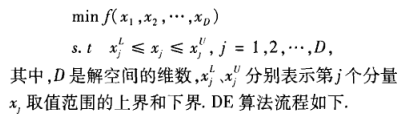
**差分算法基本原理和步骤**

<1>基本原理和过程

差分算法有三个主要流程：变异、交叉和选择。交叉原理与遗传算法比较相似。变异操作使用差分策略，利用个体之间的差异来对个体进行扰动变异。

示例操作：

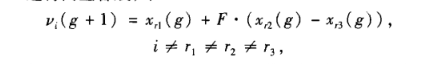
假设有优化问题：



1. 初始化种群：

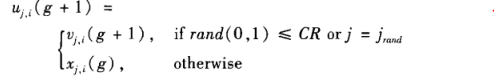
捕获

1. 变异操作：



其中F是缩放因子，如果变异的过程中产生的中间体V有任意维度不满足边界条件，那么这个中间体就重新初始化。

1. 通过变异操作产生一系列中间体后，进行交叉操作：



为了确保每个变异中间体中至少有一个基因传递到下一代，交叉时先随机选择一个基因基因点传递下去，然后对所有其他的基因点按照交叉概率来进行交叉过程。

1. 选择过程：

按照适应度值的大小，种群中的每个新染色体都是在上一代染色体和交叉后的染色体中选择出来的。

<2>算法基本参数

1:种群大小NP、缩放因子F、交叉概率CR。

**差分算法解决指派问题**

1. 问题描述

一个优化问题：有m个任务，索引为j,有n台机器，索引为i，每一台机器处理任务时的资源花费为SCij，每一台机器处理任务时花费的成本为CMij，机器自身拥有的资源量为Si，指派问题的模型如下：

Objection: C:\Users\ADMINI~1\AppData\Local\Temp\ksohtml1852\wps7.jpg

Constrain:C:\Users\ADMINI~1\AppData\Local\Temp\ksohtml1852\wps8.jpg

其中每台机器都能处理一定数量的任务，每台机器自身都有一定资源(由Si给出，一个1\*num\_mac的向量)，每台机器处理每个任务都需要花费一定资源（由SC给出，一个num\_mac\*num\_task的矩阵，SC (I,j)代表机器i（第i行）处理任务j(第j列)需要花费的资源），产生一定成本（由CM给出，一个num\_mac\*num\_task的矩阵，CM (I,j)代表机器i（第i行）处理任务j(第j列)产生的成本），每一个任务都有且只能有一个机器来处理。解由一个矩阵C:\Users\ADMINI~1\AppData\Local\Temp\ksohtml1852\wps9.jpg(其中所有元素的范围为{0,1})给出，其中*X*[I,j]代表机器i是否被安排处理任务j。

其中目标函数是要求得一个最合适的C:\Users\ADMINI~1\AppData\Local\Temp\ksohtml1852\wps10.jpg矩阵，使得所有所有任务都能够处理完成并且总成本最小。

共有三个约束：

第一个约束是指每台机器处理的任务需要消耗的资源量应小于这台机器自身所拥有的资源量。

第二个约束是指每一个任务都有且只能有一个机器来处理。

第三个约束是指解矩阵的取值范围。

1. 主要技术方案
2. 三项局部搜索算法：shifting, exchange, and k-variable move algorithms和差分进化算法的结合。
3. DE-SK算法性能更好。

**3、详细方案（技术）说明**

**1、问题的解是如何编码的？**

解编码是一列（1\*N）向量，N是任务的个数，其中向量的索引就是每个任务的索引，向量的值是一列生成的数值value，解码是两列1\*N的向量，第一行向量的值是每个任务的索引，第二行向量是与第一行向量中每个任务索引相对应的value,两个向量按照value从小到大排序。

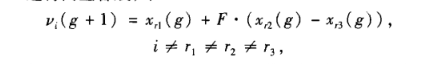
**2、解的初始化**：

(1)、随机生成这些任务的顺序，命名为omiga

(2)、按照任务的顺序列表，逐一的将所有任务分配。分配原则是对当前任务，将这个任务分配给能最小资源消耗处理这个任务的机器，如果该机器没有足够容量了，那么按照最小资源消耗原则继续往下分配，直到所有任务都被分配了机器。

**关键点是如何寻找到最优的顺序列表**omiga？

1. **变异操作**



1. **交叉操作**

**。。。。。看不下去了，文献太差浪费时间，反驳原因：**

**1、在cost matrix如下的情况**

|  |  |  |
| --- | --- | --- |
| **2** | **6** | **39** |
| **3** | **5** | **38** |

**如果omiga是[1,2,3],按这个顺序去初始化解，根本最后一个任务分配不了（解决方案，将cost matrix按列也就是任务进行求均值，按照均值从大到小分配任务，分配机器的方法不变，就可以解决这个问题）。**

**2、文献中变异操作的公式错误。**