# 基于改进蚁群算法的服务组合优化

夏亚梅[1),2)] 程 渤[1)] 陈俊亮[1)] 孟祥武[1)] 刘 栋[1)]

[1)](北京邮电大学网络与交换技术国家重点实验室 北京 100876)

[2)](北京邮电大学软件学院 北京 100876)

**摘 要** 为进行服务组合优化及适应服务组合优化过程中Web服务的动态性、不稳定性以及多种QoS属性限制等问题,提出一种多信息素动态更新的蚁群算法MPDACO,包括MPDACO局部优化算法和MPDACO全局优化算法,该算法基于建立的服务组合模型,在基本蚁群算法基础上进行研究和改进,可以适应服务组合优化过程中发生的服务无效以及服务中QoS变化等情况.另外,为使算法能较快地收敛于最优解,在实验基础上对蚁群算法策略进行了改进.为验证以上算法的有效性,在一个旅游领域的服务推荐系统中对算法进行了仿真实验,实验结果表明文中提出的算法较基本蚁群算法及一种应用于服务选择的遗传算法有更好的性能.

**关键词** 语义网;服务组合;服务选择;蚁群算法;最优化

**中图法分类号** TP311 **DOI号**: 10.3724/SP.J.1016.2012.00270

# Optimizing Services Composition Based on Improved Ant Colony Algorithm

XIA Ya-Mei[1),2)] CHENG Bo[1)] CHEN Jun-Liang[1)] MENG Xiang-Wu[1)] LIU Dong[1)]

[1)](*State Key Laboratory of Networking and Switching Technology, Beijing University of Posts & Telecommunications, Beijing* 100876)

[2)](*School of Software Engineering, Beijing University of Posts & Telecommunications, Beijing* 100876)

**Abstract** In order to optimize services composition, adapt the dynamic and instable characteristics of Web services and the limitation of multi-QoS attributes in the process of services composition, this paper puts forward an algorithm named Multi-pheromone and Dynamically Updating Ant Colony Optimization Algorithm (MPDACO), which includes one global optimizing algorithm and another local optimizing algorithm. The algorithm, which is based on the ACO and composition model that has been built, can fit for such conditions as service invalidation, QoS changing, etc. In addition, the algorithm has improved the ACO strategy on the basis of experiment to make itself be able to converge to optimal solution. In order to verify the feasibility of the above algorithms, this paper makes a simulation experiment on a prototype in tourism, and the results show that the two algorithms are more effective than ACO and the Genetic Algorithm applied to service selection.

**Keywords** semantic; services composition; service selection; ant colony algorithm; optimization

---

收稿日期:2008-05-26;最终修改稿收到日期:2012-01-05.本课题得到国家"九七三"重点基础研究发展规划项目基金(2011CB302704)与国家自然科学基金(61001118)资助.**夏亚梅**,女,1976年生,博士,讲师,主要研究方向为动态服务组合、最优化理论、云计算.E-mail:xiayamei@gmail.com.**程 渤**,男,1975年生,博士,副教授,主要研究方向为服务组合、物联网等.**陈俊亮**,男,1933年生,教授,博士生导师,中国科学院院士,中国工程院院士,主要研究领域为语义网.**孟祥武**,男,1966年生,博士,教授,博士生导师,主要研究领域为通信软件、网络服务.**刘 栋**,男,1981年生,博士,讲师,主要研究方向为语义Web服务、上下文感知.

## 1 引　言

动态服务组合技术是当前的一大研究热点.在动态服务组合过程中需要对相同功能的多个服务及组合服务进行选择,以提供最适合用户的服务.而随着提供服务的日益丰富,服务选择问题的规模急剧增长.因而动态服务组合中的服务选择问题急需有效的方法来解决.

服务选择问题属于组合优化范畴,是一个 NP 难问题,因此本文中把"服务选择"称为"服务组合优化".另外,由于 Web 服务状态不稳定、服务的 QoS 属性值有时不明确且经常发生变化等原因,要求服务组合优化算法既要具有良好的性能,又要能适应 Web 服务状态的不稳定、服务的 QoS 值不明确以及服务中 QoS 的变化等情况.

本文提出一种改进的蚁群算法对服务组合进行优化.它具有较好的性能,同时也能适应组合优化问题的动态性.这里选用蚁群算法解决以上的组合优化问题,是因为蚁群算法具有以下特点:

(1)由蚁群算法中蚂蚁选路机制决定,算法流程能够适应寻优过程中流程的动态变化,也能够适应优化过程中优化值的动态变化;另外,蚁群算法中的信息素具有信息素挥发因子 $\rho$,可以避免蚂蚁信息素的无限积累,使得蚂蚁的选路行为能根据外部环境的动态变化而变化.以上这些特点使得蚁群算法非常适合动态服务组合中的服务选择问题.

(2)可方便地对信息素机制进行改进,能设立多种信息素,以表示组合服务的多种 QoS 优化属性约束.

(3)可适应于 QoS 属性值已知的情况,同时也可以在 QoS 属性值不明确的情况下应用.这一特点使算法非常适合于服务推荐(组合过程中与用户进行交互的一种服务组合方法)系统.解释如下:服务推荐系统初始建立时,各服务的 QoS 值不明确,此时根据算法设立的启发因子进行服务的随机选择;随着服务调用记录的进一步增加,算法能根据历史记录选择越来越好的推荐服务;如果一些服务的状态变为不可用时,则暂时禁用这一服务;如果所提供服务的 QoS 发生变化,则根据信息素的挥发及更新规则重新生成新的服务 QoS 反馈,并重新调整推荐服务,较真实地模拟了服务推荐系统中的服务选择问题.

(4)蚁群算法是继遗传算法之后出现的又一种全局优化算法.同遗传算法相比,蚁群算法概念简单,需要设置的参数少,不需要交叉、变异等操作,而优化结果甚至优于遗传算法.

蚁群算法一经提出便发展迅速,现在已应用到多个组合优化领域,成为优化算法中不错的选择[1-3].

## 2 相关研究

蚁群算法优化在许多领域都有应用,如 TSP 问题、背包问题、测试集优化、神经网络、人工智能、信号传输等,有最新研究还将蚁群算法应用于网格服务可靠性保证[4]、软件测试参数的优化等[5].以上蚁群算法的应用都取得了较好的研究结果.但蚁群算法本身存在一些固有的缺陷,如收敛速度慢,易于停滞等.为了克服这些缺点,不少学者提出了改进算法,如 Stutzle 等人[6]提出了一种 MMAS (Max-Min Ant System)算法,其基本思想是对路径上的信息素进行限制,对信息素进行局部更新,以期克服停滞问题等.另外,对于蚁群算法收敛理论方法的研究,黄翰等人[7]基于吸收态 Markov 过程的数学模型,提出了蚁群算法的收敛速度分析理论,给出了估算蚁群算法期望收敛时间的几个理论方法.

目前的服务选择一般是基于服务的 QoS 属性约束进行的,且大多没有考虑服务选择问题的动态性.采用的计算方法有穷尽计算方法与启发式计算方法两种.采用穷尽计算的组合优化方法存在扩展性差、计算量相对较大的弊端,就目前的研究结果来看,启发式计算方法较穷尽计算方法有更好的性能[8].

张成文等提出一种基于遗传算法的 QoS 感知的 Web 服务选择算法,可以从所有组合路径的组合方案中选出满足用户 QoS 需求的服务方案,但该方法在优化操作前先对优化问题编码,没有考虑优化过程中的动态性问题[8].Zeng 等人[9]提出用于服务组合的 QoS 感知的中间件,他们考虑到了 QoS 值发生变化的动态性,然而没有考虑服务不可用以及由此引起的服务组合流程的动态变化问题,且该 QoS 驱动的中间件是采用穷尽计算方法,只考虑了局部优化问题,没有对组合服务进行全局的优化选择.倪晚成等人[10]用 Dijkstra 最短路径算法进行服务组合问题的优化,选择的服务能满足 QoS 需求,但对服务状态的复杂性没有考虑.

本文将蚁群算法引入到服务组合领域,提出一

种多信息素动态更新的蚁群算法 MPDACO，该算法可以动态地适应网络中服务无效以及服务的 QoS 变化等情况. 为提高优化算法的性能，根据服务组合优化问题的特点，对优化问题进行了分解，提出的 MPDACO 包含两个子算法，一种用于在单一服务中选择最优的服务，称为 MPDACO 局部优化算法，另一种用于选择最优的组合服务，称为 MPDACO 全局优化算法. 为验证 MPDACO 的有效性，进行了仿真实验，测试结果表明本文提出算法的性能不低于文献[8]中算法的性能.

本文第 2 节是相关研究介绍；第 3 节对服务组合优化问题进行建模；第 4 节给出 MDPACO 算法，并验证算法的有效性；第 5 节以一个旅游服务推荐系统中的应用为例对文中算法进行了仿真实验；第 6 节给出结论.

## 3 服务组合优化问题的建模

服务组合优化需要考虑多个 QoS 属性约束，设服务 $S$ 具有 $n$ 个 QoS 属性，表示为 $\{Q_1, Q_2, \cdots, Q_n\}$. 而人们对于服务的各个 QoS 属性，一般都有一个最低或最高能接受的约束值，设为 $q_1, q_2, \cdots, q_n$，单一服务选择的目的是寻找具体的服务实例 $s'$，$s'$ 的各个 QoS 属性满足以下约束：

$$Q_1 > q_1;$$
$$Q_2 > q_2;$$
$$\cdots$$
$$Q_n < q_n;$$

在满足以上约束的情况下寻找 QoS 总和最高的服务，QoS 总和的计算参见以下说明.

由于每个 QoS 取值之间可能有较大差别，有时甚至不属于同一个数量级，另外，有的 QoS 属性取值越高表示服务的 QoS 越高，如服务带宽，有的 QoS 属性取值越高反而表示服务的 QoS 越低，比如服务响应时间，因而简单的加总会影响某些属性值，不能准确地表达 QoS，因此给每个 QoS 属性设定相应的函数 $Q_1(s), Q_2(s), \cdots, Q_n(s)$，使各个 QoS 属性经函数运算后的结果值正比于 QoS，其值越大表示用户对相应服务的 QoS 满意度越高. 另外再设定相应的权值 $w_1, w_2, \cdots, w_n$，使各个 QoS 能合理地加总. 服务 QoS 总和的计算公式定义为 $Q_s = \sum_{i=1}^{n} Q_i(s) \times w_i$.

以上是单一服务选择问题，也称为局部优化问题[9]. 在服务组合系统中，需要对多个单一服务进行选择，以组合成为一个组合服务，而在多个组合服务之间还需要进一步选择以获取最高 QoS 的组合服务. 因此在局部优化之后，还需要进一步进行全局优化.

目前存在的动态服务组合方法有基于接口匹配的服务组合方法[11]、基于 UML 建模的服务组合方法[12]、基于实时通信的服务组合方法[13]，还有的服务组合方法在服务组合过程中考虑了用户的偏好，以提高组合服务的 QoS[14-15]. 本文对以上几种服务组合方法生成的服务组合图统一地抽象表示为图 1 的形式.

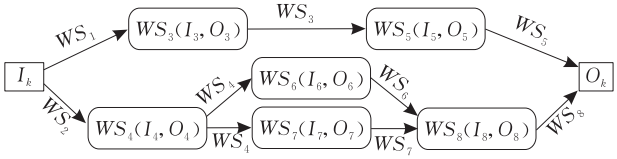


图 1 服务组合示例图

图 1 表示的服务组合示例图是伴随动态服务组合过程而动态生成的，是一个简单有向图. 图中的每个节点表示服务匹配，图中仅列出了接口匹配的情况，其中 $WS_i(i=1,2,\cdots)$ 表示服务，$I$ 表示输入接口，$O$ 表示输出接口，$WS_i$ 与 $WS_j$ 之间有节点相连，表示 $WS_i$ 的输出接口与 $WS_j$ 的输入接口匹配. 图 1 是一个简单的示例图，图中含有较少的服务节点和服务边，较容易找到最优的服务组合. 但服务组合系统中，有时存在某些复杂组合服务涉及的服务较多，因而节点和边也较多，寻优的复杂度将大幅提高. 下面对问题进行抽象，把服务组合优化问题抽象为一个网状图形寻找最佳路径的过程. 相关定义如下.

**定义 1.** Web 服务组合具有多种特性，如顺序性、并发性(并发性的情况在图 1 中未列出)等. 顺序性的各个任务按照一定的执行顺序运行，这样的组合服务在图中可用一条路径表示；并发性表示多个服务并发执行，不能用一条路径来表示，这种情况不利于使用优化算法，因此对这种情况下的组合服务进行串行化处理，使并行的服务前后相连，在逻辑上转换为顺序执行的服务. 这种转换不影响实际的服务组合流程，但有利于优化算法的运行.

如图 2 中的并行执行服务，可串行化地表示为图 3 中的形式.

**定义 2.** 服务组合图是一个动态生成的单向简单连通图，存在一个原点表示服务组合的输入，存在一个终点，表示服务组合的输出. 组合图中不存在孤立点，不存在悬点，除了起点和终点，每个节点的入度与出度均≥1，且不存在回路. 至少存在一条路

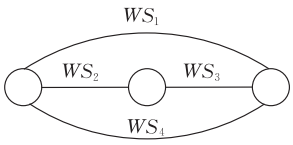


图 2　并行执行服务示意图

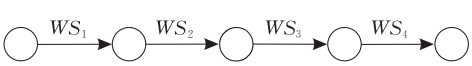


图 3　并行服务串行化示意图

径从原点通往终点.

**定义 3.**　服务组合图中的节点表示服务匹配，每条弧表示一个抽象服务(具有某一功能的同一类服务，有多个具体的候选服务实例，具体的候选服务在图中未画出)，从起点到终点的一条路径表示一个完整的组合服务，存在从起点到终点的多条路径时表示有多个不同的组合服务完成相同的功能.

**定义 4.**　每条弧具有多个 QoS 属性约束，对应相应服务的 QoS，对于每一 QoS 属性约束，相应的在 MDPACO 算法中设立一种信息素，MDPACO 算法可根据各个单一 QoS 进行服务组合优化，也可根据 QoS 总和进行服务组合优化.

服务组合模拟图表示如图 4.

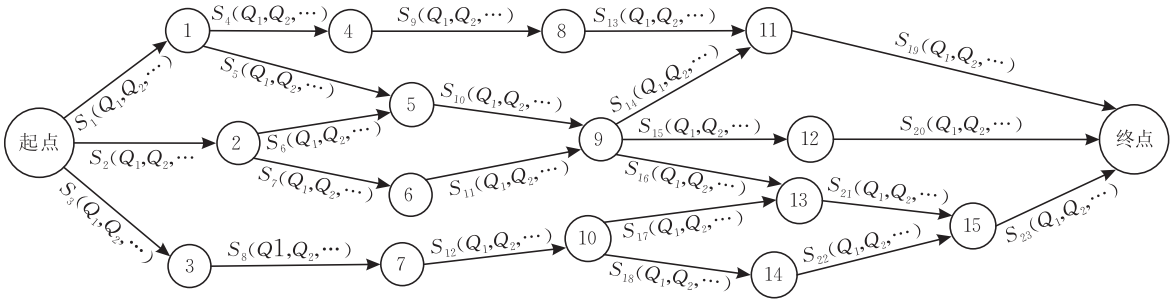


图 4　服务组合模拟图

图 4 中 $S_i(i=1,2,\cdots)$表示不同的服务，$Q_i(i=1,2,\cdots)$表示不同的 QoS 属性. 根据以上建立的服务组合优化模型，服务组合优化问题就是在以上的动态服务组合图中寻找最长路径(对于数值越低表示组合服务满意度越高的情况，是寻找最短路径). 下面就可以进行服务组合优化了. 由于基本蚁群算法(为与文中的改进蚁群算法进行区别，把根据蚁群算法原理实现的蚁群优化算法称为基本蚁群算法)存在以下局限性：

(1)基本蚁群算法中只有一种信息素，因而无法解决组合服务中多种 QoS 属性约束的问题.

(2)基本蚁群算法在求解时路径和路径权值是稳定的，因而不适用于动态的 Web 服务组合.

(3)蚁群算法主要利用正反馈原理强化较优解，当进化到一定代数后就容易因为较优解的信息量不断强化使得蚂蚁大量聚集于较少的几条路径上，出现早熟、停滞现象，使得到的最优解是局部最优的.

针对以上基本蚁群算法的不足，提出了一种改进的蚁群算法，以适应动态服务组合优化问题的动态性及提高算法性能.

本文同时实现了基本蚁群算法对服务单一 QoS 属性约束的优化，用于与 MPDACO 算法的性能进行比较. 由于篇幅所限，此处不再赘述.

## 4　多信息素动态更新蚁群算法(MPDACO)

在 MPDACO 算法中，通过改变蚁群算法中的状态转移概率以及信息素规则等完成以上需求. 状态转移概率、信息素规则这两部分在 4.1 节和 4.2 节中给出了详细说明和定义. 另外本节还包括其它 3 小节内容，其中动态性设计在 4.3 节中进行说明，4.4 节给出了算法的改进策略，4.5 节介绍具体的 MPDACO 算法，包括 MPDACO 局部优化算法和 MPDACO 全局优化算法.

### 4.1　状态转移概率

在蚁群算法中，蚂蚁以一定的概率选择信息素多的服务，表示这是更优的服务. 通过这样的正反馈机制越来越多的蚂蚁会选择更优的服务，在全局范围内也即找到了最优的服务. 设 $m$ 是蚁群中蚂蚁的数量，$n$ 表示服务组合图的节点个数，$\tau_{ij}(t)$为 $t$ 时刻路径$(i,j)$上的信息素量，$\tau_{ij}(t)$决定着状态转移概率的计算. 以下分为根据单一 QoS 进行服务组合优化和根据 QoS 总和进行服务组合优化两种情况来讨论 $\tau_{ij}(t)$的计算.

(1) 根据单一 QoS 进行服务组合优化.

根据单一 QoS 进行服务组合优化时,MDPACO 算法就退变成为了单一信息素的基本蚁群算法,因而信息素 $\tau_{ij}(t)$ 的计算公式与基本蚁群算法中的定义相同. 在基本蚁群算法中 $\tau_{ij}(t)$ 一般通过单一信息素累加得到.

(2) 根据 QoS 总和进行服务组合优化.

服务组合优化时,有时需要根据服务的各个 QoS 综合计算得到最优的组合服务. 对应于本文的 MPDACO 算法,就需要综合多种信息素来计算得到 $\tau_{ij}(t)$. 设路径$(i,j)$表示的服务为 $s$,且服务 $s$ 共包含 $n$ 个 QoS 属性值 $Q_1,Q_2,\cdots,Q_n$,对应的转换函数为 $Q_1(s),Q_2(s),\cdots,Q_n(s)$,相应的权值为 $w_1,w_2,\cdots,w_n$(参见第 3 节中 QoS 计算说明),则服务 $s$ 第 $r$ 个 QoS 属性对应的信息素 $pheromone_r$ 的计算公式为:

$$pheromone_r = Q_r(s) \times w_r, r = 1,2,\cdots,n;$$

从而,$\tau_{ij}(t)$的计算公式如下:

$$\tau_{ij}(t) = \sum_{r=1}^{n} pheromone_r;$$

综合以上(1),(2)得到 $\tau_{ij}(t)$的计算公式. 下面进一步得到状态转移概率计算公式:

$$p_{ij}^{k}(t)=\begin{cases}\dfrac{[\tau_{ij}(t)]^{\alpha}\cdot[\eta_{ik}(t)]^{\beta}}{\sum\limits_{s\subset allowed_k}[\tau_{is}(t)]^{\alpha}\cdot[\eta_{is}(t)]^{\beta}}, & j\in allowed_k\\ 0, & \text{否则}\end{cases}$$

式中,$p_{ij}^{k}(t)$表示在 $t$ 时刻第 $k$ 只蚂蚁由节点 $i$ 转移到与 $i$ 有边相连的节点 $j$ 的状态转移概率,$allowed_k$ 是所有与 $i$ 节点有边相连的节点集合. 在本文的算法中 $allowed_k$ 是一个动态生成的节点集合,表示蚂蚁 $k$ 下一步允许选择的服务. $\alpha$ 为信息启发式因子,表示轨迹的相对重要性,反映了蚂蚁在运动过程中所积累的信息在蚂蚁运动时所起的作用,其值越大,则该蚂蚁越倾向于选择其它蚂蚁经过的路径,蚂蚁之间协作越强. $\beta$ 为期望启发式因子,表示能见度的相对重要性,反映了蚂蚁在运动过程中启发信息在蚂蚁选择路径中的受重视程度,其值越大,则该状态转移概率越接近贪心规则. $\eta_{ij}(t)$ 为启发函数,若该服务无历史 QoS 信息则 $\eta_{ij}(t)$ 取经验值,以不影响程序的后续运行为原则,如可取 $\eta_{ij}(t)=1$;否则

$$\eta_{ij}(t)=\sum_{r=1}^{n} d_r, \quad \text{其中 } d_r = Q_r(s)\times w_r;$$

其中,$d_r$ 表示该服务 $s$ 最新的第 $r$ 种 QoS 属性的值.

### 4.2 信息素更新规则

由于问题规模不同,因而 MPDACO 局部优化算法与 MPDACO 全局优化算法的信息素更新规则有所不同.

(1) MPDACO 局部优化算法的信息素更新规则

$t+n$ 时刻在节点$(i,j)$弧段上第 $r$ 种 QoS 属性信息素的更新规则如下:

$$Q_{rij}(t+n) = (1-\rho)\cdot Q_{rij}(t) + \rho\cdot\Delta Q_{rij}(t+n),$$

其中:$Q_{rij}(t)$表示$(i,j)$弧段上 $t$ 时刻时第 $r$ 种 QoS 属性 $Q_r$ 的值,$\Delta Q_{rij}(t+n)$表示 $t+n$ 时刻某只蚂蚁新感受到的服务$(i,j)$的第 $r$ 种 QoS 属性 $Q_r$ 的值. $\rho$ 表示信息素更新因子,$1-\rho$ 表示信息素残留因子,这里 $\rho$ 表示服务以多大的程度借鉴新体验到的 $Q_r$ 值,$1-\rho$ 表示以多大的程度借鉴之前的 $Q_r$ 值.

(2) MPDACO 全局优化算法的信息素更新规则

$(i,j)$弧段表示服务的第 $r$ 种 QoS 对应的信息素更新规则定义如下:

$$pheromone_{rij}(t+n)=(1-\rho)\cdot pheromone_{rij}(t)+\Delta pheromone_{rij}(t+n),$$

其中:$\Delta pheromone_{rij}(t+n)=\sum_{k=1}^{m}\Delta pheromone_{rij}^{k}(t+n)$;

$\rho$ 表示信息素挥发因子,$1-\rho$ 表示信息素残留因子,为了防止信息的无限积累,$\rho$ 的取值范围为 $\rho\subset[0,1)$;$\Delta pheromone_{rij}(t+n)$表示本次循环中$(i,j)$弧段表示服务第 $r$ 种 QoS 对应信息素的增量,$\Delta pheromone_{rij}^{k}(t+n)$表示第 $k$ 只蚂蚁本次循环中留在$(i,j)$弧段表示服务第 $r$ 种 QoS 对应信息素的增量.

由以上的信息素更新规则可以看出,信息素能够根据服务 QoS 值的变化而进行动态的调整.

### 4.3 动态性设计

Web 服务具有一定的随机性、不稳定性,且服务的 QoS 属性常常发生变化,因此在服务组合优化过程中就需要考虑服务不可用及 QoS 变化的情况. 对于服务的 QoS 变化的问题,可以通过改变弧的权值来实现,参见 4.2 节. 而当某服务不可用时,则意味着包含该服务的组合服务都将不可用,这就需要在服务组合图中删除包含该服务的组合服务集合. 由于删除组合服务的情况较为复杂,我们通过以下算法来进行服务组合图的动态调整.

**算法 1.** 服务删除算法.

设待删除服务对应的弧为 $\lambda$,$\lambda$ 的初始节点为 $a$,终止节点为 $b$,设节点 $a$ 的出度为 $od(a)$,节点 $b$ 的入度为 $id(b)$. 删除弧 $\lambda$,则 $od(a)=od(a)-1$,$id(b)=id(b)-1$. 对执行减 1 操作后的入度 $od(a)$和出度 $id(b)$分情况讨论如下:

1. 如果 $od(a)\geq 1, id(b)\geq 1$，不进行其它操作，跳转到步 5；

2. 如果 $od(a)\geq 1, id(b)=0$，表示节点 $b$ 因为删除弧 $\lambda$ 而变成了悬点：删除以节点 $b$ 为初始节点的所有弧，对删除弧的所有终止节点的入度进行减 1 操作，考察这些节点，如果存在节点 $c, id(c)=0$，则对 $c$ 重复操作步 2，直到图中所有节点的入度$\geq 1$，然后跳转到步 5；

3. 如果 $od(a)=0, id(b)\geq 1$，表示节点 $a$ 因为删除弧 $\lambda$ 而变成了悬点，执行以下操作：删除以节点 $a$ 为终止节点的所有弧，对删除弧的所有初始节点的出度进行减 1 操作，考察这些节点，如果存在节点 $c$，$od(c)=0$，则重复执行步 3，直到所有节点的出度$\geq 1$，然后跳转到步 5；

4. 如果 $od(a)=0, id(b)=0$，则重复执行步 2，3，再跳转到步 5；

5. 结束.

### 4.4　算法改进策略

针对蚁群算法较早收敛于局部最优解、收敛速度慢等缺点，应用以下规则对蚁群算法进行改进以提高其性能.

**规则 1.**　一轮蚂蚁寻优结束后，比较各条路径对应信息素总和的大小，只对本轮中信息素浓度最大(表示 QoS 总和最大)的蚂蚁路径进行信息素更新. 经测试发现，算法收敛效率和寻找到最优解的概率都大大提高.

**规则 2.**　为减小算法早熟、停滞及收敛于局部最优解的概率，设定最优路径列表 $L$，设列表的长度为 $l$，$l$ 的值可根据问题的规模而定. 对每轮蚂蚁寻优中信息素浓度(指本轮新感知到的 QoS 对应的信息素浓度)最大的蚂蚁路径进行比较，并按大小顺序保存 $l$ 个最优的(信息素浓度最大)路径到 $L$ 中，最大的排在 $L$ 的第一个元素中，依次类推进行排列. 在算法到达最大循环次数之后，取 $L$ 中的第一个元素与信息素浓度最高的路径比较，如果相同，表示本次寻优算法可能收敛于全局最优解，信息素浓度最高的路径表示的服务组合即为最优服务组合；如果不相同，表示算法收敛于局部最优解，信息素浓度最高的路径应与 $L$ 中的路径进行比较后得到最优路径. 按照 $L$ 中路径由大到小的排列顺序寻找当前可用的、服务 QoS 最高的组合服务. 这里设定最优路径 $L$ 为一个列表，存放多于一条路径的原因是考虑到服务可能存在不可用，QoS 变化等情况.

以上算法的改进规则 2，能保证只要有一只蚂蚁在一次寻优过程中选择了全局最优路径，就可以使算法找到全局最优解，但如果所有的蚂蚁均未选择最优路径，这种情况下以上算法就无法寻找到最优解. 为减少这种情况发生，可在适当的运行时刻判断算法是否正在向局部最优解收敛，如果是则清除局部最优解路径上的信息素值以改变蚂蚁的选路概率，使算法以更大的概率搜索其它路径来寻找最优解. 由于篇幅所限，本文着重说明蚁群算法在服务组合中的应用，因此对进一步的算法改进不再做详细论述.

### 4.5　MPDACO 局部优化算法

由定义 3，服务组合图中的一个抽象服务有多个候选服务实例，示例图如图 5.

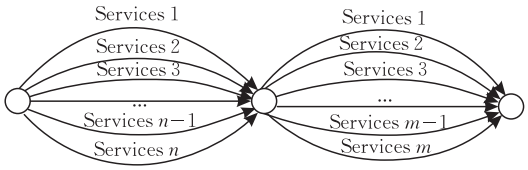


图 5　多个候选服务示例图

由图 5 中看出每两个节点之间有多个候选服务. 设图 1 和图 4 中每两个节点之间的候选服务为 100，则图 1 中存在 $100^3+2\times 100^4$ 条路径，图 2 中存在 $6\times 100^5+5\times 100^6$ 条路径，对这么多条路径进行寻优，将使得寻优问题的规模非常巨大. 而随着问题规模的增大，组合优化问题的规模将成指数级增长.

实践证明，把大规模的优化问题分解使其减小规模，能大大提高寻优的性能. 因而对问题进行转换，通过尽量缩小问题的规模，将大规模的服务选择问题分解成多个子问题进行解决. 对每两个节点之间的多个服务先进行一次服务选择，得到最优的服务，这是局部优化问题. 之后再按照 MPDACO 全局优化算法在全局范围寻找最优组合服务. 下面首先介绍 MPDACO 局部优化算法.

MPDACO 局部优化算法基于蚁群算法原理设计. 由于蚁群算法是一种全局优化算法，因而应用于这里的局部优化问题时对算法做了一些改变.

算法设计思路：MPDACO 局部优化算法只从一个节点开始选择以该节点为初始节点的弧表示的服务(如图 5)，且每只蚂蚁每轮只进行一次服务选择. 这表示优化问题虽然包含的服务数目较多，但规模较小. 另外，还由于服务的动态性、服务 QoS 不明确等因素，对蚁群算法中的状态转移概率选路策略要进行改变：不是先通过计算状态转移概率选择服务，而是先随机选取服务，然后再通过计算状态转移概率来过滤服务，这样可以把 QoS 不满足要求的服务及 QoS 值低的服务过滤掉(对应于信息素浓度低的弧)，以缩小服务选择空间，提高算法后续运行的效率.

**算法 2.** MPDACO 局部优化算法.

1. 设定最大循环次数 $N_{c\max}$,蚂蚁个数 $m$,最优服务列表 $L_{ij}$,下标 $i,j$ 对应表示本次服务选择对应的服务组合图中的抽象服务 $s_{ij}$,初始化时间片 $t=0$,循环控制变量 $N_c=0$,蚂蚁循环变量 $k=0$;

2. 循环次数 $N_c=N_c+1$,如果 $N_c>N_{c\max}$,退出循环,跳转到步 5 结束;

3. 蚂蚁数目 $k=k+1$,如果 $k>m$,退出 $k$ 循环,跳转到步 2;

4. 随机选取服务,获取蚂蚁调用该服务的 QoS 属性值 $Q_1,Q_2,\cdots,Q_n$,与各属性限定值 $q_1,q_2,\cdots,q_n$ 进行比较,如果服务不满足限定条件,则删除该服务,否则计算服务的 QoS 总和,并与最优服务列表中服务对应的信息素进行比较,使得更优的服务保存在最优服务列表 $L_{ij}$ 中,删除其它服务(参见 4.1 节,4.2 节中定义);

5. 对最优服务列表 $L_{ij}$ 中服务对应的信息素按照局部优化算法信息素更新规则进行信息素的更新;

6. 对最优服务列表 $L_{ij}$ 中的服务分别计算状态转移概率,选择最大概率计算结果对应的服务为本次选择的服务. 如果服务列表 $L_{ij}$ 为空,表示该轮服务选择中没有适合要求的服务,表示服务选择失败,应在服务组合图中删除对应的抽象服务.

该算法的复杂度是 $O(N_{c\max}\times m)$,其中 $N_{c\max}$ 为最大循环次数,$m$ 为蚂蚁个数. 该算法的最大复杂度为 $O(n)$,其中 $n$ 为候选服务个数,$O(n)$同时表示穷尽计算方法的算法复杂度. 文中提出的 MPDACO 局部优化算法可以在较少次迭代的情况下即获取较优 QoS 的服务,测试发现性能优于穷尽计算方法.

### 4.6 MPDACO 全局优化算法

在对每类抽象服务进行局部寻优后,进一步在全局内寻找最优的组合服务. 算法如下.

**算法 3.** MPDACO 全局优化算法.

1. 设定最大循环次数 $N_{c\max}$,蚂蚁个数 $m$,初始化时间片 $t=0$,循环控制变量 $N_c=0$,蚂蚁循环变量 $k=0$,禁忌表 $tabu_k(k=1,2,\cdots,m)$,路径表 $path_k(k=1,2,\cdots,m)$,最优组合服务列表 $L$ 等,将蚂蚁放置于服务组合图中的初始节点,令组合图中每条弧对应的各种 QoS 信息素浓度值 $\tau_{ij}(t)$ 为由 MPDACO 局部优化算法所选择服务对应的信息素浓度值,如果没有相应的信息素浓度值时,可令服务对应弧上的信息素 $\tau_{ij}(t)=const$,其中 $const$ 为常数,$const$ 的取值以不影响后续算法运行结果为宜;

2. 循环次数 $N_c=N_c+1$,如果 $N_c>N_{c\max}$,退出循环,跳转到步 10 结束;

3. 蚂蚁数目 $k=k+1$,如果 $k>m$,退出 $k$ 循环,跳转到步 8;

4. 寻找与蚂蚁所在节点相连的路径,根据状态转移概率公式计算转移概率,选择从此节点出发的下一个节点,即选择此节点和下一节点确定的服务;

5. 修改禁忌表 $tabu_k$,将新选择的节点放入禁忌表中;记录蚂蚁调用该服务后获得的各 QoS 属性值 $Q_1,Q_2,\cdots,Q_n$,并转换为相应的信息素进行表示(见第 3 节内容),如果发现该服务的状态为不可用,则在服务组合图中对应的删除这一服务(应用服务删除算法);

6. 如果新选择节点为服务组合图的终点,则此蚂蚁服务选择结束,跳转到步 7,否则跳转到步 4;

7. 计算并记录蚂蚁此轮选择组合服务的新增信息素总和,这里的新增信息素总和是各个子服务新增信息素的加总;如果此轮所有蚂蚁选路结束,则跳转到步 8,否则跳转到步 3;

8. 比较该轮中所有蚂蚁选择的组合服务的新增信息素总和,取新增信息素总和值最大的服务组合序列,对相应路径上的信息素按照信息素更新规则进行信息素的更新,并把该新增信息素总和值最大的组合服务与最优组合服务列表中的组合服进行比较,如果优于列表中最差的组合服务,则将该组合服务插入到最优组合服务列表中,如果最优组合服务列表已满,则在插入该服务前需先删除最差的组合服务;

9. 清空禁忌表 $tabu_k$,路径表 $path_k$ 跳转到步 2;

10. 信息素最大路径表示的组合服务与最优组合服务列表 $L$ 中的组合服务比较,得到最优的组合服务.

## 5 仿真实验及性能分析

在"平谷旅游服务系统"中对本文中提出的算法进行了应用仿真,该项目提供了一个旅游信息平台,提供的服务包括平谷地区旅游服务信息介绍,宾馆、旅店等服务的预定以及其他信息查询类业务等,提供的网络能力包括 GIS、SIP、SMS、MMS 等. 多种网络能力方便了用户与平台的连接. 另外,服务系统还具有服务推荐功能,可以根据用户的要求自动组合服务推荐给用户. 示例图如图 6、图 7 所示.


图 6 服务推荐

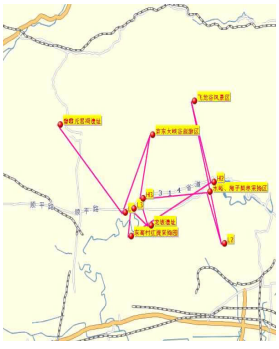

图 7 组合服务推荐

旅游服务推荐系统的建立需要有大量丰富的数据,需要大量的人力物力的投入,在运行的过程中也需要不断实时地更新数据对其进行优化,以向用户提供最优的服务推荐. 这在现实的运行过程中,会消耗太多的人力物力,不具可行性. 我们充分利用游客对所消费服务的 QoS 评价,用本文提出的优化算法

来优化服务推荐系统. 服务推荐系统建立之初有较少量的数据，推荐给用户的服务可能不是最好的，但随着游客的逐渐增多，数据会越来越丰富，推荐的服务也会越来越好，即系统具有学习功能，是一个逐步成长的系统.

设一次服务组合推荐的相关信息如下：推荐景点有京东大峡谷、金海湖、老象峰；推荐交通工具有汽车、火车、自驾车；推荐住宿有农家乐、各种宾馆等；推荐饭店有农家乐、各种餐厅等. 另外还推荐加油站、医院等. 设某组合服务图如图 8 所示.

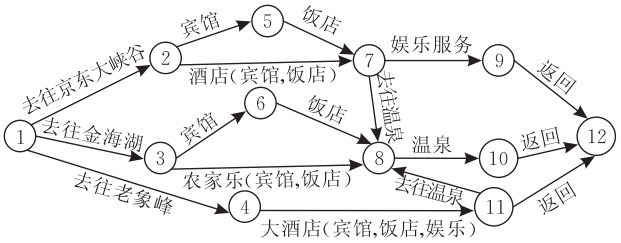


图 8　旅游服务组合图

实际生活中宾馆、饭店、娱乐服务等可能是并行发生的服务，根据文中定义 1 对这些并行服务进行了串行化处理. 图 8 中的服务同时对应相应的抽象 Web 服务，每个抽象 Web 服务同时又对应多个具体的候选服务.

下面的实验通过 MPDACO 局部优化算法和 MPDACO 全局优化算法相结合进行服务组合优化. 首先对每个抽象服务对应的多个候选服务用 MPDACO 局部优化算法寻找最优解，之后再应用 MPDACO 全局优化算法进行服务组合的全局优化，其中 5.2 节中的基本蚁群算法实验是为了与 MPDACO 算法进行实验结果的对比.

### 5.1　MPDACO 局部优化算法实验

图 8 中的每项服务都对应多个服务实例，首先通过 MDPACO 局部优化算法从多个服务实例中寻找最优的服务，为方便描述问题起见，我们设每个服务对应 1000 和10 000两种情况. 实验环境及实验数据介绍如下：

硬件环境：CPU 为 Intel Pentium IV 2.8 GHz，内存为 512 MB，操作系统为 Windows XP. 算法实现工具为 Visual C++6.0.

实验数据：采用随机生成的模拟 Web 服务及服务的参数作为测试用例. 生成两个测试数据集包含的服务总数分别为 1000，10 000，每次的测试包含 10 个不同测试集，通过对 10 个测试集测试结果的加权平均来评估算法参数，减小误差. 实验分为蚂蚁数为 5 和 10 两种情况进行测试.

由 MPDACO 局部优化算法性能测试图发现应用该算法在较少迭代次数的情况下就可以取得较高的满意度，比穷尽计算方法有更好的性能. 测试分为蚂蚁数为 5 和 10 两种情况，从图中看到蚂蚁数为 10 的性能曲线高于蚂蚁数为 5 的性能曲线，这是因为蚂蚁数多时一次搜索的路径多，图 9 中虚线上第 200 次迭代对应的满意度值有较大跳跃，是因为随机选择了较好的服务，因而性能有明显提高.

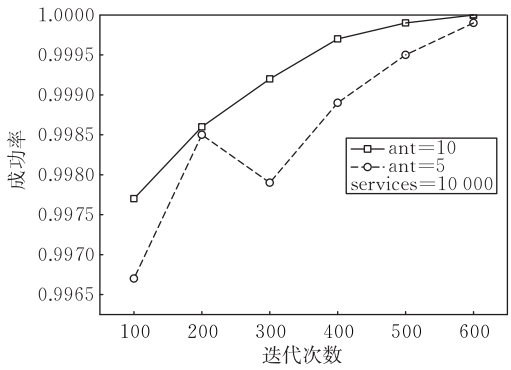


图 9　MPDACO 局部优化算法性能测试图(服务数=10 000)

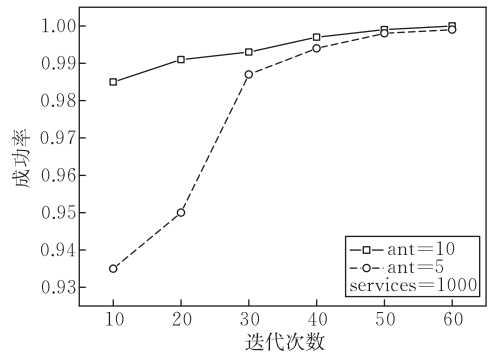


图 10　MPDACO 局部优化算法性能测试图(服务数=1000)

通过图 10 中的结果进行比较，图 10 中达到最大满意度的迭代次数是 60，按蚂蚁数为 10 的曲线进行计算，计算次数为 600，按蚂蚁数为 5 的曲线计算，计算次数为 300，而传统算法的计算次数为 1000.

### 5.2　基本蚁群算法实验

通过 5.1 节的 MDPACO 局部优化算法对服务寻优之后，产生的各最优服务 QoS 参数如图 11 所示.

图 11 与图 8 对应，弧上的数字表示服务的各 QoS 属性值，每个括号中的最后一项都表示服务价格. 下面以服务的价格属性为例应用基本蚁群算法进行服务组合优化实验.

图 11 中节点数为 12，服务数为 17，另外对节点数为 40，服务数为 100 的情况也进行了测试，图未

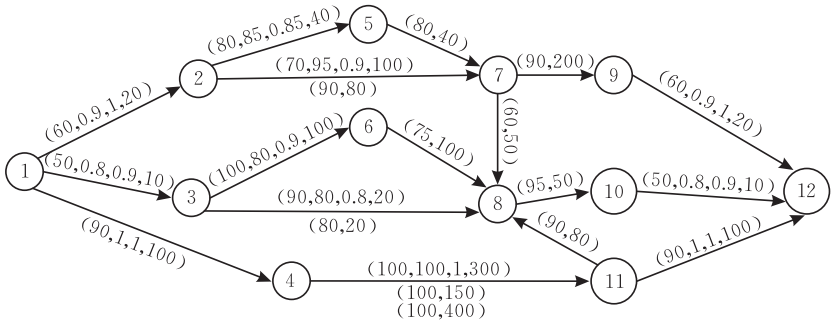


图 11 MDPACO 局部优化算法寻优产生的组合服务 QoS 数值

列出,设定参数 $\alpha=2,\beta=2$,每轮测试 10 次,取平均数,结果如图 12、图 13 中实线.

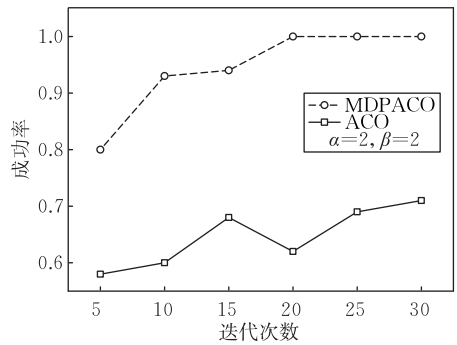


图 12 MPDACO 全局优化算法性能测试图(服务数=17)

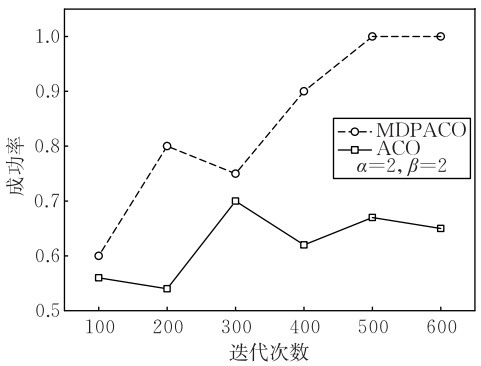


图 13 MPDACO 全局优化算法性能测试图(服务数=100)

由图中曲线看出基本蚁群算法的成功率较低,且实验结果不稳定,随着迭代次数的增加成功率曲线没有呈明显的上升趋势,这是因为算法较早的收敛于局部最优解使算法停滞的缘故.

### 5.3 MDPACO 全局优化算法实验

下面对 MDPACO 全局优化算法进行组合优化实验.由于需要和基本蚁群算法对价格属性优化的结果进行比较,因此在该节的实验中以价格属性进行组合服务优化,而以其它各 QoS 作为服务过滤条件.本文将在 5.4 节中应用 QoS 总和进行服务组合优化,并把结果与一种遗传算法的结果进行比较.

如图 8,设某一需求对各服务 QoS 的限制值如下.

交通服务:(50,0.8,0.8,20);

宾馆服务:(80,80,0.8,40);

饮食服务:(80,40);

娱乐服务:(90,100);

括号中的最后一项都表示服务价格.

下面应用 MPDACO 全局优化算法对以上服务组合进行优化,由于服务的不稳定性,在组合优化的过程中节点(3,4)之间新增加"划船"服务,QoS 属性的属性值为(0.9,40),另外还有一些服务的 QoS 值发生了变化,服务(2,7)宾馆服务的 QoS 变化为(80,95,0.9,100),服务(5,7)变为(75,40).与 5.2 的测试对应,分为节点数为 12,服务数为 17,节点数为 40,服务数为 100,两种情况进行测试,算法参数 $\alpha=2,\beta=2$,每轮测试 10 次,取平均数.性能测试图如图 12、图 13 所示.

由性能曲线图发现 MPDACO 全局优化算法具有较好的性能.另外,随着问题规模的增大,从图中看到为得到最优解所需迭代的次数急剧上升.

MPDACO 算法性能优于 ACO 的原因是 MPDACO 设置了各类抽象服务和组合服务的最优服务列表,能根据历史信息选择目前发现的最优服务,避免了 ACO 中寻优过程的盲目性.从理论角度来分析,这样的设置能使得:如果算法运行时有一只蚂蚁一次发现最优解,则能保证算法最终得到最优解.

### 5.4 与其他启发式优化算法的性能比较

张成文等人[8]将遗传算法应用于服务选择,通过 QoS 综合计算结果查找满足条件的服务.采用随机方法生成每个任务所对应的候选服务,范围从 15～50,每个服务拥有 4 个 QoS 属性值,具体属性值也是分别在一定的取值范围内随机生成[8].为描述方便起见,令张成文等人提出的算法为 ZGA 算法,本文按照 ZGA 算法的测试环境进行测试.

由于 ZGA 算法中仅对适应度值给出了性能比较,没有指出适应度值稳定时得到的是否是最优解.

这里假定文章得到的是最优解. 设任务数为 10 时 ZGA 算法中的测试图如图 14(图 14 摘自 ZGA 算法).

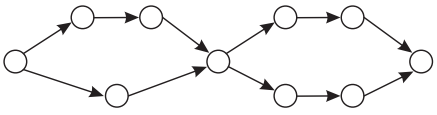

图 14　ZGA 算法中的任务关系图

由于 ZGA 算法用节点表示服务，而在本文中应用边表示服务，因此对服务关系图进行转换. 以下用本文的 MDPACO 算法对服务选择进行性能测试.

应用 MPDACO 局部优化算法首先对每个抽象服务的候选服务进行寻优，设所有候选服务的服务数均为 50，按照最多计算次数 50 次得到最优服务. 由于遗传算法一轮迭代是对所有任务的备选服务之一执行交叉、变异等操作一次，一轮迭代的算法复杂度为 $O$(抽象服务数)，因此 MPDACO 局部优化算法的执行相当于 ZGA 算法中(图 15、图 16)遗传算法迭代 50 次. 图 15、图 16 是 ZGA 算法在种群数为 400、交叉率为 0.7、变异率为 0.1 的适应度比较结果. 位于上面的线条都是矩阵编码遗传的仿真结果. 位于下面的线条都是一维编码遗传算法的仿真结果[8].

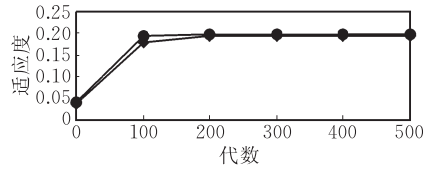


图 15　ZGA 算法性能测试图(服务数＝10)

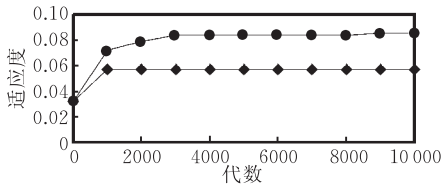


图 16　ZGA 算法性能测试图(服务数＝30)

下面用 MPDACO 全局优化算法获取图 14 中的最佳服务组合序列. 性能测试图如图 17、图 18 所示.

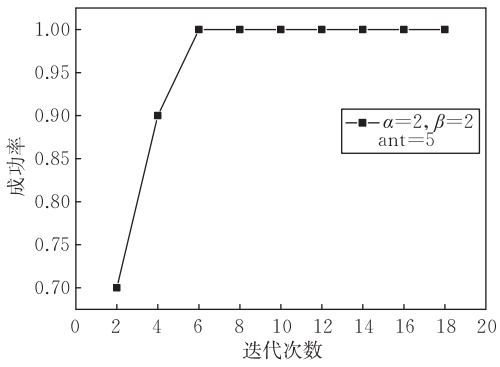


图 17　MPDACO 全局优化算法性能测试图(服务数＝10)

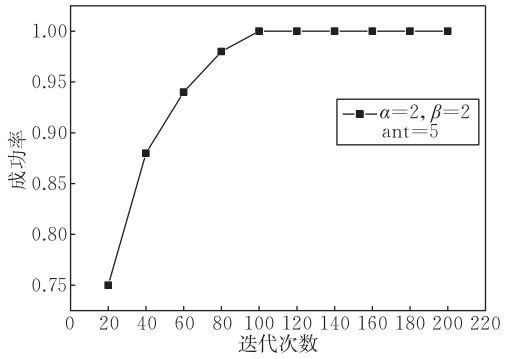


图 18　MPDACO 全局优化算法性能测试图(服务数＝30)

从图 17 看到，在 $\alpha=2$，$\beta=2$，蚂蚁数目为 5 的情况下，在迭代数为 6 时算法收敛于最优解，MPDACO 算法中每只蚂蚁每轮的运算次数≤任务数，这里按任务数计算，由于蚂蚁数目为 5，相当于遗传算法中的 $5\times6=30$ 次迭代. 从图 15 中看到，算法在第 100 次迭代时达到适应度稳定，30(全局优化算法)＋50(局部优化算法)＝80＜100，性能高于 ZGA 算法的性能. 另外，对于服务数为 30 的情况，也进行了测试，参见图 18，能看出性能也明显优于 ZGA 算法.

## 6　结论及进一步工作

本文研究了蚁群算法在服务组合优化系统中的应用，在基本蚁群算法的基础上提出了 MDPACO 算法，该算法能适应服务组合优化过程中 Web 服务的动态性、不稳定性以及多种 QoS 属性限制等问题. MDPACO 算法包含两个子算法：一个是 MDPACO 全局优化算法；另一个是 MDPACO 局部优化算法，该 MDPACO 局部优化算法用于单一功能的多服务选择时具有良好性能，较少的迭代次数下即可以获得较优解. 文章最后在一个旅游领域的服务推荐系统中对算法的有效性进行了验证，性能测试结果表明 MDPACO 较基本蚁群算法以及张成文等应用于服务选择的遗传算法有更高的性能.

下一步我们将对蚁群算法的收敛问题给予研究，并进一步寻找制约蚁群算法收敛速度的深层原因以及算法对数据敏感性的规律. 另外，还要将蚁群算法与语义网理论相联系进行研究，研究基于语义的蚁群算法，使优化算法更具智能化.

## 参　考　文　献

[1] Dorigo M, Maniezzo V, Colorni A. Ant system: optimization by a colony cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics- Part B: Cybernetics, 1996,

26(1): 29-41

[2] Dorigo M, Gambardella L M. Ant colony system: A cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation, 1997, 1(1): 53-66

[3] Dorigo M, Caro G D, Gambardella L M. Ant algorithms for discrete optimization. Artificial Life, 1999, 5(2): 137-172

[4] Guo Su-Chang, Huang Hong-Zhong, WANG Zhong-Lai, Xie Min. Grid service reliability modeling and optimal task scheduling considering fault recovery. IEEE Transactions on Reliability, 2011, 60(1): 263-274

[5] Jiang Hui-Yan, Zong Mao, Liu Xiang-Ying. Research of software defect prediction model based on ACO-SVM. Chinese Journal of Computers, 2011, 34(6): 1148-1154(in Chinese)
(姜慧研, 宗茂, 刘相莹. 基于 ACO-SVM 的软件缺陷预测模型的研究. 计算机学报, 2011, 34(6): 1148-1154)

[6] Stutzle T, Hoos H. MAX-MIN ant system and local search for the traveling salesman problem//Proceedings of the 4th IEEE International Conference on Evolutionary Computation. 1997: 309-314

[7] Huang Han, Hao Zhi-Feng, Wu Chun-Guo, Qin Yong. The convergence speed of ant colony optimization. Chinese Journal of Computers, 2007, 30(8): 1344-1353(in Chinese)
(黄翰, 郝志峰, 吴春国, 秦勇. 蚁群算法的收敛速度分析. 计算机学报, 2007, 30(8): 1344-1353)

[8] Zhang Cheng-Wen, Su Sen, Chen Jun-Liang. Genetic algorithm on Web services selection supporting QoS. Chinese Journal of Computers, 2006, 29(7): 1029-1037(in Chinese)
(张成文, 苏森, 陈俊亮. 基于遗传算法的 QoS 感知的 Web 服务选择. 计算机学报, 2006, 29(7): 1029-1037)

[9] Zeng Liang-Zhao, Boualem Benatallah. QoS-aware middleware for Web services composition. IEEE Transactions on Software Engineering, 2004, 30(5): 311-327

[10] Ni Wan-Cheng, Liu Lian-Chen, Wu Cheng, Liu Wei. Conceptual correlation based method for grid service composition. Journal of Tsinghua University, 2007, 47(10): 1581-1585(in Chinese)
(倪晚成, 刘连臣, 吴澄, 刘伟. 基于概念关联程度的网格服务组合方法. 清华大学学报(自然科学版), 2007, 47(10): 1581-1585)

[11] Li Man, Wang Da-Zhi, Du Xiao-Yong, Wang Shan. Dynamic composition of Web services based on domain ontology. Chinese Journal of Computers, 2005, 28(4): 644-650(in Chinese)
(李曼, 王大治, 杜小勇, 王珊. 基于领域本体的 Web 服务动态组合. 计算机学报, 2005, 28(4): 644-650)

[12] Timm J T E, Gannod G C. Specifying semantic Web service compositions using UML and OCL//Proceedings of the 16th International Conference on Web Services. Salt Lake City, Utah, USA, 2007: 521-528

[13] Lin Lin, Ping Lin. Orchestration in Web services and real-time communications. Web Services in Telecommunications, IEEE Communications Magazine, 2007, 45(7): 44-50

[14] Xia Ya-Mei, Meng Xiang-Wu, Chen Jun-Liang, Liu Dong. A denotation and application of preference ontology for service composition. Journal of Beijing University of Posts & Telecommunications, 2008, 31(4): 33-36(in Chinese)
(夏亚梅, 孟祥武, 陈俊亮, 刘栋. 一种面向服务组合的偏好本体表示及应用研究. 北京邮电大学学报, 2008, 31(4): 33-36)

[15] Steffen Lamparter, Anupriya Ankolekar, Rudi Studer, Stephan Grimm. Preference based selection of highly configurable web services//Proceedings of the 16th International Conference on World Wide Web. New York, USA, 2007: 1013-1022

**XIA Ya-Mei**, born in 1976, Ph. D., lecturer. Her main research interests include dynamic services composition, optimality theory, cloud computing, etc.

**CHENG Bo**, born in 1975, Ph. D., associate professor. His main research interests include services composition, the internet of things, etc.

**CHEN Jun-Liang**, born in 1933, professor, Ph. D. supervisor, member of Chinese Academy of Sciences, member of Chinese Academy of Engineering. His main research focuses on semantic Web.

**MENG Xiang-Wu**, born in 1966, Ph. D., professor, Ph. D. supervisor. His main research interests include communications software, network service.

**LIU Dong**, born in 1981, Ph. D., lecturer. His main research interests include semantic Web service, context awareness.

**Background**

This paper mainly analyzes one aspect in dynamic services composition in the field of semantic web. Dynamic services composition, a focus in semantic web research, has attracted the attention of many scholars around the world. Although fruitful results have been achieved, there are still some problems to be solved. As an important research power, State Key Laboratory of Networking and Switching Technology of Beijing University of Posts & Telecommunications

has a 10-year research experience in this field, and published many influential papers. It also provides this paper with "fertile soil".

This paper is supported by National Basic Research Program of China under Grant No. 2011CB302704 and the National Natural Science Foundation of China under Grant No. 61001118. In the research of "The Providing Mechanism and Methods of the Internet of Things Service"—the sub project of 973 (2011CB302704), the performance of service selection is an emergent problem to solve. With some inherent features, the ant colony algorithm is fit for solving this problem. This paper builds a services composition model, and changes services composition graph to a simple one-direction connected graph. For the instability of services state, the change of service state is simulated by dynamically adjusting the connected graph and changing the weight of the edge. In addition, as in many services, the Value of QoS is known only after the services are selected and implemented, the author makes improvement of the ACO, and makes the optimized algorithm being implemented without knowing the value of QoS. As the ACO matures early and tends to stagnate, which makes the optimal solution always a local one, this paper makes improvement of the ACO strategy, and profoundly enhances the convergence of the algorithm, and decreases the probability of converging to local optimal solution. To prove its effectiveness, the algorithm is used in "Pingu District Tourism Information System" to make service recommendation. The simulation experiment result shows that the algorithm has a better performance than ACO and the Genetic Algorithm used in service selection.