云计算环境中服务动态选择算法研究

张恒巍 韩继红 寇 广 卫 波
(解放军信息工程大学 郑州 450001)

摘 要 为解决云计算环境下的服务动态选择问题,设计了综合考虑反应时间和成本的适应度函数,提出了求解服务动态选择问题的分布估计蛙跳算法。在蛙跳算法的基础上,借鉴交叉操作改写蛙跳算法的进化算子,并引入分布估计进化策略改进蛙跳算法的青蛙更新模式,使改进后的新算法具有更全面的学习能力,能够有效避免算法陷入局部最优。仿真实验验证了算法的可行性和有效性,与蛙跳算法和分布估计算法相比,该算法的收敛性能和寻优能力均得到改善,能够更好地解决云计算环境下的服务动态优化选择问题。

关键词 云计算,服务动态选择,服务质量,进化算子,适应度函数,概率模型
中图法分类号 TP393.7 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2015.5.050

Research on Service Dynamic Selection Algorithm in Cloud Computing

ZHANG Heng-wei HAN Ji-hong KOU Guang WEI Bo
(PLA Information Engineering University, Zhengzhou 450001, China)

Abstract To solve the service dynamic selection problem in cloud computing environment, a fitness function which considers both the response time and the cost was designed, and an estimation of distribution-shuffled frog leaping algorithm was proposed to solve the problem of service dynamic selection. On the basis of leapfrog algorithm, evolutionary operators of leapfrog algorithm was redefined by drawing crossover operation of genetic algorithm, and distribution estimation evolutionary strategy was introduced to improve frog update mode of the leapfrog algorithms, so that the new improved algorithm has a more comprehensive learning ability and it can effectively avoid the local optimum. Simulation results demonstrate the feasibility and effectiveness of the proposed algorithm, and compared with the leapfrog algorithm and estimation of distribution algorithms, the convergence performance and optimization capabilities of the proposed algorithm are improved, and it can better solve the service dynamic selection problem in cloud computing environment.

Keywords Cloud computing, Service dynamic selection, QoS, Evolutionary operators, Fitness function, Probabilistic model

1 引言

云计算[1,2]是一种新兴的计算方式,通过将大量的网络资源(计算资源、存储资源与软件资源等)以服务的形式链接在一起,形成规模巨大的虚拟资源池,为用户和应用系统提供"能力无限"的云服务资源。云计算环境中有大量功能属性相同、非功能属性各异的云服务,而单个云服务的功能有限,因此如何将多个云服务组合起来形成新的、增值的、大粒度的组合服务来满足复杂多变的应用需要,是当前迫切需要解决的问题,而服务组合是解决这一问题的有效方法。

目前服务组合的研究工作侧重于半自动方式[3]。半自动组合实现的基础是根据任务需求所建立的服务流程模型。模型由多个服务节点组成,每个节点拥有多个功能相同、服务质量(QoS)不同的候选服务。如何从众多的候选服务中动态地选择合适的服务实例,形成最优的服务组合来满足用户的需求,是服务组合中的一个关键问题。

针对最优服务选择问题,主要存在QoS局部最优和QoS全局最优两类优化选择方法。QoS局部最优选择方法只能满足单一的QoS需求,无法从整体上获得全局最优,难以全面满足用户需求。QoS全局最优动态服务选择是NP完全问题[4],求解难度大。近年来越来越多的研究者尝试用演化学习型智能算法解决该类问题,但是现有的智能优化算法大多复杂度较高、求解效率较低,且对非支配解集的多样性考虑不足,优化解的质量还有待提高。其中,文献[5]设计了自适应变异遗传算法,即改进传统变异算子提高算法自适应性,引入指数衰减函数提高算法的收敛性能。文献[6]将人工鱼群算

到稿日期:2014-06-13 返修日期:2014-08-12 本文受国家自然科学基金项目(61303074,61309013),国家重点基础研究发展计划("973"计划)基金项目(2012CB315900),河南省科技攻关计划项目(12210231003,13210231002)资助。
张恒巍(1978-),男,博士生,讲师,主要研究方向为云资源管理、需求工程,E-mail:zhw11qd@126.com;韩继红(1966-),女,博士,教授,博士生导师,主要研究方向为网络资源管理、系统工程;寇 广(1983-),男,博士生,主要研究方向为资源动态管理;卫 波(1990-),男,硕士生,主要研究方向为云服务组合与选择。

法用于解决服务优化选择问题，相比遗传算法，其具有更好的选择结果。文献[7]提出一种改进的粒子群算法，即通过新的惯性权重动态调整机制和可选变异操作方法，提高了粒子群算法收敛性能和全局寻优能力。混合蛙跳算法（Shuffled Frog Leaping Algorithm，SFLA）是一种基于群体的启发式协同搜索群智能算法，具有参数少、计算速度快、易于实现等特点，通过与其它智能优化算法的结合，目前在众多领域得到广泛的应用。文献[8]提出了基于搜索加速参数的混合蛙跳算法，改进后的算法具有更好的收敛速率，但易于陷入局部最优，全局寻优能力较差。文献[9]提出了一种基于克隆选择的混合蛙跳算法，采用克隆选择算法对种群中最优个体进行寻优，采用 SFLA 对种群中最差个体向最优个体方向进行寻优，有效提高了算法的全局寻优能力。我们在前期工作中利用 EDA（Estimation of Distribution Algorithm）策略改进蛙跳算法[10]，以解决云资源调度问题，算法的搜索效率高于传统混合蛙跳算法。文献[11]提出一种改进的混合蛙跳算法用于求解旅行商问题（TSP），通过在全局信息交换过程中加入变异操作，算法具有更好的搜索性能。

本文在蛙跳算法的基础上，综合考虑性能和成本参数，设计了能全面反映服务质量的适应度函数，并提出了基于分布估计蛙跳算法（Estimation of Distribution-shuffled Frog Leaping Algorithm，EDSFLA）的服务动态选择方法。新算法利用遗传算法的交叉操作重新定义蛙跳算法的进化算子，使之适应整数编码的服务选择问题，同时通过分布估计进化策略改进蛙跳算法的青蛙更新模式，使算法具有更全面的学习能力，改进后的算法比标准的蛙跳算法和分布估计算法具有更好的全局寻优能力和收敛速度。

2 动态全局最优服务选择问题描述

云计算环境下的服务选择中，服务节点不是一个具体的服务，而是一个对众多相同功能、不同实现的候选服务的抽象。服务组合的本质是通过不同服务节点的合作实现不同的业务流程。由于不同的候选服务实例拥有不同的 QoS，因此采用不同服务实例的服务节点所形成的服务组合必然具有不同的 QoS。依据服务组合中不同服务节点之间的结构关系，可将其分为 4 种基本结构[12]：顺序，并行，选择，循环。组合服务的 QoS 可以利用 4 种基本结构的 QoS 综合计算得到。假设服务组合流程包含 n 个服务节点，每个节点 S_i 有 m_i 个候选服务，以图 1 所示的例子对服务组合流程进行说明，示例中有 7 个服务节点（S_1，S_2，S_3，S_4，S_5，S_6，S_7），其中节点 S_1 拥有 m_1 个候选服务（$s_1^1, s_2^1, \cdots, s_{m_1}^1$），以此类推。

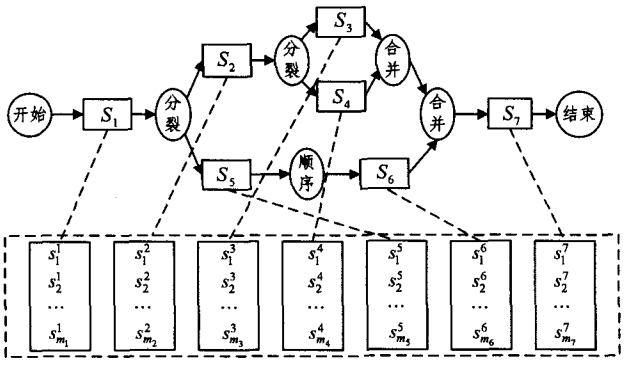

图 1 服务组合流程示例

由于服务的 QoS 属性之间具有不可公度性和矛盾性等特点，且度量单位不相同、取值范围不一致，为了更有效地对各服务的 QoS 性能指标进行综合评价，首先对服务的各 QoS 属性进行归一化处理。若 QoS 属性为响应时间、成本等负属性时，其取值越大则质量越差，则用式(1)归一化处理；若 QoS 属性为可用性、可靠性等正属性时，其取值越大则质量越好，则用式(2)归一化处理。

$$q_i' = \begin{cases} \dfrac{q_{\max} - q_i}{q_{\max} - q_{\min}}, & \text{if } q_{\max} - q_{\min} \neq 0 \\ 1, & \text{if } q_{\max} - q_{\min} = 0 \end{cases} \tag{1}$$

$$q_i' = \begin{cases} \dfrac{q_i - q_{\min}}{q_{\max} - q_{\min}}, & \text{if } q_{\max} - q_{\min} \neq 0 \\ 1, & \text{if } q_{\max} - q_{\min} = 0 \end{cases} \tag{2}$$

服务动态选择就是在服务组合流程执行过程中，从各个服务结点对应的候选服务集中选择具体的服务实例组成一个可执行的组合服务，使组合服务的 QoS 达到最优。为了全面衡量服务质量，本文选取 4 种常用的 QoS 参数综合反映性能和成本指标，即服务执行时间 T（time）、服务可靠性 R（reliability）、服务信誉 Rep（reputation）、服务执行费用 P（price）。设 cs 为组合服务，s_i 为服务节点 S 上所选择的服务实例，则 s_i 和 cs 的 QoS 分别为 $Q(s_i)=(T_i, R_i, Rep_i, P_i)$、$Q(cs)=(T_{cs}, R_{cs}, Rep_{cs}, P_{cs})$，4 种基本结构的 QoS 计算公式如表 1 所列。根据各 QoS 属性的权重，得到服务动态优化选择问题的适应度函数，如下所示：

$$F=\max(\alpha T'_{cs}+\beta R'_{cs}+\gamma Rep'_{cs}+\delta P'_{cs})$$

其中，α、β、γ、δ 分别表示服务执行时间、可靠性、信誉和执行成本所占的权重，且满足 $\alpha+\beta+\gamma+\delta=1$。

表 1 基本结构的 QoS 计算公式

	T	R	Rep	P
顺序	$\sum_{i=1}^{m} T_i$	$\prod_{i=1}^{m} R_i$	$\sum_{i=1}^{m} Rep_i/m$	$\sum_{i=a}^{m} P_i$
并行	$\max\{T_i\}$	$\min\{R_i\}$	$\sum_{i=1}^{m} Rep_i/m$	$\sum_{i=a}^{m} P_i$
选择	$\sum_{i=1}^{m} \partial_i T_i$	$\prod_{i=1}^{m} R_i \partial_i$	$\sum_{i=1}^{m} \partial_i Rep_i$	$\sum_{i=a}^{m} \partial_i P_i$
循环	$k\sum_{i=1}^{m} T_i$	$\prod_{i=1}^{m} R_i$	$\sum_{i=1}^{m} Rep_i/m$	$k\sum_{i=a}^{m} P_i$

3 相关算法及其原理

3.1 混合蛙跳算法

SFLA[13] 是在 2003 年被提出的一种模拟青蛙群体觅食过程的全新群体智能进化算法。算法综合了遗传算法和粒子群算法的优点，是一种结合了确定性方法和随机性方法的进化计算方法。

SFLA 的基本思想是：随机生成 N 只青蛙组成初始种群 $P=\{X_1, X_2, \cdots, X_N\}$，S 维解空间中的第 i 只青蛙表示为 $X_i=[x_{i1}, x_{i2}, \cdots, x_{iS}]$。生成初始群体之后，首先将种群内青蛙的个体按适应值降序排列，并将蛙群中具有最优适应值的青蛙记为 X_g；然后将整个青蛙群体分成 m 个子群，每个子群包含 n 只青蛙，满足关系 $N=m\times n$。其中，第 1 只青蛙分入第 1 子群，第 2 只青蛙分入第 2 子群，第 m 只青蛙分入第 m 子群，第 $m+1$ 只青蛙重新分入第 1 子群，依此类推。设 M^k 为第 k 个子群的青蛙的集合，其分配过程可描述如下：

$$M^k=\{X_{k+m(l-1)} \in P \mid 1 \leqslant l \leqslant n\}, 1 \leqslant k \leqslant m \tag{3}$$

每一个子群中具有最佳适应值和最差适应值的青蛙分别

记为 X_b 和 X_w。对每个子群进行局部搜索，即对子群中的 X_w 循环进行局部搜索操作。根据最初蛙跳规则，其更新方式为

$$\begin{cases} D=r\cdot(X_b-X_w) \\ X_w'=X_w+D, \| D \| \leqslant D_{\max} \end{cases} \tag{4}$$

式中，r 表示 0 与 1 之间的随机数，D 表示青蛙所允许改变位置的最大值。经过更新后，得到的青蛙 X_w' 如果优于原来的青蛙 X_w，则取代原来子群中的青蛙；如果没有改进，则用 X_g 取代 X_b，按式(4)执行局部搜索过程；如果仍然没有改进，则随机产生一个新青蛙直接取代原来的 X_w。重复上述局部搜索，将所有子群内的青蛙重新混合并排序和划分子群，再进行局部搜索，如此反复，直到定义的收敛条件结束为止。

3.2 分布估计算法

分布估计算法(Estimation of Distribution Algorithm, EDA)[14]是一种基于统计学习理论的群体进化算法，通过建立概率模型描述候选解在搜索空间的分布信息，采用统计学习手段从群体宏观的角度建立一个描述解分布的概率模型，然后对概率模型随机采样产生新的种群，如此反复以实现种群的进化[15,16]。标准 EDA 的算法流程如下：

(1)初始化种群；

(2)选择优势群体；

(3)构建概率模型；

(4)利用概率模型随机抽样产生新的种群；

(5)判断终止条件是否满足，满足则输出优化结果；否则，转 Step(2)。

4 分布估计蛙跳算法

在前期工作基础上(参见文献[10])，针对动态全局最优服务选择问题，利用遗传算法的交叉操作重新定义蛙跳算法的进化算子，使之适应整数编码的服务选择问题，同时通过分布估计模型改进蛙跳算法的青蛙更新模式，以提高分布估计蛙跳算法的全局寻优能力和加快收敛速度。

4.1 编码方式

本文采用服务节点-候选服务的编码方式。假设有 n 个服务节点，每个节点 N_i 拥有 $m_i(i=0,1,2,\cdots,n)$ 个候选服务，则候选服务的总数量 sm 为：

$$sm=\sum_{i=1}^{n} m_i \tag{5}$$

对候选服务采取顺序编码，$M[i,j]$ 表示第 i 个节点的第 j 个候选服务的编号。

$$M[i,j]=j+\sum_{k=1}^{i-1} m_k \tag{6}$$

例如，有 3 个服务节点，每个节点的候选服务数量分别是 $m_1=3, m_2=2, m_3=4$，则总的候选服务数是 $sm=9$，候选服务编号如表 2 所列。青蛙 $X_i(i=1,2,\cdots,24)$ 代表服务选择模型的可行解，如青蛙 $X_1=(2,5,8)$ 表示第 1 个节点选择 2 号候选服务，第 2 个节点选择 5 号候选服务，第 3 个节点选择 8 号候选服务，从而形成一个可行的服务选择策略。

表 2 候选服务编号

服务节点	候选服务编号	青蛙 X_1	青蛙 X_2	青蛙 X_i	青蛙 X_{24}
1	1,2,3	2	3	…	1
2	4,5	5	4	…	5
3	6,7,8,9	8	6	…	9

4.2 交叉进化算子

4.1 节设计的服务编码采用整数编码方式，为此结合遗传算法的交叉操作重新定义青蛙更新方式。利用随机编码生成长度为 d 的 0-1 向量，称为交叉算子。当子群内青蛙进行更新时，顺序检查交叉算子的编码值，当编码值为 1 时，用最优青蛙在对应位置处的编码代替最劣青蛙的编码；当交叉算子的编码值为 0 时，则保持不变。具体操作如表 3 所列。

表 3 交叉进化操作示例

最劣青蛙	$X_w=(x_1,x_2,x_3,x_4,x_5)$
最优青蛙	$X_b=(y_1,y_2,y_3,y_4,y_5)$
交叉算子	$d=(0,1,0,1,1)$
新青蛙	$X_w'=(x_1,y_2,x_3,y_4,y_5)$

如果新青蛙 X_w' 优于 X_w，则 $X_w=X_w'$；否则，用青蛙 X_g 替换 X_b，然后再进行交叉进化；若 X_w' 仍不能优于 X_w，则用随机产生的青蛙取代 X_w。

4.3 分布估计进化策略

在标准蛙跳算法中，最差青蛙只向最优青蛙学习；最优青蛙只在新生成的青蛙优于子群内的最优青蛙时进行更新，这种学习和更新模式导致算法收敛速度慢且难以达到全局最优。为此，采用分布估计进化策略进行算法改进，基于不同子群内的所有最优青蛙，建立分布概率模型。最差青蛙的更新信息不再单纯依据子群内的最优青蛙，而是依赖反映全部最优青蛙的分布概率模型，同时，依据这一模型实现子群内最优青蛙的更新进化。

在 EDSFLA 中，第 t 次迭代时，第 w 个子群的最优青蛙为 $X_{wb}(t)=(x_{w1},x_{w2},\cdots,x_{wD}), w\in[1,W]$。统计分布在各个子群中的全部最优青蛙，计算每个青蛙的编码值在各维上的出现概率，得到的概率矩阵如式(7)所示：

$$P(t)=\begin{bmatrix} p_{11} & p_{21} & \cdots & p_{D1} \\ p_{12} & p_{22} & \cdots & p_{D2} \\ \cdots & \cdots & \cdots & \cdots \\ p_{1m} & p_{2m} & \cdots & p_{Dm} \end{bmatrix} \tag{7}$$

式中，$p_{ij}(1\leqslant i\leqslant D, 1\leqslant j\leqslant m)$ 表示第 i 维上取值为 j 的概率。

在迭代过程中，$P(t)$ 依据式(8)进行更新：

$$P(t)=(1-\lambda)P(t-1)+\lambda Q \tag{8}$$

其中，$\lambda\in[0,1]$ 为学习率，$P(t-1)$ 为上一代概率矩阵信息，Q 为新概率矩阵。算法通过选择 λ 的值平衡种群的全局和局部搜索能力，当 λ 比较小时，全局搜索能力较强；反之则局部搜索能力较强。λ 的值可以根据迭代次数动态调整。

$$\lambda=k\mathrm{e}^{\frac{t-t_{\max}}{t_{\max}}}, k\in[0,1] \tag{9}$$

式中，k 为常数，t 代表当前迭代次数，$t_{\max}$ 表示子群内局部搜索次数。在算法执行的初期，λ 的值较小，由于此时子群的多样性较好，算法具有较强的全局搜索能力和较快的收敛速度；随着迭代次数增加，λ 的值增大，算法的局部搜索能力和收敛速度得到提高。

EDSFLA 通过对概率矩阵 $P(t)$ 抽样来产生下一代种群，抽样规则如图 2 所示。

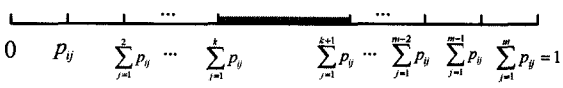

图 2 EDSFLA 的抽样规则

首先将 $X_b(t)$ 第 i 位置上不同取值的概率进行累加，累加

值在区间[0,1]上。然后在区间[0,1]上产生随机数 $Rand$，如果 $Rand$ 落在图中阴影部位即第 k 和 $k+1$ 个标度之间，则该维取值为 $k+1$。

4.4 算法步骤及分析

根据上述设计思想，给出 EDSFLA 的具体步骤如下：

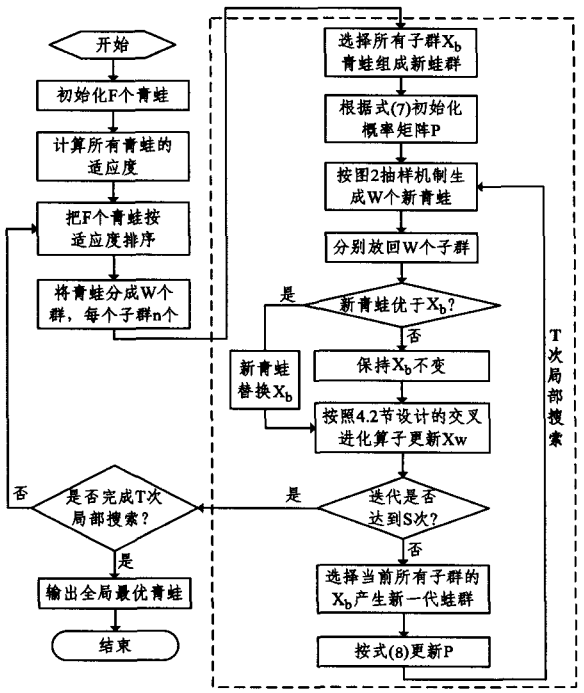

图 3 EDSFLA 流程

步骤 1 随机产生 N 只青蛙。

步骤 2 计算青蛙的个体适应度 $F_i, i=1,2,\cdots,N$。

步骤 3 按照青蛙适应度从大到小进行排序，依据 3.1 节描述的子群划分原则，将整个蛙群划分成 W 个子群，确定每个子群内最优青蛙 X_b、最差青蛙 X_w 和全局最优青蛙 X_g。

步骤 4 对每个子群进行 T 次局部搜索，每次局部索迭代 S 次。

(1)选择所有子群的最优青蛙组成新蛙群，根据式(7)初始化概率矩阵 P；

(2)根据图 2 所示的抽样规则生成 W 个新青蛙；

(3)将新生成的 W 只青蛙放回子群内，若新青蛙的适应度优于子群内的最优青蛙，则进行替换，否则保持不变；

(4)根据 4.2 节设计的交叉进化方式，对子群内最差青蛙进行更新；

(5)选择所有子群的最优青蛙对蛙群进行更新，按照式(8)更新矩阵 P；

(6)判断是否完成 T 次局部搜索，完成则进行步骤 5，否则转到(2)。

步骤 5 判断是否满足终止条件，满足则停止搜索，输出全局最优青蛙；否则转到步骤 3。

5 仿真实验与分析

本文采用云计算仿真平台 CloudSim[17] 进行仿真实验，通过扩展 CloudSim 的 DatacenterBroker 类，模拟采用本文算法(EDSFLA)、蛙跳算法(SFLA)和分布估计算法(EDA)进行对比分析。实验的微机配置为 Core2 处理器，2G 内存，操作系统为 Windows XP，算法用 Matlab2010b 实现。仿真实验采用图 1 所示的服务组合流程，根据表 1 所列的 4 种基本结构的 QoS 计算服务优化选择问题的目标函数与约束条件。服务实例的 QoS 参数采用随机方式生成，取值范围：$0<T_i\leqslant 100$，$0<R_i\leqslant 1$，$0<Rep_i\leqslant 1$，$0<P_i\leqslant 200$。算法参数设置如下：EDSFLA 的子群个数 $H=20$，子群内青蛙个数 $I=20$，子群内局部搜索次数 $T=10$，常数 $k=0.8$，青蛙种群规模根据实验的不同而设置。

5.1 算法可行性验证

本实验的目的是检验采用本文设计的 EDSFLA 求解 QoS 全局最优服务动态选择问题的可行性，每个服务节点有 100 个候选服务实例，主要设计思路是比较使用 EDSFLA 算法求得的结果与使用穷举法得到的结果，根据二者的相似程度来分析该算法的可行性。针对不同种群规模和迭代次数各重复运算 10 次求平均结果，图 4 给出了算法可行性验证的结果。

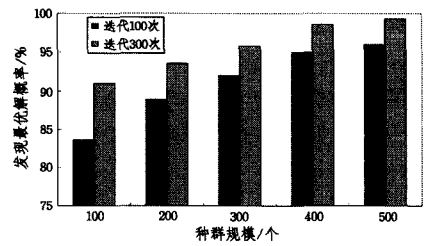

图 4 算法可行性验证实验结果

从图中可以看出，迭代次数为 300 时，能够找到最优解的概率都能达到 90%以上，而且随着种群的规模的增大，找到最优解的概率也不断增加。因此，利用 EDSFLA 求解 QoS 全局最优的服务选择问题是可行的。

5.2 算法有效性验证

本节的目的是通过分析算法求解过程中 CPU 的开销来验证 EDSFLA 的有效性。本节分为两个实验，其中实验 1 比较了分别考虑不同青蛙种群规模、不同迭代次数情况下算法的执行时间，每种情况下分别运行 10 次取平均值，图 5 示出采用该算法得到最优解的 CPU 开销情况。实验 2 中青蛙种群规模设为 $N=400$，比较在各服务节点候选服务实例规模不同的条件下，EDSFLA 与标准蛙跳算法和分布估计算法在求解该问题时的 CPU 开销情况，每种情况下分别运行 10 次取平均值，结果如图 6 所示。

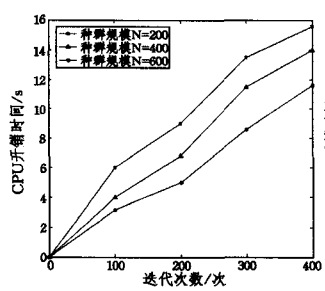

图 5 不同种群规模和迭代次数下 CPU 开销对比

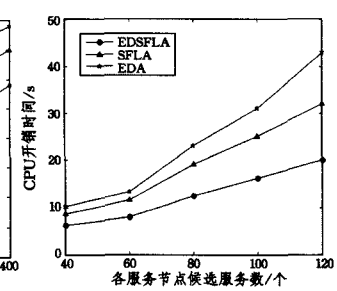

图 6 不同候选服务数下 CPU 开销对比

从图 5 可以看出，算法在不同种群规模情况下，随着迭代次数的不断增加，CPU 的运行时间没有大幅度增加，在迭代次数为 400、种群规模为 600 的情况下，求得最优解耗时仅为 15.6s，具有较高的运算效率；从图 6 可以看出，相比 SFLA 和

(下转第 269 页)

[12] Yu Y, Pedrycz W, Miao D Q. Neighborhood rough sets based multi-label classification for automatic image annotation[J]. Journal of Approximate Reasoning, 2013, 54(9): 1373-1387
[13] 王国胤. Rough 集理论与知识获取[M]. 西安: 西安交通大学出版社, 2001
[14] 苗夺谦. 粗糙集理论中连续属性的离散化方法[J]. 自动化学报, 2001, 27(3): 296-302
[15] Jensen R, Shen Q. Tolerance-based and fuzzy-rough feature selection[C]// Proceedings of the 16th International Conference on Fuzzy Systems (FUZZ- IEEE'07). 2007: 877-882
[16] Parthaláin N M, Shen Q. Exploring the boundary region of tolerance rough sets for feature selection[J]. Pattern Recognition, 2009, 42: 655-667
[17] Shen Q, Chouchoulas A. A rough-fuzzy approach for generating classification rules[J]. Pattern Recognition, 2002, 5: 2425-2438
[18] Grzymala-Busse J W. Discretization of numerical attributes[M]// Klösgen W, Zytkow J, eds. Handbook of Data Mining and Knowledge Discovery. Oxford University Press, 2002: 218-225
[19] Grzymala-Busse J W, Grzymala-Busse W J. Handling missing attribute values[M]// Maimon O, Rokach L, eds. Handbook of Data Mining and Knowledge Discovery. 2005: 37-57
[20] Min F, Zhu W. Attribute reduction of data with error ranges and test costs[J]. Information Sciences, 2012, 211: 48-67
[21] Zhao H, Min F, Zhu W. Cost-Sensitive Feature Selection of Numeric Data with Measurement Errors[J]. Journal of Applied Mathematics, 2013, 2013

(上接第 254 页)

EDA, EDSFLA 求解该问题时 CPU 耗时最少，且随着各服务节点候选服务数量规模的增加，CPU 开销仅呈现线性增加趋势，能够有效地满足大部分服务动态优化选择问题的求解。

5.3 算法收敛性能对比

本实验目的是在候选服务数量固定时，对 3 种服务选择算法的收敛性能进行对比。实验中为每个节点生成 100 个候选服务实例，实验重复 10 次取平均结果，图 7 给出了 3 个算法的进化曲线。

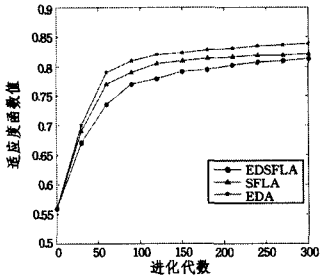

图 7 算法收敛性对比

可以看出，与 SFLA 和 EDA 相比，EDSFLA 的收敛速度最快且收敛精度更高，具有更强的全局寻优能力，这说明算法在借鉴分布估计进化思想的基础上所进行的改进是有效的。

结束语 为了更好地解决云计算环境 QoS 全局最优服务动态选择问题，本文提出了基于分布估计蛙跳算法的服务动态选择方法。该方法采用遗传算法的交叉操作重新设计了蛙跳算法的进化算子，利用分布估计进化策略改进标准蛙跳算法的青蛙更新模式，增强了各子群之间的相互学习能力，能够有效避免陷入局部最优。通过仿真实验验证了该方法的可行性和有效性，与标准蛙跳算法和分布估计算法相比，在求解过程中，本文提出的算法具有更优的收敛性能。

参 考 文 献

[1] Buyya R, Yeo C S, Venugopal S, et al. Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility [J]. Future Generation Computer Systems, 2012, 25(6): 599-616
[2] 罗军舟, 金嘉晖, 宋爱波, 等. 云计算: 体系架构与关键技术[J]. 通信学报, 2011, 32(7): 3-21
[3] Majithia S, Walker D W, Gray W A. A Framework for Automated Service Composition in Service-oriented Architectures[C]// ESWS 2008 Congress. Berlin, Germany: Springer Verlag, 2012: 269-283
[4] Yu Tao, Lin K J. Service Selection Algorithms for Composing Complex Services with Multiple QoS Constraints[C]// The 3rd International Conference on Service Oriented Computing Congress. Amsterdam, Holland: Springer Verlag, 2010: 130-143
[5] 董元元, 倪宏, 邓浩江, 等. Qos 全局最优的服务选择策略[J]. 中南大学学报: 自然科学版, 2013, 42(10): 3086-3094
[6] 刘旋, 廖明潮. 基于人工鱼群算法的 QoS 全局最优 Web 服务最优 Web 服务选择的研究[J]. 计算机应用与软件, 2013, 30(8): 87-90
[7] 孙黎阳, 林剑柠, 毛少杰. 基于改进粒子群优化算法的网络化仿真任务共同体服务选择[J]. 兵工学报, 2012, 33(11): 1393-1403
[8] Elbeltagi E, Hegazy Grierson D. A modified shuffled frog-leaping optimization algorithm. Applications to project management [J]. Structure and Infrastructure Engineering, 2012, 3(1): 53-60
[9] Antariksha B. A clonal selection based shuffled frog leaping algorithm[C]// IEEE Advance Computing Congress. New York: IEEE, 2013: 125-130
[10] 张恒巍, 卫波. 基于分布估计蛙跳算法的云资源调度方法[J]. 计算机应用研究, 2014, 31(10): 30-34
[11] 罗雪晖, 杨烨, 李霞. 改进混合蛙跳算法求解旅行商问题[J]. 通信学报, 2013, 30(7): 130-135
[12] 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[J]. 软件学报, 2011, 18(3): 646-656
[13] Xu Y, Wang L, Zhou G, et al. An effective shuffled frog leaping algorithm for solving hybrid flow-shop scheduling problem[C]// Proceedings of the 9th International Conference on Advanced Intelligent Computing. Berlin: Springer Verlag, 2013: 560-567
[14] Dong W S, Yao X. Unified eigen analysis on multivariate Gaussian based estimation of distribution algorithms [J]. Information Sciences, 2011, 178(15): 3000-3023
[15] Muhlenbein H. The equation for response to selection and its use for prediction [J]. Evolutionary Computation, 2012, 5(3): 303-346
[16] Kastegar R. On the optimal convergence probability of univariate estimation of distribution algorithms [J]. Evolutionary Computation, 2013, 19(2): 225-248
[17] The Cloud Lab. Cloudsim [EB/OL]. [2012-08-15]. http://www.cloudbus.org/cloudsim