软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2017,28(4):940−958 [doi: 10.13328/j.cnki.jos.005051]

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

融合网络环境下快速可靠的服务组合容错方法*

张俊娜, 王尚广, 孙其博, 杨放春

(网络与交换技术国家重点实验室(北京邮电大学),北京 100876)

通讯作者: 王尚广, E-mail: sgwang@bupt.edu.cn, http://www.sguangwang.com/index.html

摘 要: 针对传统容错方法在融合网络环境下服务组合的低效性,提出了一种快速可靠的服务组合容错方法.该方法首先采用模糊逻辑对服务的临时性故障进行服务重试;然后采用多属性决策理论对服务的永久性故障进行服务复制;最后,通过改进的粒子群算法对永久性故障进行服务补偿.基于真实数据集的实验结果表明,所提方法在故障排除率、故障处理时间与组合最优度方面均优于其他方法.

关键词: 服务组合;容错;服务重试;服务复制;服务补偿

中图法分类号: TP311

中文引用格式: 张俊娜,王尚广,孙其博,杨放春.融合网络环境下快速可靠的服务组合容错方法.软件学报,2017,28(4):940−958. http://www.jos.org.cn/1000-9825/5051.htm

英文引用格式: Zhang JN, Wang SG, Sun QB, Yang FC. Fast and reliable fault-tolerance approach for service composition in integration networks. Ruan Jian Xue Bao/Journal of Software, 2017,28(4):940−958 (in Chinese). http://www.jos.org.cn/1000-9825/5051.htm

Fast and Reliable Fault-Tolerance Approach for Service Composition in Integration Networks

ZHANG Jun-Na, WANG Shang-Guang, SUN Qi-Bo, YANG Fang-Chun

(State Key Laboratory of Networking and Switching Technology (Beijing University of Posts and Telecommunications), Beijing 100876, China)

Abstract: Traditional fault-tolerance approaches often result in low efficiency of service composition in integration networks. In this paper, a fast and reliable fault-tolerance approach is proposed for service composition in integration networks. This approach firstly adopts fuzzy logic to perform service retry when the transient faults of service occur. And then multi-attribute decision-making theory is employed to carry out service replicate when the permanent faults of service occur. Finally, an improved particle swarm optimization algorithm is used to implement service compensation when the permanent faults of service arise. The experimental results based on real data sets show that the proposed approach is superior to other approaches in terms of fault handling rate, fault handling time, and composition optimization.

Key words: service composition; fault-tolerant; service retry; service replication; service compensation

随着计算机、通信和网络技术的快速发展,不同网络(如移动网、互联网、传感网等)之间的融合已成为下一代网络发展的必然趋势.如今,融合网络正快速渗透到人们生活、工作的各个领域,深刻地改变着信息时代的社会生活.但是融合网络的目的绝不仅仅是把各个网络从物理意义上合为一体,而主要意义是在高层业务(服务)上的融合.其优势不仅在于可以降低运营成本、增强竞争力、提高资源利用率、不同的网络用户(也称为服务请求者)可以享受到相同的服务,更重要的是,可以方便开拓新的增值服务.

随着融合网络技术的逐渐成熟,越来越多的服务被分享到网络中,但这些单个服务(也称为原子服务)能够

* 基金项目: 国家自然科学基金(61472047, 61571066)
Foundation item: National Natural Science Foundation of China (61472047, 61571066)
收稿时间: 2015-09-01; 修改时间: 2015-11-18; 采用时间: 2016-02-24; jos 在线出版时间: 2006-03-30
CNKI 网络优先出版: 2016-03-30 11:43:40, http://www.cnki.net/kcms/detail/11.2560.TP.20160330.1143.002.html

提供的功能较单一.同时,用户对服务的要求也发展到了一个新的阶段,体现为对个性化服务的追求,即,用户对服务的要求日益多样化和复杂化.如果按照传统的模式开发服务,势必需要投入大量的人力和财力,而较长的开发周期会使开发的服务失去时效性.因此,采用面向服务的体系架构(service oriented architecture,简称SOA)是非常有必要的,它把多个功能简单的共享服务组合起来,以得到功能更为强大的增值服务(也称为组合服务),依此来加快服务开发速度,满足用户的各类需求.

然而,开放、动态、多变的融合网络环境给组合服务的可靠执行带来诸多问题.近期的实验研究[1]显示,大部分组合服务的调用失败是由于执行阶段产生的故障而造成的,所以本文研究如何处理这些故障.而对于组合服务设计阶段的正确性验证需要通过专用技术(如Petri网络[2]、自动机理论[3]、Pi演算[4]等),这已超出本文研究范围.在组合服务的执行阶段,影响组合服务可靠执行的主要原因有:

(1) 网络原因.融合网络融合了多种网络,当从一种类型的网络切换到另外一种类型的网络时,用户端容易出现网络链接断开、丢包等现象,影响原子服务的正确调用;

(2) 原子服务本身原因.在原子服务使用的高峰期,如果用户太多可能引起拥塞,造成响应速度变慢;也可能因为其处在更新阶段或者其代理服务器出现问题,上述情况将导致原子服务不可被调用[5].

因此在融合网络环境下,保证组合服务的可靠执行是当前一个亟待解决的问题.

常用的软件可靠性保障技术主要有4种,分别为故障预防(fault prevention)[6]、故障排除(fault removal)[7]、容错(fault tolerance)[8]、故障预测(fault forecasting)[9].一般情况下,用户在构建组合服务时无法获得原子服务的源代码和内部结构设计,采用故障预防和故障排除技术构造绝对无故障的组合服务是不可能的[5].又因为原子服务的彼此独立和自由演化以及组合服务执行过程中的复杂环境,组合服务的故障是很难预测的(特别是在运行阶段),所以也不能采用故障预测来保障组合服务的可靠执行.容错是提高软件系统可用性和可靠性的一种重要方法,它是系统在部分组件出现故障时,仍然能够继续正确运行并完成其设计功能的能力.容错方法通常在产生故障到最终导致系统失败的时间间隔中被使用,目的是在系统故障已经发生的情况下避免系统失败的出现(https://en.wikipedia.org/wiki/Fault_tolerance).即:容错技术通过构造健壮的软件系统来屏蔽错误(error),在用户感觉不到有故障产生的情况下,保证软件系统成功地执行完成.因此,容错技术成为目前保证组合服务可靠执行的最有效方法,得到了研究者越来越多的关注,并提出了各种容错方法[5,10−19].尽管已有的方法有效地提高了组合服务的可靠性,但仍存在一些不足.

- 时间复杂度较高.

即:在处理出现的故障时,耗费的时间较多.在融合网络环境下,用户在任何时间、任何地点都有可能使用服务,且这些服务大多为短事务(执行时间较短),所以用户希望服务能够在短时间内执行完成.如果这类服务出现了故障,采用的容错方法其时间复杂度必须很低才能满足用户要求.例如,人们在移动的过程中使用新闻服务(大多包含图片或者主题影片),而不同的智能手机对图片和影片编码格式要求不同.为了能在不同的手机上正确显示图片和影片,必须调用相应的转码服务,把它们转换为智能手机支持的编码格式.如果在调用相应转码服务时出现故障,就必须采用容错机制快速排除故障,否则将降低用户体验.

- 效率较低.

大多容错方法不区分瞬时性故障(transient fault)和永久性故障(permanent fault),出现故障时只采用同一种方法解决,导致的结果要么不能排除出现的故障,要么组合服务的组合最优度较低.瞬时性故障是持续时间较短的故障,对于此类故障,应该采取重试调用方法,等待出现故障的原子服务从故障中恢复时继续调用该服务,以此来保证组合服务的组合最优度.永久性故障是持续时间较长(比如在整个组合服务无故障执行所需时间内不能恢复)的故障.对于此类故障,应根据出现故障的组合服务是否具有事务(transactional)特性来选择是直接用复制原子服务替代出现故障原子服务[12]或者是对组合服务进行补偿操作[13],以此来保证能够排除出现的故障.

本文在克服上述不足的前提下,提出了融合网络环境下组合服务执行阶段的快速可靠容错方法 FRFTA (fast and reliable fault-tolerance approach).其在组合服务执行阶段不断地收集各种环境信息和状态信息,并针对这些信息进行分析,及时发现可能造成组合服务正常运行的故障,并针对故障的类型,采用合适的容错机制自动

地消除这些故障,保证组合服务正确、可靠地执行完成.FRFTA 主要包括 3 个模块:服务重试、服务复制、服务补偿.服务重试是当网络或服务产生临时性故障时,通过重复调用的方式探测网络或服务是否已经恢复:如果在短时间内能够恢复,则继续使用该服务,从而保证组合服务的较高组合最优度.为了保证较佳的重试时间,本文基于重试次数的原始设定值进行重试调用的原子服务处在工作流的位置以及采用重试调用时组合服务已耗费的执行时间的特点,采用模糊逻辑(fuzzy logic)的方法,自适应地调整重试次数.而当服务确实不可用,即产生了永久性故障时,如果组合服务不具有事务特性,就采用服务复制,即用实现相同功能属性不同服务质量(quality of service,简称 QoS)的原子服务替代出现故障的原子服务.对于复制方案的制定,使用层次分析法(analytic hierarchy process)确定 QoS 权重,并采用多属性决策(multi-attribute decision-making)的算法计算原子服务的综合评价分值,然后根据组合服务对可靠性的要求,选择若干个分值较高的服务作为复制服务.当组合服务具有事务特性时,采用补偿机制来进行容错.即:其需要进行一种回滚(rollback)操作,到一个可以进行补偿的状态,为具有事务特征的组合服务片段重新选择原子服务.为了降低时间复杂度,本文在重新寻找最优原子服务时采用改进粒子群算法(particle swarm algorithm)提高寻找速度,以使组合服务能够尽快地从故障中恢复.

为了验证本文所提方法的有效性,基于真实数据集 QWS(http://www.uoguelph.ca/~qmahmoud/qws/)和 WS-DREAM 的 QoSDataset2(http://www.wsdream.net/)进行了容错仿真实验:首先,采用本文所提 FRFTA 方法与其他 4 种方法进行了实验对比,结果表明,FRFTA 方法在故障排除率、故障处理时间与组合最优度方面均优于对比方法;其次,对影响 FRFTA 方法实验结果的参数进行了分析,结果显示,其对参数的变化适应良好;然后进行了模块分析实验,实验结果表明,3 个模块各司其职,并具有良好的互补性;最后对结果进行了讨论,从更加客观的角度分析了 FRFTA 的较优性能.

本文第 1 节首先介绍我们的 FRFTA 框架,然后详细描述框架中包含的 3 个主要模块——服务重试、服务复制、服务补偿的实现过程.第 2 节给出仿真实验,包括实验建立、实验对比、参数分析、模块分析和结果讨论.第 3 节介绍相关工作.最后一节总结全文并展望下一步研究工作.

1 快速可靠的服务组合容错方法 FRFTA

如图 1 所示,我们提出了用于组合服务执行阶段的、快速可靠的容错方法 FRFTA 框架,其主要包含 3 个模块.

- 模块 1 为服务重试,其作用是提供重试机制,排除出现的临时性故障.融合网络环境下大多为短事务,整个组合服务的执行时间一般不会太长,所以重试时间也不宜过长.为此,重试机制采用模糊逻辑方法,自适应地调整重试次数,以保证较佳的重试持续时间(具体实现过程见第 1.1 节).而当服务重试机制执行失败,即,出现了永久性故障时,则根据组合服务是否具有事务特性进而采用不同的容错机制;
- 当组合服务不具有事务特性时,采用模块 2 提供的服务复制机制,即,用复制服务替代出故障服务.复制方案的制定采用层次分析法(具体实现过程见第 1.2 节);
- 当组合服务具有事务特性时,采用模块 3 提供的服务补偿机制,为具有事务特征的组合服务片段重新选择原子服务.为了降低时间复杂度,采用了改进粒子群算法,以此保证组合服务继续正确地执行(具体实现过程见第 1.3 节).

为了简化问题,本文所提方法基于两个假设.

1) 根据用户的功能性需求,组合服务在设计阶段首先被定义为一个包含多个抽象服务的工作流,然后根据用户的 QoS 约束,采用最优化方法在服务代理处为每个抽象服务绑定一个最优原子服务.有关原子服务具体选择过程的研究已有很多,比如我们的前期工作[20−22],本文不再赘述,只研究组合服务执行阶段出现故障的排除方法;
2) 假定组合服务是由 N 个原子服务组合而成的顺序工作流(如果为非顺序工作流,可以通过文献[23]的方法转换为顺序工作流).

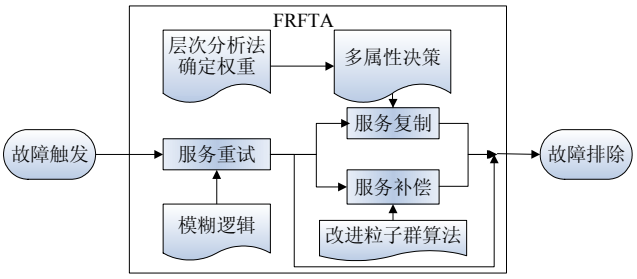


Fig.1 Framework of FRFTA

图 1 FRFTA 框架

1.1 服务重试

服务重试是对出现临时故障的原子服务采用重复调用的方式探测其是否已经恢复.理想情况下,出现临时故障的原子服务从故障中恢复的瞬间,容错框架就应该探测到,立刻调用该原子服务,继续工作流的执行.但是要想达到这种理想状态,必须以较高的频率重试调用原子服务,这样的话,势必会增加网络负载和占用更多的资源.而如果以较低的频率重试调用的话,又会出现原子服务从故障中恢复的时候没有立刻调用原子服务的问题,从而增加了容错的时间,进而增加整个组合服务的执行时间.所以,应选择合适的重试调用频率.另外,临时故障一般持续时间较短,如果经过若干次重试以后还是不能成功调用原子服务,就应该考虑是否是永久性故障:如果是,就不能采用重试策略来容错,而应该考虑其他机制容错.所以,应选择合适的重试次数.

为了简化问题,本文假设采用固定频率对原子服务进行重试调用,而把研究重点放在重试次数 r_{num} 的设置上.如果在组合服务执行前给 r_{num} 赋予初始值,那么其值即为固定值,就不能根据组合服务的具体执行情况来动态改变其重试次数,即,不能动态设定重试执行时间.因此,本文根据当前重试次数的值,进行重试调用的原子服务处在工作流的位置以及采用重试调用时组合服务已耗费的执行时间的特点,提出了基于模糊逻辑[24]的重试次数的自适应调整方法 AAM(adaptive adjustment method),用于动态调整 r_{num} 的值,使其能够根据组合服务具体执行情况,动态地调整重试次数.

一个组合服务总的执行时间由构成它的各个原子服务的响应时间(response time)和执行时间(execution time)来确定.响应时间表示从服务请求发送到接收再到响应所经历的时间;执行时间是指服务从被成功调用到执行完成所经历的时间.一个服务在没有真正调用前,只能得到根据以前用户调用该服务的历史信息统计出来的平均响应时间和执行时间,但一个服务在一次具体执行过程中的准确响应时间和执行时间与当时网络状况、执行该服务的设备硬件配置有关.所以,在预计一个组合服务总的执行时间时,除了对平均响应时间和平均执行时间求和以外,还应该给予一定的冗余时间 t_{redu},用于处理组合服务执行过程中出现的故障,或者用作原子服务响应时间和执行时间的冗余时间.t_{redu} 的取值可以根据用户对组合服务的延迟容忍程度来确定.

AAM 方法根据剩余的冗余时间 $t_{re\text{-}redu}$ 和需要重试的原子服务处于顺序工作流的位置调整重试次数.该方法有两个输入变量和一个输出变量:输入变量为重试次数 r_{num} 和重试位置性能 r_{lp},输出变量为重试次数的调整比例 cr_{num}.其中,r_{lp} 定义为

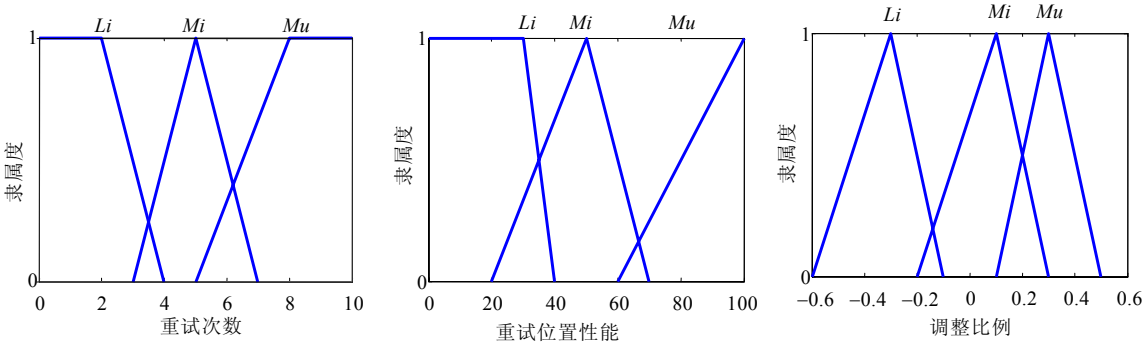
$$r_{lp}=\frac{i}{N}t_{re\text{-}redu},i=1,...,N \tag{1}$$

其中,N 为顺序工作流包含的原子服务个数,i 为需要重试的原子服务处于工作流的位置,$\frac{i}{N}$ 实际上是一个权重.i 值越小,代表采用重试调用的原子服务处在工作流的位置越靠前,那么此时就不能花费过多的时间去进行重试调用,因为工作流还有很多原子服务需要执行,要为它们的执行留下冗余时间.而当 i 值较小时,权值 $\frac{i}{N}$ 也会较小,那么 r_{lp} 也将会较小,采用模糊逻辑动态调整重试次数后,将会得到较少的重试次数,即,较短的重试持续时间.i 值越大,则反之.$t_{re\text{-}redu}$ 代表剩余冗余时间,其值为

$$t_{re\text{-}redu}=t_{redu}-\left(t_{ex}-\sum_{j=1}^{i-1}(r_j+e_j)\right) \tag{2}$$

其中,t_{ex}代表从组合服务开始执行到需要进行重试操作时共花费的时间,$\sum_{j=1}^{i-1}(r_j+e_j)$代表前 i–1 个原子服务的平均响应时间(r)和执行时间(e)的总和.

AAM 方法首先对两个输入变量 r_{num},r_{lp}和输出变量 cr_{num}进行模糊化,设其模糊子集均为{*Li*,*Mi*,*Mu*},其相应的语言变量为 *Li*(little)=少,*Mi*(middle)=中,*Mu*(much)=多.将 r_{num},r_{lp}和 cr_{num}的大小量化为不同等级,并得到各自相应的论域为 *FX*,*FY*,*FZ*,并使用三角形和梯形隶属度函数进行模糊化,如图 2(a)~图 2(c)所示,分别为两个输入变量 r_{num},r_{lp}和输出变量 cr_{num}的隶属度函数.



(a) r_{num} (b) r_{lp} (c) cr_{num}

Fig.2 Membership functions

图 2 隶属度函数

本文采用 IF-THEN 形式的模糊规则,共有 3×3=9 条规则.

1) if r_{num} is "*Li*" AND r_{lp} is "*Li*" then cr_{num} is "*Li*";
2) if r_{num} is "*Li*" AND r_{lp} is "*Mi*" then cr_{num} is "*Mi*";
3) if r_{num} is "*Li*" AND r_{lp} is "*Mu*" then cr_{num} is "*Mu*";
4) if r_{num} is "*Mi*" AND r_{lp} is "*Li*" then cr_{num} is "*Li*";
5) if r_{num} is "*Mi*" AND r_{lp} is "*Mi*" then cr_{num} is "*Mi*";
6) if r_{num} is "*Mi*" AND r_{lp} is "*Mu*" then cr_{num} is "*Mu*";
7) if r_{num} is "*Mu*" AND r_{lp} is "*Li*" then cr_{num} is "*Li*";
8) if r_{num} is "*Mu*" AND r_{lp} is "*Mi*" then cr_{num} is "*Mi*";
9) if r_{num} is "*Mu*" AND r_{lp} is "*Mu*" then cr_{num} is "*Mu*".

其曲面投影如图 3 所示.

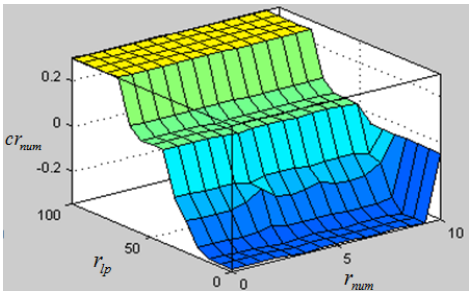


Fig.3 Fuzzy rules

图 3 模糊规则

使用上述9条规则,可以通过模糊推理得到模糊输出变量 cr_{num}.因为输出变量是模糊值,必须转变为精确量,本文采用重心法[25]对 cr_{num} 进行解模糊,从而得到精确的输出结果.得到 cr_{num} 值后,新的重试次数 r_{num} 的值可以按下面的公式进行自适应的调整.

$$r_{num}.new=\lfloor r_{num}.old\times(1+cr_{num})\rfloor \tag{3}$$

其中,$r_{num}.new\in N$.通过采用模糊逻辑,重试次数 r_{num} 可以根据组合服务的具体执行情况通过AAM方法自适应地调整,从而可以达到较佳的重试持续时间.但不是所有的故障均能通过重试操作就能解决,如果重试操作失败,就需要采用其他容错机制来排除出现的故障.

1.2 服务复制

服务复制是实现高可靠组合服务的一种有效方法,通过同时使用多个拥有相同功能属性的服务来实现同一个任务(task),在一个服务出现调用失败时,其他服务可以继续提供所需功能,从而保证组合服务的可靠执行.然而,具有相同功能属性的多个服务,QoS 属性普遍不同,因此在服务复制时,需要选择 QoS 属性较优的服务.然而,由于 QoS 属性的多样性,很难通过一个属性来判断服务的优劣.例如:可靠性高的服务,也许其响应时间较长;而响应时间较短的服务,也许可靠性较差.因此,当进行服务复制时,难以根据单个属性的优劣来选择服务,而必须根据多个 QoS 属性值对服务进行综合评价,从而选择综合评价较高的服务作为复制服务.

为了对服务进行客观评价,本文基于多属性决策[26]的方法来综合评价服务,最终为任务选择分值较高的服务作为复制服务,即,制定复制方案.该方法的思想是:将实现相同功能且满足用户约束的每一个服务看作一个方案,将服务的多个 QoS 值作为各方案的属性指标,则服务的综合评价就转化为一个多属性决策问题.下面给出具体步骤.

步骤 1.设实现一个任务且满足用户约束的服务共有 N 个,则对应的决策方案集合可表示为 $S=\{S_1,\ldots,S_N\}$.若服务的 QoS 属性有 m 个,则对应的方案属性集合记为 $Q=\{Q_1,\ldots,Q_m\}$.第 i 个服务的第 j 个指标值记为 $S_i(Q_j)$ ($i=1,\ldots,N;j=1,\ldots,m$),构成的决策矩阵为

$$X=\begin{pmatrix} S_1(Q_1) & \cdots & S_1(Q_m) \\ \vdots & \ddots & \vdots \\ S_N(Q_1) & \cdots & S_N(Q_m) \end{pmatrix} \tag{4}$$

步骤 2.由于方案的指标较多,各个指标的量纲不同,且有的是积极指标(指标值越高,表示能力越强),有的是消极指标(指标值越低,表示能力越强),为便于比较,对指标矩阵作如下标准化处理:

$$\left.\begin{aligned} r_{ij}=A_i(Q_j)/A_i(Q_j)^{\max}, &\text{ 如果该指标为积极指标,} \\ r_{ij}=A_i(Q_j)^{\min}/A_i(Q_j), &\text{ 如果该指标为消极指标} \end{aligned}\right\} \tag{5}$$

其中,$A_i(Q_j)^{\max}=\max\{A_i(Q_j)|1\leqslant i\leqslant N\}$,$A_i(Q_j)^{\min}=\min\{A_i(Q_j)|1\leqslant i\leqslant N\}$.标准化后的决策矩阵记为 $R=(r_{ij})_{N\times m}$.

步骤 3.假设第 j 个指标的权重为 $w_j\left(j=1,\ldots,m,\sum_{j=1}^{m}w_j=1\right)$(权重采用层次分析法确定[27]),与标准化决策矩阵 R 构成加权标准化矩阵.

$$Y=(y_{ij})=(w_jr_{ij})=\begin{pmatrix} w_1r_{11} & \cdots & w_mr_{1m} \\ \vdots & \ddots & \vdots \\ w_1r_{N1} & \cdots & w_mr_{Nm} \end{pmatrix} \tag{6}$$

步骤 4.根据矩阵 Y 确定正理想决策方案 S^+和负理想决策方案 S^-.

$$S^+=\{\max_{i\in L}(y_{i1},\ldots,y_{im})\}=\{y_1^{\max},\ldots,y_m^{\max}\},L=\{1,\ldots,N\} \tag{7}$$

$$S^-=\{\min_{i\in L}(y_{i1},\ldots,y_{im})\}=\{y_1^{\min},\ldots,y_m^{\min}\},L=\{1,\ldots,N\} \tag{8}$$

步骤 5.根据下面公式计算每个方案 S_i 到正理想决策方案 S^+和负理想决策方案 S^-的距离:

$$D_i^+ = \left[\sum_{j=1}^{m}(y_{ij} - y_j^{\max})^2\right]^{1/2} \tag{9}$$

$$D_i^- = \left[\sum_{j=1}^{m}(y_{ij} - y_j^{\min})^2\right]^{1/2} \tag{10}$$

步骤 6.计算理想方案的贴近度 Z_i,并按照 Z_i 值的大小排序,完成评估任务,贴近度计算公式如下:

$$Z_i = D_i^-/(D_i^- + D_i^+), 0 \leqslant Z_i \leqslant 1 \tag{11}$$

通过上式可以计算出每个方案到理想方案的贴近度 Z_i,即,服务的综合评价分值.对 Z_i 进行降序排列,则可根据组合服务对可靠性要求的高低选择若干个排在前面的服务作为复制服务.

1.3 服务补偿

服务复制机制是基于出故障的服务存在替代服务的假设,但在一些情况下也许不存在可替换服务,或者组合服务具有事务特性,即:组合服务或其部分成分具有原子特征,其要么全部执行,要么什么也不执行.这种情况下,就必须采用另外一种重要的容错机制——服务补偿,来排除故障.其原理是:当出现故障的原子服务具有原子特性时,就停止当前组合服务的执行,回溯到一个可以进行补偿的状态;从此状态开始,为具有原子特性的组合服务片段重新选择原子服务,然后继续执行工作流.

对组合服务片段重新选择要绑定的原子服务类似于对整个工作流选择最优原子服务,是一个多目标优化(multi-objective optimization)问题,也是一个 NP 难问题.为了寻找最优服务组合片段,需要对所有的候选服务进行组合.例如,对于顺序结构(只存在一条单一路径)的组合请求,如果该组合服务片段包含 m 个 task,每个 task 有 l 个候选服务,那么将会有 l^m 种组合方法.如果采用穷举搜索方法,时间复杂度会很高.容错要求耗时较少,所以,在对组合服务片段重新选择原子服务时,必须选择时间复杂度低的算法.

粒子群优化算法(particle swarm optimization,简称为 PSO)从随机解出发,通过迭代寻找最优解,通过适应度(fitness)来评价解的优劣.PSO 比遗传算法(genetic algorithm)规则更简单,没有遗传算法的交叉(crossover)和变异(mutation)操作.其通过追随当前搜索到的局部最优值和全局最优值来寻找全局最优,其实现过程为:通过随机初始化一定数目的粒子构成粒子群;然后,通过迭代寻找最优解.每个粒子代表问题的一个可能解,其具有速度与位置两个特征,粒子的位置对应的目标函数值为该粒子的适应度,通过它来衡量粒子的优劣.在每一次迭代过程中,粒子通过跟踪两个极值来更新自己:一个是粒子所找到的最优解,也就是局部最优 p_{best};另一个是当前整个粒子群找到的最优解,也就是全局最优解 p_{gbest}.粒子根据找到的这两个临时最优解,按照下面两个公式更新自己的位置和速度:

$$v_{id}^{t+1} = wv_{id}^t + c_1r_1(p_{best}(t) - x_{id}^t) + c_2r_2(p_{gbest}(t) - x_{id}^t) \tag{12}$$

$$x_{id}^{t+1} = x_{id}^t + v_{id}^{t+1} \tag{13}$$

其中,x_{id} 代表第 i 个粒子的第 d 维空间中的位置;v_{id} 代表其速度;r_1 和 r_2 是介于(0,1)之间的随机数;c_1 和 c_2 被称为学习因子(一般同时取值为 2);w 为惯性权重,取值范围为(0.1,0.9).

粒子群优化算法结构简单,运行速度很快,所以特别适合用于融合网络环境下的补偿操作中,可以很快地为组合服务片段选择最优的原子服务.为了进一步减少算法执行时间和提高结果精度,在我们前期工作[28]的基础上,继续改进了 PSO 算法.

1) 适应度函数构造

适应度函数就是最大化服务片段的效用函数.为了简化效用函数,本文只考虑消极指标的最小化,对于积极指标采用负值计算(乘上−1),可以将其转化为消极指标[29].本文采用文献[30,31]中对所有 QoS 属性进行归一化的方法,使采用不同量纲的 QoS 属性值可以进行比较,并采用文献[30,31]中的简单加权方法构建效用函数.假设服务片段中含有 m 个任务,对于每个任务要绑定的服务需要考虑 l 个 QoS 属性,q_{ij} 表示第 j 个任务绑定服务的第 i 个 QoS 归一化后的属性值,w_i 表示对第 i 个 QoS 属性选择的权重(可以根据用户偏好设置,或者根据组合服

务的特点来设置).并且,$\sum_{i=1}^{l} w_i = 1$.那么效用函数为

$$f = \max \sum_{j=1}^{m} \sum_{i=1}^{l} q_{ij} w_i \tag{14}$$

公式(14)即为 PSO 的适应度函数.

2) 调整惯性权重 w 的取值

采用文献[32]中的方法,动态地调整不同收敛阶段的惯性权重 w 的取值.文献[33]通过大量实验,其结果表明:如果惯性权重 w 随算法迭代次数的增加而线性减小,将会显著改善算法的收敛速度(减少算法执行时间).假设 $w_{\max}$ 为最大惯性权重,$w_{\min}$ 为最小惯性权重,run 代表当前迭代次数,$run_{\max}$ 为算法迭代的总次数,则有:

$$w = w_{\max} - run \times \frac{(w_{\max} - w_{\min})}{run_{\max}} \tag{15}$$

通过公式(15),使粒子在进化前期以较大的惯性权重值来保证算法的全局搜索能力,而在后期阶段以较小的惯性权重来加快收敛,从而缩短了算法的执行时间.

3) 增加变异因子

粒子群算法容易陷入局部最优,出现早熟现象,所以本文在保证搜索效率的前提下,也要提高搜索精度.因此,除了在迭代过程中对惯性权重动态变化来减少算法的执行时间外,本文又提出一种采用变异因子来对全局最优值进行动态变异,使其能够跳出局部最优,继续寻找全局最优值.

该方法在算法迭代过程中的某一时刻 t 开始,对已经收敛的粒子,即满足 $\lim_{t\to+\infty} x(t) = p$, 进行计数.如果整个粒子群中已经收敛的粒子总数超过阈值 k,则计算 $\lim_{t\to+\infty} \sqrt{E((x^t)^2) + (E(x^t))^2}$ 的值.如果满足公式(16),则认为算法出现早熟现象,增加变异算子,如公式(17).

$$\begin{cases} \sum_{t}^{t+s} 1_{\{\lim_{t\to+\infty} x(t)=p\}} \geqslant k,\ k>1, s>1, k, s \in N \\ \lim_{t\to+\infty} \sqrt{E((x^t)^2) + (E(x^t))^2} \neq 0 \end{cases} \tag{16}$$

$$p'_{gbest} = km \times (p_{gbest} - \overline{x}_{t'}) + \overline{x}_{t'} \tag{17}$$

公式(17)中:$km \in [0.1, 0.3]$; $\overline{x}_{t'}$ 代表在收敛的粒子数没有超过阈值 k 的 t'时刻,所有粒子位置的平均值,即为所有粒子的中心点.当算法陷入局部最优时,以 $\overline{x}_{t'}$ 为中心(在没有陷入局部最优时,中心点可能距离全局最优较近),对 p_{gbest} 增加一个线性算子 km,使其能够跳出局部最优位置,继续搜索其他地方,进而寻找全局最优值.为了不错过全局最优,变异速度不能太大,也不能太小,避免再次陷入局部最优位置.

自适应地调整惯性权重 w 和增加变异因子后,公式(12)的形式改变为

$v_{id}^{t+1} = w v_{id}^{t} + c_1 r_1 (p_{best}(t) - x_{id}^{t}) + c_2 r_2 (p'_{gbest}(t) - x_{id}^{t})$且

$$p'_{gbest} = \begin{cases} km \times (p_{gbest} - \overline{x}_{t'}) + \overline{x}_{t'}, & \text{if } \sum_{t}^{t+s} 1_{\{\lim_{t\to+\infty} x(t)=p\}} \geqslant k, k>1, s>1, k, s \in N \text{ AND } \lim_{t\to+\infty} \sqrt{E((x^t)^2) + (E(x^t))^2} \neq 0 \\ p_{gbest}, & \text{others} \end{cases} \tag{18}$$

PSO 执行过程与整数编码方式以及其他细节可见我们的前期工作[28].采用改进的 PSO 算法为具有事务特性的组合服务片段寻找到最优原子服务后,就可以继续执行组合服务了.

2 仿真实验

为了验证本文提出的容错方法 FRFTA 的有效性,我们进行了大量的仿真实验.首先,将我们提出的方法与其他 4 种方法在故障排除率、故障处理时间、组合最优度等性能指标上进行了实验对比;然后,对 FRFTA 方法中的重要参数和各功能模块间的作用进行了分析;最后,对实验结果进行了讨论.

2.1 实验建立

实验基于两个真实的服务数据集 QWS 和 WS-DREAM 的 QoSDataset2.QWS 数据集[34]包含了 2 507 个真实 Web 服务,每个服务包含 9 个 QoS 属性.WS-DREAM 的 QoSDataset2 数据集[1,35]包含了 339 个用户使用 5 825 个 Web 服务的响应时间(response-time)和吞吐率(throughput).实验在一台 PC 机上执行,配置为:Intel(R) Core (TM) i5-4210U 2.39GHz,8.0GB of RAM,Windows 8.1 专业版;运行环境为 Matlab R2013a.

根据实验需要,组合服务包含的任务数目设置为 10~60,候选服务的数量设置为 60.那么组合服务的执行过程为:顺序执行工作流中(如果为非顺序工作流,可以通过文献[23]的方法转换为顺序工作流)的每一个服务,当遇到故障时采用容错方法排除故障;然后继续执行工作流,直到全部服务执行完成,即组合服务成功执行.实验中,我们仿真组合服务的执行过程中出现的故障,故障的规模由原子服务的可靠性确定,故障的类型分为瞬时性故障和永久性故障.为了简化实验,瞬时性故障和永久性故障的概率均设置为 0.5.为了说明本文所提方法的有效性和便于与其他方法对比,在仿真实验过程中,我们采用故障处理时间、故障排除率以及组合最优度这 3 个性能指标.具体定义如下.

定义 1(故障处理时间). 在组合服务的执行过程中,为了排除出现的故障所耗费的时间总和为 F_t,即:

$$F_t = \sum_{i=1}^{m} f_t^i \tag{19}$$

其中,m 代表组合服务执行过程中出现的总故障次数,f_t^i 代表处理第 i 次故障所经历的时间(从故障产生到正确排除,恢复组合服务的执行所花费的时间,单位为 ms).F_t 值的大小代表了处理故障的效率:值越小,代表效率越高;值越大,代表其效率越低.

定义 2(故障排除率). 在组合服务的执行过程中,能被排除的故障数目与出现的总故障数目的比率为 F_r,即:

$$F_r = \frac{f_{handle}}{m} \tag{20}$$

其中,m 代表的意义与公式(19)相同,为一次组合服务执行过程中出现的总故障次数;f_{handle} 代表被排除的故障次数.F_r 值的大小代表了处理故障的能力:值越大,代表处理故障能力越强;值越小,代表处理故障能力越弱,其取值范围为[0,1].如果 m=0,即组合服务的执行过程中没有出现故障,那么设定 F_r=1.

定义 3(组合最优度). 为组合服务的效用函数值,采用文献[30.31]中的简单加权方法构建,即:

$$U = \sum_{i=1}^{N} (0.5r_i + 0.5th_i) \tag{21}$$

为了简化效用函数的计算,我们只考虑原子服务的响应时间和吞吐率这两个 QoS 属性,它们的权重均设定为 0.5.公式(21)中,N 为组合服务包含的原子服务个数,r_i 和 th_i 分别代表采用文献[30.31]中方法归一化后的服务 i 的响应时间和吞吐率.U 代表组合最优度,其值的大小代表组合服务的 QoS 属性(通过组成它的原子服务的 QoS 属性来体现)的优劣:值越高,代表 QoS 属性越优;值越低,则相反.

2.2 实验对比

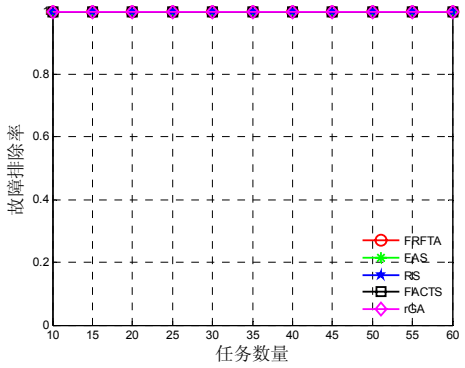
为了说明 FRFTA 的有效性,我们进行了实验对比,对比方法共有 4 种,分别为 FAS[16](采用重试策略进行容错)、RS[12](采用服务复制策略进行容错)、FACTS[15](针对具有事务特性的组合服务,提出了一个容错框架 FACTS)、rGA[13](采用改进的遗传算法进行服务补偿).限于篇幅,在实验对比中,仅对包含具有原子特性的任务数占总任务数的比率为 0%,50%,100%的 3 类实验(分别用 A 类实验、B 类实验、C 类实验代表)分别采用我们的 FRFTA 方法和 4 种对比方法进行容错实验.实验中,原子服务的可靠性均设定为 0.9.对于 3 类实验,分别按照任务数量从 10 开始,并按照以 5 递增的规律(随着任务数的增多,可能产生的故障也随之增多),分别采用 5 类方法进行实验,所有实验结果均为运行 100 次实验后的平均值.

2.2.1 A 类实验结果

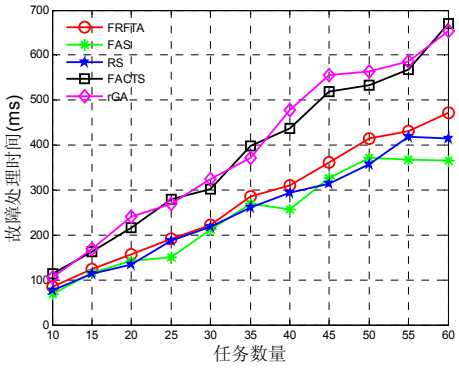
图 4 给出了 FRFTA 与其他 4 种方法在处理包含具有原子特性的任务数占总任务数的比率为 0%(即,不具有原子特性)、不同任务个数下的实验对比结果.从图 4(a)可以看出:FRFTA 可以排除所有故障,即,其故障排除

率为 1.这是因为,对于不具有原子特性的组合服务的故障排除可以采用服务重试机制排除临时性故障,采用服务复制机制排除永久性故障;而我们的 FRFTA 框架中包含有这两种容错机制,所以对于组合服务的执行过程中遇到的所有故障均能排除.

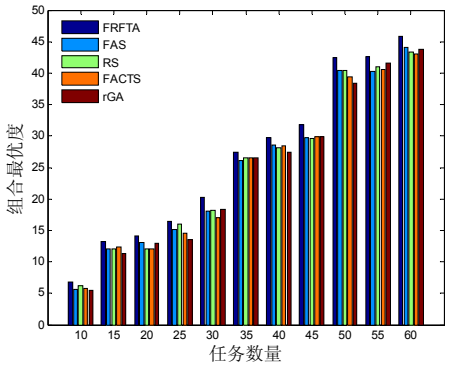
FAS 和 RS 方法也可以排除所有故障,且从图 4(b)可以看出:其故障处理时间较短,比 FRFTA 方法还要短.这是因为,FRFTA 区分临时性故障和永久性故障,在发生临时性故障时采用的是服务重试机制,临时性故障的持续时间也计入了故障处理时间;而对于 FAS 和 RS 方法,在原子服务出现故障时均采用复制服务替代出故障的服务,故障处理时间仅为调用替代服务的响应时间.但从图 4(c)可以看出,FRFTA 方法的组合最优度要高于 FAS 和 RS.这是因为,组合服务最初绑定的原子服务是在满足约束的条件下按照效用函数最优找到的原子服务,如果其能在短时间内恢复,最好还是调用最优原子服务.因为调用复制服务除了会降低组合最优度外,还有可能出现很多问题,比如也许组合服务使用者需要付出更多的成本,甚至有可能该复制服务可靠性更低,在调用过程中又出现了故障,使得在故障处理的过程中又出现故障,这样会耗费更多的时间处理新的故障.FACTS 和 rGA 方法也可以排除所有故障,但从图 4(b)可以看出,其所需故障处理时间比我们的方法要多.这是因为,这两种方法处理故障前首先要进行状态一致性检测,然后再处理故障,一致性检测增加了故障处理时间.而此类实验的组合服务不具有原子特性,所以没有必要进行一致性检测.这两种方法的组合最优度低于 FRFTA 方法的原因也是因为对产生临时性故障的服务没有采用服务重试的机制排除故障所致.



(a) 故障排除率



(b) 故障处理时间



(c) 组合最优度

Fig.4 Comparison results for experiment A

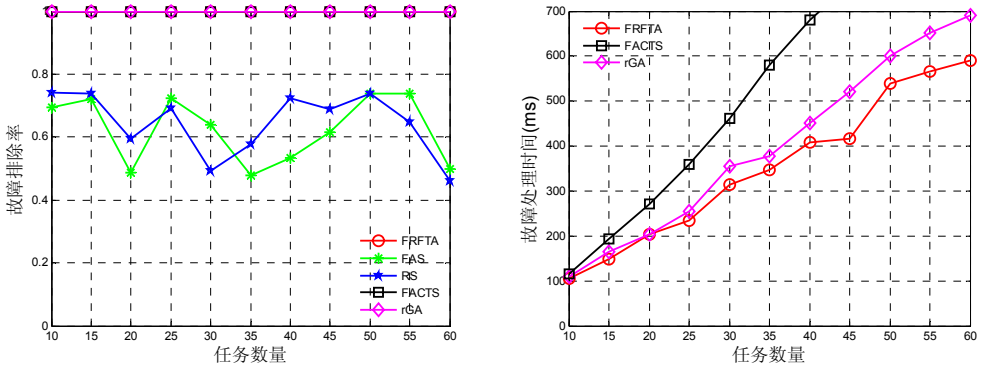
图 4 A 类实验的结果对比

2.2.2 B 类实验结果

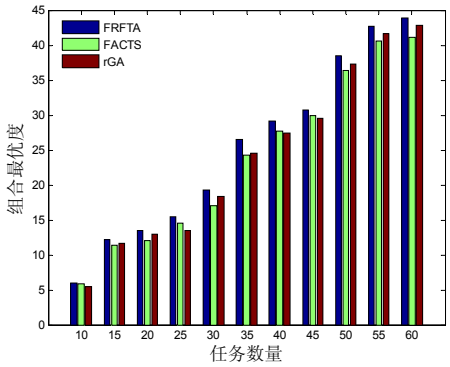
图 5 给出了 FRFTA 与其他 4 种方法在处理具有原子特性的任务数占总任务数的比率为 50%、不同任务

个数下的实验对比结果.从图 5(a)可以看出,FAS 和 RS 方法无法排除全部故障.这是因为,这两种方法在故障的排除过程中仅能排除出现的临时性故障和不具有原子特性的原子服务出现的永久性故障,而对于具有原子特性的原子服务出现的永久性故障,这两种方法均不能处理.即,这两种方法不能用于具有原子特性的组合服务的故障处理.因为其不能完全处理出现的故障,所以故障处理时间和组合最优度均无法得到,进而在图 5(b)和图 5(c)中没有这两种方法的实验结果.

FACTS,rGA 和我们的 FRFTA 方法均能处理全部故障,但从图 5(b)可知,FRFTA 方法所需故障处理时间比另外两种方法要少.其原因为:FRFTA 方法在处理具有原子特性的原子服务出现的故障时能够迅速回卷到可保证组合服务状态一致的任务处,然后采用改进的 PSO 算法,快速地为具有原子特性的组合服务片段找到最优原子服务,保证组合服务的继续执行.从图 5(b)还可以看出:在组合服务的任务个数达到 50 个时,我们的方法所需故障处理时间明显增加.这是因为,随着任务个数的增加,组合服务的状态空间也迅速增多,使得为组合服务片段寻找最优原子服务的过程耗费时间更长.但包含任务个数较多的组合服务一般为较长事务,人们对它的容忍程度也会增加.rGA 方法采用改进的遗传算法处理故障,因为其比粒子群算法实现复杂,所以故障处理时间明显比我们的方法要长;且在任务个数增加到 35 个后,故障处理时间增加明显.FACTS 方法没有采用启发式算法排除故障,其在任务个数较少时,还能在较短的时间内处理完故障;但当任务个数增加到 20 个后,故障处理时间增加得明显;且随着任务个数的继续增加,其处理时间增加得更快,使其不适合应用到实际的组合服务故障处理过程中.从图 5(c)可知,FRFTA 方法的组合最优度略高于其他两种方法.这仍然是因为 FRFTA 在原子服务出现临时性故障时,还继续调用事先绑定的原子服务导致的结果.



(a) 故障排除率 (b) 故障处理时间



(c) 组合最优度

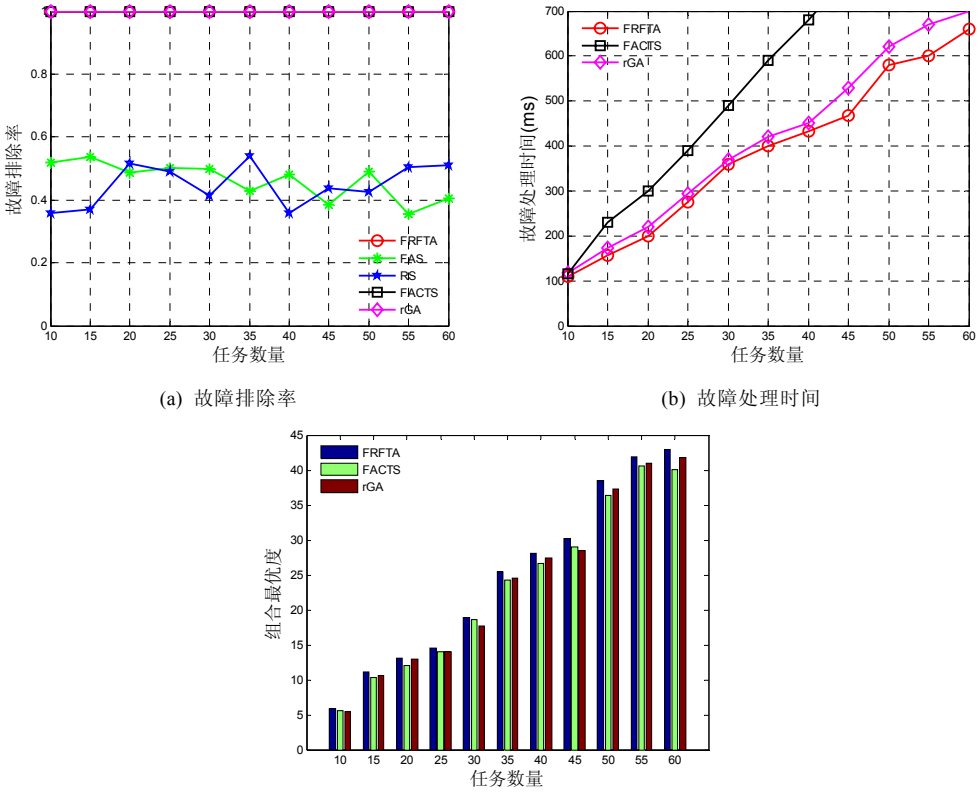
Fig.5 Comparison results for experiment B

图 5 B 类实验的结果对比

2.2.3 C 类实验结果

图 6 给出了 FRFTA 与其他 4 种方法在处理具有原子特性的任务数占总任务数的比率为 100%、不同任务个数下的实验对比结果.这种类型的实验一旦有原子服务出现永久性故障,就必须重新执行整个组合服务,且需要重新绑定原子服务.从图 6(a)可以看出,FAS 和 RS 方法的故障排除率更低,原因与第 2.2.2 节中的分析相同.同时,对这两种方法的故障处理时间与组合最优度也无法得到.而对于 FRFTA,FACTS 与 rGA 这 3 种方法,均因为含有原子特性的原子任务数量的增多,故障处理时间比上两类实验有所增加,但是 FRFTA 方法的故障处理时间还是最少的.对于组合最优度的实验结果,FRFTA 方法也略高于 FACTS 与 rGA.

从上述 3 类实验可以看出:我们的 FRFTA 容错方法可以排除出现的所有故障,且所需故障处理时间较少,组合最优度也较高.虽然在处理不具有原子特性的组合服务出现的故障时比 FAS 和 RS 方法耗费时间略长,但我们的方法在出现瞬时性故障时还继续使用最优原子服务,从而具有较高的组合最优度.总体来说,FRFTA 容错方法在处理出现的故障上的表现优于其他 4 种方法.



(a) 故障排除率 (b) 故障处理时间

(c) 组合最优度

Fig.6 Comparison results for experiment C

图 6 C 类实验的结果对比

2.3 参数分析

为了进一步验证容错方法 FRFTA 的性能,我们改变实验参数,继续进行实验,以此来观测参数变化对实验结果的影响.故障规模的大小直接影响到实验结果,而服务的可靠性决定了故障规模,所以我们通过改变可靠性值来观测其对我们实验结果的影响.在参数分析实验中,设定组合服务包含任务数为 20,每个原子服务的可靠性从 0.5(可靠性低于 0.5 的原子服务在实际的应用中已基本不可用)开始,以 0.05 递增的规律直到 0.95.出现故障时,为瞬时性故障或者永久性故障的概率均设置为 0.5.对包含具有原子特性的任务数占总任务数的比率从 0%

开始,以 10%递增,直到 100%的组合服务分别进行实验.每类实验结果取运行 100 次的平均值.

从图 7 可以看出,FRFTA 的故障排除率为 1(其原因在第 2.2 节中已给出分析).即,故障规模的变化不会影响故障排除率的值.

从图 8 可以看出:针对比率为 0%,即不具有原子特性的组合服务,FRFTA 的故障处理时间随着可靠性的增加基本上呈线性减少.这是因为,对于不具有原子特性的组合服务,故障处理时间由等待原子服务从瞬时故障中恢复所需时间和出现永久性故障时复制服务的响应时间所组成,其值会随着故障数目的减少而线性减少.从图 8 中还可以观察出:随着比率的增加,即具有原子特性的原子服务的增多,其总的故障处理时间也相应延长,相应的原因与第 2.2 节的分析相同.在比率增加到 60%时,或者可靠性低于 0.65 时,所需时间均增加较多.这是因为,随着具有原子特性的原子服务的增多或可靠性的降低,具有原子服务特性的原子服务出现故障的几率变大,需要调用粒子群算法的次数会增多,所以总的处理时间也会增加.但总的来说,增加比例不大.

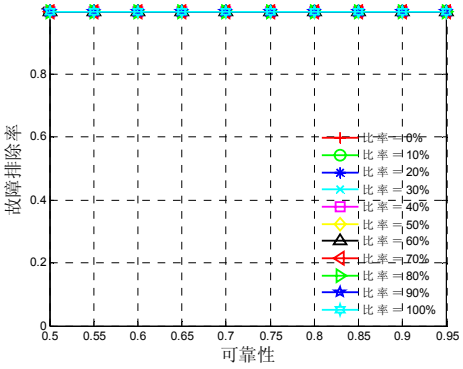


Fig.7 Experimental results of fault handling ration for different value of reliability

图 7 不同可靠性取值的故障排除率实验结果

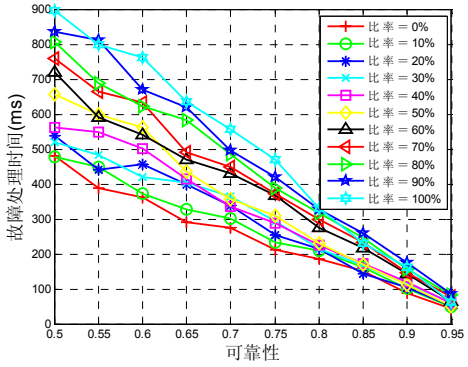


Fig.8 Experimental results of fault handling time for different value of reliability

图 8 不同可靠性取值的故障处理时间实验结果

图 9 是组合最优度的实验结果,整体来说,随着可靠性的降低或者具有原子特性的服务比率的增加,组合最优度均呈下降趋势.其原因为:随着可靠性的降低,在组合服务的实际执行过程中,最初绑定的原子服务因为不能被成功调用的几率变大,而不得不更多次地调用比其整体 QoS 值要低的服务,所以造成组合最优度的下降;随着具有原子特性的服务比率的增加,具有原子特性的服务出故障的概率也随之增加,即被替换的原子服务数目增多,从而导致组合最优度下降.但从图 9 可以看出:其下降比例较小,均不是很大.

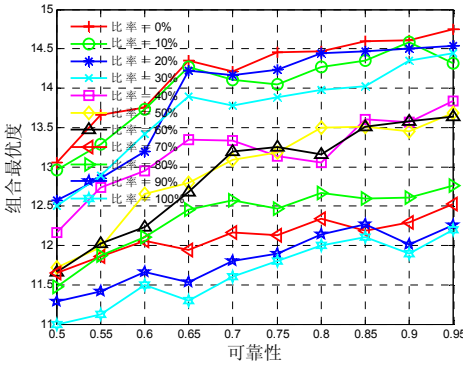


Fig.9 Experimental results of degree of optimization for different value of reliability

图 9 不同可靠性取值的组合最优度实验结果

2.4 模块分析

为了更进一步地验证 FRFTA 的性能,并为实际部署提供参考,我们设计了模块分析实验,其主要分析在缺少 3 个模块中的 1 个时对 FRFTA 性能产生的模块折损度.

定义 4(模块折损度). 采用FRFTA的实验结果与缺少一个主要模块下的实验结果的差值,与采用FRFTA的实验结果的比率为 D.其中,

- 针对故障排除率和组合最优度实验结果的模块折损度 D 为

$$D=\frac{r_{FRFTA}-r_m}{r_{FRFTA}}\times 100\% \tag{22}$$

- 针对故障处理时间实验结果的模块折损度 D 为

$$D=\frac{r_m-r_{FRFTA}}{r_{FRFTA}}\times 100\% \tag{23}$$

其中,r_{FRFTA} 表示采用 FRFTA 方法得到的实验结果,r_m 表示在缺少相应模块的情况下的实验结果.由于故障排除率和组合最优度值越大代表实验结果越佳,而故障处理时间越小代表实验结果越佳,为了能使 3 个性能指标可以相互比较,我们对其模块折损度 D 进行了不同的定义,如公式(22)和公式(23)所示.如果得到正值,代表降低了FRFTA 的实验效果;得到负值,则代表提高了 FRFTA 的实验效果;0 代表保持了 FRFTA 的实验效果.

因为篇幅的原因,我们只给出具有代表性的一类实验结果,见表 1.此实验设置为:组合服务包含任务数为20,具有原子特性的任务数占总任务数的比率为 50%,原子服务的可靠性均设定为 0.9,出现故障时,为瞬时性故障或者永久性故障的概率均为 0.5.实验结果仍取执行 100 次实验的平均值.

Table 1 Loss degree of modules

表 1 模块折损度

实验结果类型	缺少服务重试模块(%)	缺少服务复制模块(%)	缺少服务补偿模块(%)
故障排除率	0	0	28.5
故障处理时间	–2.1	10.2	–
组合最优度	7.6	1.8	–

由表 1 可以看出:如果缺少服务重试模块,组合最优度折损最厉害,达到了 7.6%.这是因为,在产生临时性故障时不能通过服务重试调用最初绑定的原子服务,而只能按照服务复制或者服务补偿机制排除故障,而这两个机制均会降低组合最优度.故障处理时间之所以会减少,是因为在产生临时性故障时直接采用了服务复制机制排除故障.从增加组合最优度 7.6%与降低故障处理时间 2.1%的程度来看,服务重试机制通过时间换取组合最优度的代价还是值得的.

如果缺少服务复制模块,故障处理时间增加 10.2%.其原因是:在出现不需要进行补偿的永久性故障时,只能通过服务补偿机制进行容错,需要进行不必要的一致性检测操作,且需要重新寻找绑定服务,所以增加了故障处理时间.组合最优度也有少许折损,是因为采用服务补偿机制对不需要进行补偿的出现故障的原子服务重新选择出的服务的 QoS 属性,与采用服务复制机制选取的最优复制服务的 QoS 属性相比略低,从而折损了一部分组合最优度.所以,FRFTA 中服务复制模块的存在会降低故障处理时间并增加组合最优度.

如果缺少服务补偿模块,故障排除率将降低 28.5%.这是因为,对于实验中出现的需要进行补偿的永久性故障将无法排除,也就是说,出现此类故障时,组合服务将无法执行完成,所以故障处理时间与组合最优度无法得到,其折损度也无法计算.

综上所述,FRFTA 在排除出现的不同故障的过程中,3 个模块均发挥了自己的作用,且能够相互配合,在取得最终较佳的实验结果的贡献上缺一不可.

2.5 结果讨论

结果讨论的目的是依据实验的结果阐述 FRFTA 的性能优于其他方法的深层原因,以便从更客观的角度分析和理解 FRFTA.

FRFTA 的性能之所以优于其他方法,从总体上来说,是因为其 3 个主要模块在处理出现的两类故障(临时性故障和永久性故障)时扮演着各自的角色,而且相互之间还起到了互补作用,因此能够具有较好的评估性能.从各个模块的角度来看,更详细的原因如下.

(1) 在融合网络环境下,可能因为无线信号或者有线线路受到瞬时的干扰,出现瞬时的网络不通的故障,或者是正在调用的服务所在的代理服务器出现了瞬时的系统故障,造成服务暂时不可被调用.此类故障的特点是在很短时间内就能够恢复,对于此类故障宜采取重试调用方法.本文中,服务重试机制采用模糊逻辑的方法得到调整后的重试次数.如果能在调整后的重试次数内成功调用到服务,则继续使用该服务,从而组合服务能够调用最初绑定的原子服务,使其具有较高的组合最优度.这也是在第 2.2 节的各类仿真对比中,FRFTA 方法始终能具有最高的组合最优度的原因.即:服务重试模块起到排除出现的临时性故障的同时,还使组合服务具有较高的组合最优度的作用.如果服务重试失败,证明故障为永久性故障,就需要考虑其他容错机制;

(2) 如果产生永久性故障的原子服务不具有原子特性,且其存在复制服务时,可以采用服务复制模块提供的服务复制机制进行容错,即:用拥有相同功能属性、不同 QoS 属性的复制服务替代出故障的服务,继续工作流的执行.但这类服务一般有多个,在制定复制方案时需要选择 QoS 属性较好的服务.但不同的 QoS 属性从不同的角度刻画了服务的特性,所以不能只从单个 QoS 属性选择复制服务.本文中的服务复制机制采用多属性决策的方法来综合评价服务,可根据组合服务对可靠性要求的高低选择若干个排在前面的服务作为复制服务.在组合服务的执行阶段,如果事先绑定的原子服务出现永久性故障,就先用贴近度最高的服务替代出故障服务:如果调用成功,则继续工作流的执行;而如果调用不成功,则用贴近度次之的服务替代.依照此规则,直至成功调用到复制服务,即,用服务复制机制排除出现的永久性故障;

(3) 如果产生的永久性故障的原子服务具有原子特性,或其不存在复制服务,则可以采用服务补偿模块提供的服务补偿机制进行容错,即,回溯到一个可以进行补偿的状态,从此状态开始,为具有原子特性的组合服务片段重新选择原子服务,然后继续执行工作流.在本文中,服务补偿机制通过改进的粒子群算法,在使重新寻找原子服务的时间降低的同时,也使服务组合片段的组合最优度值升高,从而增加整个组合服务的组合最优度.所以,通过服务补偿模块,在排除了需要补偿的永久性故障的同时,也能减少故障排除时间和使组合服务具有较高的组合最优度.

综上所述,FRFTA 独立于组合服务执行逻辑,只在出故障的时候才被触发.且其 3 个模块相互配合,在拥有较高的故障排除率的同时,也能缩短故障排除时间和拥有较高的组合最优度,所以其才能在实验对比中的结果优于其他 4 种方法.

3 相关工作

对于服务组合的容错方法,许多学者提出了相关方法或解决思路,并取得了较好的研究成果.下面将对一些与本文密切相关的研究成果予以分析.

文献[5,11]研究了带局部和全局约束的可靠面向服务系统的最优容错策略问题,提出了一个系统的和可扩展的框架,通过此框架,可以在组合服务的设计阶段选择最佳容错策略.文献把所研究问题建模为一个优化问题,同时把用户的要求转化为局部和全局约束,并把语义相关任务(task)的最佳容错策略的选择问题也形式化为优化问题的一个约束,并设计出一种启发式算法(FT-HEU)有效地求解优化问题.文献在服务组合的设计阶段,在对各个任务选择原子服务时,同时兼顾选择最优容错策略.但一些实验研究显示,大部分的故障出现在组合服务的执行阶段,所以应重点研究组合服务执行阶段的容错方法.

文献[16]引入一种外部 Web 服务,被称为故障避免服务(fault avoidance service,简称 FAS),其可以作为原子服务被直接包含进组合服务中.在组合服务的执行过程中,FAS 周期性地探测组合服务绑定的未被执行的原子服务的可用性.如果发现某个原子服务不可用,则在组合服务调用该服务前,FAS 用实现相同功能不同 QoS 的可

用原子服务替代不可用原子服务.该方法虽然避免了组合服务而去调用一个不可用的原子服务,但其不区分是临时性故障还是永久性故障.即使原子服务出现的是临时性故障,也要更新原子服务.如果在发现一个服务不可用且是临时性故障时,那么只要在组合服务真正调用该服务前能够从故障中恢复,就没有必要将其更新为其他服务,从而可以减少不必要的更新开销,且能使组合服务调用到最佳原子服务.

文献[12,14,36]采用复制策略(replication strategy),也称为冗余服务策略,来保证组合服务的可靠执行.服务复制是实现高可靠服务的一种有效方法.通过同时使用多个拥有相同功能属性的服务来实现同一个任务,在一个服务出现调用失败时,其他服务可以继续提供所需功能.文献[14]认为:在一个业务流程中,协调服务(coordination service)的不可用会直接影响所有合作伙伴(partners)的可用性,所以应该为协调服务提供更高的可用性.该文献在广域网环境下,采用服务复制机制,设计和实现了一种基础设施.该基础设施无缝地提高了协调服务的可用性.但该文献仅为协调服务提供高可用性,而为了保证融合网络环境下的组合服务的可靠执行,必须提高所有原子服务的可用性.文献[12]提出和实现了一种分布式复制策略评价和选择框架,基于此框架,可以对各种复制方案通过理论公式和实际实验进行系统的比较,方便用户找到最优复制方案.但此框架在参与构建复制策略的原子服务较少时选择最优复制方案计算量较少,但随着参与原子服务的数量的增加,计算量会呈指数级增长,会在很大程度上影响选择速度.文献[36]在文献[12]的基础上采用有向无环图(directed acyclic graph,简称 DAG)建模复制方案,通过 DAG 能够探索到所有可能的复制方案,然后采用改进遗传算法(genetic algorithm)寻找最优解.在参与复制策略的原子服务较多的情况下,执行速度也较快.但上述 3 个文献都是基于出故障的服务存在替代服务的假设,然而在一些情况下也许不存在冗余服务,如果这样,就不能采用复制策略进行容错.

文献[13]认为:虽然服务组合语言,即业务流程执行语言 BPEL 为从错误(error)处进行回滚(rollback)操作提供了补偿机制(compensation mechanism),但此类补偿存在很多问题,比如其在补偿后不能保证组合服务的功能属性.在文献[13]中提出了一种基于遗传算法的自动化算法,该方法通过计算可以得到在补偿后能够保证组合服务功能属性的恢复计划(recovery plan).给定一个拥有很大状态空间的组合服务,提出的方法不需要在组合服务的全状态空间中探索最优恢复计划,而是在部分状态空间寻找近似最优恢复计划,因此,该方法能够有效地选择恢复计划.此外,恢复计划是根据服务的 QoS 选择得到的,一个最优 QoS 的恢复计划能够保证组合服务有效地从故障的状态恢复过来.文献[15]提出了 EXTRA 机制(exception handing+transaction)来提高组合服务的可靠性,该机制是结合了异常处理(exception handing)和事务技术(transaction techniques)的混合容错机制.EXTRA 采用 8 类高层异常处理机制来修复发生在组合服务执行过程中的故障,同时也定义了一个基于服务状态转移的终止协议(service-transfer-based termination protocol,简称 STTP),如果产生了一个不可修复的故障,STTP 可以使组合服务终止在一个一致的状态.但上述两个文献提出的方法时间复杂度均偏高,如果将其应用到组合服务的实际容错中,会使组合服务的总执行时间变长,降低了用户体验.与上述研究不同,在本文中,组合服务执行逻辑与容错逻辑彼此独立.只有在执行逻辑中监测到故障的情况下,才会触发容错逻辑排除出现的故障.

4 结论与未来工作展望

本文提出了一种融合网络环境下快速可靠的服务组合容错方法.该方法在组合服务的执行阶段不断收集组合服务的执行状态:如果没有异常,容错逻辑就不阻碍组合服务的正常执行;而当发现某一原子服务出现故障时,则根据故障的类型,采用不同的机制进行容错.

- 如果为临时性故障,就采用服务重试机制进行容错,即,通过重复调用的方式探测故障原子服务的可用性;
- 如果出现故障的原子服务在重试持续时间内还是不能被成功地调用,本文就认为该原子服务出现了永久性故障,如果其不具有原子性,就采用服务复制机制进行容错;如果出故障的原子服务具有原子性,就采用服务补偿机制进行容错.

本文采用改进的 PSO 算法,快速地为组合服务片段寻找到最优原子服务,保证组合服务的继续正确执行.大

量的实验结果验证了本文所提方法的有效性.

虽然通过实验对比,我们的方法比其他方法更具有一定的优势,但仍有改进空间,例如:对于服务重试机制,我们假定采用了固定重试频率,只对重试次数进行了研究.而实际应用中,可能会根据组合服务特点选择不同的重试频率,所以需要进一步研究不同重试频率下对重试次数的控制.另外,如何在实际的融合网络环境中验证并部署 FRFTA,也是我们未来的研究工作.

致谢 在此,我们对提供 QWS 数据集的加拿大圭尔夫大学的 Eyhab Al-Masri 和 Qusay H.Mahmoud 博士以及提供 WS-DREAM 的香港中文大学的郑子彬博士和 Michael R. Lyu 教授表示感谢.

References:

[1] Zheng ZB, Zhang Y, Lyu MR. Distributed QoS evaluation for real-world Web services. In: Proc. of the 17th Int'l Conf. on Web Services (ICWS 2010). Miami: IEEE, 2010. 83−90. [doi: 10.1109/ICWS.2010.10]

[2] Hamadi R, Benatallah B. A Petri net-based model for Web service composition. In: Proc. of the 14th Australasian Database Conf. (ADC 2003), Vol.17. Darlinghurst: Australian Computer Society, 2003. 191−200. http://dl.acm.org/citation.cfm?id=820121

[3] Bultan T, Fu X, Hull R. Conversation specification: A new approach to design and analysis of e-service composition. In: Proc. of the 12th Int'l Conf. on World Wide Web (WWW 2003). New York: ACM Press, 2003. 403−410. [doi: 10.1145/775152.775210]

[4] Lucchi R, Mazzara M. A pi-calculus based semantics for WS-BPEL. The Journal of Logic and Algebraic Programming, 2007,70(1): 96−118. [doi: 10.1016/j.jlap.2006.05.007]

[5] Zheng ZB, Lyu MR. Selecting an optimal fault tolerance strategy for reliable service-oriented systems with local and global constraints. IEEE Trans. on Computers, 2015,64(1):219−232. [doi: 10.1109/TC.2013.189]

[6] Yu WD. A software fault prevention approach in coding and root cause analysis. Bell Labs Technical Journal, 1998,3(2):3−21. [doi: 10.1002/bltj.2101]

[7] Littlewood B. Stochastic reliability-growth: A model for fault-removal in computer-programs and hardware-designs. IEEE Trans. on Reliability, 1981,30(4):313−320. [doi: 10.1109/TR.1981.5221099]

[8] Randell B. System structure for software fault tolerance. In: Proc. of the Int'l Conf. on Reliable Software. New York: ACM Press, 1975. 437−449. [doi: 10.1145/800027.808467]

[9] Vergura S, Acciani G, Amoruso V. Inferential statistics for monitoring and fault forecasting of PV plants. In: Proc. of the IEEE Int'l Symp. on Industrial Electronics (ISIE 2008). Cambridge: IEEE, 2008. 2414−2419. [doi: 10.1109/ISIE.2008.4677264]

[10] Mansour HE, Dillon T. Dependability and rollback recovery for composite Web services. IEEE Trans. on Services Computing, 2011,4(4):328−339. [doi: 10.1109/TSC.2010.16]

[11] Zheng ZB, Lyu MR. A QoS-aware fault tolerant middleware for dependable service composition. In: Proc. of the 39th Int'l Conf. on the Dependable Systems & Networks (DSN 2009). Lisbon: IEEE, 2009. 239−248. [doi: 10.1109/DSN.2009.5270332]

[12] Zheng ZB, Lyu MR. A distributed replication strategy evaluation and selection framework for fault tolerant Web services. In: Proc. of the 15th Int'l Conf. on Web Services (ICWS 2008). Beijing: IEEE, 2008. 145−152. [doi: 10.1109/ICWS.2008.42]

[13] Tan TH, Chen M, André É, Sun J, Liu Y, Dong JS. Automated runtime recovery for QoS-based service composition. In: Proc. of the 23rd Int'l Conf. on World Wide Web (WWW 2014). New York: ACM Press, 2014. 563−574. [doi: 10.1145/2566486.2568048]

[14] Salas J, Perez-Sorrosal F, Patiño-Martínez M, Jiménez-Peris R. WS-Replication: A framework for highly available Web services. In: Proc. of the 15th Int'l Conf. on World Wide Web (WWW 2006). New York: ACM Press, 2006. 357−366. [doi: 10.1145/1135777.1135831]

[15] Liu A, Li Q, Huang L, Xiao M. Facts: A framework for fault-tolerant composition of transactional Web services. IEEE Trans. on Services Computing, 2010,3(1):46−59. [doi: 10.1109/TSC.2009.28]

[16] Gülcü K, Sözer H, Aktemur B, Ercan AÖ. Fault masking as a service. Software: Practice and Experience, 2014,44(7):835−854. [doi: 10.1002/spe.2255]

[17] Fan XQ, Jiang CJ, Wang JL, Pang SC. Random-QoS-Aware reliable Web service composition. Ruan Jian Xue Bao/Journal of Software, 2009,20(3):546−556 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/3339.htm [doi: 10.3724/SP.J.1001.2009.03339]

[18] Wu GQ, Wei J, Huang T. A dynamic QoS assessment approach for internetware with uncertainty reasoning. Ruan Jian Xue Bao/Journal of Software, 2008,19(5):1173−1185 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/20080509.htm [doi: 10.3724/SP.J.1001.2008.01173]

[19] Liu XZ, Huang G, Mei H. Consumer-Centric service aggregation: Method and its supporting framework. Ruan Jian Xue Bao/Journal of Software, 2007,18(8):1883−1895 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/20070804.htm [doi: 10.1360/jos181883]

[20] Wang SG, Sun QB, Zou H, Yang FC. Particle swarm optimization with skyline operator for fast cloud-based Web service composition. Mobile Networks and Applications, 2013,18(1):116−121. [doi: 10.1007/s11036-012-0373-3]

[21] Wang SG, Zheng ZB, Sun QB, Zou H, Yang FC. Cloud model for service selection. In: Proc. of the Conf. on Computer Communications Workshops (INFOCOM WORKSHOPS 2011). Shanghai: IEEE, 2011. 666−671. [doi: 10.1109/INFCOMW.2011.5928896]

[22] Ma Y, Wang SG, Sun QB, Yang FC. Web service quality metric algorithm employing objective and subjective weight. Ruan Jian Xue Bao/Journal of Software, 2014,25(11):2473−2485 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4508.htm [doi: 10.13328/j.cnki.jos.004508]

[23] Tao F, Zhao D, Hu Y, Zhou Z. Resource service composition and its optimal-selection based on particle swarm optimization in manufacturing grid system. IEEE Trans. on Industrial Informatics, 2008,4(4):315−327. [doi: 10.1109/TII.2008.2009533]

[24] Wang LX. Adaptive Fuzzy Systems and Control: Design and Stability Analysis. Upper Saddle River: Prentice-Hall, 1994.

[25] Lee CC. Fuzzy logic in control systems: Fuzzy logic controller. Part II. IEEE Trans. on Systems, Man and Cybernetics, 1990, 20(2):419−435. [doi: 10.1109/21.52552]

[26] Yu H, Liu Z, Li YJ. Key nodes in complex networks identified by multi-attribute decision-making method. Acta Physica Sinica, 2013,62(2):20204−020204 (in Chinese with English abstract). [doi: 10.7498/aps.62.020204]

[27] Saaty TL. Decision making with the analytic hierarchy process. Int'l Journal of Services Sciences, 2008,1(1):83−98. [doi: 10.1504/IJSSci.2008.01759]

[28] Wang SG, Sun QB, Yang FC. Web service dynamic selection by the decomposition of global QoS constraints. Ruan Jian Xue Bao/Journal of Software, 2011,22(7):1426−1439 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/3842.htm [doi: 10.3724/SP.J.1001.2011.03842]

[29] Alrifai M, Risse T. Combining global optimization with local selection for efficient QoS-aware service composition. In: Proc. of the 18th Int'l Conf. on World Wide Web (WWW 2009). New York: ACM Press, 2009. 881−890. [doi: 10.1145/1526709.1526828]

[30] Zeng L, Benatallah B, Ngu AH, Dumas M, Kalagnanam J, Chang H. Qos-Aware middleware for Web services composition. IEEE Trans. on Software Engineering, 2004,30(5):311−327. [doi: 10.1109/TSE.2004.11]

[31] Zeng L, Benatallah B, Dumas M, Kalagnanam J, Sheng QZ. Quality driven Web services composition. In: Proc. of the 12th Int'l Conf. on World Wide Web (WWW 2003). New York: ACM Press, 2003. 411−421. [doi: 10.1145/775152.775211]

[32] Shi Y, Eberhart R. A modified particle swarm optimizer. In: Proc. of the Int'l Conf. on Evolutionary Computation (CEC'98). Anchorage: IEEE, 1998. 69−73. [doi: 10.1109/ICEC.1998.699146]

[33] Van Den Bergh. An analysis of particle swarm optimizers [Ph.D. Thesis]. Pretoria: University of Pretoria, 2006.

[34] Al-Masri E, Mahmoud QH. Investigating Web services on the World Wide Web. In: Proc. of the 17th Int'l Conf. on World Wide Web (WWW 2008). New York: ACM Press, 2008. 795−804. [doi: 10.1145/1367497.1367605]

[35] Zhang YL, Zheng ZB, Lyu MR. Exploring latent features for memory-based QoS prediction in cloud computing. In: Proc. of the 30th IEEE Symp. on Reliable Distributed Systems (SRDS 2011). Madrid: IEEE, 2011. 1−10. [doi: 10.1109/SRDS.2011.10]

[36] Liu A, Li Q, Huang L. Quality driven Web services replication using directed acyclic graph coding. In: Proc. of the 12th Int'l Conf. on Web Information System Engineering (WISE 2011). Berlin: Springer-Verlag, 2011. 322−329. [doi: 10.1007/978-3-642-24434-6_28]

附中文参考文献:

[17] 范小芹,蒋昌俊,王俊丽,庞善臣.随机 QoS 感知的可靠 Web 服务组合.软件学报,2009,20(3):546−556. http://www.jos.org.cn/1000-9825/3339.htm [doi: 10.3724/SP.J.1001.2009.03339]

[18] 吴国全,魏峻,黄涛.基于非确定性推理的网构软件服务质量动态评估方法.软件学报,2008,19(5):1173−1185. http://www.jos.org.cn/1000-9825/20080509.htm [doi: 10.3724/SP.J.1001.2008.01173]

[19] 刘譞哲,黄罡,梅宏.用户驱动的服务聚合方法及其支撑框架.软件学报,2007,18(8):1883−1895. http://www.jos.org.cn/1000-9825/20070804.htm [doi: 10.1360/jos181883]

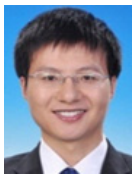
[22] 马友,王尚广,孙其博,杨放春.一种综合考虑主客观权重的 Web 服务 QoS 度量算法.软件学报,2014,25(11):2473−2485. http://www.jos.org.cn/1000-9825/4508.htm [doi: 10.13328/j.cnki.jos.004508]

[26] 于会,刘尊,李勇军.基于多属性决策的复杂网络节点重要性综合评价方法.物理学报,2013,62(2):20204−020204. [doi: 10.7498/aps.62.020204]

[28] 王尚广,孙其博,杨放春.基于全局 QoS 约束分解的 Web 服务动态选择.软件学报,2011,22(7):1426−1439. http://www.jos.org.cn/1000-9825/3842.htm [doi: 10.3724/SP.J.1001.2011.03842]

张俊娜(1979−),女,河南周口人,博士生,副教授,主要研究领域为服务计算.

孙其博(1975−),男,博士,副教授,CCF 专业会员,主要研究领域为网络服务与网络智能化,物联网应用技术.



王尚广(1982−),男,博士,副教授,博士生导师,CCF 高级会员,主要研究领域为服务计算,移动云计算,车联网,网络安全.

杨放春(1957−),男,博士,教授,博士生导师,CCF 杰出会员,主要研究领域为融合通信软件,网络安全,网络智能化.