一种求解 SLA 等级感知服务组合问题的多目标离散粒子群优化算法

尹 浩,张长胜,张 斌,孙若男,刘婷婷
(东北大学信息科学与工程学院,辽宁沈阳 110819)

摘 要: 针对 SLA 等级感知服务组合问题,本文提出了一种求解该问题的多目标离散粒子群算法(MDPSO),建立了多目标粒子群算法优化模型.根据该问题的特征,对粒子更新策略进行重新设计;并且提出粒子变异策略以抑制群体的早熟收敛增强群体的全局搜索能力.另外,提出了一种基于约束支配关系的局部搜索策略并将其结合到 MDPSO 算法,形成算法 MDPSO+.最后对 MDPSO 算法的参数设值进行了分析,并将算法 MDPSO、MDPSO+ 与最近提出的求解该问题的 E^3-MOGA 算法及 NSGA-II 算法在不同规模的测试用例上进行了实验对比,结果表明算法 MDPSO+ 能够更加有效的解决该问题.

关键词: 多目标离散粒子群优化 (MDPSO); 服务等级; 群体多样性; 局部搜索
中图分类号: TP311 **文献标识码:** A **文章编号:** 0372-2112 (2014)10-1983-08
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2014.10.018

A Multi-Objective Discrete Particle Swarm Optimization Algorithm for SLA-Aware Service Composition Problem

YIN Hao, ZHANG Chang-sheng, ZHANG Bin, SUN Ruo-nan, LIU Ting-ting
(*College of Information Science & Engineering, Northeastern University, Shenyang, Liaoning* 110819, *China*)

Abstract: For SLA-aware service composition problem (SSC), a multi-objective discrete particle swarm optimization algorithm (MDPSO) is proposed in this paper and an optimization model for this algorithm is also built. According to the character of this SSC problem, a particle updating strategy is redesigned by introducing crossover operator. A particle mutation strategy is proposed to increase the swarm diversity and restrain particle swarm's premature convergence. In addition, algorithm MDPSO+ is formed by incorporating a local search strategy based on constraint-domination into the algorithm MDPSO. At last, some parameters in algorithm MDPSO are analyzed and set with relative proper values, and then the algorithm MDPSO and the algorithm MDPSO+ are compared with the recently proposed algorithm E3-MOGA and NSGA-II on different-scale cases; the results show that algorithm MDPSO+ can solve the SSC problem more effectively.

Key words: MDPSO; service level agreement; swarm diversity; local search

1 引言

SLA 等级感知服务组合问题(SSC)是面向服务的体系结构(SOA)中的一个关键问题[1,2],对其进行研究具有重要意义.尽管有大量的研究利用线性规划解决 SSC 问题[3],但是计算代价很大.为了降低计算代价,出现了很多启发式算法[4,5],但这些已有的研究都将该问题转化为单目标优化问题,只考虑单个服务等级并未考虑多个服务等级的情况.直到 2012 年 Hiroshi Wada[6] 等人才开始同时考虑多个服务等级,将该问题转化为多目标优化问题,并提出求解该问题 E^3-MOGA 算法.这个算法收敛速度慢,易陷入局部最优,且当问题规模较大时难以获得令人满意的候选解.而作为一种被广泛应用的群智能算法[7,8],粒子群算法(Particle Swarm Optimization, PSO)具有容易实现,精度高,收敛快等优点[9,10].因此本文拟采用粒子群算法进行求解,并针对该问题提出了的多目标离散粒子 MDPSO.在算法 MDPSO 中,引入了遗传算法中的交叉算子对粒子更新策略进行了重新设计;提出了粒子变异策略,通过粒子的变异引入新信息增强群体的全局搜索能力.为了加快获得满足约束条件的可行解,设计了将局部搜索策略融入 MDPSO 的 MDPSO+ 算法.最后,对 MDPSO 算法的参数值进行了分析并通过实验

收稿日期:2013-08-23;修回日期:2014-3-20;责任编辑:孙瑶
基金项目:国家自然科学基金(No.61100090,No.61073062,No.61100027);中央高校基本科研业务费专项资金(No.N11024006,No.N110604002,No.N120604003)

验证了 MDPSO 算法和算法 MDPSO + 的有效性.

2 服务组合问题模型

SLA 等级感知的服务组合问题(SSC)是一个查找抽象服务与可用具体服务之间最优绑定的组合优化问题,是一个 NP 难问题,该问题的详细描述请参考文献[5].图 1 为问题的模型,包括抽象流程,抽象服务,具体服务,具体服务实例,以及实例流程.通过为抽象流程中的每个抽象服务部署具体服务形成实例流程,通过将抽象流程实例化为多个流程实例来为用户提供多个 SLA 等级的组合服务的实例.图 1 中给出了抽象服务的一个流程实例,其中为同一个抽象服务部署多个具体服务实例并行执行的情况为冗余并行,而且必须保证为每个抽象服务至少部署一个具体服务实例,并且在组合服务中需完成对所有抽象服务的具体服务部署.

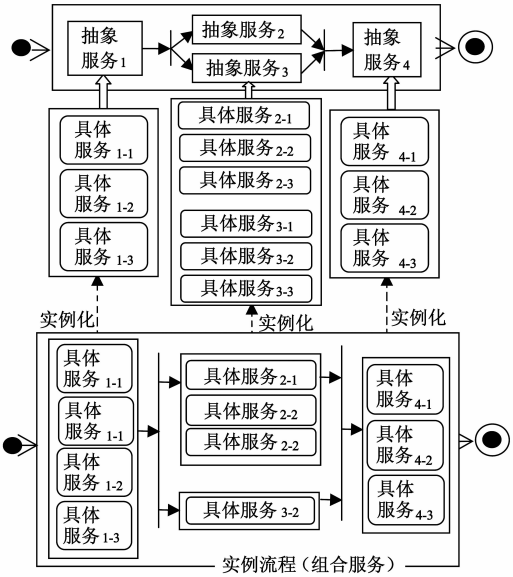

图1 SSC问题的服务组合模型

定义 1(SSC 问题的解) 对于 SSC 问题的抽象流程 $\mathcal{S}$,问题的解由多个组合服务(抽象流程的实例流程)部署方案构成.组合服务的部署方案为实例化过程中所需为每个抽象服务部署的相应具体服务的数量.

假设三维向量 $\boldsymbol{X}=(cs_1,\cdots,cs_L)=(x_{111},\cdots,x_{1NM},\cdots,x_{L11},\cdots,x_{LNM})$表示问题的解,其中 L,N,M 表示服务等级数量,抽象服务个数,和与每个抽象服务关联的具体服务的个数.二维向量 $\boldsymbol{S}=(s_{11},\cdots,s_{NM})$表示所有具体服务.则向量 $\boldsymbol{X}$ 中的分量 cs_i 表示与等级 i 关联的服务组合,x_{ijk} 代表服务组合 cs_i 中,抽象服务 j 中第 k 个具体服务 s_{jk} 部署的数量,并且 $i\in[1,L]$ && $i\in Z$, $j\in[1,N]$ && $j\in Z$, $k\in[1,M]$ && $k\in Z$.

采用文献[5]相同的问题假设,即存在三类用户分类:白金卡用户类,金卡用户类和银卡用户类,则等级数量 $L=3$ 并且组合服务 cs_1,cs_2,cs_3 分别表示与白金卡用户,金卡用户和银卡用户关联的服务组合.假设每个抽象服务与三个具体服务相关联,则 $M=3$ 且它们分别为高性能,低性能,以及中等性能的具体服务.并且 SSC 问题模型涉及到的 QoS 属性为吞吐量(throughput),延迟时间(latency),费用(cost),分别由 Q_1,Q_2,Q_3 表示.

为了判断问题的解是否满足给定的 SLA 等级约束,依据文献[5]的 QoS 聚合函数计算服务组合端到端的 QoS 属性.抽象服务间不同 QoS 属性(Q_1,Q_2,Q_3)聚合函数分别用 Π_T,Π_L,Π_C 分别表示,并且选择关系在文中被分解为多个顺序结构执行.则问题的十个目标函数白金卡用户服务组合的 throughput, latency, cost,金卡用户服务组合的 throughput, latency, cost,银卡用户服务组合的 throughput, latency, cost,以及这三个服务组合总的 cost,可以表示为向量 $\boldsymbol{Q}=\{Q_1(cs_1),Q_2(cs_1),Q_3(cs_1),Q_1(cs_2),Q_2(cs_2),Q_3(cs_2),Q_1(cs_3),Q_2(cs_3),Q_3(cs_3),Q_3(\boldsymbol{X})\}$即

$$Q_1(cs_1)=\prod_{j=1}^{N}{}_T\Big(\sum_{k=1}^{M}x_{1jk}Q_1(s_{jk})\Big) \tag{1}$$

$$Q_2(cs_1)=\prod_{j=1}^{N}{}_L\Big(\sum_{k=1}^{M}x_{1jk}Q_2(s_{jk})\Big) \tag{2}$$

$$Q_3(cs_1)=\prod_{j=1}^{N}{}_C\Big(\sum_{k=1}^{M}x_{1jk}Q_3(s_{jk})\Big) \tag{3}$$

$$Q_1(cs_2)=\prod_{j=1}^{N}{}_T\Big(\sum_{k=1}^{M}x_{2jk}Q_1(s_{jk})\Big) \tag{4}$$

$$Q_2(cs_2)=\prod_{j=1}^{N}{}_L\Big(\sum_{k=1}^{M}x_{1\backslash 2jk}Q_2(s_{jk})\Big) \tag{5}$$

$$Q_3(cs_2)=\prod_{j=1}^{N}{}_C\Big(\sum_{k=1}^{M}x_{2jk}Q_3(s_{jk})\Big) \tag{6}$$

$$Q_1(cs_3)=\prod_{j=1}^{N}{}_T\Big(\sum_{k=1}^{M}x_{3jk}Q_1(s_{jk})\Big) \tag{7}$$

$$Q_2(cs_3)=\prod_{j=1}^{N}{}_L\Big(\sum_{k=1}^{M}x_{3jk}Q_2(s_{jk})\Big) \tag{8}$$

$$Q_3(cs_3)=\prod_{j=1}^{N}{}_C\Big(\sum_{k=1}^{M}x_{3jk}Q_3(s_{jk})\Big) \tag{9}$$

$$Q_3(\boldsymbol{X})=Q_3(cs_1)+Q_3(cs_2)+Q_3(cs_3) \tag{10}$$

三类用户 SLA 等级约束用向量 $\boldsymbol{C}=(C_1,\cdots,C_7)$表示,分别为对白金卡和金卡用户 throughput 和 latency 的最低约束,对银卡用户 throughput 和 cost 的最低约束,以及对三类用户所产生的总费用的最低约束,即:

$$Q_1(cs_1)\geqslant C_1 \tag{11}$$

$$Q_2(cs_1)\leqslant C_2 \tag{12}$$

$$Q_1(cs_2)\geqslant C_3 \tag{13}$$

$$Q_2(cs_2)\leqslant C_4 \tag{14}$$

$$Q_3(cs_3)\geqslant C_5 \tag{15}$$

$$Q_3(cs_3)\leqslant C_6 \tag{16}$$

$$Q_3(\boldsymbol{X}) \leqslant C_7 \tag{17}$$

定义 2(可行解)　对于抽象流程 $\mathcal{S}$ 和 SLA 等级约束 $\boldsymbol{C}=(C_1,\cdots,C_7)$,如果与问题的解 $\boldsymbol{X}=(cs_1,cs_2,cs_3)$ 相关的服务组合满足 SLA 等级约束,则认为这个解是可行的.

定义 3(SLA 等级感知的服务组合)　对于一个给定的抽象流程 $\mathcal{S}$ 和 SLA 等级约束,SLA 等级感知的服务组合就是找到该问题的可行解 $\boldsymbol{X}$ 并且使它们的十个目标函数 $\boldsymbol{Q}$ 整体最优.

3　多目标离散粒子群优化(MDPSO)

为采用 PSO 算法解决该问题,在 MDPSO 算法中,定义了粒子更新策略,以实现对解的空间进行全局搜索;设计了变异策略用于抑制算法的早熟收敛;并给出了该算法的描述.另外,为了加快获得满足约束条件的可行解,本文还设计了将局部搜索策略融入 MDPSO 的 MDPSO+算法.

3.1　粒子位置的表示

依据 SSC 问题假设,算法 MDPSO 中的每个粒子位置代表问题的一个解 $\boldsymbol{X}=(cs_1,\cdots,cs_L)=(x_{111},\cdots,x_{1NM},\cdots,x_{L11},\cdots,x_{LNM})$,如图 2 所示,它表示图 1 中抽象流程的一个实例流程.其中包括三个用户等级 $L=3$,抽象流程中包括 4 个抽象服务 $N=4$,且每个抽象服务与 3 个具体服务相关联 $M=3$.

因此粒子位置所表示的解 $\boldsymbol{X}$ 由 $L\times N\times M=3\times4\times3=36$ 个分量组成.

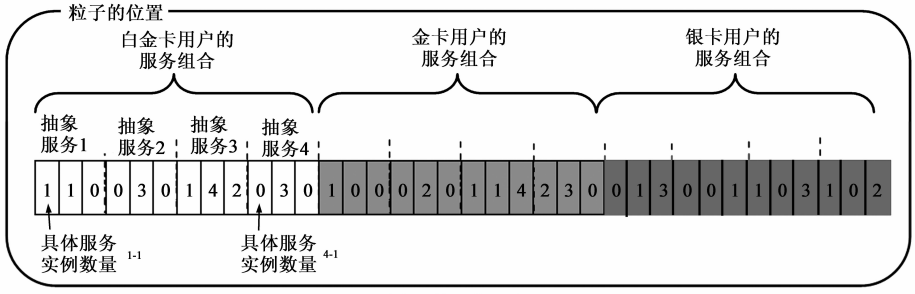

图2　粒子位置表示

3.2　粒子更新策略

粒子更新策略包括两部分,即粒子速度更新和粒子位置更新,并在更新过程中引入交叉算子.粒子速度更新,如图 3 所示,由三个粒子位置交叉得到,即粒子的当前速度,粒子的个体最优位置和粒子的全局最优位置.以抽象服务为单位,所有抽象服务的节点都可能成为交叉点.在所有的交叉点中,随机选取两个交叉点将粒子分为 3 个部分,新的当前速度由从每个粒子位置随机挑选出一个不同的部分重新组成.粒子位置更新由粒子的当前位置和粒子的速度交叉来实现.

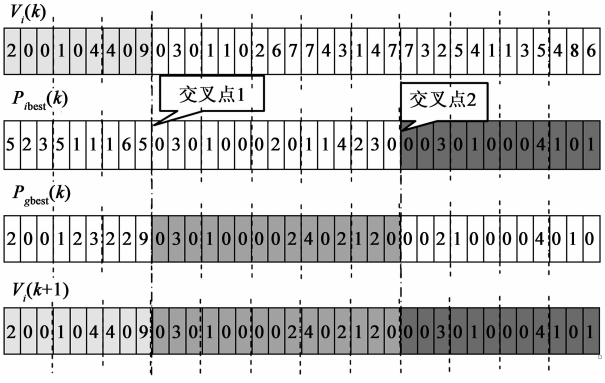

图3　粒子速度更新过程

由粒子更新策略可以看出,每个粒子的行为和传统的粒子群算法一样,粒子的行为主要受其当前动量项、个体认知部分及群体认知部分的影响,它同样具有传统粒子群的收敛快等优点.更新策略具体公式如下:

$$V_i(k+1)=V_i(k)\otimes P_{ibest}\otimes P_{gbest} \tag{18}$$

$$P_i(k+1)=P_i(k)\otimes V(k+1) \tag{19}$$

其中符号$\otimes$表示交叉操作,式(1)和式(2)分别为速度更新公式和位置更新公式.

3.3　粒子变异策略

从粒子更新的过程可以看出,每个粒子追随其当前个体最优解及全局最优解运动.与传统的 PSO 算法一样,它具有快速收敛,计算简单等优点.但是粒子会迅速逼近个体最优和全局最优位置,即粒子个体最优解,全局最优解与其当前位置相同,易于陷入局部最优解.粒子陷入局部最优,主要是因为群体的能量不断减小,有用的指导信息不断丢失,使得粒子没有能力跳出局部最优位置[8].为此,本文定义了衡量群体多样性的指标.

定义 4　群体多样性指标为群体中每个粒子的能量和,给定粒子 P_i, $i\in[0,N\text{-}1]$,其能量根据当前位置和当前速度计算如下:

$$\text{energy}(P_i)=\frac{\sum_{u=1}^{\dim}\sum_{v=u+1}^{\dim}\text{same}(P_i(u),V_i(u))}{\dim*(\dim-1)} \tag{20}$$

其中 $P_i(u)$, $V(u)$ 分别为 P_i 和 V_i 的第 u 个分量,并且

$$\text{same}(P_i(u), V_i(u)) = \begin{cases} 0, & \text{if} \quad P_i(u) = V_i(u) \\ 1, & \text{if} \quad P_i(u) \neq V_i(u) \end{cases} \tag{21}$$

群体多样性为粒子能量的和：

$$\text{DiversityCalculate} = \text{Div} = \sum_{i=1}^{N} \text{energy}(P_i) \tag{22}$$

可以看出群体的多样性指标在一定程度上可以反映出当前群体所具有的全局搜索能力.在迭代过程中群体多样性不断减小,当群体多样性小于某个给定的阈值 α 时,对粒子的个体最优解执行变异操作,如公式所示：

$$P_{i\text{best}}(k+1) = \text{mutation}(P_{i\text{best}}(k+1)) \tag{23}$$

通过变异信息能够给群体重新引入新的信息,增加群体的多样性,指导粒子搜索那些未曾搜索过的区域,抑制算法的早熟收敛.

3.4 算法描述及分析

根据前面所设计的策略,将算法的具体步骤归纳如算法 1 所示.其中粒子群 P^k 用于存储粒子的状态,包括当前位置 P_i, $P_{i\text{best}}$;粒子群 G^k_{best}用于存储粒子的全局最优位置;粒子群 Q^k 用于临时存储每一代新产生的粒子位置,从而计算粒子位置间的支配关系;文献[5]中的函数 AssignFitnessValue()计算粒子群粒子位置的适应度值.

算法 1 MDPSO

$g \leftarrow 0$
$P^0 \leftarrow$随机产生 μ 个粒子($X_i(0), P_{i\text{best}}(0)$)
$G^0_{\text{best}} \leftarrow$随机产生 λ 个粒子($X_i(0)$)为全局最优位置
AssignFitnessValues(G^0_{best})
repeat until $k = k_{\max}$ {
$Q^k \leftarrow \varnothing$
div = CalculateDiversity(P^k)
for each particle i in P^k {
//粒子变异
if(div < α)
$P_{i\text{best}}(k) \leftarrow$Mutation($P_{i\text{best}}(k)$)
endif
//通过 binary tourname 方式选取全局最优解
$G_a(k), G_b(k) \leftarrow$RandomSelection($G^k_{\text{best}}$)
$G_v(k) \leftarrow$BTSelection($G_a(k), G_b(k)$)
//更新 P^k 中的粒子位置
$V_i(k+1) = V_i(k) \otimes P_{i\text{best}}(k) \otimes G_v(k)$
$P_i(k+1) = P_i(k) \otimes V_i(k+1)$
//通过 $P_i(k+1)$与对比更新 $P_{i\text{best}}(k)$
$P_{i\text{best}}(k+1) = P_i(k+1)$ if $P_{i\text{best}}(k+1) \prec P_i(k+1)$
Add $P_{i\text{best}}(k)$ to Q^k if Q^k does not contain $P_{i\text{best}}(k)$
}
AssignFitnessValues($G^k_{\text{best}} \cup Q^k$)
$G^{k+1}_{\text{best}} \leftarrow$Top λ of $G^k_{\text{best}} \cup Q^k$
$k \leftarrow k+1$
}

由于粒子的支配度值由 NSGA-II[11]相同的方式得到,则它的复杂度可以表示为 $O(mN^2)$,其中 m 为目标的个数,N 为粒子群的规模.对于 MDPSO 中粒子的更新操作,函数 Mutation, randomselection, BTselection 的复杂度均为常数,离散粒子更新过程和支配关系的判断也为常数,所以 MDPSO 中粒子的更新过程复杂度为 $O(N)$.因此 MDPSO 的复杂度为 $K_{\max}(O(mN^2) + O(N)) = O(mN^2 K_{\max})$

由于 MDPSO 适应度函数与 E^3-MOGA 相同,所以 E^3-MOGA 的算法复杂度也为 $O(K_{\max} mN^2)$.当 NSGA-II 的迭代次数为 $K_{\max}$时,它复杂度仍然是 $O(K_{\max} mN^2)$,所以算法 MDPSO, E^3-MOGA, NSGA-II 算法复杂度相同.

3.5 局部搜索策略

由于 SSC 问题的模型设置了十个优化目标,而一般当问题目标超过三个时,群体中的多数个体趋于相互不支配的状态,为了减轻粒子优化的压力,加快获取满足问题约束条件的候选解,在算法 MDPSO 中加入局部搜索策略.它是基于抽象服务的约束支配关系提出的,以抽象服务为单位局部改善粒子位置的约束满足状态,从而加快获取满足问题约束条件的粒子位置.

算法 2

Procedure Localsearch
Input: particle i's individual best solution $P_{i\text{best}}$
to be update and Global best particle k's solution
$P_{g\text{best}}$ for updating
Output: $P_{i\text{best}}$ be updated
{
for each abstract service j as $P_{i\text{best}}(S_j)$ and $P_{g\text{best}}(S_j)$
in $P_{i\text{best}}$ and $P_{g\text{best}}$
if ($P_{g\text{best}}(S_j) \prec P_{i\text{best}}(S_j)$)
update $P_{i\text{best}}(S_j)$ with $P_{g\text{best}}(S_j)$
endif
end for
}

对于 throughput,由属性聚合的特点可知,只有每个抽象服务的 throughput 满足约束条件时,组合服务的 throughput 才满足约束条件;对于 latency 和 cost,只有每个抽象服务的 latency 和 cost 尽量小时,组合服务的 latency 和 cost 才更容易满足约束条件.因此以下条件满足时,则视为抽象服务 i 支配抽象服务 j:

(1)抽象服务 i 的 throughput 满足对应的约束条件;
(2)抽象服务 i 的 latency 小于抽象服务 j 的 latency;
(3)抽象服务 i 的 cost 小于抽象服务 j 的 cost.

Procedure 给出了以抽象服务约束支配为基础的局部搜索的过程,比较 P_{ibest}和 P_{gbest}中每个抽象服务的支配关系,当 P_{gbest}中抽象服务支配 P_{ibest}对应的抽象服务时,即 $P_{gbest}(S_j) \prec P_{ibest}(S_j)$,则用 $P_{gbest}(S_j)$替换 $P_{ibest}(S_j)$.

在算法 MDPSO + 中,根据局部搜索策略,利用变异后产生的个体最优位置,对从解集 G_{best}中随机选取的全局最优解 $G_r(k)$进行更新.由于局部搜索策略的复杂度为常数,所以算法 MDPSO + 的复杂度仍然为 $O(mN^2K_{max})$.

4　实验设计

为验证算法的有效性,本文将提出的算法在 4 个不同规模的 Case 上进行了测试,并讨论了参数群体规模 swarm_size 和群体多样性阈值 α 对算法 MDPSO 的影响,最后将提出的算法 MDPSO、MDPSO + 与最近提出的算法 E^3-MOGA[5]及算法 NSGA-II[11]进行了对比分析.实验过程中根据目标函数值的变化情况和 hyper-volume 指标[12]对各算法的求解质量进行评价.所有算法用 C 语言实现,电脑配置 core(TM)2,2.00GH,3 GBRAM.

4.1　测试用例设计

实验设计了三种不同的抽象流程结构,对应 4 个不同的 Case,对于每个 Case,算法 MDPSO + 和 MDPSO 运行得到的解集与算法 NSGA-II,E^3-MOGA 运行得到的解集进行比较,并且每个解集由 10 次独立运行的均值得到.

对于本文所设计的 4 个 Case.根据各个 Case 的抽象流程结构和目标函数向量 $\boldsymbol{Q}$ 公式得到各 Case 的目标函数,同样依据各个 Case 的 SLA 等级约束和约束条件的公式,得到各 Case 的约束条件.由于所有对比算法具有相同的复杂度,所以将适应度函数评价次数作为终止条件,将它设置为 $L \times N \times 10^4$,其中 L 等级个数,N 为抽象服务的个数.

在 Case2 中的流程结构和等级约束条件如图 4 和表 1 所示;数据集由 Case1 中的数据集与文献[5]中数据集共同组成;并且在 Case2 中增加了选择结构.它的终止条件设置为适应度函数评价次数 1.2×10^5.

在 Case1 中的流程结构和等级约束条件如图 1 和表 1 所示;并且数据集与文献[5]中的数据集相同.依据终止条件设置,将 Case1 的适应度函数评价次数设置为 1.2×10^5.

在 Case3 和 Case4 中,它们流程结构分别为 10 和 15 个抽象服务的顺序连接,每个抽象服务与表 2 中的 3 个具体服务相关联,所有具体服务属性值都是确定的,每个抽象服务都与这些具体服务相关联.并且 Case3 和 Case4 的 SLA 等级约束条件如表 1 所示.依据终止条件设置,将 Case3 和 Case4 的适应度函数评价次数设置为 3.0×10^6 和 4.5×10^6.

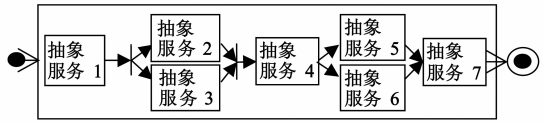

图4　Case2的流程结构

表 1　SLA 等级约束

用户分类		等级约束(上界/下界)			
		Throughput (下界)	Latency (上界)	Cost (上界)	Total cost (上界)
Case1	白金卡	12000	100	–	2000
	金卡	6000	130	–	
	银卡	2000	–	250	
Case2	白金卡	12000	150	–	3000
	金卡	6000	195	–	
	银卡	2000	–	375	
Case3 /Case4	白金卡	40000	80 * M	–	1000 * M
	金卡	20000	120 * M	–	
	银卡	15000	–	200 * M	

表 2　Case3 和 Case4 的具体服务

具体服务	QoS 属性		
	Throughput	Latency	Cost
1	10000	60	100
2	5500	100	50
3	2000	200	20

4.2　参数选取

本文所提算法 MDPSO 中,主要参数包括粒子群的

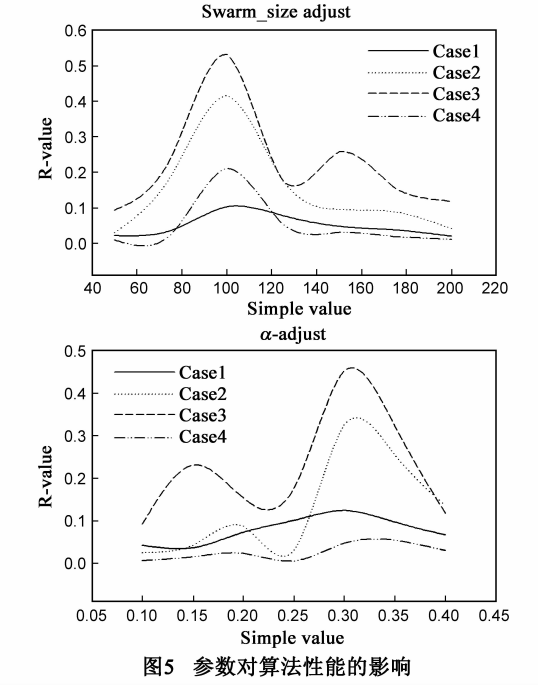

图5　参数对算法性能的影响

群体规模 Swarm_size 和群体多样性阈值 α. 参数 Swarm_size 对算法有显著影响,当粒子群规模过大时,粒子群不易收敛,反而影响求解的效果,实验将群体规模的取值范围设定为从 50 到 200 以 25 为增量的 7 个值. 群体多样性阈值 α 比参数 Swarm_size 敏感,将它的取值范围设置为从 0.10 到 0.40 以 0.05 为增量的 7 个值.

由于不同 Case 所得解集的 hyper-volume 值的数量级别不同,将它按如下的公式进行转换为 r 值:

$$r_{ij}^{k}=\frac{H_{ij}^{k}}{(10^{G_{ij}(\max(H_{ij}^{k}))})} \tag{24}$$

其中 H_{ij}^{k}为算法 MDPSO 在 Case i 上第 j 个参数取第 k 个值时所得解集的 hyper-volume 值, i 的取值为 1 到 4 中的一个整数, j 的取值为 1 或 2, k 为 1 到 7 中的一个整数. 当 i, j 固定, k 取遍所有取值时,所得 7 个解集的 hyper-volume 中最大的 hyper-volume 值由 $\max(H_{ij}^{k})$ 表示, $G_{ij}(\max(H_{ij}^{k}))$表示 $\max(H_{ij}^{k})$的数量级别,如 0.005 的数量级别为 0.001. 在这个过程中,算法 MDPSO 对于每种参数设置,在每个 Case 上运行 10 次,得到的 hyper-volume值经过式(24)进行转换,结果如图 5 所示. 先讨论 Swarm_size 的影响,设定 α 的初始值为 0.35,从图 5 中 Swarm_size 的参数调整图可以看出, Swarm_size = 100 效果较好. 设定 Swarm_size 为 100,调整 α 的取值,得到图 5 中 α 的参数调整图,可以看出 α = 0.35 时效果较好,所以认为 Swarm_size = 100 和 α = 0.35 为一种相对较好的设置.

4.3 与相关算法比较

关于 SLA 等级感知服务组合问题,将多目标离散粒子群算法 MDPSO 以及在该算法加入局部搜索策略后的 MDPSO + 算法与近期提出的 E^3-MOGA(多目标算法遗传算法)和 NSGA-II(快速非支配排序遗传算法)在 4 个不同流程结构的 Case 上进行实验,这些算法的参数和终止条件为前面小节所设定. 实验分为两部分,第一部分在 Case1 上评估所有算法对问题各个目标函数的优化情况;第二部分将所有算法实验所得解集的 hyper-volume 值进行比较.

4.3.1 目标函数值比较

图 6,7,8 分别给出了可行解对应的目标函数白金卡用户,金卡用户和银卡用户服务组合的 throughput 最大值的变化情况. 算法开始阶段为随机产生初始粒子,所以它们所对应的解为不可行解,大概迭代 60 次以后各算法找到满足 SLA 等级约束的可行解. 然后,随着可行解的优化,目标函数的最大值,即白金卡用户,金卡用户和银卡用户服务组合的 throughput 的最大值不断增大,直到接近平稳的状态. 由于各算法在迭代过程中按照适应度函数值的大小保留较优的解集,并且多目标的适应度函数值不与单个目标函数一致变化,所以解集对应的目标函数 throughput 的最大值存在波动. 由于算法 MDPSO 能有效抑制早熟收敛的情况,所以如图 6 ~ 8 所示,与算法 E^3-MOGA 和算法 NSGA-II 相比,算法 MDPSO 能找到 throughput 更优的解. 由于局部搜索策略能加快获得满足约束条件的候选解并且提升优化效果,所以如图 6 ~ 8 所示,在算法 MDPSO 的基础上,算法 MDPSO + 能找到效果更好的解.

图 9 ~ 11 分别给出了可行解所对应的目标函数白金卡用户,金卡用户和银卡用户服务组合的 latency 最小值的变化情况. 和 throughput 类似,算法 MDPSO 找到了比算法 E^3-MOGA 和算法 NSGA-II 更优的解,算法 MDPSO + 找到了比算法 MDPSO 更优的解. 图 12 ~ 14 分别给出了可行解所对应的目标函数白金卡用户,金卡用户和银卡用户服务组合的 cost 最小值的变化情况.

图 15 给出三类用户总的 cost 最小值的变化情况. 如表 3 所示,SLA 等级协议对白金卡用户和金卡用户的 cost 没有约束,但是对白金卡用户组合 throughput 和 latency 的要求高于金卡用户组合的 throughput 和 latency,所以白金卡用户服务组合趋向于选择高性能的具体服务,而高性能具体服务的 cost 也会较高,因此白金卡用户的 cost 要高于金卡用户和银卡用户的 cost 值. 并且与算法 E^3-MOGA、算法 NSGA-II 和算法 MDPSO 相比,算法 MDPSO + 能得到 cost 更优的解,表明局部搜索策略让算法具有更强的搜索能力.

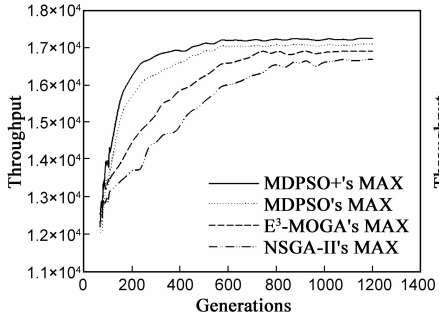

图6 白金卡用户的throughput

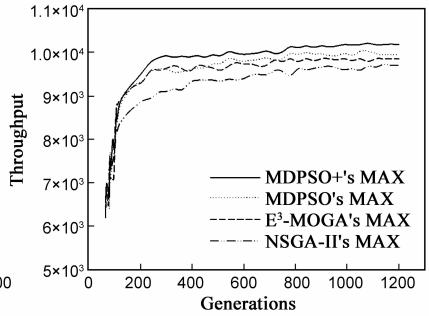

图7 金卡用户的throughput

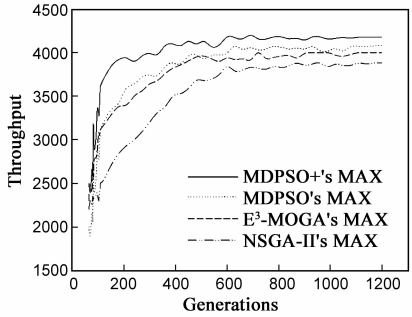

图8 银卡用户的throughput

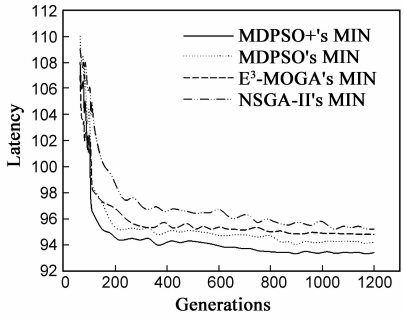

图9　白金卡用户的latency

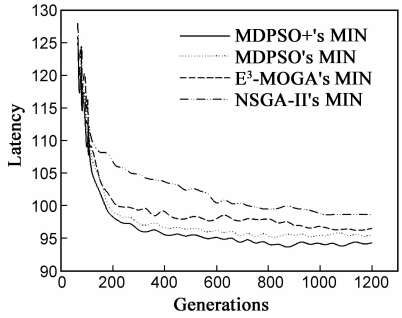

图10　金卡用户的latency

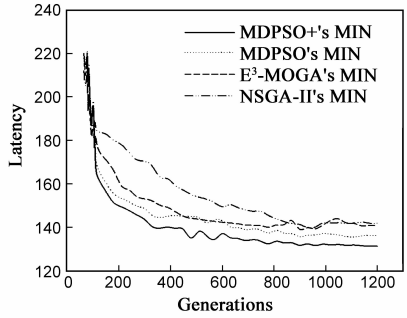

图11　银卡用户的latency

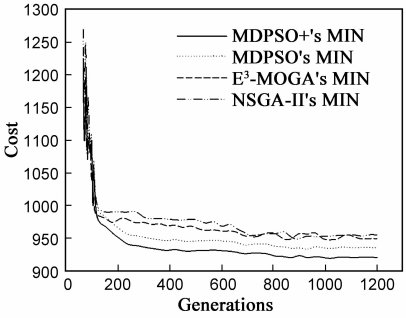

图12　白金卡用户的cost

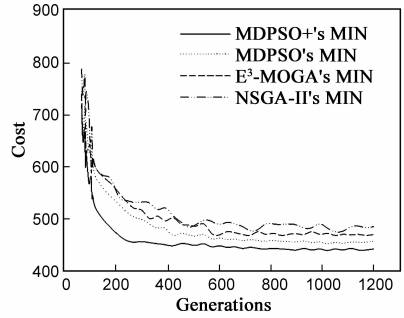

图13　金卡用户的cost

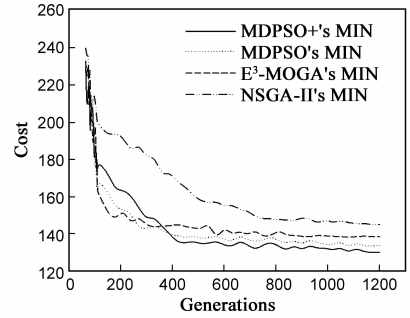

图14　银卡用户的cost

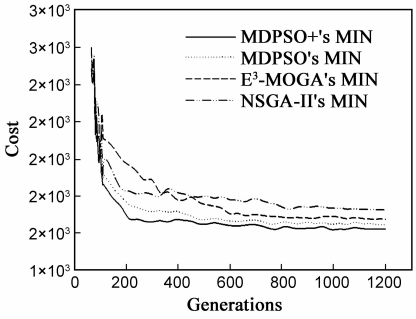

图15　所有用户总的cost

4.3.2　Hyper-volume 值比较

将所有算法在 4 个不同流程结构的 Case 上进行实验,所得解集的 hyper-volume 值进行比较.表 3 给出了每个算法在所有 Case 上运行得到 20 次所得解集 hyper-volume 值的,最大值,最小值,均值和从表中能看出来算法 MDPSO 的求得解集的最大 hyper-volume 值,最小 hyper-volume 值和均值都比算法 E^3-MOGA、算法 NSGA-II 的相应值大,而且在 Case1, Case2, Case4 中变异系数(CV%)最小,在 Case3 中算法 E^3-MOGA 变异系数最小,但是算法 MDPSO 的变异系数与之相对.因此认为算法 MDPSO 比 E^3-MOGA 和算法 NSGA-II 求解质量高而且性能稳定.同样如表 3 所示,算法 MDPSO + 所求解集的 hyper-volume 值的各统计量比算法 MDPSO 的相应值更优,所以认为算法 MDPSO + 比算法 MDPSO 更有效.这些主要由于算法 MDPSO 的粒子更新策略使得粒子跟随个体最优和全局最优解运动,在全局范围内搜索;粒子变异策略为粒子群引入变异信息,增加群体多样性,使得搜索不易陷入局部最优.算法 MDPSO + 中的局部搜索策略从局部改变粒子位置的约束状态,使得该算法能更快的找到问题的可行解,并且不断优化.

表 3　不同算法 Hyper-volume 值对比(最大/最小/均值/变异系数)

Algorithm	Case 1	Case 2	Case 3	Case 4
NSGA-II	0.0002/0.00009/0.00008(0.7003)	0.0015/0.00004/0.0006(0.7152)	0.3731/0.00313/0.1079(0.80997)	0.0012/0.00002/0.0005(0.692)
E^3-MOGA	0.00043/0.00001/0.0002(0.7227)	0.0308/0.00035/0.0081(0.9941)	1.8372/0.0835/0.5515/(0.78966)	0.0176/0.0006/0.0046 (0.9939)
MDPSO	0.0009/0.00012/0.00035(0.6869)	0.0406/0.00269/0.0137(0.6963)	3.2691/0.0556/1.1251(0.78185)	0.0302/0.0027/0.0119(0.7366)
MDPSO +	0.00134/0.00015/0.0005(0.6763)	0.0606/0.0043/0.0228/(0.6863)	6.7691/0.4071/2.5601/(0.77937)	0.0622/0.0071/0.0215(0.6892)

5　结语

针对 SLA 等级感知服务组合问题,本文提出了多目标离散粒子群算法(MDPSO),并设计了将局部搜索策略融入该算法的 MDPSO + 算法.在算法 MDPSO 中定义了离散粒子更新策略,用于对问题解空间的全局搜索;

提出了粒子变异策略增加群体多样性,抑制算法的早熟收敛的情况.在算法 MDPSO + 中,融入了局部搜索策略,它利用抽象服务约束支配关系从局部改善粒子位置对约束的满足程度,加快问题可行解的查找速度;最后实验表明算法 MDPSO + 在所求解集质量方面效果显著.此外,粒子群算法、遗传算法解决在个问题中都没有很好地处理分支结构,因此在该问题中对分支结构的合适处理是我们下一步的研究目标.

参考文献

[1] Yu T, Zhang Y, Lin K J. Efficient algorithms for Web services selection with end-to-end QoS constraints [J]. ACM Transactions on the Web (TWEB), 2007, 1(1): 6.

[2] 龙军,袁鑫攀,桂卫华.基于环境感知的可信 QoS 评价与服务选取策略[J].电子学报,2012,40(6):1133 – 1140.
Long Jun, Yuan Xinpan, Gui Weihua. A policy for trusted QoSevaluation and service selection with environment aware [J]. Acta Electronica Sinica, 2012, 40(6): 1133 – 1140. (in Chinese)

[3] Cardellini V, Casalicchio E, Grassi V, et al. Moses: A framework for qos driven runtime adaptation of service-oriented systems [J]. IEEE Transactions on Software Engineering, 2012, 38(5): 1138 – 1159.

[4] 夏亚梅,程渤,陈俊亮,等.基于改进蚁群算法的服务组合优化[J].计算机学报,2012,35(2):270 – 281.
Ya-Mei X I A, CHENG B, Jun-Liang C, et al. Optimizingservices composition based on improved ant colony algorithm [J]. Chinese Journal of Computers, 2012, 35(2): 270 – 281. (in Chinese)

[5] 万长林,韩旭,牛温佳,等.基于动态描述逻辑的服务组合及质量模型[J].电子学报,2010,38(8):1923 – 1928.
WAN Chang-lin, HAN Xu, NIU Wen-jia et al. Dynamic Description Logic Based Web Service Composition and QoS Model[J]. Acta Electronica Sinica, 2010, 38(8): 1923 – 1928. (in Chinese)

[6] Wada H, Suzuki J, Yamano Y, et al. E3: Amultiobjective optimization framework for SLA-aware service composition [J]. IEEE Transactions on Services Computing, 2012, 5(3): 358 – 372.

[7] 温涛,盛国军,郭权,等.基于改进粒子群算法的 web 服务组合[J].计算机学报,2013,36(5):1031 – 1046.
Tao W, Guo-Jun S, Quan G. Webservice composition based on modified particle swarm optimization [J]. Chinese Journal of Computers, 2013, 36(5): 1031 – 1046. (in Chinese)

[8] 张长胜,孙吉贵,欧阳丹彤.一种自适应离散粒子群算法及其应用研究[J].电子学报,2009,37(2):299 – 304.
ZHANG Chang-sheng, SUN Ji-gui, OUYANG Dan-tong. A self-adaptive discrete particle swarm optimization algorithm [J]. Acta Electronica Sinica, 2009, 37(2): 299 – 304. (in Chinese)

[9] Kennedy, J Eberhart R C. Particle swarm optimization [A]. Proceedings of the 1995 IEEE International Conference on Neural Networks. Piscataway [C]. NJ, Perth, Australia: IEEE Service Center, 1995. 1942 – 1948

[10] 彭喜元,彭宇,戴毓丰.群智能理论及应用[J].电子学报,2003,31(12):1982 – 1988.
Peng Xi-yuan, Peng Yu, Dai Yu-feng. Swarm intelligence theory and applications [J]. Acta Electronica Sinica, 2003, 31(12): 1982 – 1988. (in Chinese)

[11] Deb K, Pratap A, Agarwal S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II [J]. IEEE Transactions on Evolutionary Computation, 2002, 6(2): 182 – 197.

[12] While L, Bradstreet L, Barone L. A fast way of calculating exact hypervolumes [J]. IEEE Transactions on Evolutionary Computation, 2012, 16(1): 86 – 95.

作者简介

尹 浩 女,1985 年 5 月出生,辽宁沈阳人.2008 年毕业于大连交通大学信息与计算科学 + 软件工程专业,2009 年获得伊利诺大学芝加哥分校 MBA 学位,现在为东北大学计算机应用技术的博士研究生,从事服务计算方面的研究.

E-mail: orange258312241@163.com

张长胜 男,1980 年 6 月出生,辽宁沈阳人.分别于 2003 年、2006 年和 2009 年在吉林大学获工学学士、工学硕士和工学博士学位.现为东北大学副教授、硕士生导师,主要从事智能算法、约束程序及 Web 智能信息处理技术等方面的研究工作.