

# 基于梯度的改进动态 Web 服务选择算法

杨丽琴<sup>1 2</sup>, 康国胜<sup>2</sup>

(1. 上海中医药大学 计算机综合教研室, 上海 201203;

2. 复旦大学 计算机科学技术学院, 上海 201203)

**摘 要:** 随着 Web2.0 的迅速发展, 互联网上发布的 Web 服务越来越多, 不同服务供应商提供的服务通过整合以提供功能更强大的组合服务。每个服务节点上功能相似的 Web 服务的 QoS (quality of service) 不同, 因此, QoS 全局最优动态 Web 服务选择成为了服务组合中的一大挑战。在传统的粒子群优化算法的基础上引入梯度的思想, 文中设计了一种用于解决动态 Web 服务选择问题的改进算法 gPSO-GODSS。将问题抽象为带 QoS 约束的多目标组合优化问题, 并进一步将其向单目标转化。利用梯度的方法改进粒子群算法的更新速度, 从而改进算法的收敛速度, 最终产生一组满足约束条件的优化服务组合流程集。理论分析和实验结果证明了该算法的可行性和有效性, 且 gPSO-GODSS 算法收敛的执行效率和收敛速度均优于已有的 PSO-GODSS 算法。

**关键词:** 服务组合; 服务选择; QoS 全局优化; 梯度; 粒子群算法

中图分类号: TP311

文献标识码: A

文章编号: 1673-629X(2019)05-0032-06

doi: 10.3969/j.issn.1673-629X.2019.05.007

## An Improved Dynamic Web Services Selection Algorithm Based on Gradient

YANG Li-qin<sup>1 2</sup>, KANG Guo-sheng<sup>2</sup>

(1. Computer Teaching and Research Office, Shanghai University of Traditional Chinese Medicine, Shanghai 201203, China;

2. School of Computer Science, Fudan University, Shanghai 201203, China)

**Abstract:** As the rapid development of Web2.0, there are more and more Web services on the Internet. Web services from different service providers can be integrated to form a composited service. As the Web services on each node with similar functions have different QoS (quality of service), dynamic Web service selection with global QoS optimization becomes a critical issue in Web service composition. In order to solve the problem, based on the basic particle swarm optimization (PSO) and the thought of gradient, we propose a gPSO-GODSS (global optimal of dynamic Web services selection based on PSO with gradient). We abstract the original Web service selection problem into a multi-objective services composition optimization with global QoS constraints, which is further transformed into a single-object. The gradient method is used to improve the update speed of basic PSO, thus improving the convergence speed of the PSO-GODSS algorithm. Then the improved PSO is exploited to produce a set of optimal services composition process with QoS constraints. Theoretical analysis and experimental results indicate the feasibility and efficiency of this algorithm, and the execution efficiency and convergence rate of gPSO-GODSS algorithm are both better than the existing PSO-GODSS algorithm.

**Key words:** service composition; service selection; QoS global optimization; gradient; PSO

### 1 概述

随着服务计算技术的迅猛发展, 互联网上的可用服务的数量迅速增长。然而, 单个 Web 服务提供的功能往往比较单一, 为了实现更复杂的功能, 需要将共享的多个 Web 服务组合成为用户所需要的复杂服务, 满

足用户定制化的需求。因此, Web 服务组合优化问题成为服务计算研究的一个挑战。

支持 QoS 的 Web 服务选择主要分为三大类。第一类, 研究单个服务请求的 Web 服务优化选择<sup>[1-3]</sup>; 第二类, 研究面向多个相同或相似功能服务请求的全局

收稿日期: 2018-05-10

修回日期: 2018-09-18

网络出版时间: 2019-03-06

基金项目: 国家自然科学基金(60873115); 上海中医药大学预算内资助项目(2016YSN81)

作者简介: 杨丽琴(1982-), 女, 硕士, 讲师, 研究方向为服务计算、业务流程管理。

网络出版地址: <http://kns.cnki.net/kcms/detail/61.1450.TP.20190306.0907.024.html>

优化 Web 服务选择<sup>[4-5]</sup>; 第三类, 研究组合服务中动态 Web 服务优化选择<sup>[6-7]</sup>。文中主要考虑第三类情形。通常, 对一个复杂的功能, 会建立一个业务流程来实现, 该业务按一定的流程结构组成, 每个流程节点通过调用服务来完成, 最终整个流程就形成了一个组合服务。然而, 互联网环境复杂多变, 带来 Web 服务质量的动态变化。即使 Web 服务的功能相同或相似, 但是它们的服务质量(quality of service, QoS)参数(如执行费用、信誉等级、执行时间等)可能不同。而且, 同一服务在不同的时间点服务质量也可能不同。因此, 如何从满足各节点功能需求的服务中选出最终的服务, 形成一个组合服务来完成用户的定制化需求成为一个难题, 该问题文中称为动态 Web 服务选择。尤其是在当前大数据、云计算环境下, 服务选择问题显得尤为严峻。

目前, 关于组合服务中的动态 Web 服务选择的研究大多是基于 QoS 局部最优的原则<sup>[8]</sup>, 即独立地为流程中的每个节点选择一个满足业务功能且服务的综合服务质量最优的服务。这种方法没有考虑单个服务节点质量对组合服务中的其他节点质量的影响, 即考虑了满足局部约束但未考虑满足全局约束, 因此无法解决多目标多约束的 QoS 全局优化问题。比如: 对一部分服务质量需要满足约束条件即可, 而对另一些服务质量需要尽量优化。一个具体的例子如下: 在对组合流程中的费用和执行时间做约束的条件下使得可靠性和信誉等级最高。针对组合服务中的 QoS 全局最优 Web 服务选择问题, 目前已有一些相关工作<sup>[9-12]</sup>, 但均具有以下缺点: 集中在解决局部优化问题以及单目标问题, 无法解决多目标的优化问题; 只提供单个的优化方案, 用户无法从多个优化方案中进行选择; 计算量大、算法复杂度高, 随着流程中业务节点以及节点对应的候选服务的增多, 组合优化问题的求解效率将大大降低。因此, 使用启发式算法来提高 Web 服务组合优化问题的求解是一个可行方案。针对以上服务组合优化选择算法的缺点, 文献[13]基于多目标遗传算法提出了一种动态选择 QoS 全局最优化算法。首先将服务动态组合优化建模为一个带 QoS 约束的多目标优化问题; 然后, 采用遗传算法启发式原理求解多目标优化问题; 最后, 产生一组满足约束的 Pareto 优化服务组合方案。这种启发式算法解决了多目标优化求解的问题, 然而算法的执行效率还需要改进。基于粒子群优化算法, 文献[6]出了解决动态 Web 服务优化选择问题的算法 PSO-GODSS, 该算法不仅解决了前述的 3 个问题, 并改进了动态 Web 服务选择的执行效率。实验结果证明相比遗传算法, PSO-GODSS 算法的执行效率和收敛速度均较优, 但依然还有待提高。尤其在

当前的大数据环境下, Web 服务的规模非常庞大, 算法的求解效率显得尤为重要。

在文献[6]的基础上, 针对多目标多约束服务组合优化问题, 文中引入梯度的方法改进传统的粒子群算法, 提出了解决该问题的改进算法 gPSO-GODSS (gradient PSO-GODSS)。相对于文献[6]中的 PSO-GODSS 算法, 该算法收敛速度更快, 能够快速找到全局最佳服务组合决策。

## 2 服务组合优化问题

定义(多约束多目标的优化路径, 即 MCOOP)<sup>[6]</sup>: 对于图  $G = (N, E, W)$ , 其中  $N$  为顶点集,  $E$  为链路集,  $W$  为链路权重, 设存在  $k(k \geq 2)$  个约束  $C_i(i = 1, 2, \dots, k)$ , 从源点  $S$  到目的点  $T$  的全部路径称之为解空间的集合  $\Omega$ , 对  $\forall P \in \Omega$ , 具有  $m(m \geq 2)$  个非负的性能度量指标  $f_1, f_2, \dots, f_m$ , 若  $P^* \in \Omega, \forall P \in \Omega(P \neq P^*)$ , 在  $P$  和  $P^*$  均满足约束  $C_i$  的条件下, 对于所有的度量准则均使得  $f_i(P^*) \geq f_i(P)$ , 至少存在一个严格不等式  $f_i(P^*) > f_i(P)(i = 1, 2, \dots, m)$  成立, 则称  $P^*$  为多约束条件下多目标问题的 Pareto 优化解。其中  $\geq$  和  $>$  是两个偏序关系, 分别表示不劣于和优于关系。

服务节点(service node, SN)是一个抽象概念, 各服务节点只包含对 Web 服务的功能描述和接口信息, 不绑定具体的 Web 服务。服务组合按照流程模型的方式来建模, 由多个业务或服务节点按照一定的顺序结构组成, 每个服务节点对应一个 Web 服务群(service group, SG)。同一服务群中的多个 Web 服务具有相同或相似的功能, 而 QoS 不尽相同。一个包含  $n$  个服务节点的串行结构服务组合流程模型如图 1 所示。图中, 第  $i$  个服务节点  $SN_i$  含有  $n_i$  个服务, 每个业务节点对应一组具有相同或相似功能的候选 Web 服务, 称之为一个服务群, 将其表示为  $SG_i = (S_{i,1}, S_{i,2}, \dots, S_{i,n_i})(i = 1, 2, \dots, n)$ 。



图 1 串行服务组合流程

基于动态 Web 服务选择的组合服务 QoS 全局优化问题, 即在业务流程的执行过程中, 在各业务节点对应的候选服务集合中分别选择针对全局是优化的服务实例, 使得整个流程可以执行完成, 实现整体的业务流程功能, 同时满足组合服务流程的全局服务质量约束, 使得多目标达到最优。然而, 该问题的求解是一个挑战。

结合前面介绍的 MCOOP 问题, 可把 Web 服务组合中的动态服务选择 QoS 全局优化问题进行转化, 将其建模为一个求解服务组合流程图中带 QoS 条件约

束的多目标优化路径问题,从而解决服务组合优化问题。每个服务节点所对应的服务群中的服务对应图中的顶点。为方便分析,在图中添加两个虚节点  $S$  和  $T$ ,则图 1 对应的串行服务组合流程如图 2 所示。

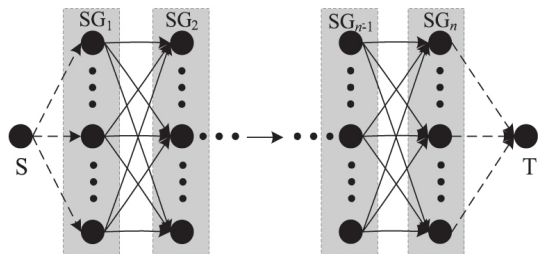


图 2 服务组合流程(串行结构)

文中将提出求解动态 Web 服务选择 QoS 全局优化问题的 gPSO-GODSS 算法。如前所述,服务组合流程常建模为一个基于 Web 服务的业务流程。业务流

程由不同的流程结构组成,常用的流程结构主要有如下四大类型:串联模式、并联模式、选择模式和循环模式。大多数服务组合流程均可由这四种基本模型组合而成。针对这四种基本模型的服务组合 QoS 参数计算方法,已有一些相关工作<sup>[14-15]</sup>。为了计算组合服务在各维度的综合质量,文中对四种基本模式的 QoS 进行聚合<sup>[6]</sup>。下面,以执行时间  $T$ 、执行费用  $C$ 、信誉等级  $Rep$  和可靠性  $R$  四种服务质量为例介绍其聚合方法。在组合服务  $CS$  中, $S_i$  为执行组合服务中满足单个业务节点功能的具体服务, $S_i$  和  $CS$  的 QoS 服务质量分别建模表示为  $Q(S_i) = (T_i, C_i, Rep_i, R_i)$ 、 $Q(cs) = (T_{cs}, C_{cs}, Rep_{cs}, R_{cs})$ ,四种基本模式的 QoS 参数聚合方法如表 1 所示。其中,设选择模式中的第  $i$  个分支被选择的概率为  $\alpha_i$ ,循环模式中的循环次数为  $k$ 。

表 1 四种基本模式的 QoS 参数计算方法

基本模式	QoS 参数计算方法
串联模式	$T_{cs} = \sum_{i=1}^n T_i, C_{cs} = \sum_{i=1}^n C_i, Rep_{cs} = \sum_{i=1}^n Rep_i/n, R_{cs} = \prod_{i=1}^n R_i$
并联模式	$T_{cs} = \max\{T_i\}, C_{cs} = \sum_{i=1}^n C_i, Rep_{cs} = \sum_{i=1}^n Rep_i/n, R_{cs} = \min\{R_i\}$
选择模式	$T_{cs} = \sum_{i=1}^n T_i \alpha_i, C_{cs} = \sum_{i=1}^n C_i \alpha_i, Rep_{cs} = \sum_{i=1}^n Rep_i \alpha_i, R_{cs} = \prod_{i=1}^n R_i \alpha_i$
循环模式	$T_{cs} = k \sum_{i=1}^n T_i, C_{cs} = k \sum_{i=1}^n C_i, Rep_{cs} = \sum_{i=1}^n Rep_i/n, R_{cs} = \prod_{i=1}^n R_i$

### 3 gPSO-GODSS 算法描述

#### 3.1 MCOOP 问题描述

在 MCOOP 问题中,将信誉等级  $Rep$  和可靠性  $R$  两个质量属性作为约束,要求其值大于一定的值;目标则是使组合服务的执行费用  $C(P)$  和执行时间  $T(P)$  最小化。因此,该问题可形式化为:

$$\begin{aligned} \min f(P) = & \{ [T(P), C(P)] \\ \text{s.t.} & \begin{cases} Rep(P) \geq Rep_0 \\ R(P) \geq R_0 \end{cases} \end{aligned} \quad (1)$$

其中,  $T(P)$ 、 $C(P)$ 、 $Rep(P)$  和  $R(P)$  分别表示服务组合流程的 QoS 聚合公式,根据第 1 节表 1 中的 QoS 聚合方法计算。

式 1 表示使  $T(P)$  和  $C(P)$  两个目标函数同时最小化。在实际场景中, QoS 属性的个数可能多于 4 种,可根据具体的场景选择合适的 QoS 参数作为优化模型的目标属性和约束属性。因此,文中的 MCOOP 模型具有可扩展性。

#### 3.2 MCOOP 问题转化为单目标问题

在解空间中寻找多目标的优化解,需要有一个评价机制来评估可行解的好坏。文中使用欧氏距离来构造评价函数。首先分别求解单目标的解,将其作为理

想点,然后计算可行解对应的组合服务的综合执行时间和执行费用到理想点的欧氏距离作为评价可行解的适应值函数,这样便将多目标问题转化为单目标问题。最后,利用单目标的求解方法求出最优解,并把这些最优解作为多目标的最优解。构造的 MCOOP 问题的评价函数如下:

$$\begin{aligned} \min f(P) = & [(T(P) - T^*)^2 + (C(P) - C^*)^2]^{\frac{1}{2}} \\ \text{s.t.} & \begin{cases} Rep(P) \geq Rep_0 \\ R(P) \geq R_0 \end{cases} \end{aligned} \quad (2)$$

其中,  $(T^*, C^*)$  为一个理想点。由此可知,  $N$  个目标就对应一个由  $N$  维向量表示的理想点。文中的理想点  $(T^*, C^*)$  是对  $T(P)$  和  $C(P)$  在各约束条件下分别求出的单目标最优值,它们的计算方法如下:

$$T^* = \min\{T(P) \mid Rep(P) \geq Rep_0, R(P) \geq R_0\} \quad (3)$$

$$C^* = \min\{C(P) \mid Rep(P) \geq Rep_0, R(P) \geq R_0\} \quad (4)$$

为了求解理想点,根据组合服务 QoS 属性的聚合计算方法可得,在各候选服务集合中均选择执行时间最短的服务形成的组合服务的综合执行时间为  $T^*$ ;在各候选服务集合中均选择执行费用最低的服务形成

的组合服务的综合执行费用为  $C^*$ , 从而得到组合服务的综合执行时间和执行费用的理想值。

### 3.3 PSO 算法

粒子群优化算法(PSO)是一种进化计算技术,由 Russel Eberhart 和 James Kennedy 于 1995 年提出,源于对鸟群捕食的行为研究<sup>[16]</sup>。算法最初受飞鸟和鱼类等集群活动规律的启发,模拟种群间个体协作来搜索问题的最优解。该算法的优势是易于实现同时又有深刻的智能背景,既适合科学研究,又适合工程应用,且没有许多参数的调节。关于 PSO 的更多详情可参考文献[17-18]。MCOOP 问题是一个 NP 难问题<sup>[17]</sup>,因此使用智能进化算法求解是一个合适的选择。PSO 算法作为一种启发式的智能优化方法,具有群体寻优、并行计算等特点。因此,可用于各种 NP 难问题的近似求解。

PSO 算法的初始种群为一群随机粒子,在每一次迭代中粒子通过跟踪局部“极值”和全局“极值”(  $g_{best}$  和  $p_{best_i}$  )来更新粒子的速度和位置,其迭代公式如下:

$$v_i(t+1) = wv_i(t) + c_1r_1(p_{best_i} - x_i(t)) + c_2r_2(g_{best} - x_i(t)) \quad (5)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (6)$$

其中,  $r_1$  和  $r_2$  是(0,1)之间的随机数;  $c_1$  和  $c_2$  是两个学习因子,分别表示将每个微粒向局部最佳位置和全局最佳位置移动的统计加速项的权重。式5的第一项为记忆项,第二项为自身认知项,第三项为群体认知项。为了使粒子维持一定的惯性,在原来速度基础上乘以一个惯性权重因子  $w$ ,使粒子既能有原来的运动趋势又能探索新的区域。 $w$  值越大,全局寻优能力越强,局部寻优能力越弱; $w$  值越小,则反之。 $w$  采用较多的是线性递减权重策略进行设置,其公式如下:

$$w = w_{max} - \frac{w_{max} - w_{min}}{iter_{max}} \times iter \quad (7)$$

其中,  $w_{max}$  和  $w_{min}$  的经典取值应为:  $w_{max} = 0.9$ ,  $w_{min} = 0.4$ 。

在每一维,粒子都有一个最大的限制速度  $v_{max}$ ,决定当前位置与最好位置之间区域的分辨率(或精度)。当粒子在某一维度的速度超过了最大限制速度  $v_{max}$ ,则将其速度设定为最大的限制速度  $v_{max}$ 。因为粒子运动速度过快,则单次迭代时越过的位移会比较大,从而容易越过极小值;相反,如果粒子的运动速度太慢,则单次迭代移动的位移太小,一方面可能越陷入局部极小值,另一方面可能导致长时间无法收敛。

### 3.4 gPSO-GODOSS 算法设计

假设组合服务流程模型中有  $m$  个业务节点,第  $i$  个业务节点对应  $n_i$  个服务实例,即  $SG_i = (S_{i1}, S_{i2}, \dots, S_{in_i})$ 。其中  $S_{ij}$  为第  $i$  个业务节点对应的某个服务实例

的编号。第  $i$  个业务节点选择的服务实例表示为  $x_i = S_{ik}$ , 则最终的组合服务方案可形式化为一个  $m$  维的向量  $x_i = (x_{i1}, x_{i2}, \dots, x_{im})$ , 其中  $x_{ik} \in [1, n_k]$ 。

文中考察的速度、位移模型均是离散型数据,因此需要对  $x_i$  的编码进行离散化,改进原始粒子群算法中粒子的速度、位移模型,保证粒子在整数空间内移动。具体改进的公式如下:

$$v_{id}(t+1) = \text{int}(wv_{id}(t)) + \varnothing_1 + \varnothing_2 \quad (8)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \quad (9)$$

其中,  $\varnothing_1$  和  $\varnothing_2$  为均匀分布的整数。 $\varnothing_1 \in [a_1, b_1]$ ,  $p\{\varnothing_1\} = 1/m_1$ ,  $m_1 = \lfloor b_1 - a_1 + 1 \rfloor$ ;  $\varnothing_2 \in [a_2, b_2]$ ,  $p\{\varnothing_2\} = 1/m_2$ ,  $m_2 = \lfloor b_2 - a_2 + 1 \rfloor$ , 其中

$$a_1 = \begin{cases} 0 & p_{id} > x_{id}(t) \\ c_1r_1(p_{id} - x_{id}(t)) & p_{id} \leq x_{id}(t) \end{cases}$$

$$b_1 = \begin{cases} c_1r_1(p_{id} - x_{id}(t)) & p_{id} > x_{id}(t) \\ 0 & p_{id} \leq x_{id}(t) \end{cases}$$

$$a_2 = \begin{cases} 0 & p_{gd} > x_{gd}(t) \\ c_2r_2(p_{gd} - x_{gd}(t)) & p_{gd} \leq x_{gd}(t) \end{cases}$$

$$b_2 = \begin{cases} c_2r_2(p_{gd} - x_{gd}(t)) & p_{gd} > x_{gd}(t) \\ 0 & p_{gd} \leq x_{gd}(t) \end{cases}$$

式8和式9反映了PSO进化计算的基本思想,并将进化控制在整数空间内。需注意:两式表示的是粒子在某一维的速度和位置变化。式8和式9使得PSO可应用到离散的情形。收敛速度快是PSO的一个优点,但若在起始阶段过快,反而可能会收敛不到全局最优解。为保证PSO不仅收敛速度快,且收敛到全局最优解,文中基于梯度的思想进行改进,指导速度的变化。将粒子的变化速率作为梯度,定义如下:

$$\nabla f(x_i(t), x_i(t-1)) = \begin{pmatrix} \frac{f(x_i(t)) - f(x_i(t-1))}{x_{i1}(t) - x_{i1}(t-1)} \\ \dots \\ \frac{f(x_i(t)) - f(x_i(t-1))}{x_{id}(t) - x_{id}(t-1)} \end{pmatrix} \quad (10)$$

将粒子  $x_i$  在第  $d$  维的梯度记为  $\varnothing_3$ , 则

$$\varnothing_3 = \frac{f(x_i(t)) - f(x_i(t-1))}{x_{id}(t) - x_{id}(t-1)} \quad (11)$$

将梯度信息引入式8,可得:

$$v_{id}(t+1) = \text{int}(wv_{id}(t) + \alpha(t)(\varnothing_1 + \varnothing_2) + (1 - \alpha(t))c_3\varnothing_3) \quad (12)$$

式12中包含了梯度的信息,其中  $\alpha(t)$  和  $1 - \alpha(t)$  用来平衡PSO和梯度,  $c_3$  为一个给定的常数。将结合梯度信息的PSO称为gPSO。文中根据  $x_i$  采用Sigmoid函数来定义,如式13所示,函数图像如图3所示。



$$\alpha(t) = \frac{1}{1 + \exp(-f(x_i(t)))} \quad (13)$$

由此可见,  $\alpha(t)$  的值和  $x_i$  相关。当粒子的  $f(x_i(t))$  越大时,  $\alpha(t)$  越大, 表示 gPSO 在一个大的解区域进行搜索; 当粒子的  $f(x_i(t))$  越小时,  $\alpha(t)$  越小, 表示 gPSO 在一个小的解区域进行搜索, 使得 gPSO 较早达到收敛, 且不跳过最优解。

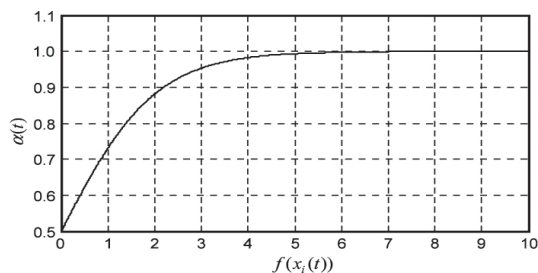


图3 Sigmoid 函数图像

基于前面的 QoS 属性聚合方法和位置向量  $x_i$  的编码表示, 以及基于梯度改进的速度、位移模型, 结合基本 PSO 的智能进化思想, 文中提出求解 MCOOP 问题的改进算法 gPSO-GODOSS, 具体的过程描述如下:

Step1: 将多目标问题向单目标优化问题转化;

Step2: 初始化所有粒子的速度和位置, 并给出参数赋值;

Step3: 根据粒子的位置解码成对应的服务组合方案, 计算组合服务的综合信誉等级和可靠性, 判断其是否满足约束条件。若满足, 则按照评价函数计算该粒子的适应值; 否则, 赋予该粒子一个最差的适应值即可, 并重新优化;

Step4: 计算粒子  $x_i$  的局部最佳位置  $p_{best_i}$ 。具体方法如下: 将其适应值与其本身经历过的局部最佳位置  $p_{best_i}$  的适应值进行比较, 若更优, 则将  $x_i$  的值赋给  $p_{best_i}$ ;

Step5: 计算粒子  $x_i$  的全部最佳位置  $g_{best}$ 。具体方法如下: 将其适应值与所有粒子经历过的全局最佳位置  $g_{best}$  的适应值进行比较, 若更优, 则将  $x_i$  的值赋给  $g_{best}$ ;

Step6: 更新粒子的速度和位置, 然后检查新位置和速度的范围, 限制粒子的搜索范围和最大速度;

Step7: 循环计算所有粒子, 即重复 Step3~Step6;

Step8:  $t = t + 1$ ;

Step9: 判断是否满足终止条件。若不满足, 则返回 Step3。

gPSO-GODOSS 的时间复杂度为  $O(mN^2T)$ , 其中  $T$  为迭代次数,  $N$  为粒子群规模,  $m$  为抽象服务数。

#### 4 实验结果与分析

针对文献[6]中提出的组合服务流程例子进行比

较实验, 其中计算机配置为 Pentium D 3 400 MHz 处理器 4 G 内存, Windows 7 操作系统, 算法用 Matlab7.0 实现。仿真实验的流程如图 4 所示。

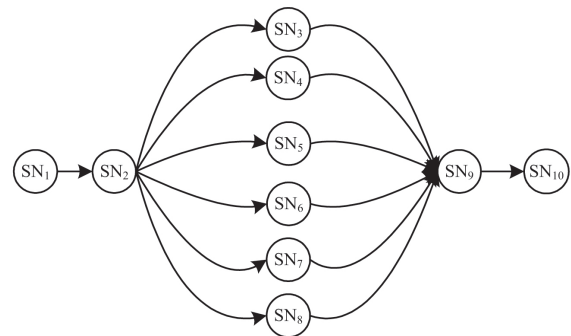


图4 服务组合流程

图 4 中各服务节点对应一个候选服务集合, 每个候选服务集合中的服务实例的 QoS 各属性值随机生成, 并采用极大极小方法标准化。根据第 2 节中服务组合流程基本模式及 QoS 聚合方法, 构造适应值函数和约束条件。实验中设  $Rep_0 = 0.8$ ,  $R_0 = 0.5$ 。其他参数设置如下:  $c_1 = c_2 = 2$ ,  $c_3 = 0.05$ ,  $iter_{max} = 100$ ,  $w$  按式 7 进行计算。实验中的候选服务集合的服务数量规模为 10, 迭代次数分别为 100、200、300、400。文中根据执行时间和收敛速度来评价 gPSO-GODOSS 算法的可行性和有效性, 与文献[6]中的多目标粒子群算法 PSO-GODOSS 的执行结果进行对比及分析。

首先, 比较每一代种群的平均适应值。如图 5 所示, PSO-GODOSS 在第 63 代收敛到最优值, 而 gPSO-GODOSS 在第 34 代已收敛到最优值, 表明 gPSO-GODOSS 的收敛速度优于 PSO-GODOSS 的收敛速度。因此, 从收敛速度上看, gPSO-GODOSS 优于 PSO-GODOSS。

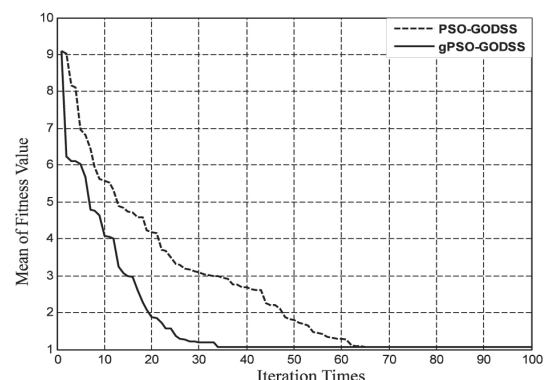


图5 算法收敛速度的比较

下面比较算法收敛的执行效率, 如图 6 所示。在相同迭代次数下, gPSO-GODOSS 的执行时间和 PSO-GODOSS 的执行时间非常相近。但由图 5 可知, gPSO-GODOSS 的收敛速度较快, 找到最优解时的迭代次数更少, 故达到收敛的时间开销会更短。因此, 从收敛的执行效率上说明了 gPSO-GODOSS 更具可行性。

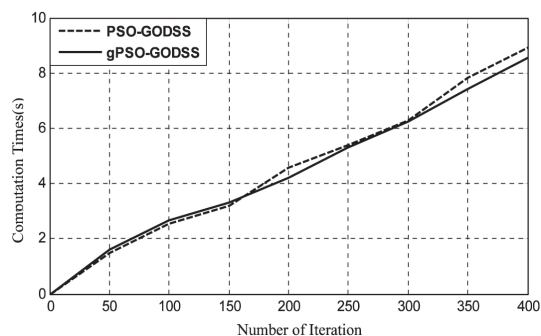


图6 算法执行时间比较

## 5 结束语

针对组合服务流程中 QoS 全局最优动态 Web 服务选择问题, 基于粒子群算法及梯度下降的思想, 提出了求解该问题的 gPSO-GODSS 改进算法。利用粒子群算法的智能优化原理进行优化, 通过在 PSO 的速度更新公式中引入梯度的方法, 不仅保证算法快速收敛且收敛到全局最优解。实验结果证明了 gPSO-GODSS 算法的可行性和有效性, 且算法收敛的执行效率和收敛速度均优于传统的离散粒子群算法。

## 参考文献

- [1] KANG Guosheng, LIU Jianxun, TANG Mingdong, et al. Web service selection algorithm based on principal component analysis[J]. Journal of Electronics (China), 2012, 30(2): 204-212.
- [2] LIU Yutu, NGU A, ZENG L. QoS computation and policing in dynamic web service selection[C]//Proceedings of 13th international world wide web conference on alternate track papers and posters. New York, USA: ACM, 2004: 66-73.
- [3] 康国胜, 刘建勋, 唐明董, 等. 考虑 QoS 属性相关性的 Web 服务选择[J]. 小型微型计算机系统, 2014, 35(4): 786-790.
- [4] KANG Guosheng, LIU Jianxun, TANG Mingdong, et al. Web service selection for resolving conflicting service requests[C]//IEEE international conference on web services. Washington, USA: IEEE, 2011: 387-394.
- [5] HEINRICH B, LEWERENZ L, MAYER M. Enhancing decision support in multi user service selection[C]//Proceedings of thirty sixth international conference on information systems. Las Vegas, USA: IEEE, 2015: 1-20.
- [6] KANG Guosheng, LIU Jianxun, TANG Mingdong, et al. An effective dynamic web service selection strategy with global optimal QoS based on particle swarm optimization algorithm[C]//Proceedings of 2012 IEEE 26th international parallel and distributed processing symposium workshops & PhD forum. Shanghai, China: IEEE, 2012: 2280-2285.
- [7] ARDAGNA D, PERNICI B. Global and local QoS guarantee in web service selection[C]//Proceedings of third international conference on business process management. Nancy, France: Springer, 2005: 32-46.
- [8] RAN Shuping. A model for web services discovery with QoS[J]. ACM SIGecom Exchanges, 2003, 4(1): 1-10.
- [9] LIU Shulei, LIU Yunxiang, JING Ning, et al. A dynamic web service selection strategy with QoS global optimization based on multi-objective genetic algorithm[C]//4th international conference on grid and cooperative computing. Beijing, China: Springer-Verlag, 2005: 84-89.
- [10] 范小芹, 蒋昌俊, 方贤文, 等. 基于离散微粒群算法的动态 Web 服务选择[J]. 计算机研究与发展, 2010, 47(1): 147-156.
- [11] 夏虹, 李增智. 粒子群算法求解 Web 服务组合中基于 QoS 的服务选择[J]. 北京邮电大学学报, 2009, 32(4): 63-67.
- [12] 刘莉平, 陈志刚, 刘爱心. 基于粒子群算法的 Web 服务组合研究[J]. 计算机工程, 2008, 34(5): 104-106.
- [13] 刘书雷, 刘云翔, 张帆, 等. 一种服务聚合中 QoS 全局最优服务动态选择算法[J]. 软件学报, 2007, 18(3): 646-656.
- [14] QI Lianying, TANG Ying, DOU Wanchun, et al. Combining local optimization and enumeration for QoS-aware web service composition[C]//Proceedings of international conference on web services. Miami, Florida, USA: IEEE, 2010: 34-41.
- [15] ALRIFAI M, RISSE T. Combining global optimization with local selection for efficient QoS-aware service composition[C]//Proceedings of the 18th international conference on world wide web. Madrid Spain: ACM, 2009: 881-890.
- [16] KENNEDY J, EBERHART R. Particle swarm optimization[C]//Proceedings of IEEE international conference on neural networks. Perth, Australia: IEEE, 1995: 1942-1948.
- [17] SHI Y. Particle swarm optimization[J]. IEEE Neural Network Society, 2004, 2(2): 8-13.
- [18] EBERHART R, SHI Y. Particle swarm optimization: developments applications and resources[C]//IEEE congress on evolutionary computation. Seoul, South Korea: IEEE, 2002: 81-86.