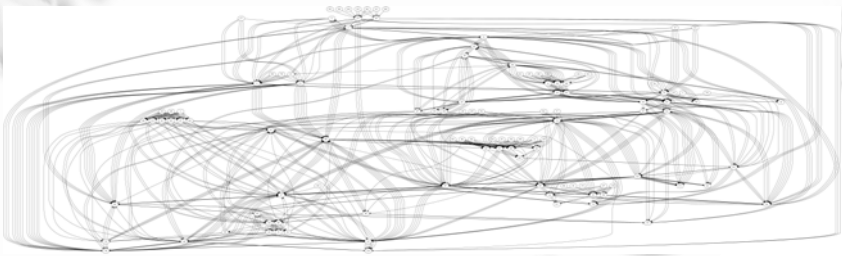


Fig.12 An example service network generated by SCA

图 12 使用方案合并策略得到的服务网络示例

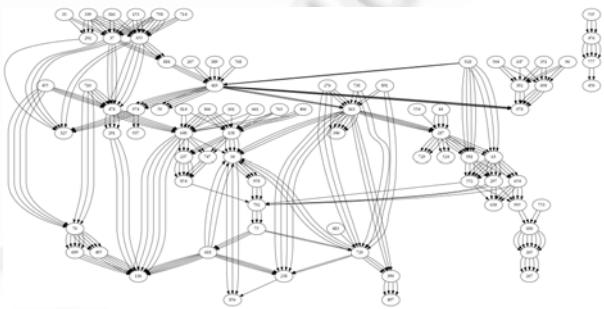
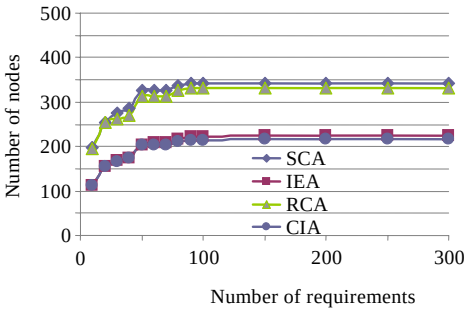


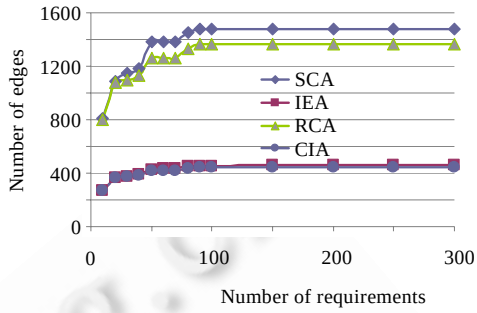
Fig.13 An example service network generated by IEA

图 13 使用渐进迭加策略得到的服务网络示例

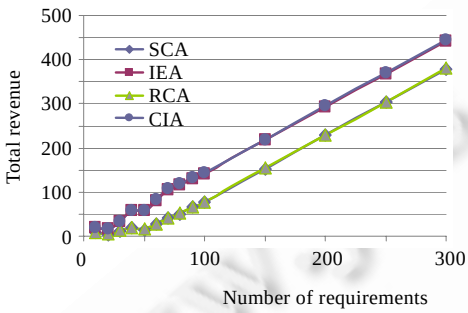
分别选择 10,20,30,…,300 个需求,在同样的候选服务集下进行 4 种算法策略的实验,从最终生成服务网络的节点数、边数、成本有效性、运行时间 4 个方面进行对比,如图 14 所示.



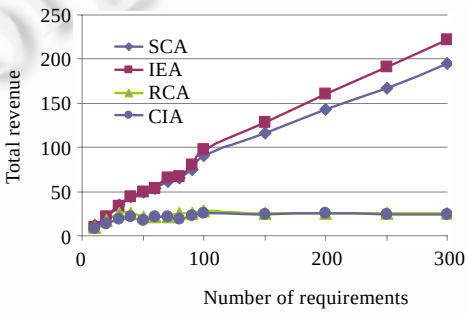
(a) 服务网络节点数目相对于需求数目的变化趋势



(b) 服务网络边的数目相对于需求数目的变化趋势



(c) 服务网络的总收益相对于需求数目的变化趋势



(d) 算法运行时间相对于需求数目的变化趋势

Fig.14 Comparisons between four strategies

图 14 4 种算法策略的对比分析

从图 14(a)、图 14(b)可以看出,传统方法 SCA 和 RCA 所得到的服务网络规模(包含的服务节点数、包含的边数)均大于本文的 IEA 和 CIA,这是因为后者考虑了需求之间的相似性,利用迭代机制在已有服务网络的基础上扩展来满足新的需求,而无需单独构造方案,从而提高了服务的利用率,降低了服务网络的规模.RCA 和 SCA 的趋势线几乎重合在一起,在不同的需求数目下,二者的服务网络节点数仅相差 3.3%~4.2%,边数相差 8.1%~9.5%,RCA 稍好,这是由于在需求分组的时候考虑了一部分需求相似性.IEA 和 CIA 的趋势线也几乎重合,在不同的需求数目下,二者的服务网络节点数仅相差 0.5%~3.2%,边数相差 1.4%~4.1%,CIA 稍好,CIA/IEA 所得到的两组服务网络的规模和复杂度明显小于 RCA/SCA,IEA 的节点数平均为 SCA 的 65.3%,边数平均为 SCA 的 31.1%,CIA 的节点数平均为 RCA 的 65.4%,边数平均为 RCA 的 32.3%,这表明渐进迭代比需求分组在算法中起的作用更大.另一点需要说明的是,不管哪种算法,网络的规模均是先增加后趋于平稳,在需求数超过 100 之后,服务网络的节点数与边数不再显著增加.以 SCA 为例,在需求数为 100 和 300 时所得到的服务网络的节点数和边数分别仅相差 0.3%和 0.14%,这意味着后续的大量需求不会再引入新的服务节点和关系,依靠之前已有的网络即可充分满足.这个临界值取决于多需求之间的相似度,相似性越大,该临界值就会出现得越早.

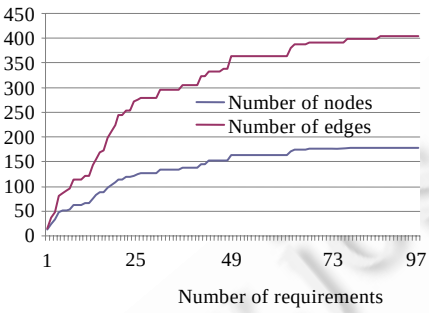
从图 14(c)可以看出,4 种方法构造的网络从满足需求上获得的收益在需求数超过 100 以后,几乎以线性的方式增加(对 4 种算法的收益进行线性拟合,得到的斜率分别为 1.507,1.503,1.507,1.504),但在初期需求数量比较少的时候,收益增加的速度慢甚至存在负增长(如图 14(c)中前两个取样点与第 4、第 5 个取样点出现的下滑),这是由于初期网络规模增长较快而不得不付出较大成本.IEA/CIA 得到的收益基本相同,在不同的需求数目下,CIA 仅比 IEA 高 0.05%~1.03%;SCA/RCA 方法的收益也基本相同,在不同的需求数目下 RCA 比 SCA 高 0.24%~6.19%.

图 14(d)为 4 种方法的时间复杂度对比,这体现出了需求分组策略的优势.SCA/IEA 算法没有对需求分组,需要逐个考虑所有需求,故计算时间基本上与需求数目成正比关系增长(线性拟合的斜率分别为 0.62 和 0.72);

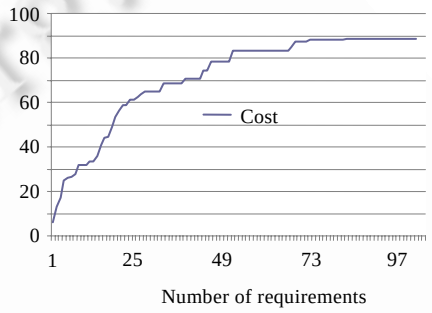
由于 IEA 算法既要分析已有网络对新需求的可满足性以及对缺失参数的方案构造,故其复杂性略高于 SCA 单纯为需求构造方案的情况.RCA/CIA 算法对需求进行了分组,这极大地降低了计算时间,故二者的执行时间在需求数较少的时候线性增加,而后续则保持平稳不再增加.

综合起来,需求分组策略可降低计算复杂度,渐进迭加策略可提升服务复用度并降低网络规模,故二者相结合的 CIA 是最优的求解算法.

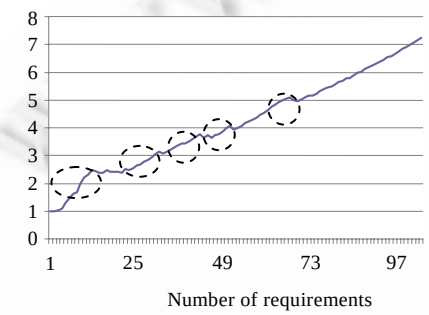
接下来考虑在确定数目的个性化需求下进行单次实验(IEA,CIA)的结果,观察算法执行过程中每个新需求对当前正在构造的服务网络的规模、成本和复用度的影响.从上述实验可知,在需求数量较少时服务网的规模变化较大,故本实验里使用了 100 个需求作为测试数据.实验结果如图 15 所示.



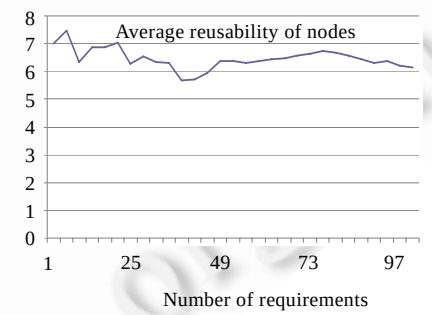
(a) IEA 得到的服务网络节点数/边数的变化趋势



(b) IEA 得到的服务网络总成本的变化趋势



(c) IEA 得到的服务网络节点平均复用度变化趋势



(d) CIA 得到的服务网络节点平均复用度变化趋势

Fig.15 Variation trends of the gererated service network for total 100 requirements

图 15 面向 100 个需求的服务网络生成过程中的变化趋势

从图 15(a)看出,服务网络的规模和复杂度在逐步迭代扩展过程中体现出了阶梯型增长特征,且最初增长得快,后续越来越缓.以节点数为例,在需求数目[1,25],[26,50],[51,100]这 3 个区间内,线性拟合的斜率分别为 10.5,2.97,1.01.这一现象与 IEA 算法设计策略是一致的:最先进入队列的需求对后续需求具有较大的覆盖度,故对其构造网络需要较多的服务节点和边;另外,排在队列前面的各个需求之间较少存在相互的覆盖,这导致算法运行初期基本上与传统 SCA 一致(为每个需求分别构造方案,再合并到一起),故网络规模增长较快;后续需求则越来越可能被已有网络满足,故增长速度趋于平稳.从图 15(b)的总成本变化曲线上也可以看出类似规律,3 个区间内成本增长率从 2.23 降低到 0.73,0.15.

节点平均复用度是指网络中每个节点平均参与满足的需求个数,该值越大,表明多需求之间共享使用的服务节点就越多.图 15(c)是 IEA 算法执行过程中网络节点平均复用度的变化趋势,最初阶段(需求数小于 5 时)需求复用度基本为 1(原因与上段中的分析一致),然后快速增加(表明后续需求可被已有网络所满足);但在增加过程中经常出现小波动,甚至出现短暂的降低(图中用虚线圈标出),这是由于该处进入队列的新需求倾向于引入

若干新服务节点而不是复用已有节点;即使如此,复用度整体上仍然呈线性增加趋势,增长率约为 0.057.

图 15(d)是 CIA 算法执行过程中网络节点平均复用度的变化趋势.在该实验中,100 个需求经过分组之后得到 30 组,每组内包含的需求数在 1~10 之间.实验结果与 IEA 有很大差异,不再是线性增加,而是变成了一种高位波动的形态.这个形态也很好理解:CIA 为每个虚拟需求构造方案,其实是满足了该虚拟需求所代表的分组内的全部需求,故平均复用度始终在较高水平.

针对 300 个需求进行了 CIA 实验,统计了最终生成的服务网络中各服务节点复用度,并按由大到小的顺序排列起来,如图 16 所示.可以看出,节点复用度的分布服从幂律分布(图中的曲线是对其进行幂律拟合的结果,$y=276.07x^{-0.725}$).这表明:对候选服务集中的某些少数服务,由于其本身输入/输出参数集的重要性,它们参与了很多需求的满足;对其他大部分候选服务而言,其参与需求的数目并不高.该结论有助于实际领域应用时找出关键服务并对其进行可靠性增强,以确保服务网络满足需求的能力.这将是后续研究的内容之一.

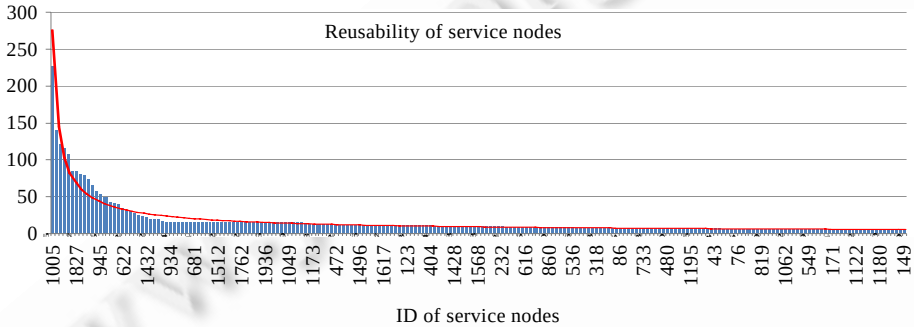


Fig.16 Distribution of the reusability degree of services in the network

图 16 生成的服务网络中各节点复用度的分布情况

4 结束语

在客户群体所形成的大规模个性化需求的场景下,传统服务组合算法往往难以有效应对.本文考虑各需求之间功能上的相似性,采用渐进迭加的策略将对服务组合方案进行逐步改进和扩展,构造得到一个可满足多项个性化需求的服务网络,进而可通过配置来生成满足各需求的特殊方案.算法优化目标是成本有效性,以尽最少的服务来满足尽可能多的需求,达到优化的成本有效性.另一方面,采用需求合并策略来降低算法的时间复杂性.从理论研究的角度,本文工作扩展了传统服务组合的问题空间.

本文方法的应用场合既可定位于互联网上公开发布的 Web 服务,也可面向现实世界中的特定服务领域,在可用服务的规模较大、组合逻辑复杂的情况下,针对领域内的大量顾客需求进行服务网络构建.以阿里巴巴电子商务服务生态系统为例,其中涉及到各种 Web 服务(如淘宝提供的 API)、物流服务、支付服务、团购服务、B2B、B2C、C2B 服务、现实中人工和物理服务等,客户需求规模巨大,彼此之间既有领域相关性也有差异性.再以海运物流服务领域为例,根据作者在山东威海区域所做调查,中韩区域的海运物流服务中,货主企业(客户)的数量约 10 余万家,船公司/港口/货代/场站/车队等相关服务提供者共 3 000 余家,而目前很多货主的个性化需求需靠货代人工方式将各种服务组合成个性化服务方案.若利用本文方法,可降低领域服务成本、提高服务效率.在在线旅游服务中,服务商考虑大量客户的个性化需求,将机票、酒店、景点门票等离散服务聚合起来形成可同时满足多顾客需求的度假方案,当特定顾客提出新需求时,通过对度假方案的配置而得到具体方案.类似的例子也可见于健康医疗服务、制造服务等领域.本文给出的方法将有助于这些领域的服务提供商更好地构建其大粒度服务产品与方案.

References:

[1] Milanovic N, Malek M. Current solutions for Web service composition. IEEE Internet Computing, 2004,8(6):51−59. [doi: 10.1109/MIC.2004.58]

[2] Kumar S, Mishra RB. An approach to multi-attribute negotiation between semantic Web services. Int'l Journal of Web Engineering and Technology, 2009,5(2):162−186. [doi: 10.1504/IJWET.2009.028619]

[3] Lin SY, Lin GT, Chao KM, Lo CC. A cost-effective planning graph approach for large-scale Web service composition. Mathematical Problems in Engineering, 2012. 1−21. http://www.hindawi.com/journals/mpe/2012/783476/ [doi: doi:10.1155/2012/783476]

[4] Wang XZ, Wang ZJ, Xu XF. Price heuristics for highly efficient profit optimization of service composition. In: Proc. of the IEEE Int'l Conf. on Services Computing. Washington: IEEE, 2011. 378−385. [doi: 10.1109/SCC.2011.11]

[5] Sun W, Zhang X, Guo CJ, Sun P, Su H. Software as a service: Configuration and customization perspectives. In: Proc. of the IEEE Congress on Services (Part II). Beijing: IEEE, 2008. 18−25. [doi: 10.1109/SERVICES-2.2008.29]

[6] Bylander T. The computational complexity of propositional STRIPS planning. Artificial Intelligence, 1994,69(1-2):165−204. [doi: 10.1016/0004-3702(94)90081-7]

[7] Wang XZ, Wang ZJ, Xu XF. Semi-Empirical service composition: A clustering based approach. In: Proc. of the IEEE Int'l Conf. on Web Services. Washington: IEEE, 2011. 219−226. [doi: 10.1109/ICWS.2011.15]

[8] Gao Y, Zhang B, Shi SW, Zhu HN, Na J, Zhou FC. A user requirement-driven service dynamic personalized QoS model. In: Proc. of the Int'l Conf. on Dependability. Venice: IEEE, 2010. 17−23. [doi: 10.1109/DEPEND.2010.11]

[9] Wang ZJ, Xu F, Xu XF. A cost-effective service composition method for mass customized QoS requirements. In: Proc. of the 9th IEEE Int'l Conf. on Services Computing. Honolulu: IEEE, 2012. 194−201. [doi: 10.1109/SCC.2012.5]

[10] Wang JS, Li JZ, Li MJ. Composing semantic Web services with description logics. Ruan Jian Xue Bao/Journal of Software, 2008, 19(4):967−980 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/19/967.htm [doi: 10.3724/SP.J.1001.2008.00967]

[11] Zheng XR, Yan YH. An efficient syntactic Web service composition algorithm based on the planning graph model. In: Prof. of the IEEE Int'l Conf. on Web Services. Beijing: IEEE, 2008. 691−699. [doi: 10.1109/ICWS. 2008.134]

[12] Hatzi O, Vrakas D, Nikolaidou M, Bassiliades N, Anagnostopoulos D, Vlahavas I. An integrated approach to automated semantic Web service composition through planning. IEEE Trans. on Services Computing, 2012,5(3):319−332. [doi: 10.1109/TSC.2011.20]

[13] Kuzu M, Cicekli NK. Dynamic planning approach to automated Web service composition. Applied Intelligence, 2012,36(1):1−28. [doi: 10.1007/s10489-010-0238-z]

[14] Karpagam GR, Bhuvaneswari A. AI planning-based semantic Web service composition. Int'l Journal of Innovative Computing and Applications, 2011,3(3):126−135. [doi: 10.1504/IJICA.2011.041913]

[15] Wan CL, Chen LM, Wang ZX, Wang WJ, Shi ZZ. Semantic model and planning algorithm for Web service composition. CAAI Trans. on Intelligent Systems, 2009,4(6):490−496 (in Chinese with English abstract). [doi: 10.3969/j.issn.1673-4785.2009.06.004]

[16] Peer J. Web service composition as AI planning—A survey. Technical Report, University of St. Gallen, 2005.

[17] Oh SC, Lee D, Kumara S. A comparative illustration of AI planning-based Web services composition. ACM SIGecom Exchanges, 2005,5(5):1−10. [doi: 10.1145/1124566.1124568]

[18] Digiampietri LA, Pérez-Alcázar JJ, Medeiros CB. AI planning in Web services composition: A review of current approaches and a new solution. In: Proc. of the VI Encontro Nacional de Inteligencia Artificial (ENIA). 2007. 983−992. http://www.cos.ufrj.br/~ines/enia07_html/pdf/27495.pdf

[19] Cardellini V, Casalicchio E, Grassi V, Presti FL. Flow-Based service selection for Web service composition supporting multiple QoS classes. In: Proc. of the IEEE Int'l Conf. on Web Services. Salt Lake City: IEEE, 2007. 743−750. [doi: 10.1109/ICWS.2007.91]

[20] Oh SC, Lee DW, Kumara SRT. Web service planner (WSPR): An effective and scalable Web service composition algorithm. Int'l Journal of Web Services Research, 2007,4(1):1−23. [doi: 10.4018/jwsr.2007010101]

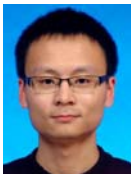
附中文参考文献:

[10] 王杰生,李舟军,李梦君.用描述逻辑进行语义 Web 服务组合.软件学报,2008,19(4):967−980. http://www.jos.org.cn/1000-9825/19/967.htm [doi: 10.3724/SP.J.1001.2008.00967]

[15] 万长林,陈立民,王竹晓,王文杰,史忠植.语义 Web 服务组合中的服务建模及规划算法.智能系统学报,2009,4(6):490−496. [doi: 10.3969/j.issn.1673-4785.2009.06.004]

王忠杰(1978－),男,山东龙口人,博士,教授,博士生导师,CCF 高级会员,主要研究领域为服务计算,服务工程,软件体系结构.

E-mail: rainy@hit.edu.cn



徐飞(1988－),男,硕士生,主要研究领域为服务组合,服务大规模定制.

E-mail: xf_01001@126.com

徐晓飞(1962－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为企业计算,服务计算,商务智能.

E-mail: xiaofei@hit.edu.cn