面向云数据中心的虚拟机部署时延优化算法研究*

敬 超[a,b†], 程小辉[a,b]
(桂林理工大学 a. 广西高校重点实验室; b. 信息科学与工程学院, 广西 桂林 541004)

摘 要: 考虑了服务器内资源容量及虚拟机多类型资源需求时虚拟机部署最优化时延问题。首先,将最优化虚拟机部署时延问题进行了形式化建模,并证明了该问题为一个 NPC 问题;通过遗传结合贪心策略提出了一种高效的虚拟机部署算法优化时延。该算法的主要特点在于:结合了贪心策略指导个体在初始化、选择、交叉、变异时形成最优解,并且在交叉过程中采用奇、偶数位个体交叉的方式形成新个体,既避免了个体间的重复交叉,又通过多样化的新个体形成使得算法的解不会陷入局部最优;另外,由于遗传算法在交叉过程中会存在交叉冲突问题(服务器容量超载),还设计了一种交叉冲突检查方法避免了交叉冲突后非法个体的生成;最后,通过实验比对,将提出的算法分别与最新研究成果 VMPDN、粒子群优化算法等算法比较,结果表明提出算法有效地缩短了虚拟机的部署时延。同时通过不同资源类型数量、迭代次数及种群大小的情况下,分析和考察了算法的性能,结果表明提出算法的性能优好于其他的算法。

关键词: 云计算; 数据中心; 虚拟机部署; 遗传算法; 时延优化
中图分类号: TP391.9 **文献标志码**: A **文章编号**: 1001-3695(2017)12-3792-05
doi:10.3969/j.issn.1001-3695.2017.12.062

Research of virtual machine placement algorithm for latency optimization on cloud data centers

Jing Chao[a, b†], Cheng Xiaohui[a, b]
(*a. Guangxi Key Laboratory of Embedded Technology & Intelligent Information Processing, b. School of Information Science & Engineering, Guilin University of Technology, Guilin Guangxi* 541004, *China*)

Abstract: This paper investigates the the problem of minimizing the latency of VM placement on data centers with the considerations of physical machine capacity and multi-type resource. First, the problem is strictly formulated and proved to be an NPC problem. Then, based on genetic algorithm, we propose an efficient VM placement algorithm by integrating with greedy scheme. The main distinct of the proposed algorithm is as follows: to obtain the optimal results, it combines with the greedy algorithm for individuals' initialization, selection, crossover and mutation. Also the crossover is taken place between the individual with odd and even number to avoid the repeat crossover and trap-in local optimum. Besides, we design a method to check the collision in the process of crossover (VMs are overloading on certain server). Finally, we compared the proposed algorithm with the latest algorithm VMPDN and Particle Swarm Optimization (PSO). The results have shown that the proposed algorithm outperforms than that of the other algorithm in minimizing the latency. Moreover, with different type of resource, various number of iteration and size of population, the proposed algorithm still demonstrates better performance than that of the other algorithms.

Key words: cloud computing; data center; virtual machine placement; genetic algorithm; latency optimization

0 引言

随着互联网技术的兴起,网络业务服务请求的猛增,许多企业都以云计算为核心技术建立数据中心以满足用户的需求。基于云计算技术数据中心不仅可以为在线请求服务提供质量保证,而且还提供了可靠和安全的服务。这是因为在云数据中心里主要采用的技术包括:虚拟化技术、存储技术以及节能技术等,其中虚拟化技术做为核心技术可以有效地提高数据中心资源的利用率,同时根据用户的服务请求将虚拟机部署在合理的服务器上为用户提供服务。

然而,由于云数据中心内的服务器数量成百上千,数据中心网络带宽受限,请求数量的日益增加以及虚拟化资源的管理,使得虚拟机部署时延优化问题一直以来为很多专家和学者所研究。目前这一方面的研究主要是集中在最小化虚拟机迁移时延上,2009 年 Li 等人[1]提出了一种启发式的节能迁移算法,将待虚拟机组按照资源空间进行等级划分,合理利用零散空间提高资源利用率。2010 年刘鹏程等人[2]考虑了不同操作系统异构性及管理难度大的问题,提出了一种虚拟机动态迁移框架缩短总迁移时间。文献[3]中提出了一种虚拟机选择策略,从长远的角度考虑选择迁移代价最小的虚拟机进行迁移。为降低虚拟机过度聚集造成的服务器热点问题,文献[4]从降

收稿日期: 2016-10-17; **修回日期**: 2016-11-25 **基金项目**: 广西自然科学基金资助项目(2015GXNSFBA139260);国家自然科学基金资助项目(61563012,61540054);广西高校重点实验室主任基金资助项目(2016-01-05);桂林理工大学博士启动基金资助项目(002401003456)

作者简介:敬超(1983-),男(通信作者),河南长葛人,讲师,博士,主要研究方向为云计算与大数据处理(jingchao@alumni.sjtu.edu.cn);程小辉(1961-),男,教授,博士研究生,主要研究方向为嵌入式系统与物联网技术.

低能耗的角度出发,提出一种静态的虚拟机部署方法,并提高了资源的利用率。文献[5]则是从数据传输方法的角度,降低单个虚拟机迁移过程的停机时间。文献[6]将虚拟机迁移问题抽象为装箱问题,提出了一种基于 NSGA-Ⅱ迁移算法,该算法的目标是使得服务器数量最少和虚拟机迁移次数最小。

在数据中心由于虚拟机的部署会对数据的处理时延产生影响,所以 2013 年 Alicherry 等人[7]提出了一种智能感知的虚拟机部署方法,缩短了数据连接时延,在此基础上,文献[8]提出了一种虚拟机部署主要是为了最小化数据连接时延问题,将虚拟机部署问题描述为一个混合整数的规划问题,并以阈值调整为核心思想设计了一种虚拟机部署的逼近算法,主要是通过设定阈值 t 作为最小时延,在每一轮计算过程中选择数据节点到虚拟机的最小时延。还有的研究则是从流量感知的角度来解决虚拟机部署的问题并提出了相关的虚拟机部署算法,如文献[9,10]。

与这些工作相比,本文研究的是云数据中心虚拟机部署的问题,如图 1 所示,目标是最小化虚拟机的部署时延,主要包括虚拟机的迁移时延和目标服务器的配置时延,同时考虑了虚拟机多种资源的需求,提出了一种基于遗传算法结合贪心策略的虚拟机部署算法(GGA)。该算法的主要特点是在遗传算法的架构下,根据数据中心部署时延差异性,采用贪心策略让虚拟机选择目标服务器。其中由于服务器资源受限,会导致使用遗传算法中交叉操作可能出现的交叉冲突问题,GGA 还借鉴了文献[6]的工作设计了一种解决交叉冲突的策略,它主要是通过检查交叉的合法性(即服务器实际容量是否满足虚拟机资源需求)来避免交叉冲突。

1　相关工作

目前对于虚拟机迁移问题的研究工作有很多:张彬彬等人[5]从系统的角度来考虑虚拟机迁移代价的问题,提出了三阶段迁移方法,该方法主要是通过减少关键数据传送造成的虚拟机停机时间,来降低虚拟机迁移代价。有的工作则是从多目标优化的角度设计迁移算法,权衡了服务器数量、SLA (service level agreement)违背率和虚拟机后期迁移次数解决虚拟机的迁移问题。其中胡元元等人[3]则是将迁移方案分成两个部分进行考虑,即虚拟机的迁入和迁出。前者是将迁移代价最小的服务器当作目标服务器将虚拟机迁入,后者是从虚拟机的角度挑选迁移代价最小的虚拟机迁出,通过这种方式达到迁移代价最小化的目的。李强等人[6]将虚拟机迁移问题抽象为装箱问题(bin-packing problem)。他们将数据中心的服务器当做箱子,虚拟机视作要装进箱子的物品的组合优化问题。针对这个问题,他们在工作中提出了一种基于 NSGA-Ⅱ遗传算法,来减少服务器的数量、虚拟机迁移次数,从多目标的优化角度提高和保障数据中心的性能。

还有的研究则是解决数据中心内虚拟机部署对数据处理时延优化的问题,2013 年 Alicherry 等人[7]提出了一种智能感知的虚拟机部署方法最优化数据连接时延。台湾交通大学的 Kuo 等人[8]发表了关于最小化数据连接时延的虚拟机部署最优逼近算法。ICDCS'14 年 Tso 等人[9]提出了一种基于流量感知的虚拟机管理方法,该方法主要特点是不仅考虑了底层网络的带宽分配,还将高层的核心网络带宽分配也考虑在内;Infocom'14 南京大学的 Li 等人[10]基于物理机间流量的开销以及网络的消耗,提出了一种流量感知的虚拟机部署方法。文献[11]在高性能计算云中提出了一种利用合理分配虚拟机部署策略来降低数据中心的能耗费用,该方法主要考虑数据中心的静态耗电、动态耗电、冷却耗电以及峰值耗电的情况,从气候、电价分布以及冷却延时的角度,在整体上降低数据中心的总耗电价。文献[12]的主要贡献主要是将工作负荷呈负相关的虚拟机组成一组,以组为单位迁移虚拟机,这样做的优点是提高了资源利用率,减少了 SLA 违背率和能耗。

然而这些研究仅仅是从单资源限制的角度,并没有把虚拟机多重资源的需求考虑在内。与他们的工作不同,本文的虚拟机部署问题不仅需要最小化虚拟机的迁移时延,而且还考虑了虚拟机多种资源的需求问题,同时通过在遗传算法中结合贪心思想减少算法搜索空间效率高。

2　问题建模与描述

本章主要是对问题进行建模,主要包括云数据中心内的物理服务器模型、资源模型、网络模型、虚拟机模型等,最后对问题进行了形式化的描述和复杂度分析。

2.1　云数据中心模型

2.1.1　物理服务器及资源模型

本文假设云数据中心内由 N 个异构物理服务器节点组成,可以形式化表示为:$P=\{P_1, P_2, \cdots, P_N\}$,其中服务器 P_i,$i\in N$ 的容量包含资源的有计算资源(如 CPU)、内存资源以及存储资源(如硬盘大小)等,分别采用 C_i、M_i、S_i、$i\in N$ 表示服务器 S_i 中三种资源的可用数量。

2.1.2　网络模型

本文研究的数据中心网络结构主要是基于 Fat-Tree 模型,各个物理服务器之间通过数据中心内的互连网络相互通信。对于数据中心的互连网络模型,主要是通过路由器的通信链路实现虚拟机数据迁移,假设共有 L 条通信链路,每条通信链路的带宽 w_l, $l\in L$。

2.1.3　虚拟机模型

假设在 t 时刻,有 M 台虚拟机到达缓存区($M>N$),虚拟机集合表示为 $VM=\{VM_1, VM_2, \cdots, VM_M\}$。每个虚拟机所需要的资源为计算资源、内存资源以及存储资源,可以分别用 c_j、m_j、s_j、$j\in M$ 表示所需各个资源的数量。

2.1.4　虚拟机部署时延

每台虚拟机中的数据量为 Data_j,所以当虚拟机 VM_j 从缓存 B 到服务器 P_k,$k\in N$ 的虚拟机部署时延包括:虚拟机数据迁移时延 $T_m(B, k)$ 和虚拟机在目标服务器的配置时延 $T_{\mathrm{con}}(k)$,具体的计算方式如下:

$$\tau_j(B,k)=T_m(B,k)+T_{\mathrm{con}}(k)=\sum_{q\in L}\frac{\mathrm{Data}_j}{w_q}+T_{\mathrm{con}}(k) \tag{1}$$

2.2　问题定义

本文要解决的问题就是多类型资源需求的虚拟机部署时延优化问题,具体的问题描述为:基于云数据中心的物理服务器模型、网络模型、资源模型、虚拟机模型等,假设在 t 时刻,需要将 M 台虚拟机部署到 N 个物理服务器,目标是最小化虚拟机部署时延 Γ,具体的形式化描述如下:

$$\text{objective minimize}:\Gamma=\{\max(\sum_{j\in M}\tau_j(B,1)\cdot x_{(1,j)},\ \sum_{j\in M}\tau_j(B,2)\cdot x_{(2,j)},\cdots,\sum_{j\in M}\tau_j(B,N)\cdot x_{(N,j)})\} \tag{2}$$

$$\text{s. t.}\qquad x_{(i,j)}=0 \text{ or } 1 \tag{3}$$

$$\sum_{j\in M} c_j \cdot x_{(i,j)} \leqslant C_i, i \in N \tag{4}$$

$$\sum_{j\in M} m_j \cdot x_{(i,j)} \leqslant M_i, i \in N \tag{5}$$

$$\sum_{j\in M} s_j \cdot x_{(i,j)} \leqslant S_i, i \in N \tag{6}$$

其中:式(2)表示将某个服务器中最大的部署时延做为虚拟机的总体部署时延,所以问题的目标就是最小化虚拟机总体时延 Γ;式(3)中 $x(i,j)=1$ 表示把虚拟机 i 部署到目标服务器 j,为 0 时表示不部署到目标服务器 j;式(4)~(6)分别表示目标服务器的容量限制分别包括计算资源、内存资源及存储资源。

2.3 复杂性分析

定理 1 多类型资源需求的虚拟机部署时延最优化问题为 NPC 问题。

证明 要证提出的问题为 NPC 问题,分两个步骤:

a)证其为 NP 问题。给定问题的上界,猜一个可能的解 T_b,在多项式时间内验证该解,T_b 小于等于上界,该问题属于 NP 问题;

b)证明问题为 NPC 必须将一个已知的 NPC 归约于该问题。在这里将要证问题进行了简化,假设资源需求仅为一种,构造实例输入需要将 M 台虚拟机部署到 N 个物理服务器,虚拟机中的数据量为 Data_j, L 条通信链路,每条通信链路的带宽 w_l, $l \in L$,若问题的上界为 κ,那么该问题可以表示为

$$\Omega(M,N,\text{Data}_j,L,w_l) \leqslant \kappa \tag{7}$$

式(7)是将要证问题转换成了是否问题,即是否存在一种虚拟机部署方案满足时延为 κ 的上界。

将二维装箱问题规约于该问题。假设箱子底边为 D,要装入箱子的小盒子数量为 $\tilde{N}$,盒子底边为 d_j,高为 h_j,放入箱子时盒子不可以翻转,问是否存在一种装箱方法使得将所有的小盒子装入箱子后高度小于等于给定上界 $\tilde{H}$,该问题可以表示为

$$\Pi(\tilde{N},D,d_j,h_j) \leqslant \tilde{H} \tag{8}$$

对比式(7)(8)发现小盒子数量可以归约为虚拟机的数量,小盒子底边和高可以分别规约为需求资源数量和时延,而箱子底边可以规约为总的物理机数量。因此,从以上证明二维装箱问题可以在多项式时间内规约到要证明问题,说明要证问题为一个 NPC。

3 算法设计

3.1 算法概述

由于遗传算法常被用来解决组合优化问题[13~15],所以本文所提的虚拟机部署算法主要是基于遗传算法来实现的。然而与传统的遗传算法不同,GGA 算法在虚拟机部署时结合了贪心思想选择要部署的目标服务器,同时设计了一种交叉个体检查方法避免了交叉冲突。因此,GGA 算法的主要步骤为:

a)初始化基于贪心思想生成大小为 $|P|$ 的种群,种群内每个个体对应一个部署方案,长度为 $|M|$。

b)从种群中按照随机概率选择个体两两比较后保留适应度较好的个体形成种群。

c)将种群内的奇数位置的个体与偶数位置的个体以随机进行交叉,并对交叉后的新个体检查是否冲突。

d)对新个体随机选择一个变异位,保留种群中最优个体。

e)反复执行 a)~d)达到规定迭代次数,算法结束输出结果。

3.2 算法设计细节

提出算法的核心思想是基于遗传算法,算法的第一步就是进行初始化并对个体进行编码形成种群。在这里采用的编码方式主要是先随机生成虚拟机的部署序列如图 2 所示。

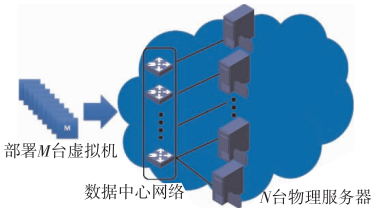

图 1 云数据中心虚拟机部署示意图

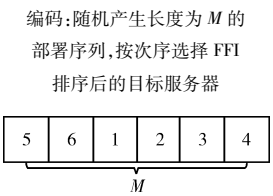

图 2 GGA 编码方式示意图

由于网络带宽的受限,虚拟机网络迁移时延大于配置时延,所以采用 FFI (first-fit increasing)贪心思想,即按式(1)计算得到虚拟机到目标服务器的部署时延,把可部署的目标服务器部署时延由小到大排列,根据虚拟机多种资源的需求和容量限制,按照 FFI 思想依次将虚拟机部署到目标服务器形成长度为 $|M|$ 的个体,反复执行该步骤形成个体大小为 $|P|$ 的种群。

之后,从种群中两两选择个体,保留适应度较好的个体。这里的适应度就是个体中虚拟机到目标服务器的总体部署时延 Γ,若 Γ 越短,说明适应度越高个体越好。接着对种群内奇数位的个体遍历并生成[0, 1]的随机数,把随机数小于 P_c(交叉概率)的个体与其下一个偶数位的个体交叉。这样做的好处是可以增加新的交叉个体,而且避免个体间重复交叉。由于在交叉的过程中会存在交叉冲突,出现虚拟机对资源的需求超过目标服务器容量的问题,所以在这一步中设计了一种冲突检查方法,若交叉后虚拟机无法部署到目标服务器,则待其他虚拟机部署完毕后,将该虚拟机到目标服务器部署时延最小,且满足虚拟机资源需求的目标服务器。

完成交叉操作后进入变异过程,遍历所有个体生成[0, 1]随机数,把随机数小于 P_m(变异概率)进行变异,这里的变异操作是随机地对个体中某一位进行变异,然后根据 FFI 将虚拟机部署至时延最小且满足虚拟机资源需求的目标服务器。

所有操作完成后对种群中所有个体计算适应度并进行比较,保留本代中最好的个体。反复迭代上述过程,当最大迭代次数为规定的迭代次数 $|G|$ 后算法停止,并输出最佳的个体,即虚拟机部署方案 VPSchedule。具体的算法步骤如下:

算法 1 GGA 算法

输入:虚拟机数量 $|M|$,目标服务器数量 $|N|$,网络带宽 w。

输出:虚拟机目标服务器的部署方案 VPSchedule。

```
参数初始化
调用 FFI 算法
for i < M
    for j < N
        计算 M 个虚拟机通过 L 条通信链路到 N 台服务器的部署时延
    end for
end for
FFI 算法将 M 个虚拟机对应的 N 台目标服务器部署时延按照从小到大进行排列。随机产生虚拟机调度序列,按照顺序和 FFI 算法排序结果形成种群。
for i < |G|
for j = 1 : 2 : |P| - 1
    个体两两选择,保留适应度较好的一组
end for
保留最优个体
for j = 1 : 2 : |P| (选择奇数)//交叉操作
    if 个体产生的随机数 < Pc
        奇数位和偶数位个体交叉
        if 交叉基因冲突
            冲突处理
```

```
        end if
    end if
end for
保留最优个体
for j = 1 : 1 : |P|//变异操作
    if 个体产生的随机数 < P_m
        随机对个体的某一位变异
    end if
end for
保留最优个体
for j = 1 : |P|//群体数据更新
        更新个体适应度
        if 个体比当前最优优秀
            VPSchedule←更新最优个体
        end if
    end for
end for
```

3.3　算法分析

GGA 的主要思想是基于贪心策略的基因遗传算法,所以在此将对每个步骤的算法复杂度进行分析。首先分析贪心策略 FFI 由于虚拟机的数量为 M,物理服务器的数量为 N,有 L 条通信链路,所以 FFI 排序算法时间复杂度为 $O(N \cdot M \cdot L)$。接着再分别对选择、交叉、变异操作复杂度进行分析。在选择操作中种群大小为 $|P|$,将种群个体均分进行两两互不重复的比较保留适应度较好的个体,其算法的复杂度为 $O(P^2)$。在交叉操作时是对种群内奇数位个体与偶数位个体按照交叉概率进行交叉形成新个体,所以在最坏的情况下,假设所有偶数位个体被交叉,则该操作的最坏复杂度为 $O(P^2)$;在最好的情况下只有一个个体被交叉,那么复杂度仅为 $O(P)$。而对于变异操作主要是对种群内个体某个位进行变异,算法复杂度最坏为 $O(M)$。

基于以上各个步骤,对 GGA 算法复杂度进行分析可以发现:当种群中个体数为 P,算法规定的迭代次数为 G,所以 GGA 算法在最坏的情况下复杂度为:$O(G \cdot (NML + 2P^2 + M))$。

4　实验仿真

4.1　实验参数设置

采用 MATLAB r2010b 对算法进行了实验仿真,实验主机为 Lenovo ThinkPad PC,系统 Windows 8,CPU 配置 i3 2.40 Ghz 处理器,内存 6 GB。使用 MATLAN r2010b 版本实现。其中 GGA 的交叉概率 P_c 为 0.8,变异概率 P_m 为 0.05。其中涉及的相关参数采用均匀随机分布的方式产生包括链路带宽和虚拟机数据量取整数为[1, 10],虚拟机资源需求和服务器容量在计算资源、内存资源和存储资源分别为([1,100], [1, 50], [1, 100])和([1,300], [1, 150], [1, 300])。表 1 中列出了不同虚拟机和服务器数量的三组数据。

4.2　比较的算法

将 GGA 分别和贪心算法(greedy)、粒子群优化(PSO)算法、文献[8]算法 VMPDN 作比较:

贪心算法首先将虚拟机预部署到 N 台目标服务器,选择总体部署时延最小的一组作为预分配方案,反复执行当把 M 台虚拟机部署到服务器后得到 $M \times N$ 组虚拟机—服务器预分配方案,保留部署时延最小的一组并最终将该虚拟机部署到服务器,接着再对剩余的虚拟机执行上述操作,当把所有的虚拟机都部署到服务器后算法输出结果形成部署方案。该算法的时间复杂度为 $O(M \cdot N)$。

粒子群优化算法的基本思想是,固定大小的粒子群通过记录自己和群体离目标的最近的距离,学习有这样的距离时群体最优和自己最优的速度,调整自己的速度,以达到最后群体都能向目标靠拢的目的。算法中 PSO 的惯性权重 $w = 0.8$,迭代次数 $G = 50$,全局/部分参考参数:$C_1 = C_2 = 2$。PSO 算法复杂度为 $O(P \cdot G \cdot M)$(符号意义与 GGA 类似)。

文献[8]中提出虚拟机到数据节点的部署算法是针对单资源的,所以本文对算法进行了改进,可以处理多类型的资源问题 MVMPDN。主要思想还是阈值法,阈值 $\hat{t}$ 的确定会受虚拟机多种资源需求的影响,通过这种方式每轮将资源需求最小的虚拟机部署到资源最丰富物理机上,从而形成虚拟机部署的优化方案。

4.3　算法结果比较

4.3.1　性能比较

本组实验的数据中不改变 PSO 和 GGA 的迭代次数和种群大小,在三组数据中 PSO 和 GGA 的迭代次数均为 $G = 50$,$P = 10$。实验结果采用均值的方法取 10 次实验后的平均结果,算法性能比较结果如下。

表 1　虚拟机、服务器数量

	虚拟机数量	服务器数量
数据 1	10	6
数据 2	50	30
数据 3	200	120

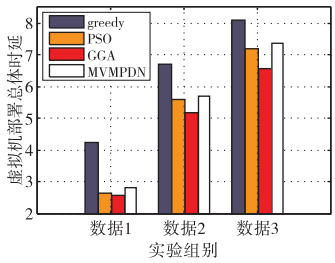

图 3　四种算法虚拟机部署总体时延比较示意图

表 2　四种算法时间复杂度比较

算法名称	算法平均运行时间/s		
	数据 1	数据 2	数据 3
greedy	0.001 566	0.005 239	0.008 655
PSO	0.122 6	0.570 9	2.474 2
GGA	0.127 7	0.589 6	2.566 5
MVMPDN	0.002 577	0.006 588	0.013 567

图 3 将四种算法的虚拟机部署总体时延进行了比较。从图中可以明显发现,greedy 算法得出总体时延要远远大于 PSO、GGA、MVMPDN,这是因为 greedy 算法只是得到了一个局部最优解,相比之下虽然 MVMPDN 总时延好于 greedy,但是相比于 PSO 和 GGA 由于 MVMPDN 中阈值选择是以虚拟机资源需求最小一组确定的,其空间搜索范围较小,所以虽然优于 greedy 但是总体时延不如 PSO 和 GGA。当数据规模输入较小(数据 1)时,PSO 和 GGA 得到总体部署时延相当,但是随着输入规模的扩大,由于在 GGA 中采用了基于贪心思想的 FFI 所以更加容易收敛到最优解,而 PSO 中没有采用此种策略造成搜索空间大影响了最终结果。

从表 2 可以观察到,算法时间复杂度 greedy 和 MVMPDN 运行时间较短,PSO 算法和本文的提出的 GGA 算法花销时间相当,并且随着虚拟机数量和目标服务器的增加运行时间增大。

4.3.2　资源类型数量对算法的影响

a)图 4 主要是考察了当资源类型不同时对算法性能的影响。在这里本文将资源类型数分为三种,类型 1、2、3 分别表示资源类型数为 1、2、3,并且将三组资源类型数分别在三组数据

上运行,从得到的结果可以发现:

b)当资源类型数最少的时候,MVMPDN 得到最优时延和 GGA 相差不多,而随着资源数量的增加其获得的最优时延要大于 GGA,这是因为 MVMPDN 的设计思想主要是基于单类型资源的,无法处理多类型资源时的情况。相比之下,无论在哪一种情况下 GGA 都能获得最佳的部署时延,而缺乏足够的搜索范围和灵活变化功能的贪心算法获得的时延最差。

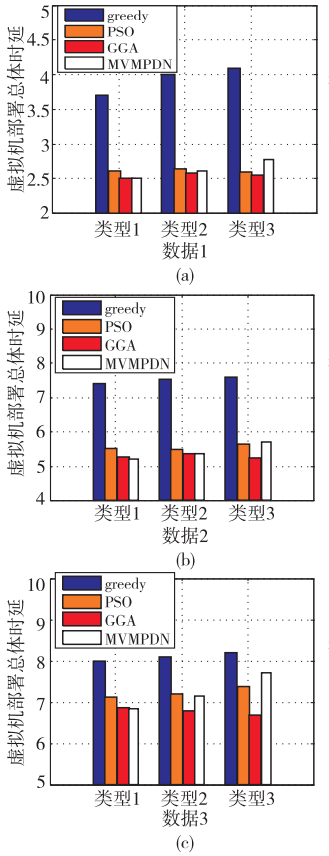

图 4 资源类型数量对算法性能的影响示意图

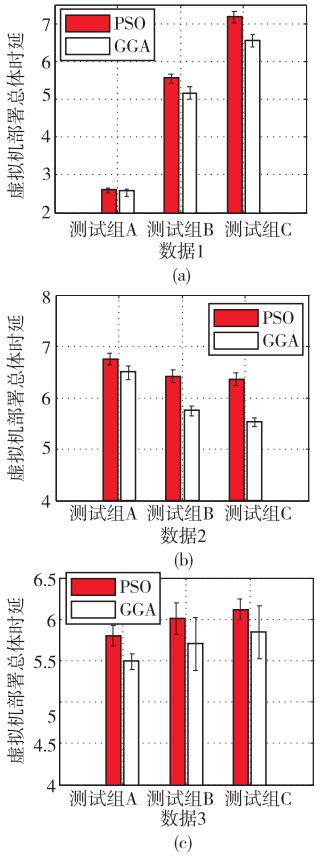

图 5 三个测试组在不同数据集中得到的虚拟机部署总体时延示意图

4.3.3 迭代次数和种群大小对算法的影响

接下来本文还考察了当不同的迭代次数,种群大小测试时对 PSO 和 GGA 算法的影响,如表 3 所示。

表 3 种群大小和迭代次数测试数据

组别	*P* 种群大小	*G* 迭代次数
测试组 A	10	50
测试组 B	20	100
测试组 C	40	200

图 5 展示了当迭代轮次和种群个体数目不同时候,分别对三组数据 1 – 3 的测试,从中可以观察到,当 *G* 和 *P* 相同时,GGA 相对 PSO 总能找到更好的近似最优解。这是因为 PSO 算法每次迭代的过程中为了得到最佳的个体,算法在解空间中的搜索范围过大无法最终收敛到最优个体。而 GGA 算法中由于采用了贪心思想,为个体形成最优解有指导作用,并且交叉和变异过程增减了个体的多样性,所以提出算法相对 PSO 更容易搜索到最优解,从而最终确定最佳的虚拟机部署方案。

表 4 列出了三种算法的时间复杂度,可以观察由于算法复杂度都与最大迭代次数、虚拟机数目、物理机数量等因素有关,所以当迭代次数和种群个体数目相同的时候,三种算法在不同测试组和数据组其运行时间比较接近。

表 4 种群大小和迭代次数不同时算法的平均复杂度 /s

	数据 1		数据 2		数据 3	
	PSO	GGA	PSO	GGA	PSO	GGA
测试组 A	7.179 0	6.808 1	6.720 8	6.469 2	6.089 4	6.083 6
测试组 B	6.827 1	6.490 4	6.424 9	5.941 0	6.048 4	5.844 2
测试组 C	6.767 5	6.540 0	6.350 0	5.421 2	5.854 2	5.505 6

5 结束语

针对云数据中心内虚拟机部署的时延优化问题,本文提出了基于遗传算法的虚拟机部署算法 GGA,该算法主要结合了贪心思想有效地指导最优部署方案的形成。其中的贪心算法主要是 FFI,将虚拟机部署到服务器的部署时延按照由小到大排列,每次选择时延最小的服务器部署形成个体。同时还利用遗传算法中交叉和变异,增加了解空间个体形成的多样性。通过实验,将 GGA 分别与文献[8]、经典的 PSO 算法比较后发现:GGA 算法不仅最小化虚拟机部署的时延,而且算法运行时间合理,为云数据中心虚拟机部署的研究提供了理论依据和实现方法。

参考文献:

[1] Li Bo, Li Jianxin, Huai Jinpeng, *et al*. EnaCloud. an energy-saving application live placement approach for cloud computing environments [C]//Proc of IEEE International Conference on Cloud Computing. 2009: 17-24.

[2] 刘鹏程, 陈榕. 面向云计算的虚拟机动态迁移框架[J]. 计算机工程, 2010, 36(5): 37-39.

[3] 胡元元, 林浒, 李鸿彬. IaaS 云中最小迁移代价的虚拟机放置算法[J]. 小型微型计算机系统, 2014, 35(4): 878-882.

[4] Xu Jing, Fortes J A B. Multi-objective virtual machine placement in virtualized data center environments[C] //Proc of IEEE/ACM International Conference on Green Computing and Communications & International Conference on Cyber, Physical and Social Computing. 2010: 179-188.

[5] 张彬彬, 罗英伟, 汪小林, 等. 虚拟机全系统在线迁移[J]. 电子学报, 2009, 37(4): 894-899.

[6] 李强, 郝沁汾, 肖利民, 等. 云计算中虚拟机放置的自适应管理与多目标优化[J]. 计算机学报, 2011, 34(12): 2253-2264.

[7] Alicherry M, Lakshman T V. Optimizing data access latencies in cloud systems by intelligent virtual machine placement[C]//Proc of IEEE INFOCOM. 2013: 647-655.

[8] Kuo J J, Yang H H, Tsai M J. Optimal approximation algorithm of virtual machine placement for data latency minimization in cloud systems[C]//Proc of INFOCOM. IEEE Gouference on Computer Communications. 2014.

[9] Tso F P, Oikonomou K, Kavvadia E, *et al*. Scalable traffic-aware virtual machine management for cloud data centers[C]//Proc of the 34th International Conference on Distributed Computing Systems. 2014:238-347.

[10] Li Xin, Wu Jie, Tang Shaojie, *et al*. Let's stay together: towards traffic aware virtual machine placement in data centers [C] //Proc of IEEE INFOCOM. 2014:1482-1850.

[11] Le K, Bianchini R, Zhang Jingru, Meng J, *et al*. Reducing electricity cost through virtual machine placement in high performance computing clouds[C]//Proc of International Conference for High Performance Computing, Networking, Storage and Analysis. New York: ACM Press, 2011.

[12] Lin Hao, Qi Xin, Yang Shuo, *et al*. Workload-driven VM consolidation in cloud data center[C] //Proc of the 29th International Parallel and Distributed Processing Symposium. 2015: 207-216.

[13] 赵振勇, 王力, 王保华, 等. 遗传算法改进策略的研究[J]. 计算机应用, 2006, 26(S2): 189-191.

[14] 马永杰, 云文霞. 遗传算法研究进展[J]. 计算机应用研究, 2012, 29(4): 1201-1206.

[15] 何大勇, 查建中, 姜义东. 遗传算法求解复杂集装箱装载问题方法研究[J]. 软件学报, 2001, 12(9): 1380-1385.