基于多目标混合粒子群算法的

虚拟机部署问题研究

邵旖洁

（东南大学网络空间安全学院）

**摘 要** 针对虚拟机部署问题，为将虚拟机部署在适当的物理机上，达到负载均衡的目标，本文提出了一种基于多目标混合粒子群算法的部署策略。本文以各物理机的CPU占用率的标准差和内存占用率的标准差来反映负载均衡状况，为同时优化两个目标，在算法中引入了Pareto理论，最终可得到一组非支配解集，并且在算法中引入遗传算法中的交叉操作和变异操作以及模拟退火算法中的温度系数概念以避免陷入局部最优解。经实验验证，与其它算法相较，本文算法能够使得各物理机之间资源占用率的标准差同时达到更小，实现负载均衡的效果。

**关键词** 多目标优化；混合粒子群算法；虚拟机部署；负载均衡

Research on Virtual Machine Deployment Based on Multi-Objective Hybrid Particle Swarm Optimization

Shao Yijie

(School of Cyber Science and Engineering, Southeast University)

**Abstract** In order to deploy virtual machines on appropriate physical machines to achieve the goal of load balancing, a virtual machine deployment strategy based on multi-objective hybrid particle swarm algorithm was proposed. In this paper, the standard deviation of the CPU usage rate and the standard deviation of the memory usage rate of each physical machine is used to reflect the load balancing situation. To optimize two goals at the same time, the Pareto theory is introduced into the algorithm, and finally a non-dominated solution sets can be obtained. Moreover, the crossover operation and mutation operation in the genetic algorithm and the temperature coefficient concept in the simulated annealing algorithm are introduced into the algorithm to avoid falling into the local optimal solution. By experiment, compared with other algorithms, the algorithm in this paper can make the standard deviation of resource occupancy between physical machines smaller at the same time, and achieve the effect of load balancing.

**Key word** Multi-objective optimization; Hybrid particle swarm optimization; Virtual machine deployment; Load balancing

# 引言

随着对计算资源需求的日益增长，云计算服务应运而生。云计算可以为用户提供动态的、可扩展的虚拟资源。虚拟化技术是云计算中的一个关键技术，它将物理服务器虚拟化为多个逻辑机器。每台逻辑机器都可以运行不同的操作系统，称为虚拟机，用户可以在虚拟机上部署自己的应用程序[1]。虚拟机和物理机之间的映射是近年来广泛研究的主题，如何合理地将虚拟机部署在适当的物理机上，使得每个物理机之间的资源得到充分利用，以达到负载均衡的效果，这对于当前云计算服务和虚拟化技术研究具有重要意义。

由于虚拟机部署问题是一个NP难的组合优化问题，基于简单规则的方法能够快速得到解决方案，但是不能保证该方案是最优解[2]，因此已有较多研究采用人工智能算法来解决该问题。文献[3]提出了一种基于遗传算法的虚拟负载均衡技术，证明了算法结果优于传统策略；文献[4]提出了一种多目标蚁群系统算法使得总资源浪费和功耗降至最低；文献[5][6]提出了基于粒子群算法的虚拟机调度策略，验证了该算法的有效性，但粒子群算法在处理复杂的问题时，仍然存在缺陷，本文在粒子群算法的基础上加以改进。

本文提出了一种多目标混合粒子群算法来优化虚拟机部署问题。首先，对虚拟机部署问题进行建模，通过各物理机的各项性能指标的标准差来反映负载均衡状况，本文选取的优化的目标有2个，包括各物理机之间CPU占用率的标准差以及内存占用率的标准差。其次，为解决多目标问题无法实现所有的优化目标都同时达到最优的情况，本文算法引入了Pareto理论和外部存储库来改进粒子群算法。再者，为避免粒子群算法陷入局部最优解，本文算法引入遗传算法中的交叉操作和变异操作以及模拟退火算法中的温度系数概念，以进一步改进粒子群算法。相较于其它算法，该多目标混合粒子群算法对于虚拟机部署问题更能接近全局最优解，使得各物理机之间CPU占用率的标准差以及内存占用率的标准差同时达到更小，实现负载均衡的效果。

# 虚拟机部署问题的建模

对于将台虚拟机部署到台物理机的问题进行建模，虚拟机集合为，第台虚拟机的资源占用率需求记为、，其中为该虚拟机的CPU占用率需求，为该虚拟机的内存占用率需求；物理机的集合记为，第台物理机的资源占用率记为、，其中为该物理机的CPU占用率，为该物理机的内存占用率。用表示将第台虚拟机部署到第台物理机上，不部署则，表示每台虚拟机只能部署在一台物理机上，而每台物理机可以部署多台虚拟机。

对于第台物理机，某种资源占用率为当前某种资源占用率与部署在其上的所有虚拟机所需某种资源之和。记第台物理机的当前资源占用率为、，其中为该物理机当前的CPU占用率，为该物理机当前的内存占用率。

则第 j 台物理机的CPU占用率为（2.1）：



（2.1）

第 j 台物理机的内存占用率为（2.2）：



（2.2）

记台物理机之间CPU占用率的标准差为（2.3）



（2.3）

记台物理机之间内存占用率的标准差为（2.4）



（2.4）

若要达到负载均衡，即求标准差和标准差最小化，因此模型可以表示为：

目标函数：



（2.5）



（2.6）

约束条件：



（2.7）



（2.8）



（2.9）

其中式（2.7）表示每台虚拟机只能部署在一台物理机上，式（2.8）、（2.9）表示物理机的资源占用率不能超过100%。

# 多目标混合粒子群算法

## 3.1 粒子群算法

粒子群算法（Particle Swarm Optimization, PSO）由Kennedy和Eberhart于 1995 年提出[7]。该算法的基本概念源自于鸟类觅食行为的研究，具有计算简单、收敛快、顽健性强等优点，因此在求解较复杂的问题时展现出良好的优越性。

该算法的基本思想是在一个维搜索空间中，每个优化问题的解作为搜索空间中的一个粒子，个粒子构成种群，其中,表示第个粒子在维搜索空间中的位置。每个粒子都有一个取决于被优化的函数的适应度函数和一个速度决定它们飞翔的方向和距离。

首先初始化为一群随机粒子，然后通过迭代找到最优解。在每一次迭代过程中，粒子通过跟踪个体极值和全局极值来更新自己的位置，更新公式如（3.1）、（3.2）所示。



（3.1）



（3.2）

其中，；为惯性系数；和称为加速因子，分别称为认知学习率和社会学习率；是分布于的随机数。

## 3.2 算法粒子编码

在应用粒子群算法处理虚拟机部署问题时，合适的编码方式至关重要。粒子编码是为了将问题的解和粒子位置相映射，将虚拟机简化成粒子并相应编码的过程[8]。本文采用多值编码，假设有台虚拟机和台物理机，第个粒子表示问题的一个解，例如，表示第1台虚拟机部署在第1台物理机上，第2台虚拟机部署在第3台物理机上，以此类推。

## 3.3 混合粒子群算法

为了避免陷入局部最优解，可以将遗传算法的交叉操作和变异操作应用到粒子群算法中，称为混合粒子群算法[9]。

### 3.3.1 交叉操作

交叉操作在遗传算法中是把两个父代个体的部分结构加以替换重组从而生成新个体的操作。在混合粒子群算法中，交叉操作是对当前个体与全局极值进行操作，采用整数交叉法，随机选择位置进行交叉操作。

例如粒子和粒子，假设随机选择位置为2、4、5，则通过交叉操作得到的新粒子个体为。

### 3.3.2 变异操作

变异操作在遗传算法中是指对群体中的个体串的某些基因座上的基因值作变动，可维持群体多样性。在混合粒子群算法中，变异操作是指在群中随机选择一个粒子个体，对于该粒子随机选取位置，并在粒子位置范围内随机选取值进行变异操作。

例如粒子，假设随机选择位置为2、4、5，在粒子位置范围内随机选取值为1、3、5，则通过交叉操作得到的新粒子个体为。

### 3.3.3 温度系数

本文引入了模拟退火算法中温度系数的概念[10]，用于在更新粒子位置时对交叉或者变异操作进行选择。

温度系数随着迭代次数的增加而逐步减小，在迭代过程中当前温度的计算公式见（3.3）。



（3.3）

其中为在当前迭代次数下的温度系数；为初始温度系数；为迭代结束时的温度系数；为当前迭代次数；为总迭代次数。

温度系数同时影响选择交叉操作或变异操作的概率以及操作位置个数。在迭代过程初期，温度系数较高，为保证粒子群的多样性，此时应该以较大概率选择变异操作，且对应的操作位置个数应较多。随着迭代次数的增加，温度系数将有所下降，群体最优解也逐渐趋近于最优，此时应该以较大概率选择与最优解的交叉操作，且对应的操作位置个数应较多。

## 3.4 多目标混合粒子群算法

对于多目标问题的求解，若粒子个体的有些目标更好，但有些目标更差，就无法严格对比得出极值，因此我们引入了Pareto理论和外部存储库来解决多目标问题[11]。

### 3.4.1 Pareto理论

Pareto解又称非支配解（Non-dominated Set），其定义为：对于多目标问题，假设任何二解和，若对所有目标而言，均优于，则称支配；若没有被其它解所支配，则称为非支配解，也称为Pareto解。一组目标函数最优解的集合称为Pareto最优集，最优集在空间上形成的曲面称为Pareto前沿面。

### 3.4.2 多目标混合粒子群算法的实现

多目标混合粒子群算法的输入为虚拟机性能参数矩阵、物理机性能参数矩阵、粒子的个数，惯性系数、加速因子和，初始温度系数，结束温度系数，总迭代次数，输出为Pareto解集。

多目标混合粒子群算法的实现步骤为：

1. 初始化群体和Archive集；
   1. 随机生成个粒子的位置，初始化粒子速度为0;
   2. 根据式s计算每个粒子的适应度；
   3. 初始化每个粒子的个体极值为当前粒子的解；
   4. 求出当前非支配解集，并存入外部存储Archive集中；
   5. 对于每一个粒子，从Archive集中随机选择出一个解作为全局极值。
2. 根据式（3.1）和（3.2）更新粒子位置和速度；
3. 根据式（3.3）计算当前温度系数，并根据温度系数选择交叉操作或变异操作，得到新解；
4. 根据式（2.3）和（2.4）计算当前解的适应度；
5. 更新个体极值

对于当前解和历史个体极值进行非支配关系评估，若当前解能够完全支配历史最优解，则更新；反之，若历史最优解能够完全支配当前解，则不更新；若两者互不支配，则随机选择一个作为个体极值。

1. 更新Archive集；
   1. 计算当前粒子群的非支配解集，并存入外部存储Archive集中；
   2. 对于Archive集进行筛选，去除受其它解完全支配的解。
2. 对于每一个粒子，从Archive集中随机选择出一个解作为全局极值；
3. 迭代次数加1，若迭代次数小于，则跳转到步骤（2），否则算法结束，Archive集即为最终得到的Pareto解集。

# 仿真实验及其结果分析

## 4.1 实验准备

本仿真实验设计将50台虚拟机部署到25台物理机上，为模拟虚拟机部署的不同情形，在每一次仿真实验中每台虚拟机的CPU需求占用率与内存需求占用率都是采用间的随机数。

## 4.2 实验验证分析

为了验证多目标混合粒子群算法在部署虚拟机负载均衡问题上的有效性，将多目标混合粒子群算法、多目标普通粒子群算法以及一些基于简单规则的虚拟机部署算法，如最小连接数法（选择运行虚拟机数目最少的物理机）以及最小负载法（选择负载最低的物理机）进行对比。

多目标混合粒子群算法参数设置为：粒子的个数，惯性系数、加速因子、，初始温度系数，结束温度系数，总迭代次数。

首先比较多目标混合粒子群算法与多目标普通粒子群算法所得的Pareto解，结果如图 4.1所示：

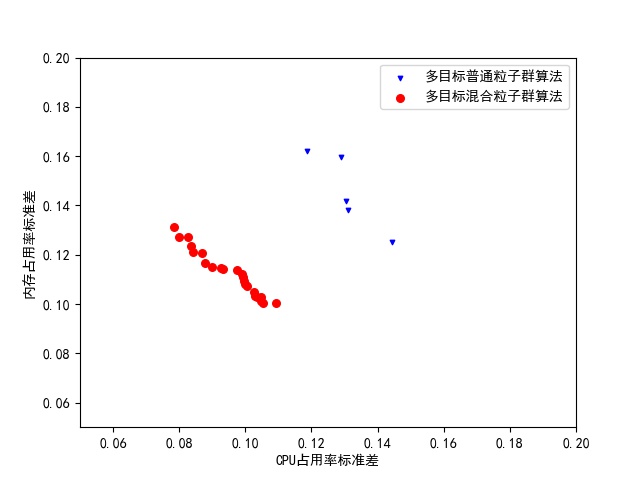


图 4.1多目标混合粒子群算法与多目标普通粒子群算法所得的Pareto前沿面

由图 4.1可以看出，混合粒子群算法的非支配解集优于普通粒子群算法的非支配解，即所得解的CPU标准差与内存标准差均小于普通粒子群算法的所得解。由此说明混合粒子群算法可以避免陷入局部最优解，搜索效果良好。

其次，使用各方法分别对100组数据进行试验，比较各方法所得结果的各物理机的CPU占用率标准差和内存占用率标准差之和，结果如图 4.2所示：

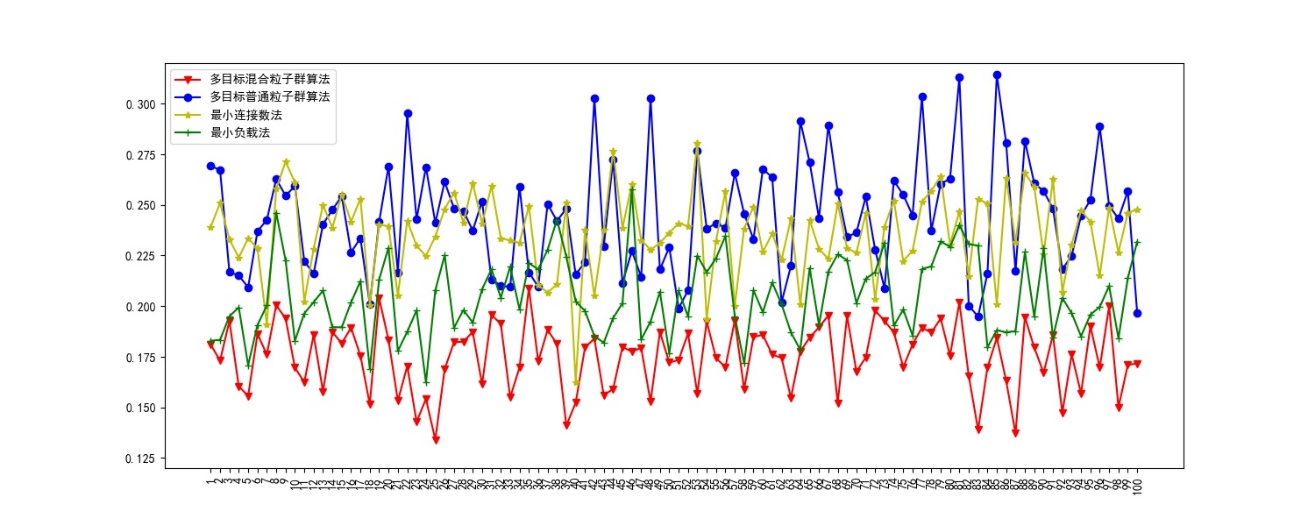


图 4.2 各虚拟机部署方法的两种性能的标准差之和

在100次试验中，各物理机的资源占用率标准差的均值如表 4.1所示：

表 4.1 各物理机资源占用率标准差的均值

|  |  |  |
| --- | --- | --- |
|  | CPU占用率标准差 | 内存占用率标准差 |
| 多目标混合粒子群算法 | 0.086 | 0.089 |
| 多目标普通粒子群算法 | 0.122 | 0.122 |
| 最小连接数法 | 0.118 | 0.118 |
| 最小负载法 | 0.101 | 0.104 |

从图 4.2和表 4.1可以看出，改进后的多目标混合粒子群算法虽然具有一定的随机性，但是与其它三种方法相比，更能接近最优解，在虚拟机部署达到负载均衡的问题上表现更好。

# 结论及下一步工作

在本文中，我们对虚拟机部署问题进行建模，并应用多目标混合粒子群算法以达到负载均衡的目标。在粒子群算法的基础上，引入Pareto理论来解决多目标优化问题，引入遗传算法中的交叉操作和变异操作以及模拟退火算法中的温度系数概念来避免陷入局部最优解。经实验验证，本文算法在虚拟机部署负载均衡方面优于普通粒子群算法和基于简单规则的部署算法。

本文主要考虑虚拟机的静态部署问题，运行一次多目标混合粒子群算法耗时远高于基于简单规则的部署策略，在下一步工作中，希望能够优化算法的收敛速度，将算法应用于动态部署虚拟机，满足时延要求。

**参考文献**

1. S. Anish Babu, M. J. Hareesh, John Paul Martin.System Performance evaluation of Para virtualization, Container virtualization and Full virtualization using Xen, OpenVZ and XenServer[C].Proceedings of the fourth International Conference on Advances in Computing and Communications, 2014, pp. 247-250.
2. Caili Wen, Wenyi Jiang. Research on Virtual Machine Layout Strategy Based on Improved Particle Swarm Optimization Algorithm[C]. 2019 IEEE 21st International Conference on High Performance Computing and Communications,2019.
3. Mala Yadav, Jay Shankar Prasad. An Enhanced Genetic Virtual Machine Load Balancing Algorithm for Data Center[J]. Communications in Computer and Information Science,2019, v1045, p244-253.
4. Yongqiang Gao, Haibing Guan, Zhengwei Qi, et.al. A multi-objective ant colony system algorithm for virtual machine placement in cloud computing[J]. Journal of Computer and System Sciences,2013,79:1230-1242.
5. D. Kumar, Z. Raza. A PSO Based VM Resource Scheduling Model for Cloud Computing[C]. 2015 IEEE International Conference on Computational Intelligence & Communication Technology, Ghaziabad, 2015, pp. 213-219.
6. 杨靖,张宏军,赵水宁, 等.基于粒子群优化算法的虚拟机部署策略[J].计算机应用,2016,36(1):117-121.
7. J Kennedy, R Eberhart. Particle Swarm Optimization[C].IEEE International Conference on Neural Networks, 1995, vol.4.
8. 陈强,刘彩霞,李凌书.基于粒子群优化算法的5G网络切片功能迁移机制[J].网络与信息安全学报,2018,4(08):47-55.
9. R.J. Kuo, Y.J. Syu, Zhen-Yao Chen, et.al. Integration of particle swarm optimization and genetic algorithm for dynamic clustering. Information Sciences, 2012,195:124-140.
10. BERTSIMAS D, TSITSIKLIS J. Simulated Annealing[J].Statistical Science,1993,8(1):10-15.
11. Carlos A. Coello. Handling Multiple Objectives With Particle Swarm Optimization[C]. IEEE Transactions On Evolutionary Computation, 2004,VOL. 8, NO. 3.

# 附件-问题回答

**问题1：**如何判定粒子群算法求解问题时可能陷入局部最优？出现局部最优的因素会是什么？

**回答：**（1）可以通过测试函数来判定粒子群算法求解问题时可能陷入局部最优解，本文选取ZDT系列测试函数，该系列测试函数为双目标函数，实验结果如图 6.1所示：

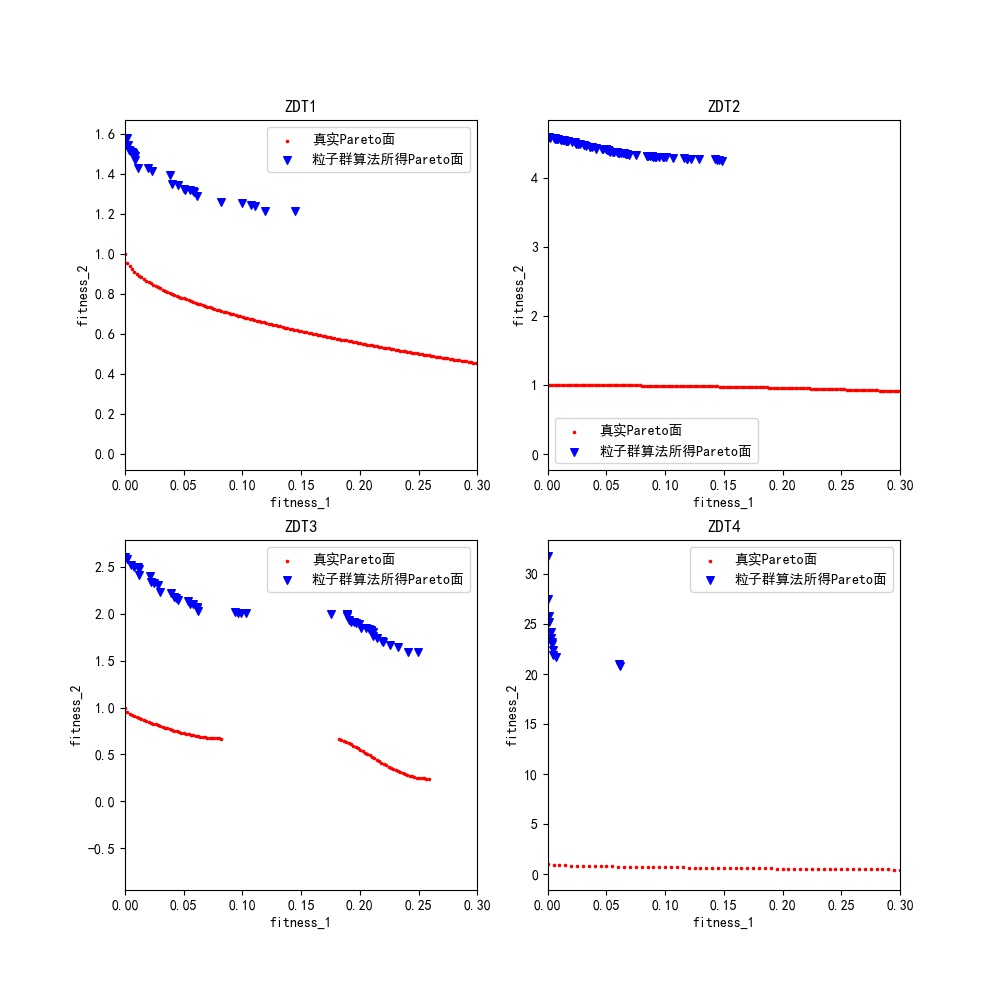


图 6.1 真实Pareto前沿面与粒子群算法所得Pareto前沿面比较

由图 6.1可知，粒子群算法所得Pareto前沿面与真实Pareto前沿面有一定差距，即粒子群算法没有求得最优解，由此可以判断粒子群算法陷入了局部最优解。

在实际虚拟机部署问题中，由于粒子初始位置具有随机性，若粒子初始位置为局部最优解，在随后的迭代过程中粒子将跟踪该局部最优解更新自己，就可能使算法陷入局部最优。因此，我们可以改变粒子位置的初始化方式，即在粒子初始位置中加入不同局部最优解。我们对粒子群算法进行两次实验，每次实验中初始化过程中加入的局部最优解如表 6.1所示，实验结果如图 6.2所示：

表 6.1两次实验中初始化过程中加入的局部最优解

|  |  |  |
| --- | --- | --- |
|  | CPU占用率标准差 | 内存占用率标准差 |
| 实验1 | 0.114 | 0.126 |
| 实验2 | 0.081 | 0.102 |

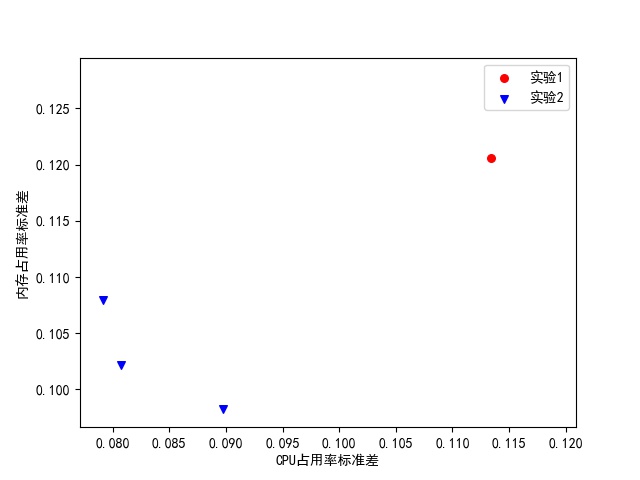


图 6.2 两次实验的Pareto前沿面

由图 6.2可知，由于粒子群算法陷入局部最优，无法跳出，所以对于同一数据所得问题的最优解不同。

（2）粒子群算法出现局部最优的因素：

在粒子群算中，粒子通过跟踪个体极值和全局极值来更新自己的位置，若全局极值陷入局部最优，如图 6.3所示：

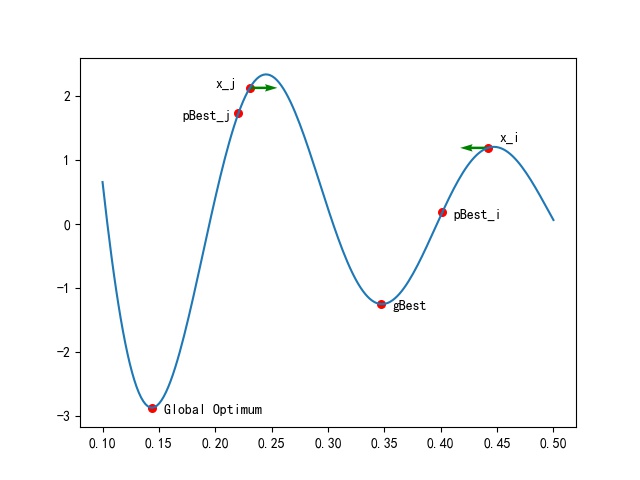


图 6.3 全局极值陷入局部最优

对于粒子，由于和都位于局部最优区域内，所以粒子将向局部最优方向移动。而对于，尽管和都位于局部最优区域之外，但是由于全局极值所驱动，粒子仍然会向局部最优方向错误地移动。

由此可知，若出现全局极值陷入局部最优的情况，粒子会被吸引至局部最优区域，最终只能获得局部最优解

**问题2：**本文的双目标（CPU占用率和内存占用率）之间存在何种冲突，从而需要考虑引入Pareto理论？能否转换为一个目标？为什么？

**回答：**（1）由于每台虚拟机所需的CPU占用率和内存占用率并不成比例，即存在一台虚拟机所需的CPU占用率极高而所需内存占用率极低或者相反的情况，并且无法判断何者处于更重要的地位，因此两者之间存在双趋冲突，即面对同时出现的两个各有优劣的目标进行选择时难以取舍，例如有两个解如表 6.2所示：

表 6.2 两个解双目标比较

|  |  |  |
| --- | --- | --- |
|  | CPU占用率标准差 | 内存占用率标准差 |
| 解1 | 0.116 | 0.096 |
| 解2 | 0.123 | 0.092 |

解1的CPU占用率标准差优于解2，但内存占用率标准差劣于解2，因此无得出哪个解更优，从而需要考虑引入Pareto理论。

（2）若想要转换为一个目标，可以考虑以下方法，但以下方法各有缺点：

1）权重法：



(6.1)

如式(6.1)所示，可以通过权重赋值的方法将虚拟机的多个性能指标整合为一个目标，但是此法的权重系数选取困难，具有非常强的主观性，且每个目标的量纲不同，要精确的给出所有目标函数的标量信息，就必须有每一个目标的全局先验知识，这实现起来非常困难。

2）约束法

对于多个目标，若能够确定一个主要的目标，而其它目标只要满足一定的约束条件即可，此方法与权重法相同，难以选择主要目标，具有非常强的主观性。

3）距离法

我们可以将粒子个体解与理想值之间的距离来作为其适应值，但是这个理想值难以确定。

**问题3：**在本文中，如何以多目标粒子群优化算法为主体，融合遗传操作和温度系数？粒子群算法中的温度变化如何体现？相应的概率模型如何选择，概率值如何确定？

**回答：**在多目标粒子群算法的基础上，要融合遗传操作和温度系数，需在每次迭代过程中更新位置之后增加遗传操作，具体过程如下：

温度系数与迭代次数有关，首先根据式（6.2）计算当前温度系数，温度系数决定着执行何种遗传操作和操作位置的个数。



（6.2）

其中为在当前迭代次数下的温度系数；为初始温度系数；为迭代结束时的温度系数；为当前迭代次数；为总迭代次数。

遗传操作部分的伪代码如下：

|  |
| --- |
| **Input:**当前温度系数w\_now, 非支配解集pareto,各粒子位置particles, 粒子位置最大值max,  粒子位置最小值min，虚拟机个数vm\_num  **Output:**遗传操作后的新粒子位置particles |
| /\* 根据w\_now选择何种遗传操作，rand(a,b)为区间(a,b)之内的随机数\*/   1. **If** w\_now > rand(0,1) **Then**   /\*变异操作\*/   1. i ← rand(0,len(particles)) //随机选择一个粒子进行变异操作 2. **For**  j ← 1 **to** w\_now \* vm\_num **do** //根据w\_now选择操作位置个数 3. vary\_pos ← rand(0, vm\_num) 4. particles [i][vary\_pos] ← rand(min, max) 5. **End For** 6. **Else**   /\*交叉操作\*/   1. **For** i ← 1 **to** len(particles) **do** 2. ref ← rand() in pareto //在非支配解集中选择一个解作为交叉目标 3. **For** j ← 1 **to** (1-w\_now) \* vm\_num **do** //根据w\_now选择操作位置个数 4. cross\_pos ← rand(0, vm\_num) 5. particles [i][ cross\_pos] ← ref [cross\_pos] 6. **End For** 7. **End For** 8. **End If** |

本文的随机函数rand()是符合均匀分布的，根据是否大于随机数来选择何种遗传操作，因此选择变异操作的概率分布函数应为，如图 6.4所示：

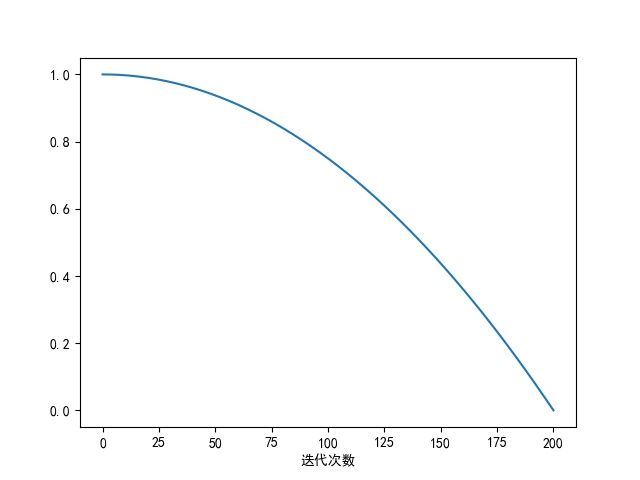


图 6.4选择变异操作的概率分布函数

如图 6.4所示，迭代次数越多，选择变异操作的概率越低，选择交叉操作的概率越高。