一种动态自适应蚁群算法

李开荣[1] 陈宏建[1] 陈 峻[1,2]
[1](扬州大学计算机科学与工程系,江苏扬州 225009)
[2](南京大学软件新技术国家重点实验室,南京 210093)
E-mail:yzlkr@yzcn.net

摘 要 针对传统蚁群算法容易出现早熟和停滞现象的缺陷,提出了一种动态自适应蚁群算法。该算法对传统的MMAS蚁群算法中的信息素进行自适应调整。实验结果表明,该算法比传统的蚁群算法和传统的MMAS蚁群算法具有更好的搜索全局最优解的能力,并具有更好的稳定性和收敛性。

关键词 蚁群算法 自适应 信息素 优化

文章编号 1002-8331-(2004)29-0149-04 **文献标识码** A **中图分类号** O141.3

A Dynamic and Adaptive Ant Algorithm

Li Kairong[1] Chen Hongjian[1] Chen Ling[1,2]
[1](Department of Computer Science and Engineering, Yangzhou University, Yangzhou, Jiangsu 225009)
[2](State Key Laboratory for New Software Technology, Nanjing University, Nanjing 210093)

Abstract: This text advances a dynamic and adaptive ant algorithm in accordance with the defect of early variety and stagnation for traditional algorithm.The algorithm adjusted the message units of traditional MMAS ant algorithm adaptively.The test results indicate that this algorithm has more excellent ability in searching the whole best solution than the traditional ant algorithm and the traditional MMAS ant algorithm.In addition,this algorithm has much better stability and convergency.

Keywords: ant algorithm, adaptive, message units, optimization

1 引言

蚁群算法是最近几年由意大利学者Dorigo等人首先提出的一种新的启发式优化算法[1],是目前国内外启发式算法研究的热点和前沿课题,它已被成功地应用于求解TSP问题[2]、job-shop调度问题[3]、指派问题[4,5]、序列求序[6](sequential ordering)等NP难问题。种种实验表明,蚁群算法在求解复杂优化问题方面具有很大的优越性和广阔的应用前景,但由于蚁群中多个蚂蚁的运动是随机的,当群体规模较大时,很难在短时间内从复杂无章的路径中找出一条较好的路径。为此,Dorigo等人在基本蚁群算法的基础上提出了称为Ant-Q System的蚁群算法[7,8],仅让每一次循环中最短的路径上的信息素作更新。为了克服在Ant-Q中可能出现的停滞现象,Stutzle等人提出了MAX-MIN Ant System(MMAS)[9],允许各个路径上的信息素在一个限定的范围内变化。Gambardella等人提出了一种混合型蚁群算法HAS[6],在每次循环中蚂蚁建立各自的解后,再以各自的解为起点用某种局部搜索算法求局部最优解,作为相应蚂蚁的解,这样可以迅速提高解的质量。另外,Bilchev等人在使用遗传算法解决工程设计中连续空间的优化问题时,使用了蚁群算法对遗传算法所得到的初步结果进行精确化,取得了较好的效果[10],为了提高蚁群算法的速度和精确度,吴斌、史忠植等提出了基于蚁群算法的分段求解算法[11],丁建立、陈增强等提出了先利用遗传算法再利用蚁群算法融合的方法[12]。文章针对蚁群算法容易出现停滞和收敛速度慢等现象,提出了一种自适应蚁群算法,通过自适应地调整运行过程中的挥发因子来改变路径中的信息素,从而有效地克服了传统蚁群算法中容易陷入局部最优解和收敛速度慢的现象,实验证明,该蚁群算法比传统的蚁群算法和传统的MMAS算法具有更好的搜索全局最优解的能力,并且其收敛性和稳定性较传统的蚁群算法也有了明显的提高。

2 基本蚁群算法

以TSP问题为例说明基本蚁群算法的框架。设有m个城市,n只蚂蚁,采用$d_{ij}(i,j=1,2,\ldots,m)$表示城市i和城市j之间的距离,$\tau_{ij}(t)$表示在时刻t城市i和城市j之间的路径上的残留信息素强度,以此来模拟实际蚂蚁的分泌物。蚂蚁k在行进过程中,根据各条路径上的信息素强度来决定下一步所行进的路径,采用$p_{ij}^{k}(t)$表示在时刻t蚂蚁k由城市i转移到城市j的概率,则有:

$$p_{ij}^{k}(t)=\begin{cases}\dfrac{\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum_{r\in Table_k}\tau_{ir}^{\alpha}(t)\eta_{ir}^{\beta}(t)}, & j\in Table_k\\ 0, & \text{否则}\end{cases}\tag{1}$$

其中$Table_k$表示蚂蚁k下一步允许行进的城市集合,它随蚂蚁k的行进过程而动态改变。信息素强度$\tau_{ij}(t)$随时间的推移会逐步消逝,用ρ表示它的消逝程度。蚂蚁选择转移到哪个城市采用伪随机概率选择规则,按照这种规则,每当蚂蚁要选

基金项目:国家自然科学基金项目(编号:60074013);国家高性能计算基金项目(编号00210);江苏省教育厅自然科学基金;南京大学软件新技术国家重点实验室开放基金资助

择转移的城市时，就产生一个在[0,1]范围内的随机数，根据该随机数按下列公式确定蚂蚁转移的方向。

$$s=\begin{cases}\arg\max\{[\tau(i,j)][\eta(i,j)]^{\beta}\} & \text{若 } q\le q_0\\ S & \text{否则}\end{cases} \tag{2}$$

其中，q 为一个在区间[0,1]内的随机数，q_0 为[0,1]内的一个参数，S 由公式(1)确定。这样，经过 m 时刻，蚂蚁走完所有的城市，即完成了一次循环。此时根据下式对各路径上的信息素进行更新：

$$\tau_{ij}(t+1)=(1-\rho)\cdot\tau_{ij}(t)+\Delta\tau_{ij} \tag{3}$$

其中：$$\Delta\tau_{ij}=\sum_{k=1}^{m}\Delta\tau_{ij}^{k} \tag{4}$$

$\Delta\tau_{ij}^{k}$ 表示蚂蚁 k 在本次循环中在城市 i 和城市 j 之间的路径上留下的信息素，其计算方法可以根据计算模型而定，在最常用的 Ant Circle System 模型中：

$$\Delta\tau_{ij}=\begin{cases}\dfrac{Q}{L_k}, & \text{若蚂蚁 } k \text{ 在本次循环中经过城市 } i \text{ 和城市 } j\\ 0, & \text{否则}\end{cases} \tag{5}$$

其中：Q 为常数，L_k 为蚂蚁 k 在本次循环中所行走路径的总长度，在式(1)中，α 表示蚂蚁在行进过程中所积累的信息素对它选择路径所起的作用程度，η_{ij} 表示由城市 i 转移到城市 j 的期望程度，可根据某种启发算法而定，例如，可以取 $\eta_{ij}=1/d_{ij}$，β 表示 η_{ij} 的作用。当 $\alpha=0$ 时，算法就是传统的贪心算法，而当 $\beta=0$ 时，算法就成了纯粹的正反馈的启发式算法。可以用试验的方法确定参数 α,β 的最优组合。在经过若干次循环以后，可以根据适当的停止条件来结束计算。

3 自适应的信息更新策略

蚂蚁在行进过程中常常选择信息量较大的路径，但当许多蚂蚁选中同一条路径时，该路径中的信息量就会陡然增大，从而使得多只蚂蚁集中到某一条路径上，造成一种堵塞和停滞现象，表现在使用蚁群算法解决问题时就容易导致早熟和局部收敛。为了解决这一问题，提高蚁群算法的全局收敛能力和搜索速度，许多文献提出了各种不同的更新信息量的策略，如引言中所谈到的蚁群算法，它们在更新信息量时，要么采取只要蚂蚁遍历时选择此路径就更新其路径上的信息量，要么只让最优适应度的路径上的信息量增强，而其余路径上的信息量被削减，要么就是基于等级变化的算法，让适应度相对较好的蚂蚁固定在若干条路径上，根据其解的优劣程度决定信息量的增加幅度，这些算法虽然各不相同，但它们主要采用固定信息量增减的比例来进行信息量的更新，它们都忽视了解的分布特征，在一定程度上虽对蚁群算法的特性进行了一定的改进，但它们一般只适合于处理较小规模的问题，为了解决较大规模的问题，这里从解的分布状态入手，提出了一种新的自适应的信息量更新策略。

当问题规模较大时，由于信息量挥发系数的存在，使那些从未被搜索过的路径上的信息量减小到接近于 0，从而降低了算法在这些路径上的搜索能力，反之，当某条路径中信息量较大时，这些路径中的信息量增大，搜索过的路径再次被选择的机会就会变得较大，这也影响了算法的全局搜索能力，此时通过固定地变化挥发系数虽然可以提高全局搜索能力，但却使算法的收敛速度降低，因而该文提出一种自适应的改变 τ 值的方法，将信息素更新公式：

$\tau_{ij}(t+1)=(1-\rho)\cdot\tau_{ij}(t)+\Delta\tau_{ij}$

变为当 $\tau>\tau_{max}$ 时　$\tau_{ij}(t+1)=(1-\rho)^{1+\psi(m)}\cdot\tau_{ij}(t)+\Delta\tau_{ij}$

而当 $\tau<\tau_{min}$ 时　$\tau_{ij}(t+1)=(1-\rho)^{1-\psi(m)}\cdot\tau_{ij}(t)+\Delta\tau_{ij}$

其中 $\psi(m)$ 是一个与收敛次数 m 成正比的函数，收敛次数 m 越多，$\psi(m)$ 的取值越大，如：

$\psi(m)$=连续收敛次数 m/c

这里 c 为常数，这样，根据解的分布情况自适应地进行信息量的更新，从而动态地调整各路径上的信息量强度，使蚂蚁既不过分集中也不过分分散，从而避免了早熟和局部收敛，提高全局搜索能力。

改进的蚁群算法框架描述如下：

```
{初始化 α、β、ρ、c 等参数;
  for(i=0;i<m;i++)// m 为城市的数量
        for(j=0;j<m;j++)τ(i,j)=τ0
  for(k=0;k<n;k++)// n 为蚂蚁的个数
    {
        设 ik0 为蚂蚁 k 的开始城市;
        Tablek={0,1,…,m-1}-ik0;
        ik=ik0;
    }
  while(not 结束条件)
    {
      for(p=0;p<m;p++)
        {
          if(p<m)
            for(k=0;k<n;k++)
              {
                按公式(1)和公式(2)选择下一个城市 jk;
                Tablek=Tablek-jk;
                Tourk(p)=(ik,jk);
              }
          else
            for(k=0;k<n;k++)
              {
                jk=ik0;
                Tourk(p)=(ik,jk);
              }
  for(k=0;k<n;k++)ik:=jk;
  }
  for(k=0;k<n;k++)
      计算 Lengthk 并求出最优长度的平均值 AveL。
  if(平均值 AveL 与前一次 AveL 计算值相同)m=m+1;else m=1;
   ψ(m)=m/c;
  for(每条边 edge(i,j))
    {
      if(τ(i,j)<τmax)τ(i,j)=(1-ρ)^(1+ψ(m))τ(i,j)+Δτij
      if(τ(i,j)<τmin)τ(i,j)=(1-ρ)^(1-ψ(m))τ(i,j)+Δτij
    }
  }
  输出最佳结果;}
```

上述算法根据解的分布情况自适应地进行信息量的更新，从而动态地调整了各路径上的信息量强度，增加了解空间的多样性，提高了全局搜索能力，避免了局部收敛和早熟现象，实验证明改进后的蚁群算法比传统蚁群算法和传统的 MMAS 算法具有更好的搜索最优解的能力。

4 仿真实验与结果

为了验证改进的蚁群算法比传统的蚁群算法和MMAS蚁群算法具有更好的搜索全局最优解的能力和较好的稳定性，文章以TSPLIB中提供的TSP问题Eil51，Kroa100，Gr120及Pcb442为例进行实验。实验中所采取的各项参数如下：蚂蚁的数目为15只，$\alpha=3$，$\beta=5$，$\rho=0.1$，$c=20$。系统信息素强度的初始值为0.1，常数为1，统计次数为15，最大迭代次数为3000次。

4.1 解的进化情况

笔者对上述4个TSP问题分别采用上述参数以不同的蚁群算法进行实验，其最优解和平均解的进化情况如表1所示，各问题所找到的最优解的路径如图1至图4所示。

表1 几种蚁群算法的最优解和平均解的进化情况

算法及解的情况		Eil51	Kroa100	Gr120	Pcb442
标准蚁群算法	最优解	426	21282	6942	50778
	平均解	432.9	21899.3	7218.6	51173.8
MMAS	最优解	426	21282	6942	50778
	平均解	428.06	21346.8	7124.1	50929.3
本文改进蚁群算法	最优解	426	21282	6942	50778
	平均解	426	21285.4	7022.9	50783.5

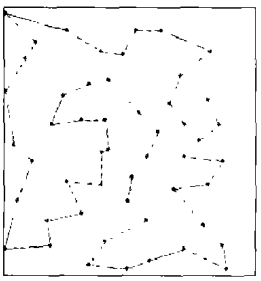

图1 Eil51找到的最优路径

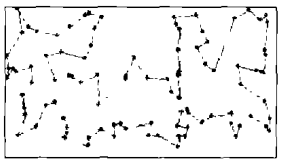

图2 Kroa100找到的最优路径

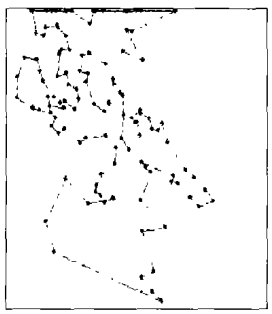

图3 Gr120找到的最优路径

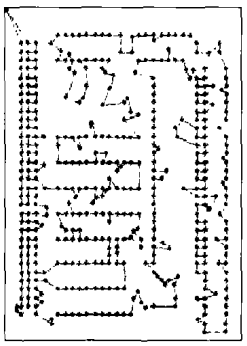

图4 Pcb找到的最优路径

4.2 收敛性

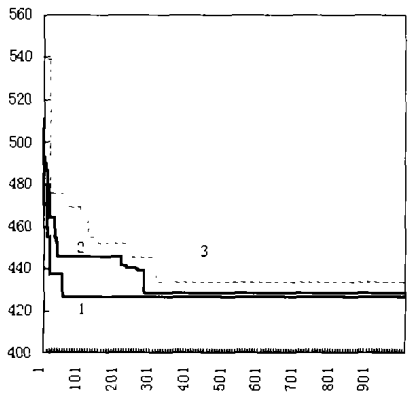

图5 Eil51解的各次进化曲线

为了测试改进算法的收敛性，笔者对问题Eil51，Kroa100，Gr120分别采用上述参数以不同的蚁群算法进行实验，各算法的最优解进化曲线如图5至图7所示。各图中横坐标表示迭代次数，纵坐标表示最优解的平均路径长度，各图中的曲线1表示该文算法的进化曲线，曲线2和曲线3分别表示MMAS算法和标准蚁群算法的进化曲线。

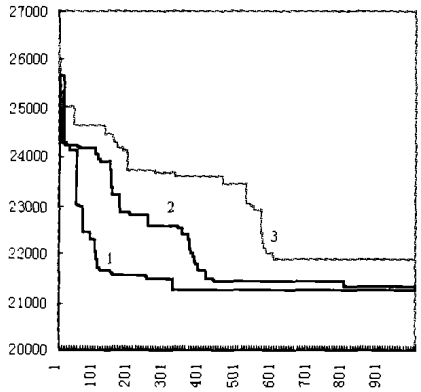

图6 Kroa100解的各次进化曲线

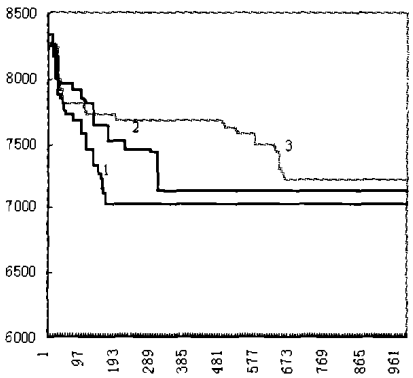

图7 Gr120解的各次进化曲线

4.3 稳定性

为了测试改进算法的稳定性，笔者对问题Eil51，Kroa100，Gr120分别采用上述参数对各种不同的蚁群算法分别连续运行10次，每次进化1000代，各次达到上述平均最优解的进化代数以及它们的平均值和标准如表2所示。

表2 各种算法达到平均最优解的进化代数的平均值与标准差

问题	算法	取得平均最优解的进化代数	平均进化代数	标准差
Eil51	标准蚁群算法	506，433，306，398，521，439，678，354，414，552	460.1	101.93
	MMAS算法	363，301，444，288，309，278，290，322，563，459	361.7	90.76
	该文算法	64，82，54，179，54，77，59，54，98，67	78.8	36.07
Kroa100	标准蚁群算法	883，635，678，984，605，821，766，609，882，638	750.1	129.23
	MMAS算法	437，566，632，578，459，677，529，455，726，478	553.7	95.09
	该文算法	544，384，365，320，462，497，432，392，363，346	410.5	67.92
Gr120	标准蚁群算法	785，763，923，647，686，856，742，694，722，698	751.6	79.95
	MMAS算法	464，396，345，300，422，527，341，593，462，387	423.7	85.36
	该文算法	211，179，156，354，165，194，156，246，222，243	212.6	54.06

由上述实验可知，改进后的蚁群算法具有比传统蚁群算法和MMAS蚁群算法更强的搜索全局最优解的能力，并具有更好的稳定性和收敛性。

5 结束语

该文针对传统蚁群算法容易出现早熟和停滞现象的缺陷，

提出了一种动态更新信息素的蚁群算法，实验表明，改进的蚁群算法具有比传统蚁群算法和 MMAS 蚁群算法更强的搜索全局最优解的能力，并具有更好的稳定性和收敛性。蚁群算法作为一种新的生物进化算法，目前还没有象遗传算法、模拟退火等那样形成较系统的分析方法和坚实的数学基础，各种参数的确定也没有一定的理论指导，相信随着蚁群算法研究的不断深入，蚁群算法也将会同其他生物进化算法一样获得越来越广泛的应用和坚实的理论根据。（收稿日期：2004 年 7 月）

参考文献

1.Dorigo M，Maniezzo V，Colorni A.Ant system：optimization by a colony of cooperating agents[J].IEEE Transactions on SMC，1996；26(1)：8~41

2.Dorigo M ，Gambardella L M.Ant colony system：a cooperative learning approach to the traveling salesman problem[J].IEEE Transactions on Evolutionary Computing，1997；1(1)：53~56

3.Colorni A，Dorigo M，Maniezzo V.Ant colony system for job-shop scheduling[J].Belgian Journal of Operations Research Statistics and Computer Science，1994；34(1)：39~53

4.Maniezzo V.Exact and approximate nondeterministic tree search procedures for the quadratic assignment problem[J].Informs Journal of Computer，1999；11(4)：358~369

5.Maniezzo V，Carbonaro A.An ANTS heuristic for the frequency assignment problem[J].Future Generation Computer Systems，2000；16(8)：927~935

6.Gambardella L M，Dorigo M.HAS-SOP：an hybrid ant system for the sequential ordering problem[R].Technique Report，No IDSIA 97-11，IDSIA，Lugano，Switzerland，1997

7.Gambardella L M，Dorigo M.Ant-Q：a reinforcement learning approach to the traveling salesman problem[C].In：Prieditis A，Russell S eds.Proceedings of the 12th International Conference on Machine Learning，Tahoe City，CA：Morgan Kaufmann，1995：252~260

8.Dorigo M，Luca M.A study of some properties of Ant-Q[R].Technical Report TR/IRIDIA/1996-4，IRIDIA，Université Libre de Bruxelles，1996

9.Stutzle T，Hoos H H.Improvements on the ant system：introducing the MAX-MIN ant system[C].In：Artificial Neural Networks and Genetic Algorithms，New York：Springer-Verlag，1988：245~249

10.Bilchev G，Parmee I C.Adaptive search strategies for heavily constrained design spaces[C].In：Proceedings of 22nd International Conference on Computer Aided Design'95，Yelta：Ukraine，1995：230~235

11.吴斌，史忠植.一种基于蚁群算法的 TSP 问题分段求解算法[J].计算机学报，2001；24(12)：1328~1333

12.丁建立，陈增强，袁著祉.遗传算法与蚂蚁算法的融合[J].计算机研究与发展，2003；40(9)：1351~1356

（上接 46 页）

表 3　指令总线的位翻转次数的比较（单位：次）

测试程序	PISA 2.0			PISA 3.0			操作码改变导致的指令总线位翻次数减少比例	指令总线上总的位翻转次数减少比例
	操作码改变导致的指令总线位翻转	操作数改变导致的指令总线位翻转	指令总线上总的位翻转	操作码改变导致的指令总线位翻转	操作数改变导致的指令总线位翻转	指令总线上总的位翻转		
quick	174619	303305	477924	95342	303305	398647	45.4%	16.6%
heap	264495	600841	865336	169502	600841	770343	35.9%	11.0%
bubble	1019115	1686541	2705656	600240	1686541	2286781	41.1%	15.5%
hanoi	577241	1193599	1770840	363208	1193599	1556807	37.1%	12.1%
queens	988336	2255370	3243706	664040	2255370	2919410	32.8%	10.0%

要 5 位即可表示，而 PISA2.0 对 111 条指令编码，操作码字段用了 16 位。

指令集经过重新编码后，指令总线上的位翻转次数也会随之改变，从而影响处理器速度和功耗。表 3 是模拟得到的测试程序集合在 PISA2.0 与 PISA3.0 两种指令集体系结构下指令总线的位翻转次数的比较。观察表 3 可知，对所有测试程序 PISA3.0 比 PISA2.0 指令总线上的位翻转次数有明显减少，因此功耗也能随之降低，运行速度能得到提高。

4　结束语

ASIP 行为级设计方法适用于 SoC 设计与开发的初期，是整个设计的基础。该文提出了一种 ASIP 行为级设计方法，并详细描述了一个可视化的基于体系结构描述语言的 ASIP 行为级设计平台 xptools 的功能、结构、工作机制。现在已经完成了 ASIP 行为级设计平台中 ASIP 行为级设计平台的 GUI、行为级体系结构描述语言 xpADL 的建模、可重定向模拟器生成器和可重定向指令集模拟器的设计。下阶段的主要研究问题是指令集体系结构的优化算法。

参考文献

1.Steven Bashford，Ulrich Bieker，Berthold Harking et al.The MIMOLA Language Version 4.1[EB/OL].http：//ls12-www.cs.uni-dortmund.de/research/mimola-4.1.ps.gz，1994

2.Mark R Hartoog，James A Rowson，Prakash D Reddy et al.Generation of Software Tools from Processor Descriptions for Hardware/Software Codesign[C].In：Proceedings of the 34th Design Automation Conference，1997：303~306

3.George Hadjiyiannis，Srinivas Devadas.Techniques for Accurate Performance Evaluation in Architecture Exploration[J].IEEE Transactions on VLSI Systems，2003；11(4)：601~615

4.Shinsuke Kobayashi，Kentaro Mita，Yoshinori Takeuchi et al.Rapid Prototyping of JPEG Encoder using the ASIP Development System：PEAS-III[C].In：Proceedings of ICASSP'03，Hong Kong，2003.II-485-488

5.Andreas Hoffmann，Tim Kogel，Achim Nohl et al.A Novel Methodology for the Design of Application-Specific Instruction-Set Processors(ASIPs) Using a Machine Description Language[J].IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems，2001；20(11)：1338~1354

6.Ashok Halambi，Peter Grun，Vijay Ganesh et al.EXPRESSION：A Language for Architecture Exploration through Compile/Simulator Retargetability[C].In：Proceedings of DATE'99，Munich，1999：485~490

7.Doug Burger，Todd M Austin.The SimpleScalar Tool Set，Version 2.0 [EB/OL].http：//www.simplescalar.com/docs/users_guide_v2.pdf，1997