第33卷 第12期 计 算 机 工 程 2007年6月
Vol.33 No.12 Computer Engineering June 2007
·人工智能及识别技术· 文章编号：1000—3428(2007)12—0208—03 文献标识码：A 中图分类号：TP391.72

改进模拟退火算法在模块划分中的研究及应用

单 泉，闫光荣，雷 毅
(北京航空航天大学机械工程及自动化学院，北京 100083)

摘 要：模块划分是产品模块化设计的关键技术之一。目前大多采用非数值方法划分模块，数值划分方法主要是使用模拟退火算法或遗传算法。模拟退火算法虽可以一次性得到模块划分最优方案，但是操作困难，效率不高。而遗传算法容易陷入局部最优解。该文在模拟退火算法的基础上，融入遗传算法的种群思想，提出了基于改进模拟退火算法的模块划分方法，研究了其实现的关键技术，并通过 VC++6.0 将其实现。通过具体的模块划分实例，证实了该方法的高效性和易操作性。
关键词：模块划分；模拟退火算法；遗传算法；改进模拟退火算法

Research and Application of Improved Simulated Annealing Algorithms in Module Identification

SHAN Quan, YAN Guangrong, LEI Yi
(School of Mechanical Engineering and Automation, Beijing University of Aeronautics and Astronautics, Beijing 100083)

【Abstract】 Module identification is one of the key technique for product modularization design. While non-numerical value module identification method is more common, the method of numerical value mainly uses simulated annealing algorithms or genetic algorithms. Using simulated annealing algorithms can get the best solution of module identification at one time, but it is hard to operate and has low efficiency, using genetic algorithms easy gets into local optimization. This paper puts forward module identification way which is based on improved simulate annealing algorithms and researches the key techniques, blends the genetic algorithms' population idea. This method is realized with Visual C++ 6.0. Using one example, this method is validated easy operation and high efficiency.
【Key words】 Module identification; Simulated annealing algorithm; Genetic algorithm; Improved simulated annealing algorithm

面对目前日益激烈的市场竞争，企业必须增强产品创新能力、缩短设计周期和提高用户满意度，同时又要降低成本、保证质量和相对延长产品生命周期。传统的产品设计方法已经无法满足上述要求，而模块化设计技术能满足以上要求，已经成为目前产品设计技术发展的重要方向之一。模块划分是模块化设计的关键技术之一。目前，对模块划分非数值方法，如启发式方法[1]、模糊聚类分析法[2]或最大树图法[3, 4]等的研究较多，这些方法都较复杂，可操作性较差；对模块划分数值方法研究较少，集中在单独使用模拟退火算法[4, 5]或遗传算法[6]方法上。模拟退火算法有很强的全局搜索能力和收敛性，但是通过随机迭代出邻域值，一次只对一个邻域值操作，算法效率较低，且借助Matlab等软件[4]，对用户及计算机配置都要求较高，操作也比较复杂；遗传算法强调对染色体种群进行操作，算法效率较高，但遗传算法容易陷入局部最优解。本文基于模拟退火算法，融入遗传算法的种群思想，并针对优化目标函数不能处理单一模块等问题，得到基于改进模拟退火算法的模块划分方法，在VC++6.0 上将该方法实现。用户只需打开零部件关联矩阵，程序将快速得到产品模块划分最优方案，整个模块划分过程对使用者来说非常方便。将此方法和基于模拟退火算法的划分方法进行比较，优化结果表明，该方法的确能大大提高模块划分的优化速度。

1 基于改进模拟退火算法的模块划分方法

模块划分是模块化设计的核心问题之一，它是一个复杂、影响因素多的综合优化过程，而产品元件(零部件)之间关联值的形成是使用数值方法划分模块的前提。通过层次分析及相关性分析方法得到产品元件两两之间的交互数值，一般是采用[0，1]的平均分布数值来定义元件之间的关联关系，0 表示没有任何关联关系，1 表示关联关系密不可分，如元件和自身的关联关系。有了这些关联值后，就可以构成产品关联矩阵，因此关联矩阵是一个主对角元素为 1 的对称方阵。关联矩阵的具体形成过程可以参考文献[4]，在此不再赘述。值得一提的是，关联矩阵的确立必须综合考虑产品生命周期各阶段的影响，并且针对具体的产品，各阶段的影响权重也不相同，这些权重值应由人工确定。

模块划分问题是一个典型的组合优化问题，可以使用现代优化方法对其进行优化，遗传算法和模拟退火算法都属于现代优化方法。模拟退火算法和遗传算法可以参考文献[7]。结合模拟退火算法以一定概率接受邻域值且随温度下降确保算法收敛性的特点和遗传算法对种群操作效率较高的特点，改进模拟退火算法具有较高的效率和较强的全局搜索能力。使用改进模拟退火算法进行产品模块划分优化的过程如下：

Step1 读入关联矩阵值，生成初始染色体，具体编码过程详见关键技术部分；

Step2 随机交换最初染色体的基因，生成一组不含相同

基金项目：国家“863”计划基金资助项目(2003AA4Z3130)
作者简介：单 泉(1977－)，男，博士生，主研方向：CAD，产品生命周期管理；闫光荣，副研究员；雷 毅，教授、博导
收稿日期：2006-07-30 **E-mail：**shanquan2001@126.com

染色体的染色体组，将其设为当前染色体组；

Step3 得到当前最佳染色体 C1 及其对应目标值 F1；

Step4 计算初始温度并将其设为当前温度，设定最低温度、内部最大迭代次数和接受概率 K 值；

Step5 如果当前温度小于最小温度，则转向 Step12；否则进入下一步；

Step6 如果当前内部迭代数大于内部最大迭代数，则降温，将降温后的温度设为当前温度，内部迭代次数置 0，转向 Step5；否则进入下一步；

Step7 依次随机得到当前染色体组中的每个染色体的邻域染色体，如果邻域染色体的目标函数值更好，则接受此邻域染色体；如果目标函数值更差，则计算当前接受概率和随机概率，如果当前接受概率大于随机概率，则接受此邻域染色体；如果接受概率小于等于随机概率，则依旧使用原来的染色体，由此生成新染色体组 1；

Step8 得到当前最佳染色体 C2 及其对应目标值 F2；

Step9 如果 F2 小于 F1，则将 F2 赋给 F1，C2 赋给 C1，进入下一步；否则直接进入下一步；

Step10 依次计算染色体组 1 中每个染色体的适应函数值，根据适应函数值选取染色体，生成新染色体组 2；

Step11 将新染色体组 2 设为当前染色体组，内部迭代次数＋1，转向 Step6；

Step12 输出最优染色体 C1 及其对应目标值 F1，然后结束计算。

整个计算过程如图 1 所示。

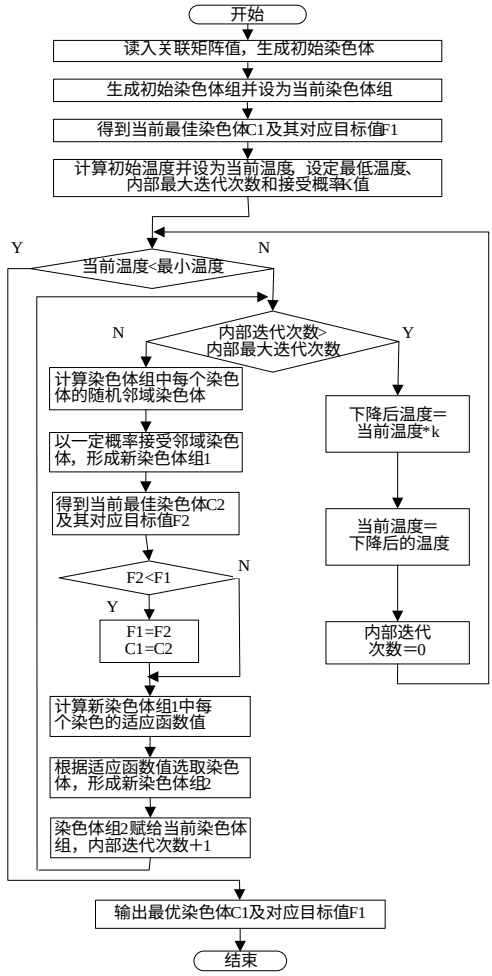


图 1 基于改进模拟退火算法的模块划分优化过程

2 关键技术

下面介绍单一模块处理、染色体编码、接受概率 K 的确定和迭代后新染色体的有效性判断这 4 个实现本划分方法的关键技术。

2.1 单一模块

一个模块中只有一个元件，这个模块叫作单一模块。本文使用文献[4]中的模块划分目标函数，如下所示：

$$F=\sum_{m=1}^{V}\frac{fm}{fm\max}*\frac{fm}{Tf\ \max} \tag{1}$$

其中，$fm=\sum_{i=1}^{Nm-1}\sum_{j=i+1}^{Nm}A(i,j)$；$A(i,j)$是第 i 个元件和第 j 个元件之间的关联值；fm 表示第 m 个模块内所有元件两两之间关联值之和，如果 $A(i,j)=1$，则

$$fm\max=\sum_{i=1}^{Nm-1}\sum_{j=i+1}^{Nm}1=\frac{Nm(Nm-1)}{2} \tag{2}$$

$$Tf\ \max=\sum_{m=1}^{V}fm\max \tag{3}$$

其中，V 为模块数；Nm 为第 m 个模块中所包含的元件数。

从式(2)可知，如果存在单一模块，则 $fm\max=\frac{Nm(Nm-1)}{2}=\frac{1*0}{2}=0$，这样在目标函数 F 中将出现分母为 0 的情况，在运算中这是不允许的。所以，仅仅通过目标函数的计算是无法处理单一模块的问题。解决方法是通过读入的产品关联矩阵，分析是否存在下列两种情况：

(1)某一元件和其他元件的关联值都小于 0.2，表示此元件和其他元件的关联都非常弱，则此元件应为单独形成一个模块；

(2)某一元件和其他元件的关联值都大于 0.8，表示此元件和其他元件的关联都非常强，此元件为一个总线(BUS)模块，所以此元件应单独形成一个模块。

通过这两项原则，可以首先确定是否存在单一模块，如果有，将此模块单独列出，只对剩下的元件进行优化计算。

2.2 编码

编码是算法运算的基础，编码过程详述如下：

Step1 读入关联矩阵值开始，关联矩阵是一个主对角元素为 1 的对称方阵，因此只需要读入除主对角元素以外的一半关联值即可；

Step2 根据读入关联值的个数算出元件个数 n。假设读入某个产品关联矩阵值的个数是 m，通过 m=n*(n-1)/2 可求得 $n=(1+\sqrt{1+8*m})/2$，同时可以判断，如果 $\sqrt{1+8*m}$ 并非整数，则读入的关联矩阵错误；

Step3 分析关联矩阵，是否有单一模块，如果有且数目为 SingleM，则将其单独列出，剩下的元件数 n=n-SingleM；

Step4 计算默认模块数目 Modulenumber=[$\sqrt{n}$](表示不大于 $\sqrt{n}$ 的最大整数)，也可以由用户自行设定模块数目；

Step5 编制最初染色体代码，本算法中的染色体采用自然数编码，规则如下：首先染色体的第 1 位为 0；然后依次插入从 1 到 n 的 n 个自然数，代表 n 个元件的代号，染色体末端再插入 0；

Step6 计算 0 插入位 Insertsite=n/Modulenumber，如果 Insertsite 可以整除，则在从 1 到 n 的 n 个自然数中，每隔 Insertsite 个位数后插入一个 0，一共插入(Modulenumber-1)个 0；如果 Insertsite 不能整除，则在从 1 到 n 的 n 个自然数中，每隔(Insertsite+1)个位数后插入一个 0，同样一共插入(Modulenumber-1)个 0，完成了初始染色体的编码工作。

按照上述规则，假设某个产品有 24 个元件，没有单一模块，则初始染色体为(0，1，2，3，4，5，6，0，7，8，9，10，11，12，0，13，14，15，16，17，18，0，19，20，21，22，23，24，0)，两个 0 之间的元件构成一个模块。通过元件的代号，结合已读入的关联矩阵，可以很方便地得到两个元件之间的关联值。采用这种编码方式简单明了，易于算法的运算。

2.3 接受概率系数 K

染色体接受概率的计算公式 $e^{-\frac{K*\Delta}{t}}$，其中Δ是两个染色体目标函数的差值；t 是当前温度；K 是系数，这个系数非常重要，直接影响到改进模拟退火算法的全局搜索能力和收敛性。K 值的选取要遵循下面原则：

(1)在初始温度时，要求接受概率趋向 1，即 $\frac{K*\Delta}{t}$ 趋向 0，此时接受概率大，基本上所有的迭代值不论好坏都接受，这样可以避免早熟；

(2)随着温度的下降，要求接受概率逐步下降，当温度降到最低时，此时的接受概率较小，由此确保算法的收敛性。笔者建议此时的 K*Δ和 t 最好处于同一数量级，并且 K*Δ要大于 t。

通过上述两个原则，可以有效地确定 K 的值。

2.4 新染色体的有效性

所有迭代出的新染色体，必须进行如下有效性判断：

(1)染色体的第一位和最后一位必须为 0；

(2)染色体中没有连续的两个 0；

(3)染色体中不存在单一模块。

3 实例

为了验证本方法的正确性和高效性，以文献[4]中 24 个元件的齿轮减速器为实例，其最优模块划分结果是{1，5，9，15，21}、{2，6，10，13，16，19，22}、{3，7，11，14，17，20，23}和{4，8，12，18，24}，最优目标函数值是-0.32749，元件关联矩阵如表 1 所示。

表 1 齿轮减速器的元件关系矩阵

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	1	0.197	0.197	0.197	0.803	0.197	0.197	0.197	0.803	0.197	0.197	0.197	0.197	0.197	0.653	0.197	0.197	0.197	0.197	0.197	0.803	0.197	0.197	0.197
2		1	0.197	0.197	0.197	0.803	0.197	0.197	0.197	0.803	0.197	0.197	0.653	0.197	0.197	0.653	0.197	0.197	0.803	0.197	0.197	0.803	0.197	0.197
3			1	0.197	0.197	0.197	0.803	0.197	0.197	0.197	0.803	0.197	0.197	0.653	0.197	0.197	0.653	0.197	0.197	0.803	0.197	0.197	0.803	0.197
4				1	0.197	0.197	0.197	0.803	0.197	0.197	0.197	0.803	0.197	0.197	0.197	0.197	0.197	0.653	0.197	0.197	0.803	0.197	0.197	0.803
5					1	0.197	0.197	0.197	0.43	0.197	0.197	0.197	0.197	0.197	0.43	0.197	0.197	0.197	0.197	0.197	0.197	0.197	0.197	0.197
6						1	0.197	0.197	0.197	0.43	0.197	0.197	0.43	0.197	0.197	0.43	0.197	0.197	0.43	0.197	0.197	0.43	0.197	0.197
7							1	0.197	0.197	0.197	0.43	0.197	0.197	0.43	0.197	0.197	0.43	0.197	0.197	0.43	0.653	0.197	0.43	0.197
8								1	0.197	0.197	0.197	0.43	0.197	0.197	0.197	0.197	0.197	0.43	0.197	0.197	0.197	0.197	0.197	0.43
9									1	0.197	0.197	0.197	0.197	0.197	0.43	0.197	0.197	0.197	0.197	0.197	0.197	0.197	0.197	0.197
10										1	0.197	0.197	0.43	0.197	0.197	0.43	0.197	0.197	0.43	0.197	0.653	0.43	0.197	0.197
11											1	0.197	0.197	0.43	0.197	0.197	0.43	0.197	0.197	0.43	0.197	0.197	0.43	0.197
12												1	0.197	0.197	0.197	0.197	0.197	0.43	0.197	0.197	0.197	0.197	0.197	0.43
13													1	0.57	0.197	0.43	0.197	0.197	0.803	0.197	0.803	0.43	0.197	0.197
14														1	0.197	0.197	0.43	0.197	0.197	0.803	0.197	0.197	0.43	0.197
15															1	0.57	0.197	0.197	0.197	0.197	0.197	0.197	0.197	0.197
16																1	0.197	0.197	0.43	0.197	0.197	0.803	0.197	0.197
17																	1	0.53	0.197	0.43	0.803	0.197	0.803	0.197
18																		1	0.197	0.197	0.197	0.197	0.197	0.803
19																			1	0.197	0.197	0.43	0.197	0.197
20																				1	0.197	0.197	0.43	0.197
21																					1	0.197	0.197	0.197
22																						1	0.197	0.197
23																							1	0.197
24																								1

本文使用的计算机配置为：CPU 赛扬 1GHz，内存 512MB，硬盘 40GB。除本文使用的改进模拟退火算法模块外，笔者还编写了模拟退火算法模块，选取了 3 次不同优化过程，使用 Origin6.0 读入优化过程中产生的数据，绘制出优化曲线。3 次优化使用的算法、参数和优化的结果详述如下。

(1)使用改进模拟退火算法，其中最小温度 1，内部最大迭代次数为 24*24=576 次，温度下降系数 k=0.9，接受概率系数 K=1000，计算所得的初始温度＝175.74，共进行 50 次降温，总迭代次数为 50*576=28800 次，整个计算花费时间是 1min7s，在第 24 678 次迭代后(第 43 次降温)，得到最优模块划分结果，优化曲线如图 2 所示。

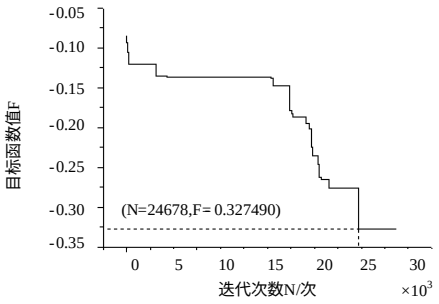


图 2 基于改进模拟退火算法的优化结果

(2)使用模拟退火算法，其中最小温度 0.001，内部最大迭代次数 24*24=576，温度下降系数 k=0.9，接受概率系数 K=1 000，计算所得的初始温度＝175.74，共进行 115 次降温，总迭代次数为 115*576=66 240 次，整个计算花费时间是 1min12s，得到的最优值为-0.294 795，未能得到最优模块划分结果，优化曲线如图 3 所示。

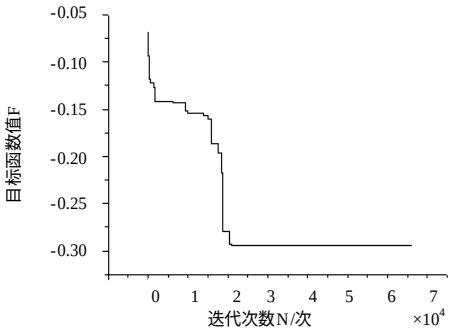


图 3 基于模拟退火算法的优化结果 1

(3)使用模拟退火算法，其中最小温度 1，内部最大迭代次数 24*24*24=13824 次，温度下降系数 k=0.9，接受概率系数 K=1000，计算所得的初始温度＝175.74，共进行 50 次降温，总迭代次数为 50*13824=691200 次，整个计算花费时间是 10min36s，在第 505 610 次迭代后(第 37 次降温)，得到最优模块划分结果，优化曲线如图 4 所示。

从上面实例的优化结果来看，本文采用的基于改进模拟退火算法的模块划分方法可以准确得出产品模块划分的最优结果，并且划分效率要大大高于基于模拟退火算法的模块划分方法。

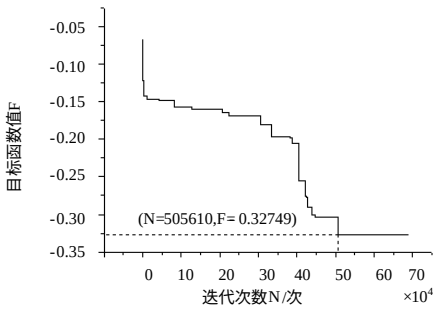


图 4 基于模拟退火算法的优化结果 2

(下转第 213 页)