李强,刘晓峰,孔德瑾,等.基于烟花算法的云作业调度策略[J].湖南科技大学学报(自然科学版),2023,38(4):88-96. doi:10.13582/j.cnki.1672-9102.2023.04.011

LI Q, LIU X F, KONG D J, et al. Cloud Job Scheduling Strategy Based on Fireworks Algorithms[J]. Journal of Hunan University of Science and Technology (Natural Science Edition),2023,38(4):88-96. doi:10.13582/j.cnki.1672-9102.2023.04.011

基于烟花算法的云作业调度策略

李强[1],刘晓峰[2*],孔德瑾[1],秦华伟[1],吴瑞芳[1]

(1.山西省财政税务专科学校 大数据学院,山西 太原 030024;
2.太原理工大学 计算机科学与技术学院(大数据学院),山西 太原 030600)

摘　要:典型的云作业调度策略不能满足现有云系统工作效率的需求,为了进一步提升其性能,提出一种基于烟花算法的调度策略.首先,分析限制云作业调度的节点性能指标,并将这些节点性能指标作为调度器的决策因素;然后,通过数学问题建模,使用模拟植物生长的算法改进烟花算法中爆炸烟花的分布方式,使其按照植物的生长方式分布烟花;最后,将所提算法与 4 种典型的云作业调度算法进行试验对比,分析所提算法的性能.结果表明:与典型的云作业调度算法相比,所提算法可以更好地提升系统的性能.

关键词:调度算法;模拟植物生长算法;云计算;负载均衡;烟花算法
中图分类号:TP391.9　　**文献标志码**:A　　**文章编号**:1672-9102(2023)04-0088-09

Cloud Job Scheduling Strategy Based on Fireworks Algorithms

LI Qiang[1], LIU Xiaofeng[2], KONG Dejin[1], QIN Huawei[1], WU Ruifang[1]

(1. College of Big Data, Shanxi Finance and Taxation College, Taiyuan 030024, China;
2. College of Computer Science and Technology (College of Data Science), Taiyuan University of Technology, Taiyuan 030600, China)

Abstract: Typical cloud job scheduling algorithms can not meet the efficiency requirements of existing cloud systems, and the improvement of algorithms needs to be further implemented, a cloud job scheduling strategy based on fireworks algorithms is proposed. Firstly, the performance indicators of nodes that restrict cloud job scheduling are analyzed, and are used as decision-making factors of the scheduler. Secondly, through mathematical problem modeling, the plant growth simulation algorithm is used to improve the distribution of explosive fireworks in traditional fireworks algorithm, so that fireworks can be distributed according to the growth mode of plants. Finally, four typical cloud job scheduling algorithms are compared with the proposed algorithm by experiments. Experiments show that the proposed algorithm can better improve the performance of the system than the typical scheduling algorithm for cloud job.

Keywords: scheduling algorithm; plant growth simulation algorithm; cloud computing; load balancing; fireworks algorithm

随着大数据应用的不断发展,作为大数据支撑的云计算平台显得格外重要,云计算的作业调度效率直

收稿日期:2021-06-10
基金项目:国家自然科学基金资助项目(61502330);山西省高等学校科技创新项目资助(2020L0743;2022-676);山西省软科学计划研究项目资助(2016041008-5)
* **通信作者**,E-mail:122824940@qq.com

接关系到云计算平台的性能.以 MapReduce 为典型代表的大规模云计算模型为例,该模型的默认调度算法仅考虑计算节点的本地性、机架的本地性和机架的距离这 3 个方面,而现实调度环境中还有很多需要考虑的因素,如环境的异构性、系统整体的负载能力和负载均衡等,因此,该模型还有很多可以改进的地方[1].

针对云计算环境具有异构性的特点,PANDEY 等[2]分析异构环境如何影响云作业调度器并进行设计;GHOMI 等[3]对云计算环境下的负载均衡算法进行调查研究;KHEZR 等[4]针对 MapReduce 的未来发展趋势和应用,提出自己的观点.边缘计算作为云计算的一种典型代表,MACH 等[5]对移动通信中的云任务卸载进行了比较系统的阐述;SAHNI 等[6]为提升边缘计算的系统调度能力,提出一种基于数据敏感性的云作业调度模型;LI 等[7]采用 Ad-Hoc 的自组织方式建立调度器,将系统资源耗费和时间开销作为优化目标;KANG 等[8]提出一种 Neurosugeon 的神经网络调度器,可实现移动设备和云计算中心之间的任务协调,且能降低系统功耗和时间开销;ZHANG 等[9]采用社会感知模型降低系统延迟,实现调度器的优化;宋杰等[10]提出一种采用动态能耗评估策略实现任务调度的方法,以减少系统能耗为目标实现作业调度.

为了解决多用户异构环境下 Hadoop 作业调度效率缺乏反馈机制的问题,马莉等[11]提出一种动态反馈模型来提升云作业调度的效率,通过心跳包实时计算作业的平均到达率和平均完成率,并将这 2 个参数反馈到作业队列调度器来决策作业的调度优先级,缩短作业调度队列的平均长度;马文等[12]在 CloudSim 环境下,提出一种混合云作业调度算法,提升混合云的效率;李强等[13]采用能量动力改进 Logistic 模型,提出一种受自然环境影响的模拟植物算法来实现云作业调度,并通过理论和试验选择适合云作业调度的植物类型,该方法在云作业调度效率方面有显著提升;王满英[14]提出一种通过评价 QoS 服务质量实现缩短作业完成时间的方法,该方法可以显著提升系统作业的调度性能,满足用户服务质量的要求;XU 等[15]以弹性资源作为云作业调度的条件,提出一种非线性问题解决方案来保证服务质量;SELVAM 等[16]通过改进副本策略、下一个任务到达时间和磁盘运行情况决策云作业的调度顺序,但是该算法没有考虑远程任务的调度情况;RASOOLI 等[17]考虑作业到达率和作业完成周期两个方面,在异构环境下改进云作业调度器的实现算法,但是该方法没有综合考虑其他因素的影响,还有很多改进空间;ZHANG 等[18]综合考量多方面云作业调度环境因素,提出一种基于多目标的云作业调度模型,可以较好地提升系统的整体性能.

提升任务调度性能的方法很多,夏家莉等[19]提出一种动态优先级的任务调度算法,可以综合提升作业调度的实时效率;王万良等[20]使用 Hopfield 神经网络来解决车间作业调度问题;王秀利等[21]利用神经网络来提升多处理机作业调度的效率.

很多人工智能算法在解决任务调度问题方面表现突出,如粒子群算法、蚁群算法和神经网络等,其中,烟花算法是近年内被学术界所推崇的新算法,在任务调度性能中该算法的效果显著[22].为了进一步提升烟花算法的性能和应用效果,出现了大量的改进算法.张以文等[23]采用高斯变异方法,通过增加烟花种群的多样性来提升算法的效率,并在 Web 服务组合优化中得到很好的应用;余冬华等[24]提出峰值火花的概念并将其应用到烟花的选择策略上来改善搜索效率,试验证明该方法要优于粒子群算法和普通的烟花算法;朱启兵等[25]通过优化粒子群算法,提出一种带有引力的搜索算法来提升算法性能.

通过提升种群的多样性来提高算法的效率,白晓波等[26]利用高斯函数实现混合变异算子,达到精英粒子的优化,提出一种烟花算法优化粒子波的方法,可以有效解决精英粒子的优化问题.云作业的调度问题可以视作是在有限条件下对任务进行组合优化.本文模拟植物的生长过程来改进烟花算法的分布,优化调度云作业的先后顺序,实现云作业调度效率的整体提升.

1　云作业调度的资源环境

云作业调度的本质是在多个资源条件下的动态分配过程,对应的数学问题为资源的组合优化过程.

1.1　调度器模型设计

作业调度模型由作业、节点、调度器和调度算法构成,系统模型如图 1 所示.其中,调度器配置了本文提出的调度算法,所提交的作业将根据本文算法进行优先级的决策和资源分配.

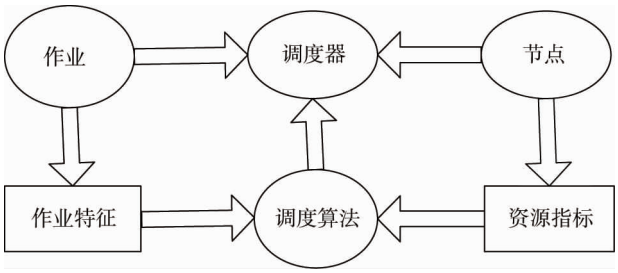

图 1 系统模型

1.2 性能指标

试验中进行作业和资源分配的 5 个性能指标如表 1 所示.

表 1 资源利用率

名称	单位
CPU 平均利用率	%
内存平均利用率	%
I/O 平均利用率	%
网络平均利用率	%
磁盘平均利用率	%

1.3 问题描述

文本经过分析,作业调度模型问题的表达可以转化为多个条件函数的极值问题.被处理的作业集合为 $J=\{J_1, J_2,\cdots, J_j\}$,$J_j$ 为第 j 个作业,$j=1, 2,\cdots, n$; M 为计算节点集合为 $M=\{M_1,M_2,\cdots,M_i\}$, M_i 为第 i 个计算节点,$i=1, 2,\cdots, m$.

定义 1 f 为目标函数,定义为在所有条件组合中用时最小的排列,该函数表达式如式(1)所示.

$$f=\min\left(\max_{T}\sum_{j=1}^{n}\sum_{i=1}^{m}\sum_{k=1}^{T}Y_{jik}\right). \tag{1}$$

式中:T 为时间片数;Y_{jik} 为任务当前计算节点 i 上第 k 个时间片上被分配第 j 个作业,取值为 0 或 1.

定义 2 每个作业的最长时间约束 λ:

$$\lambda=\min\sqrt{\sum_{j=1}^{n}\left(\sum_{i=1}^{m}\sum_{k=1}^{T}Y_{jik}-T_j\right)^2}. \tag{2}$$

作业 j 的运行时间预期为 T_j 个时间片完成,若每个作业都可以按预期时间完成,则 λ 接近 0.

定义 3 计算节点时间限制条件 τ_1,定义为每个计算节点上一个时间片周期内只允许单个作业运行,如式(3)所示.

$$\tau_1=\sum_{j=1}^{n}\sum_{i=1}^{m}\sum_{k=1}^{T}\sum_{n}Y_{jik}\cdot Y_{hik}=0. \tag{3}$$

式中:Y_{hik} 为在第 i 个计算节点第 k 个时间片上运行第 h 个作业.

定义 4 作业互斥条件 τ_2,定义为每个作业在同一个时间片周期内只能在单个计算节点上运行,如式(4)所示.

$$\tau_2=\sum_{j=1}^{n}\sum_{i=1}^{m}\sum_{k=1}^{T}\sum_{m}\sum_{l=1}^{T}Y_{jik}\cdot Y_{jhl}. \tag{4}$$

式中:l 为时间片序列; Y_{jhl} 为在第 l 个时间片上第 j 个作业运行在第 h 个计算节点上.

定义 5 空闲节点的限制条件 τ_3,定义为在任意组合的任务序列中,要求不存在空闲节点,以保证系统的最大工作效率,如式(5)所示.

$$\tau_3=\sum_{i=1}^{m}\sum_{k=1}^{T}\left(\sum_{j=1}^{n}Y_{jik}-1\right)^2. \tag{5}$$

定义 6 限制每个任务最长时间 τ_4:

$$\tau_4 = \sum_{j=1}^{n}\sum_{i=1}^{m}\sum_{k=1}^{T} Y_{jik}K^2G(L_{jik}); \tag{6}$$

$$G(L_{jik}) = \begin{cases} 0, L_{jik} > 0; \\ 1, L_{jik} \leqslant 0. \end{cases} \quad K = k - t_j. \tag{7}$$

式中:K 为执行时间与最大时间约束之差;$G(L_{jik})$ 为单位函数;L_{jik} 为第 k 个时间片上第 j 个作业执行在第 i 个节点上的超时标志;t_j 为作业 j 的最长时间.

定义 7　资源约束条件 τ_5,保证任意两个任务不会同时争用同一资源,如式(8)所示.

$$\tau_5 = \sum_{j=1}^{n}\sum_{i=1}^{m}\sum_{k=1}^{T}\sum_{n}\sum_{l=1}^{m}\sum_{w=1}^{F} Y_{jik}R_{jikw}Y_{hlk}R_{hlkw} = 0. \tag{8}$$

式中:w 为资源序号;F 为最大资源数;R_{jikw} 为第 k 个时间片上第 j 个作业执行在第 i 节点上占用第 w 个资源.

1.4　问题的数学表达

将上述函数和条件公式联合表达为 $L(\ )$ 函数,表达式如式(9)所示.

$$L(\ ) = f + h + \sum_{i=1}^{5}\theta_i\tau_i. \tag{9}$$

式中:θ_i 为权重系数,其值在 0~1.

通过 $L(\ )$ 函数将问题转换为函数表达后,就可以按本文提出的改进烟花算法来实现问题的求解.

2　烟花算法

2.1　算法描述

烟花算法是通过烟花爆炸的过程模拟空间搜索,与其他群体智能优化算法类似,烟花爆炸的过程可以理解为全局优化的一种模拟烟花算法,通过随机函数生成若干个种群,然后种群中的每个烟花通过爆炸或者高斯变异方法生成新的烟花因子,最后经过选择算法优选精英粒子,生成下一次烟花火种.通过反复迭代,直到计算精度可以满足条件或者达到预设的最大迭代次数.

2.2　烟花爆炸

假设在烟花算法的迭代过程中,S_i 为第 i 个烟花种,每个烟花种在空间爆炸产生烟花,在每次迭代过程中可以产生 $S_{\max}$ 个爆炸烟花.该爆炸烟花的数量与烟花的适应度值成正比,即适应度值高的产生较多的爆炸烟花,适应度值低的产生较少的爆炸烟花.每个爆炸烟花产生的范围被限定在每个烟花的爆炸半径内,同时每个烟花的爆炸半径的长度与该烟花的适应度值成反比,即适应度值较小的烟花爆炸半径比较大,这样可以保证烟花的寻优过程不被限制在局部,可以在更大的空间内搜索结果.第 i 个烟花种 S_i 和第 i 个烟花的爆炸半径 E_i 的计算公式为

$$S_i = C\frac{g_{\max} - g_i + \rho}{\sum_{i=1}^{Z}(g_{\max} - g_i) - \rho}; \tag{10}$$

$$E_i = E\frac{g_i - g_{\min} + \rho}{\sum_{i=1}^{Z}(g_i - g_{\min}) - \rho}. \tag{11}$$

式中:C 为爆炸烟花种数;$g_{\max}$ 为种群中的最大适应度值;g_i 为适应度值函数;ρ 为一个小数,保证分子不为 0;E 为烟花的爆炸半径;$g_{\min}$ 为种群中的最小适应度值,是一个小数,用来保证除数不为 0.

通过取整函数可以保证每次烟花的种群数为整数,可以通过式(12)控制取整过程和烟花的数量,同时可以实现控制较低适应值的烟花数过多且较高适应值的烟花数过少的情况.

$$S_i = \begin{cases} \mathrm{Rnd}(\alpha S_{\min}), S_i < \alpha S_{\min}; \\ \mathrm{Rnd}(\beta S_{\max}), S_i > \beta S_{\max}; \ \alpha < \beta < 1. \\ \mathrm{Rnd}(S_i), \text{其他情况}. \end{cases} \tag{12}$$

式中:Rnd( )为四舍五入取整函数;α 和 β 为预设参数,一般情况下,设定 $\alpha=0.1$,$\beta=0.2$; $S_{\min}$ 为最少烟花种数.

2.3 爆炸烟花的计算

第 i 个烟花将生成 S_i 个烟花种,每个烟花具有 γ 个维度,对于一个烟花种子的第 μ 个维度记为 S_i^{μ}($\mu=1,2,\cdots,\gamma$). 按式(13)可以计算出每个爆炸烟花的空间距离:

$$S_{i,j+1}^{\mu}=S_{i,j}^{\mu}+R_iU(-1,1). \tag{13}$$

式中: R_i 为第 i 个种子的分布宽度系数;$U(\ )$为均匀分布函数.

当某个爆炸烟花的第 μ 个 维度超出上下界时,通过式(14)将该维度值 S_i^{μ} 映射到一个边界内的位置.

$$S_{i,j+1}^{\mu}=S_{i,j}^{\mu}\%(B_{\text{up}}-B_{\text{down}})+B_{\text{down}}. \tag{14}$$

式中 :B_{up} 和 B_{down} 分别为该维度的上下界;%为求余运算.

2.4 高斯变异

烟花算法通过高斯变异提升种群的多样性,首先在烟花种群中任意选取 p 个烟花,在每个烟花种的 γ 个维度根据式(15)进行高斯变异.

$$S_{i,j+1}^{\mu}=S_{i,j}^{\mu}\text{GS}(1,1). \tag{15}$$

式中:GS(1,1)为均值和方差都为 1 的高斯分布.

同理,当某烟花的维度超过界限时,需要将空间距离通过式(14)重新转换到一个合理距离.

2.5 选择概率

在每次迭代过程中,为了将精英烟花种子遗传到下一次迭代过程中,需要从烟花、爆炸烟花和高斯变异烟花构成的集合 J 中通过选择规则保留 1 个精英种子和随机选取 $n-1$ 个种子.该选择概率方法如式(16)所示.

$$p_i=\sum\frac{\text{dist}(S_i-S_j)}{\sum_{i=1}^{n}S_i}. \tag{16}$$

式中:dist( )为距离函数,用于计算任意两个烟花种子的距离.烟花种子的距离越大,p_i 值越大,概率越大,越容易被选中;反之,越易被淘汰.

3 算法改进

传统的烟花算法中的爆炸烟花是通过均匀分布实现的,而现实中的爆炸烟花并不是按此分布.为了提高爆炸烟花的模拟效果,本文通过模拟植物生长算法来更新爆炸烟花.

3.1 模拟植物生长算法描述

为了使爆炸烟花的分布更加合理,采用模拟植物生长的算法来代替高斯均匀分布,使种子的确定更加合理.植物的整个生长过程[27]:首先,植物的茎从原始发芽点开始生长,并且茎上某些位置还可以生长出其他新芽,该位置被定义为新的生长点;然后,在这些新的生长点上还可以长出新的枝条,并且新的枝条上还会生长出新的芽;最后,通过同样的生长过程反复长出新的枝条,构成类似的结构.

根据文献[27],植物上的发芽位置与该点的生长激素有关,而这些激素水平决定该生长点是否可以长成新芽.第 i 个生长点的激素浓度 q_i 可以按式(17)表示.

$$q_i=\frac{g(o)-g(i)}{\sum_{i}^{D}[g(o)-g(i)]+\sum_{j}^{Q}[g(o)-g(j)]}. \tag{17}$$

式中: $g(o)$ 为原始生长点目标函数;$g(i)$,$g(j)$为第 i 个和第 j 个生长点的目标函数;D 为总生长点数;Q 为枝生长点数.

设每个茎的长度为 L', 并具有 q 个生长点,表示为 $C_N=\{C_{N_1},C_{N_2},\cdots,C_{N_q}\}$.此外,生长点 C_{N_i}对应的激素浓度为 q_{N_i}.每个枝条上有 l 个生长点,且生长点距离相同,该枝条上的生长点 $B_L=\{B_{L_1},B_{L_2},\cdots,B_{L_l}\}$,其

对应的激素浓度为 q_{L_i}.茎和枝的生长激素浓度可以按式(18)和式(19)表示.

$$q_{N_i}=\frac{g(o)-g(C_{N_i})}{\sum_{i}^{q}[g(o)-g(C_{N_i})]+\sum_{j}^{l}[g(o)-g(B_{L_j})]};\tag{18}$$

$$q_{L_i}=\frac{g(o)-g(B_{N_i})}{\sum_{i}^{q}[g(o)-g(C_{N_i})]+\sum_{j}^{l}[g(o)-g(B_{L_j})]}.\tag{19}$$

式中:q 为茎生长点数;l 为枝上生长点数,茎和枝的生长点共有 $q+l$ 个,同时对应 $q+l$ 个激素.

每次迭代生长过程中会产生新的生长点,并添加到生长点集合 S 中,并且按式(18)和式(19)更新每个生长点的激素水平.所有的激素浓度和为 1,如式(20)所示.

$$\sum_{i}^{q}q_{N_i}+\sum_{j}^{l}q_{L_j}=1.\tag{20}$$

通过模拟植物算法,烟花爆炸的形态将改变分布,爆炸烟花的形状如图 2 所示.

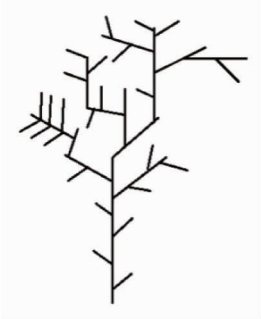

图 2　爆炸烟花的形状

3.2　算法步骤

基于模拟植物生长改进的烟花算法步骤如下:

第 1 步:随机产生 n 个烟花种子.

第 2 步:烟花种子分别爆炸,并计算每个烟花爆炸的半径以及爆炸烟花的个数.

第 3 步:依据式(17)~式(20)模拟植物生长算法形成爆炸烟花的分布.

第 4 步:依据植物模拟生长算法产生变异烟花.

第 5 步:由烟花、爆炸烟花和变异烟花组成生长点集合 S,选择 1 个精英种子并通过式(16)选取 $n-1$ 个烟花种子.

第 6 步:若满足计算精度或达到迭代上限,则输出结果;否则,重复执行第 2 步~第 6 步.

第 7 步:输出最优值.

4　试验分析

4.1　试验条件

为了测试算法的效果,试验在 11 台联想 PC 机上搭建 Hadoop 集群,每台计算机采用相同的配置,CPU3.2 GHz,8 G 内存,500 G 硬盘,千兆光纤网络.其中 1 台为管理节点(Master),剩余 10 台为从节点(Slave),运行操作系统为 Linux.

采用标准的测试用例 WordCount,Terasort 和 Pi Estimator.其中,WordCount 和 Pi Estimator 用例为 CPU 密集型,而 Terasort 兼有 IO 密集型和 CPU 密集型的综合用例.分别运行以上测试用例,并设置 Map 和 Reduce 的任务数都为 10.该算法部署在管理节点上,在执行的动态过程中实时监测并获取每个从节点的资源参数,设定管理节点的采样周期为 0.1 s.

4.2 结果分析

试验依次运行 WordCount,Terasort 和 Pi Estimator 这 3 个标准测试用例,系统执行效率如表 2 所示.由表 2 可知:这 3 个标准测试用例的执行时间分别为 835, 733 和 137 ms.

表 2 系统执行效率

用例	执行时间/ms	Map 计时/ms	Reduce 计时/ms
WordCount	835	791	820
Terasort	733	695	721
Pi Estimator	137	113	130
总计	1 705	1 599	1 671

为了验证算法效果,将本文算法与其他典型算法进行比较,包括烟花算法、能耗算法、动态调用算法以及 Hadoop 系统中缺省的公平调度算法.算法结果如表 3 所示.由表 3 可知:5 种算法的时间开销依次是 1 705,2 097,2 443,2 531 和 2 915 ms,前 4 种算法比 Hadoop 缺省的公平调度算法在效率上分别可以提升 42%, 28%, 16%和 13%,Hadoop 缺省的公平调度算法的性能提升空间较大,本文算法比其他算法具有明显的性能优势.同时,本文算法比烟花算法在效率上可以提高 18.69%,说明改进烟花算法所采用的模拟植物生长的爆炸烟花分布方式,其效果显著.

表 3 算法结果

算法名称	执行时间/ms	Map 计时/ms	Reduce 计时/ms	提升率/%
本文算法	1 705	1 599	1 671	42
烟花算法	2 097	2 013	2 085	28
能耗算法	2 443	2 383	2 431	16
动态调用算法	2 531	2 513	2 527	13
公平调度算法	2 915	2 757	2 897	–

根据系统 Monitor 程序记录和统计系统的状态,5 种算法的系统整体资源利用率(CPU 平均利用率、内存平均利用率、I/O 平均利用率、磁盘平均利用率和网络平均利用率)如图 3 所示.由图 3 可知:本文算法比其他 4 种算法具有更加明显的优势.本文算法的 5 个指标均在 75%~81%,说明该算法还有较强的弹性,系统还能承载更多的负载.而公平调度算法的 4 个指标和动态调度算法的 1 个指标已经超出 90%,系统已经进入瓶颈.

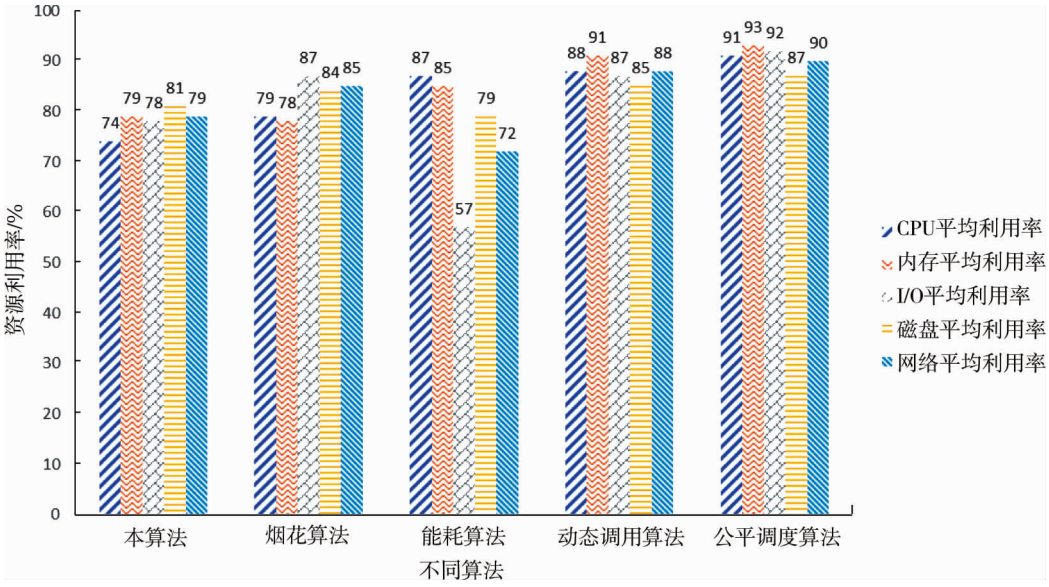

图 3 各算法的资源利用率

从主机内的角度考虑,本文算法、烟花算法和能耗算法的 CPU 平均利用率和内存平均利用率都可以

控制在 85%以下,说明这 3 种算法在运算上的花销基本相同.

再从主机外的角度分析,能耗算法的 I/O 平均利用率、磁盘平均利用率和网络平均利用率明显最低,而该算法的 CPU 平均利用率和内存平均利用率这 2 个性能指标较高,都达到 85%左右,可知该算法为了显著地降低能耗开销,将大量的计算留在本地执行,导致系统在分配任务时偏重距离近的节点分配.而动态调用算法的 5 个指标都在 85%以上,比公平调度算法的指标略低,说明动态调用算法在各个节点之间经常调度作业,外部开销过大也会增大内部开销.

试验节点各指标的利用率如图 4 所示.图 4 中第 1 个节点是管理节点(Master 节点),主要负责执行本文算法和决策器,即作业的分配和调度,该节点的 5 个指标性能都是最高的,因此,大量的通信任务和作业记录都在该节点执行,导致该节点与其他节点相比,其任务最重.而其他节点的每个指标性能基本相似,说明本文算法基本上可以比较公平地将作业分配到各个计算节点,实现自动负载均衡,使系统的整体性能可以较好地发挥.

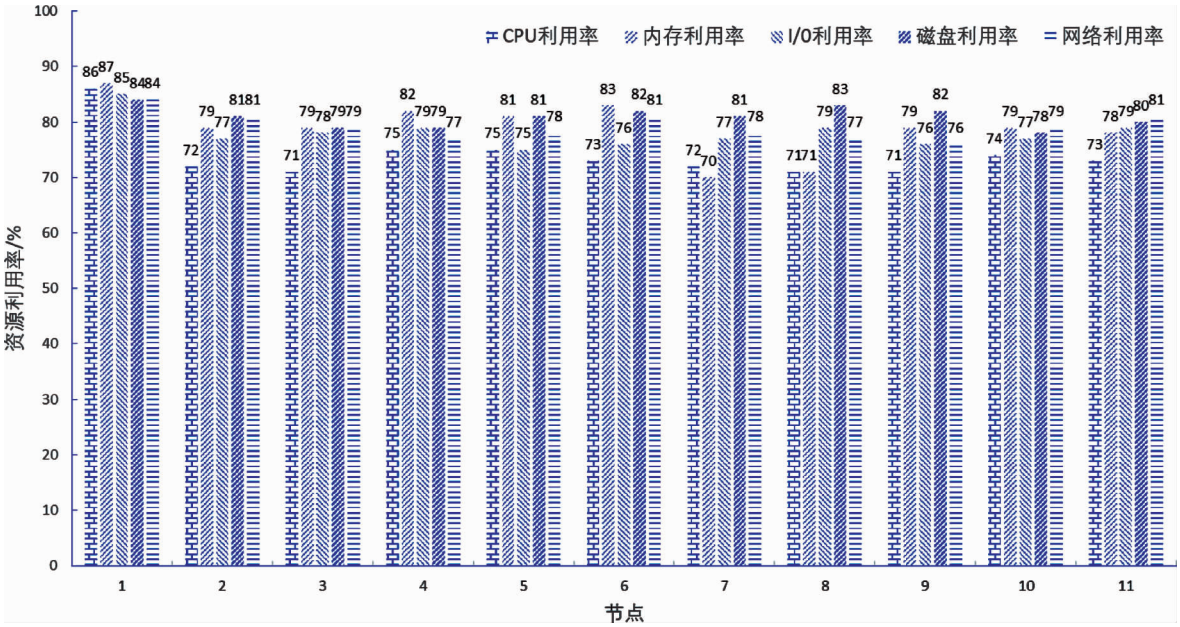

图 4　试验节点的指标利用率

由图 4 可知:11 个计算节点的 CPU 平均利用率和内存平均利用率基本分布在 70%~80%,说明计算任务的负载是比较均衡的.它们的 I/O 平均利用率、磁盘平均利用率和网络平均利用率也基本在 80%左右,可以证明这些计算节点的主机外开销也比较接近,可以比较合理地承担外部开销,说明本文提出的改进烟花算法可以更好地分配系统资源,且能够充分调度系统资源并比较合理地分配云作业.

5　结论

1)所提方法可以从整体提升系统的调度性能,具有更好的资源平衡性.

2)与单一使用高斯变异改善爆炸烟花分布的方法相比,本文算法具有更好的适应性.

3)烟花种子的起点位置对整个系统的性能有较大的影响.因此,针对烟花种子起点位置的确定算法是下一步的研究目标.

参考文献:

[1] 黄山,王波涛,王国仁,等.MapReduce 优化技术综述[J].计算机科学与探索,2013,7(10):885-905.

[2] PANDEY V, SAINI P. How Heterogeneity Affects the Design of Hadoop MapReduce Schedulers: A State-of-the-Art Survey and Challenges[J].Big Data,2018,6(2):72-95.

[3] GHOMI E J, RAHMANI A M, QADER N N. Load-balancing algorithms in cloud computing: A survey[J]. Journal of Network

and Computer Applications,2017,88:50-71.
[4] KHEZR S N, NAVIMIPOUR N J. MapReduce and its applications, challenges, and architecture: a comprehensive review and directions for future research[J]. Journal of Grid Computing,2017,15(3):295-321.
[5] MACH P, BECVAR Z. Mobile edge computing: A survey on architecture and computation offloading [J]. IEEE Communications Surveys & Tutorials, 2017, 19(3):1628-1656.
[6] SAHNI Y, CAO J N, YANG L. Data-aware task allocation for achieving low latency in collaborative edge computing[J]. IEEE Internet of Things Journal, 2019, 6(2): 3512-3524.
[7] LI T Z, WU M Q, ZHAO M, et al. An overhead-optimizing task scheduling strategy for ad-hoc based mobile edge computing[J]. IEEE Access, 2017, 5: 5609-5622.
[8] KANG Y P, HAUSWALD J, GAO C, et al. Neurosugeon: collaborative intelligence between the cloud and mobile edge[J]. ACM SIGARCH Computer Architecture News, 2017, 45(1):615-629.
[9] ZHANG D, MA Y, ZHENG C, et al. Cooperative-competitive task allocation in edge computing for delay-sensitive social sensing[C]// 2018 IEEE/ACM Symposium on Edge Computing(SEC). IEEE, 2018: 243-259.
[10] 宋杰,徐澍,郭朝鹏,等.一种优化 MapReduce 系统能耗的任务分发算法[J].计算机学报,2016,39(2):323-338.
[11] 马莉,唐善成,王静,等.云计算环境下的动态反馈作业调度算法[J].西安交通大学学报,2014,48(7):77-82.
[12] 马文,耿贞伟,张莉娜.基于数据挖掘的混合云作业调度算法[J].现代电子技术,2017,40(19):49-51.
[13] 李强,刘晓峰.基于模拟植物生长算法的云作业调度模型[J].系统仿真学报,2018,30(12):4649-4658.
[14] 王满英.基于 QoS 模型感知的云作业调度算法[J].计算机工程与应用,2014,50(8):57-60.
[15] XU X Y, TANG M L, TIAN Y C. QoS-guaranteed resource provisioning for cloud-based MapReduce in dynamical environments[J]. Future Generation Computer Systems,2018,78:18-30.
[16] SELVAM S, GOPALAN N P. An optimal task selection scheme for Hadoop scheduling[J]. IERI Procedia, 2014, 10:70-75.
[17] RASOOLI A, DOWN D G. COSHH: A classification and optimization based scheduler for heterogeneous Hadoop systems[J]. Future Generation Computer Systems,2014,36:1-15.
[18] ZHANG F, CAO J W, LI K Q, et al. Multi-objective scheduling of many tasks in cloud platforms[J]. Future Generation Computer Systems, 2014,37:309-320.
[19] 夏家莉,陈辉,杨兵.一种动态优先级实时任务调度算法[J].计算机学报,2012,35(12):2685-2695.
[20] 王万良,吴启迪,徐新黎.基于 Hopfield 神经网络的作业车间生产调度方法[J].自动化学报,2002,28(5):838-844.
[21] 王秀利,吴惕华.一种求解多处理器作业调度的 Hopfield 神经网络方法[J].系统工程与电子技术,2002,24(8):13-16.
[22] 谭营,郑少秋.烟花算法研究进展[J].智能系统学报,2014,9(5):515-528.
[23] 张以文,吴金涛,赵姝,等.基于改进烟花算法的 Web 服务组合优化[J].计算机集成制造系统,2016,22(2):422-432.
[24] 余冬华,郭茂祖,刘晓燕,等.改进选择策略的烟花算法[J].控制与决策,2020,35(2):389-395.
[25] 朱启兵,王震宇,黄敏.带有引力搜索算子的烟花算法[J].控制与决策,2016,31(10):1853-1859.
[26] 白晓波,邵景峰,田建刚.改进的烟花算法优化粒子滤波研究[J].计算机科学与探索,2018,12(11):1827-1842.
[27] 曹庆奎,刘新雨,任向阳.基于模拟植物生长算法的车辆调度问题[J].系统工程理论与实践,2015,35(6):1449-1456.