文章编号:1671-4598(2020)06-0247-05 DOI:10.16526/j.cnki.11—4762/tp.2020.06.050 中图分类号:U495 文献标识码:A

雾计算平台的任务调度算法研究

黄思宇，何通能
（浙江工业大学 信息工程学院，杭州 310023）

摘要：雾计算平台中的任务调度问题是无法在多项式时间复杂度内求取精确解的NP—问题；通过对雾计算任务调度流程的分析，在构建雾计算平台任务调度数学模型基础上，采用改进人工蜂群算法，将任务调度映射为蜂群寻找蜜源的过程，在种群初始化阶段过引入混沌思想，改善了人工蜂群算法缺陷，扩大了蜂群搜索范围，避免陷入局部最优解；实验结果表明，改进后的人工蜂群算法具有更快的算法收敛速度，算法解所对应的任务调度策略，也具有更高的任务处理总性能，表明改进人工蜂群算法，达到了提高雾计算资源利用率，提高雾计算任务处理效率的目的。

关键词：雾计算；任务调度；人工蜂群算法；混沌思想

Research on Task Scheduling Algorithm of Fog Computing Platform

Huang Siyu，He Tongneng
（Zhejiang University of Technology，Information College，Hangzhou 310023，China）

Abstract：The task scheduling problem in fog computing platforms is an NP—problem that cannot be solved accurately within the polynomial time complexity. On the basis of analyzing the task scheduling process of fog computing platform，the task scheduling mathematical model of fog computing platform is built，and the improved artificial bee swarm algorithm is used to map the task scheduling to the process of searching for honey source by bees. Chaos thought is introduced in the initial stage of population，which improves the defect of artificial bee swarm algorithm，enlarges the search range of bees and avoids trapping into local optimal solution. The experimental results show that the improved artificial bee swarm algorithm has faster convergence speed，corresponding task scheduling strategies，and higher overall task processing performance. It shows that the improved artificial bee swarm algorithm can improve the utilization rate of fog computing resources and improve the efficiency of fog computing task processing.

Keywords：fog computing；task scheduling；artificial bee swarm algorithm；chaos thought

0 引言

雾计算是一个高度虚拟化的计算平台，在传统的云计算数据中心与终端设备间，提供计算、存储和网络服务，以提高物联网计算性能。雾计算利用附件设备完成数据处理，而将数据上传至“遥远”的云端进行处理，可节约远程传输带宽，降低计算延时，缓解云端计算压力。同时，雾计算技术的位置感知能力更强，延迟更低，适用于多节点部署，因此在车联网等物联网和交通指挥系统等互联网系统中广泛应用[1]。

云计算模式中的资源高度集中，导致数据中心远离用户端，容易造成传输拥塞与传输延迟。而雾计算通过将云计算的部分功能转移到网络边缘，在更靠近终端的地方，为用户提供实时交互、移动性支持和外置感知等服务，减少了系统内部数据流量，提高终端的响应能力。但是，由于不同终端、不同任务的资源需求不同，需要雾计算有适当的任务调度策略对海量的雾计算节点资源进行分配与调度。因此，构建合理的资源管理、任务调度体系，可进一步提高雾计算服务质量，在信息技术领域具有重要的理论意义以及现实应用价值。

当前，这对雾计算特点的任务调度算法研究较少。汪成亮利用Rete算法构造推理网络，将推理节点优化分配至各计算节点，提高了节点的资源利用率[2]。汤琳煜基于稳定匹配的思想，解决任务与服务设备的分配问题[3]。韩奎奎提出一种改进的遗传算法，进行任务调度与分配[4]。熊凯采用强化学习算法有效提高异构车联雾架构下的资源优化[1]。从雾计算任务调度的研究现状来看，其研究较少，缺少一种较广泛应用的雾计算任务调度算法。为此，本文提出一种基于改进蜂群算法，为雾计算的任务调度提供一种新的思路[5]。

1 雾计算概述

雾计算是由Cisco公司所提出的一种分布式计算结构，自2011年被提出以来，这种技术被广泛地应用于智能服务机器人、智能家居、智能电网、智能交通等领域。雾计算的整体架构如图1所示。

如图1所示，雾计算利用雾集群中的边缘闲散计算资源，为终端用户提供计算服务。无集群的整体结构设计如图2所示。

其中，雾中心节点负责整个雾集群节点的任务内编排与控制，雾服务节点负责提供Docker容器和雾计算能力，

收稿日期:2020-04-14； 修回日期:2020-04-27。

作者简介:黄思宇(1994-)，男，河北霸州人，硕士生，主要从事模式识别与智能控制方向的研究。

通讯作者:何通能(1962-)，男，浙江义乌人，副教授，主要从事模式识别与智能控制方向的研究。

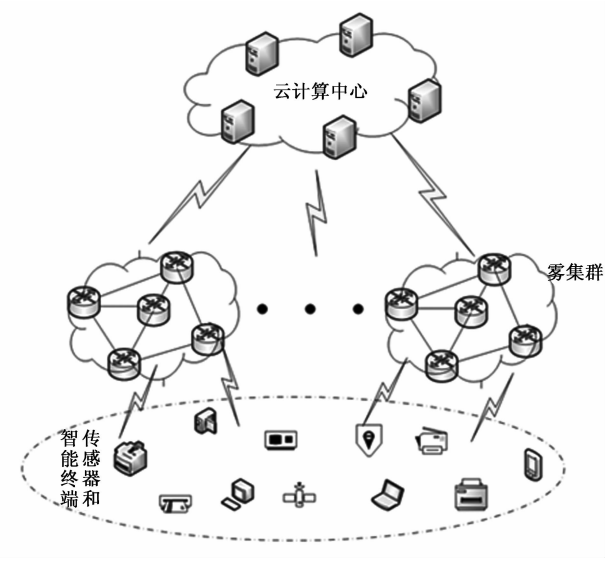

图 1 雾计算平台整体结构

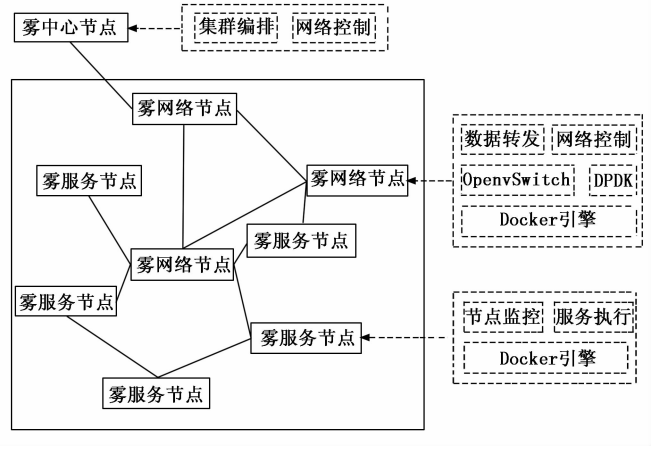

图 2 雾集群结构

雾网络节点负责网络拓扑发现与数据转发。雾服务节点均基于 Docker 引擎，运行不同数量的容器，而这些容器构成了容器集群，提供雾计算服务。

雾计算体系内的调度问题，其实质为任务请求过程中的重叠问题。在雾计算为用户请求任务分配计算资源时，每个任务的处理必然会持续一段时间，而任务请求时间及其后的持续时间段，会导致任务处理时间重叠。而雾计算任务的角度的目的为：对于处理时间上有重叠的任务集合，制定合理的资源、任务调度方案，以保证所有任务处理的总体时间最短。

2 雾计算任务调度模型

2.1 任务调度流程

雾计算任务调度过程如图 3 所示，由雾中心节点在调度任务与计算资源的约束下，通过调度算法，选择合适的雾服务节点，在雾服务节点的 Pod 容器内进行任务计算。

调度服务不断向雾中心节点查询待调度的 Pod 列表，以及当前雾节点集群中的可用雾服务节点 Node 列表，根据

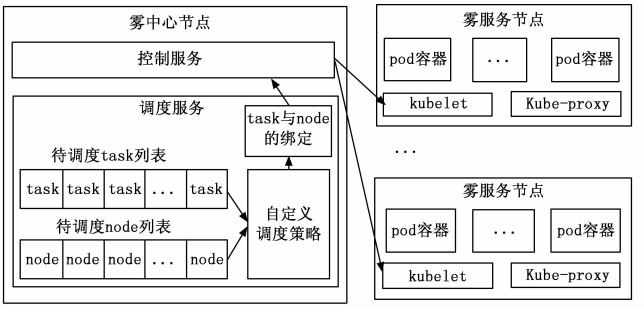

图 3 任务调度过程

调度策略，优选雾服务节点资源进行任务处理[6]。

默认的调度流程为以下两个步骤：

1）根据雾中心节点与雾服务节点的通信，了解雾服务节点的可用状态，构建可用服务节点 Node 列表；

2）按照改进人工蜂群算法对可用的雾服务节点进行评价，选择最合适的雾服务节点，完成雾服务节点的调度。

2.2 任务调度数学模型

雾计算任务调度模型的描述可使用五元组 $G=(V, P, E, W, C)$ 来描述。

其中，$V=\{v_i \mid 1\leqslant i\leqslant n\}$ 表示待处理的任务集合；$P=\{p_i \mid 1\leqslant i\leqslant m\}$ 为可用的雾服务节点；$E=\{e_{ii} \mid 1\leqslant i\leqslant n, 1\leqslant j\leqslant m\}$ 为 DAG 图中的有向边集合，表示任务 v_j 是否需要任务 v_i 的处理结果；$\boldsymbol{W}$ 为式（1）所示的一个 $n\times m$ 的矩阵。

$$\boldsymbol{W}=[w_{ij}]=\begin{bmatrix} w_{1,1} & w_{1,1} & \cdots & w_{1,m} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ w_{n,1} & w_{n,2} & \cdots & w_{n,m} \end{bmatrix} \tag{1}$$

w_{ij} 表示采用节点 p_j 处理任务 v_i 时的系统开销。$\boldsymbol{C}$ 为如式（2）所示的一个 $m\times m$ 的矩阵。

$$\boldsymbol{C}=[c_{ij}]=\begin{bmatrix} c_{1,1} & c_{1,1} & \cdots & c_{1,m} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m,1} & c_{m,2} & \cdots & c_{m,m} \end{bmatrix} \tag{2}$$

c_{ij} 表示，若 $e_{ij}\neq 0$ 时，v_i 完成后，到 v_j 开始处理前的通信开销。若 v_i 和 v_j 在同一个节点上完成，或 $e_{ij}=0$ 时，$c_{ij}=0$。

综上，雾计算的任务调度过程看作是如图 4 所示的一个有向无环图。

在雾服务节点集合 $P=\{p_1, p_2, \cdots, p_i, \cdots, p_m\}$ 上，执行任务集合 $V=\{v_i \mid 1\leqslant i\leqslant n\}$ 的过程，是一个在任务模型 DAG 图上，具有时间有限约束的调度函数 f，函数 f 以特定的开始时间，将任务映射到处理单元上，一个任务的调度可描述如式（3）所示：

$$f: T\rightarrow\{1,2,\cdots,m\}\times[0, \infty) \tag{3}$$

若，有 $v\in T$，且 $f(v)=(i, t)$，表示任务 v 在时间 t，被分配到雾服务节点 p_i 上执行。

任务调度函数 f，可通过 Gantt 图（甘特图）来表示。

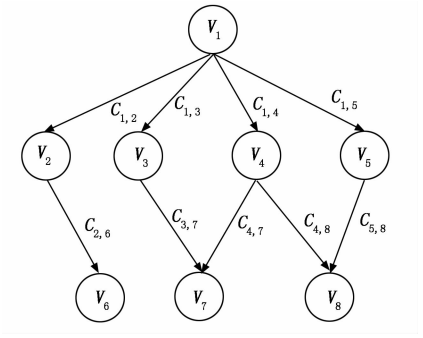

图 4　任务调度过程

在 Gantt 中，存储了一个处理单元表，对于每一个处理单元，每个任务都有处理单元与之相对，可直观地表示任务的开始时间与结束时间，按照分配到该雾服务节点上的所有任务，按照先后执行的顺序进行排列。

Gantt 图可使用如式（4）所示的四元表来表示：

$$Gant(p_i, v_j, ST(v_i, p_i), FT(v_j, p_i)) \tag{4}$$

其中：$ST(v_j, p_i)$ 表示在任务 v_j，在雾服务节点 p_i 上的开始执行时间，$FT(v_j, p_i)$ 表示在任务 v_j，在雾服务节点 p_i 上的结束执行时间。

在任务调度系统中，所有雾节点上的最大调度长度 SL 定义为：

$$SL = \max_{i}\left\{\sum_{j} FT(v_j, P)\right\} \tag{5}$$

而雾计算任务调度策略的目标即为：最大调度长度 SL 的最小化。

3　雾计算任务调度算法设计与优化

3.1　算法思路

在以性能优先的雾计算平台中，其调度的目标就是在雾计算平台的资源和待处理任务的约束下，使得如图 4 所示的有向无环图的遍历时间最短。已有的研究成果表明，DAG 图的遍历问题是一个 NP－完全问题，在多项式时间复杂度内，无法找到问题精确解。对此本文将采用人工蜂群（artifical bee colony，ABC）算法，将任务分配给雾服务节点的过程，模拟为派遣蜜蜂前往蜜源采蜜的过程，以求解雾计算任务调度数学问题，在较短时间内，找到较优的雾计算任务调度策略，以提高任务的调度和处理效率[7]。

ABC 模拟蜂蜜的觅食采蜜机制，将觅食的蜂群划分为侦查蜂、观察蜂、雇佣蜂 3 个类型。雇佣蜂的数量与环境中食物点（蜜源）数量相同，雇佣蜂在环境区域内随机搜索蜜源，存储所找到的蜜源信息；雇佣蜂在完成区域搜索后，与观察蜂分享区域内的蜜源地址信息；观察蜂在蜂巢内，接收侦查蜂传递的信息，确定环境内蜜源。若蜜源长时间内没有进化，则暂时放弃该蜜源，随机搜索新的、更具价值的蜜源，以取代没有进化的蜜源[8]。

ABC 通过模拟蜂蜜的觅食采蜜机制，将觅食的蜂群划分为侦查蜂、观察蜂、雇佣蜂 3 个类型，其具体步骤如下所示。

1）初始化阶段：初始化蜜源数量 N，观察蜂数量和雇佣蜂数量 N，蜜源最大开采次数 $Limit$，最大迭代次数 Maxcycle。在初始阶段，侦查蜂随机搜索 N 个蜜源，采用 N 个 D 维向量表示搜索所得的 N 个蜜源，蜜源的产生公式如式（5）所示：

$$x_{ij} = x_{\min j} + \alpha(x_{\max j} - x_{\min j}) \tag{5}$$

其中：x_{ij} 表示蜜源 $i \in \{1,2,\cdots,N\}$ 的第 $j \in \{1,2,\cdots,D\}$ 个维度，α 为 (0,1) 间的随机数，$x_{\min j}$ 和 $x_{\max j}$ 分别表示第 j 个维度空间内的最小值和最大值。

2）雇佣蜂阶段：当雇佣蜂在区域内搜集到新的蜜源后，根据贪婪规则更新区域内的蜜源，若新蜜源的适应度，较原蜜源的适应度更高，则将新蜜源替代原有蜜源，其更新的公式如式（6）所示：

$$v_{ij} = x_{ij} + \varphi_{ij}(x_{ik} - x_{kj}) \tag{6}$$

其中：x_{ij} 为原有的蜜源位置，v_{ij} 为雇佣蜂所搜索到的新的蜜源位置，φ_{ij} 为一个 [0,1] 间的随机数，x_{kj} 为不同于当前蜜源的另一蜜源，$k \neq i$，且 $k \in [1,N]$。

3）观察蜂阶段：侦查蜂发现新蜜源后，通过适应度函数计算蜜源的适应度值，以衡量蜜源的质量，蜜源适应度的计算公式如式（7）所示：

$$fit(x_i) = \begin{cases} \dfrac{1}{1+f_i}, & f_i \geqslant 0 \\ 1 + abs(f_i), & f_i < 0 \end{cases} \tag{7}$$

其中：f_i 为蜜源 x_i 的适应度值观察蜂通过雇佣蜂的所传递的信息，用于判断新蜜源位置，与新蜜源质量。新蜜源质量越高，前往采蜜的观察蜂越多，并采用轮盘赌概率，派遣前驱采蜜的观察蜂，观察蜂选择新蜜源的概率公式如式（8）所示：

$$p_i = \frac{fit_i}{\sum_{i=1}^{N} fit_i} \tag{8}$$

4）侦查蜂阶段：在 $Limit$ 次循环后，若蜜源仍未被更新，则选择放弃该蜜源。此时，雇佣蜂就会变为侦查蜂，重回侦查蜂阶段，寻找新的蜜源。

最终，在寻找到适应度值满足预设要求，或者当迭代次数超出了预设的最大迭代次数后，将适应度值最高的蜜源作为算法结果，并退出人工蜂群算法的迭代。

3.2　人工蜂群算法改进与优化

虽然 ABC 算法有较好的收敛性。但对初始值质量的要求较高，若初始方案的质量较低，将影响 ABC 算法的收敛精度与收敛速度等问题。但在雾计算的任务调度过程中，初始的任务分配方案是随机的，可能影响 ABC 算法效果。

针对该问题，本文主要通过在初始阶段引入正弦映射，以提高初始值质量，和在迭代过程中，引入混沌思想，以提高搜索范围的方式，进行 ABC 算法的优化[9]。

混沌具有非线性动力系统的固有特性，混沌空间中的变化过程看似是杂乱无章，但在这个看似杂乱无章的混乱过程中，又体现了精致的内在规律性，其数学描述如下所示：

$$x_{k+1} = \tau(x_k) \tag{9}$$

其中：$k=0,1,2,\cdots,\tau$ 为一个非线性的映射，$0<x_k<1$，从 x_0 初始值开始，通过反复的 τ 非线性映射，得到元素值呈混沌分布的序列 $\{x_k\}$。

如式（10）所示的正弦非线性映射，通过正弦变化，可得到所需的混沌序列。

$$x_{n+1}=a\sin(bx_n) \tag{10}$$

其中：a 和 b 为正弦非线性映射模型的相关参数。

在人工蜂群算法的初始化阶段，采用正弦非线性映射，引入到人工蜂群的初始化过程。基于正弦变化的人工蜂群的种群变量初始化如式（11）所示：

$$Ch_{k+1}=\sin(\pi Ch_k) \tag{11}$$

其中：k 为迭代计数器，$k=0,1,2,\cdots,m$，m 为最大混沌迭代次数，$Ch_k\in(0,1)$。将式（11）引入到人工蜂群的初始化过程，采用混沌变量后的种群变量转换公式如式（12）所示：

$$x_{ij}=x_{\min j}+Ch_{kj}(x_{\max j}-x_{\min j}) \tag{12}$$

其中：$x_{\min j}$ 和 $x_{\max j}$ 分别表示第 j 个维度空间内的最小值和最大值。

3.3 资源调度算法流程

基于改进人工蜂群算法，快速实现任务与雾计算资源间的匹配，实现雾计算任务的快速调度。为了让改进人工蜂群算法适用于雾计算资源的调度分配，还需要算法中的解和适应度函数进行重构[10]。

1）解的重新构造：假设雾计算层的可用雾服务节点数为 m，待任务数为 n。任务调度的目的就是将 n 个计算任务分配到 m 个雾服务节点上进行处理。因此，蜂群解描述如式（13）所示：

$$X_i=(x_{i1},x_{i2},\cdots,x_{in}),i=1,2,\cdots,m \tag{13}$$

例如，将任务（t_0,t_1,t_2,t_3）分配到雾计算虚拟资源（p_0，p_1）上，最终的解表述为 $X_0=(1,0,0,1),X_1=(0,1,1,0)$，即将任务 t_0 和 t_3 分配到雾服务节点 p_0 上进行处理，在雾服务节点 p_1 上计算资源 v_1 上计算任务 t_1 和 t_2。

2）适应度函数的计算：对蜜源质量进行评价的适应度函数设计如式（13）所示：

$$fit=\frac{1}{SL} \tag{13}$$

即完成所有任务的时间越短，该调度策略的适应度函数值越大，以确保找到最优的任务调度策略[11]。

基于改进蜂群算法的雾计算资源优化调度策略流程如图 5 所示。

1）蜂群规模初始化，采用混沌映射初始化人工蜂群的种群，产生初始解，根据将 n 个计算任务分配到 m 个虚拟雾服务中心上，构成人工蜂群算法的初始解，并设置人工蜂群算法的最大迭代次数 $Maxcycle$ 等初始化参数。

2）雇佣蜂按照贪婪规则，搜寻区域内的新蜜源位置，基于适应度函数计算新蜜源适应度，通过新蜜源适应度值与当前解适应度值的对比，选择适应度更优的蜜源位置，获得更优的雾计算资源分配方案。

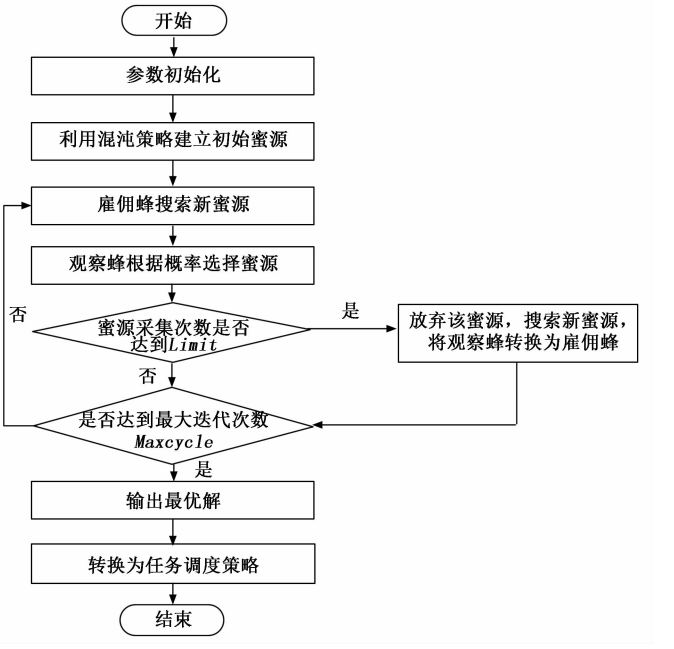

图 5 基于改进人工蜂群的任务调度

3）在迭代过程中引入混沌思想，让适应度函数越高的蜜源，能招募到更多的观察蜂，通过新蜜源的不断更新，保留条件最优蜜源。

4）若某蜜源超过了预设临界值 $Limit$，则将观察蜂转换为侦查蜂，产生新的蜜源，替代被放弃的蜜源。

5）当人工蜂群的迭代次数超过了最大迭代次数 $Maxcycle$ 时，算法结束，将适应度最高的蜜源，作为雾计算任务调度策略；否则继续执行第 2）步。

4 基于改进人工蜂群的资源调度策略

在基于改进人工蜂群算法的雾计算平台任务调度过程如图 6 所示。

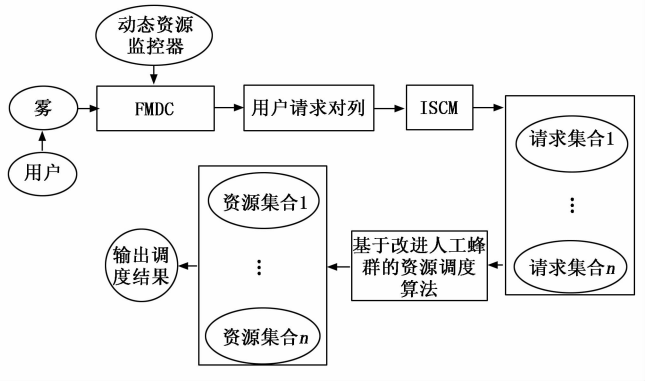

图 6 基于改进算法的资源调度过程

雾计算平台终端通过雾计算将服务提交到 FMDC 雾中心节点，设置资源监控器，监测当雾层的资源使用状况，并将资源信息存储到雾中心节点，将待处理任务进行集中管理，通改进人工蜂群的资源调度算法，制定任务调度策

略，最终完成待处理任务的计算。

基于改进人工蜂群的任务调度过程具体实现如下所示：

算法：改进人工蜂群的任务调度

输入：人工蜂群初始化参数(N，D，Maxcycle)

输出：资源调度方案

Step 1. 在初始化阶段引入正弦映射

Step 2. for $i=1$：N

Step 3. for $j=1$：D

Step 4. ch(1，j) = rand

Step 5. for $k=1$：$c_{\max}$

Step 6. ch($k+1$,j) = sin($pi \times ch(k, j)$)

Step 7. end

Step 8. $x_{ij}=x_{\min j}+ch_{kj}(x_{\max j}-x_{\min j})$

Step 9. 计算适应度值

Step 10. end

Step 11. end

5　仿真实验分析

为证明本文所研究的改进人工蜂群算法在任务调度中的有效性，采用 Matlab 工具，对 ABC 人工蜂群算法和改进人工蜂群算法进行对比仿真实验。

仿真实验过程中相关参数设置如下：蜜源停留最大次数 $Limit=100$，蜜源数目 $N=20$，最大迭代次数 $Maxcycle=2\,500$，随机生成 100～700 个不同的任务，对比分析不同任务数量下，两种不同算法的任务完成时间。为确保算法实验结果的精确性，对分别使用两种算法分别独立运行 20 次，取其平均值作为算法的执行结果[12]。

在不同任务数量下，两种不同算法的任务完成时间对比情况如图 7 所示。

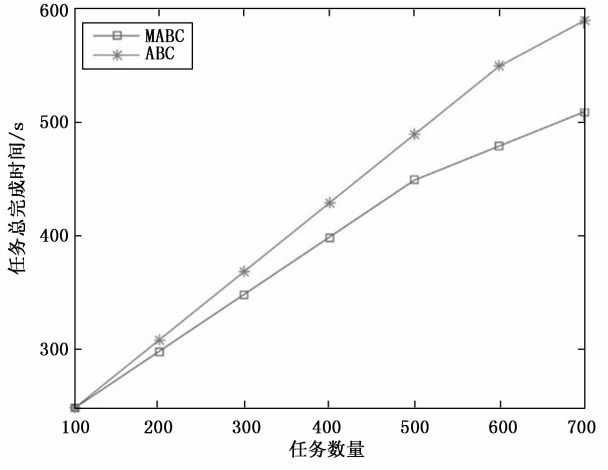

图 7　两种算法的任务完成时间对比

如图 7 所示，当任务量较少时，两种算法的优化效果并未有明显区别，根据不同调度算法完成所有任务的时间也相差不大，但是随着任务数量的不断增长，改进人工该算法完成所有任务的时间明显更优。究其原因，主要是因为随着任务数量的增加，人工蜂群算法容易陷入局部最优解，从而消耗较长时间也难以获得全局最优解，而在改进人工蜂群算法中，通过引入混沌策略，极大地提高了算法的全局寻优能力，提高了任务调度效率，以优化雾计算资源使用更优，和提高雾计算平台的任务处理效率与性能。

6　结束语

雾计算资源的任务调度策略是雾计算技术的研究重点之一，通过任务调度策略的优化，有助于雾计算资源的应用优化，进一步提升雾计算性能。本文所采用的改进人工蜂群算法，不仅能在多项式时间内找到较优解，提高了任务调度性能，同时通过在初始阶段引入混沌思想，提高了初始值质量，扩大了搜索范围，提高了解的质量，优化了任务调度策略，有利于雾计算性能的进一步提升。

参考文献：

[1] 熊　凯，冷甦鹏，张　可，等. 车联雾计算中的异构接入与资源分配算法研究 [J]. 物联网学报，2019，3 (2)：20－27.

[2] 汪成亮，黄心田. 智能环境下基于雾计算的推理节点优化分配研究 [J]. 电子学报，2020，48 (1)：35－43.

[3] 汤琳煜，蒋加伏，谷　科. 基于雾计算的计算资源分配方案 [J]. 计算机工程与应用，2019，55 (19)：96－104.

[4] 韩奎奎，谢在鹏，吕　鑫. 一种基于改进遗传算法的雾计算任务调度策略 [J]. 计算机科学，2018，45 (4)：137－142.

[5] Jamil B，Shojafar M，Ahmed I，et al. A job scheduling algorithm for delay and performance optimization in fog computing [J]. Concurrency and Computation：Practice and Experience，2020，32 (7)：76－83.

[6] Li G S，Liu Y C，Wu J H，et al. Methods of resource scheduling based on optimized fuzzy clustering in fog computing [J]. Sensors (Basel，Switzerland)，2019，19 (9)：132－139.

[7] Lei D M，Liu M Y. An artificial bee colony with division for distributed unrelated parallel machine scheduling with preventive maintenance [J]. Computers &amp；Industrial Engineering，2020，141 (12)：34－46.

[8] Aslan S. Time－based dance scheduling for artificial bee colony algorithm and its variants [J]. International Journal of Computational Intelligence Systems，2019，12 (2)：64－69.

[9] Li Y L，Li F，Pan Q K，et al. An artificial bee colony algorithm for the distributed hybrid flowshop scheduling problem [J]. Procedia Manufacturing，2019，39 (C)：71－75.

[10] 于立婷，谭小波，解　羽. 基于改进人工蜂群优化 K－means 的入侵检测模型 [J]. 沈阳理工大学学报，2019，38 (06)：8－14，27.

[11] 李志敏，张　伟. 基于差分进化人工蜂群算法的云计算资源调度 [J]. 计算机工程与设计，2018，39 (11)：3451－3455.

[12] Peng K K，Pan Q K，Zhang B. An improved artificial bee colony algorithm for steelmaking－refining－continuous casting scheduling problem [J]. Chinese Journal of Chemical Engineering，2018，26 (8)：1727－1735.