第20卷第5期 半 导 体 学 报 Vol. 20, No. 5
1999年5月 CHINESE JOURNAL OF SEMICONDUCTORS May, 1999

基于协同工作方式的一种蚁群布线系统*

庄昌文 范明钰 李春辉 虞厥邦

(电子科技大学光电子技术系 成都 610054)

摘要 基于协同学习机制的蚁群算法ACS已成功应用于求解TSP问题.本文基于蚁群的协同工作机制,提出了一种其内涵扩充了的增强蚁群算法(IACS).利用此新算法设计了一个开关盒(Switchbox)布线程序,并用JAVA语言加以实现.针对几个算例计算的结果,证明该程序可获得比WEAVER、MIGHTY、BEAVER、GAP基准例低的计算复杂度.

EEACC: 1130B, 2210B

1 引言

随着超大规模集成电路的发展,通道布线在物理设计中的作用日趋重要,它是布线过程中的最后步骤,主要完成总体布线单元内部线网的物理连结.通道布线具体又分为两边通道布线、开关盒布线和L-型通道布线,都是NP完全问题.本文首次将蚁群系统(ACS,即Ant Colony System)用于解决通道布线问题,并取得了很好的效果.

Dorigo于1992年提出了一种仿生算法体系——蚂蚁系统(AS,即Ant System)[1],最近他和Gambardella又将其发展成为ACS算法,并将之应用于求解TSP问题,获得了较遗传算法(GA),模拟退火算法(SA)和演化规划算法(EP)为优的结果[2].

ACS算法的主旨是模仿蚁群的协同学习机制.蚁群受到食物气味的吸引,能借协同学习机制搜索到由蚁巢(出发点)到食物(目的地)之间的最短路径.在每个蚂蚁朝目的地奔走时,借随机搜索方式找到较短路径,就在该路径注入蚁激素,若多数蚂蚁借自己体验也认可该路径,也仿此注入自己的蚁激素,这样其余蚂蚁就能在各路径分岔点选择激素气味浓的路径朝目的地走去,少走弯路,最后获得一条最短路径供蚁群使用.按这种构思,ACS算法就能以较小的时间复杂度代价获得TSP问题的全局最优解.

本文将协同工作机制加入Dorigo基于协同学习的ACS中,扩充了其内涵,形成了一种新的算法体系——IACS(Intensified ACS)即增强的蚁群系统,并将IACS应用到开关盒布线问题中,形成了一个开关盒布线算法,命名为IACR(Intensified Ant Colony Router)-S1.

* 中国博士后基金资助项目

庄昌文 博士研究生,专业兴趣在计算智能,VLSI物理设计
范明钰 博士后研究人员,专业兴趣在计算智能应用,VLSI物理设计,通信系统理论等
李春晖 博士研究生,专业兴趣在计算智能,VLSI物理设计
虞厥邦 教授,主要从事非线性动力系统、神经网络及计算智能、集成电路CAD方面的研究

1998-03-18收到,1998-05-11定稿

通道布线问题与TSP问题有相似之处，也有不同之处，相似之处在于：在TSP问题中，旅行推销商遍历各城市时要寻求一最短总路径长，而在通道布线问题中，各线网总长多半也需为最短. 不同之处在于：TSP问题中旅行推销商的旅行路线为一闭合路径，而通道布线形成的图是一些互不连通的树的集合；此外，通道布线所加的约束条件较多，且随布线类型而异.

我们将ACS用于求解通道布线问题时，发现只要选择适当的模型表述和蚁群任务调度策略，该算法也可用于问题求解，但仅用协同学习机理时，寻优速度慢，布通率也较低；若加上协同工作机制，则算法质量产生质的飞跃，存在的上述问题也就迎刃而解. 以下各节将依次叙述IACR-S1的算法、编程实现及算例.

2　IACR-S1算法

2.1　开关盒布线问题

开关盒SB是一个$M\times Q$的纵横网格. 设SB含N个线网，给它们依次标记为1，2，…，k，…，$N-1$，N. 各线网的引脚按设计要求分布在此网格的上、下、左、右方，如图1所示. 对这些引脚按如下的方式作标记：第k个线网在SB四周的引脚分别记为P^r_{ek}，P^l_{wk}，P^b_{sk}，P^t_{nk}，右下标注明其方位(东、西、南、北)和线网编号，右上标注明此线网在SB的右、左、下、上方边上引脚的编号. 开关盒双层布线的要求是：顺着SB的$M\times Q$网格中走线，以形成对应于各线网且互不连通的N个树形，树形T_k须以k线网的各个引脚为其顶点集V_k.

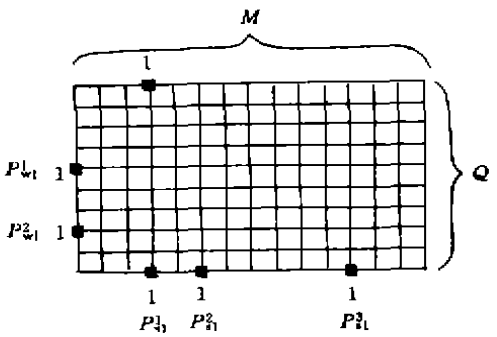

图1　开关盒

$$V_k=\{P^r_{ek},P^l_{wk},P^b_{sk},P^t_{nk}\}\quad k=1,2,\cdots,N \tag{1}$$

其中　r,l,b,t各取0,1,2,…,0取值表示线网k在该边上无引脚. 如图1所示，$V_1=\{P^1_{w1},P^2_{w1},P^1_{s1},P^2_{s1},P^3_{s1},P^1_{n1}\}$.

各树形的树支可分布在SB的上、下两层，其间可借过孔(Vias)连通.

2.2　IACR-S1算法描述

```
Algorithm
  read positions of pins on switchbox;
  initialization;
  generate N ant colonies;
  create V_k for each ant colony;
  select start point for each ant colony;
  start all ant colonies;
  wait for all ant colonies stopping;
  report routing result;
end
```

图2　启动所有蚁群的概略算法

IACR-S1用于求解SB布线问题时，就是组织N个蚁群，分别去生成各个线网的树形，蚁群之间和各个蚁群内的蚂蚁，均按协同学习和协同工作的方式去布线. IACR-S1算法主要由两部分组成，第一部分启动所有蚁群，第二部分为各个蚁群的行进过程.

启动所有蚁群　启动所有蚁群的概略算法如图2所示. 在为k($k=1,2,\cdots,N$)线网生成蚁群时，将建立该蚁群的V_k(k线网的引脚集)和选定蚁群的起始位置. 每个蚁群的蚂蚁数目，可选为本线网的引脚数，例如第k个线网交由第k个蚁群去布线，其蚂蚁数为P_k，P_k是第k个线网的引脚数. 从哪一个引脚

出发可按一定规则(如概率的随机方式)选定.然后启动所有蚁群,各蚁群将以异步方式各自去布通自己的线网.具体在高密度开关盒算例中,$N=19$,即有19个线网,算法将生成19个蚁群,并启动所有蚁群.

蚁群的行进过程 IACR-S1在蚁群的行进过程中引进了引力、停等机制等新思想,每个蚁群都以走走停停的方式前进,具体可分如下几个步骤:

STEP 1:目标和路况检测 在布线过程中,各蚁群到达一个新的位置后,若V_k为空,则蚁群停止布线,如果所有蚁群都已停止布线,则整个开关盒布线结束.若V_k不为空则蚁群将搜集目标和路径情况的信息,以作选择行进策略和蚁群内任务分配的依据.这些检测信息包括:

(1) 正前方和左、右两侧有无本线网的引脚;

(2) 左前方和右前方有无属于本线网的引脚,如有,有多少个;

(3) 在蚁群当前位置与本线网的某个引脚相当靠近时,就要作该目标的可到达性检测,即路况检测.当|cp.x−p.x| < detectx or |cp.y−p.y| < detecty时,即为相当靠近,在我们的算例中detectx=8, detecty=8,式中cp.x, cp.y分别表示蚁群当前位置的横、纵坐标;p.x和p.y则分别表示引脚p位置的横坐标和纵坐标,下同.

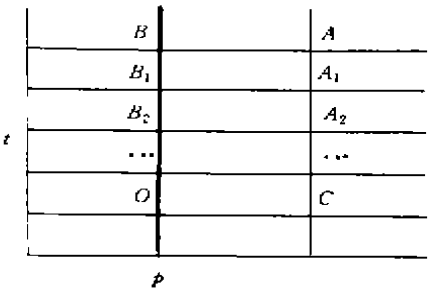

图3 底边上引脚的可到达性检测

参看图3,若蚁群k从北边南下,到达位置A,距其右前方的p已相当靠近,为到达此目标,蚁群须派遣蚂蚁先爬行到O点;因此,此蚁群就须依次检测ABO、ACO、AA_1B_1O、AA_2B_2O、…等路径是否通畅,是否有其它蚁群布线阻断了这些通路.

STEP 2:直达处理 通过检测(1),发现正前方/左侧/右侧有属于本线网的引脚或者是通过可到达性检测(3),发现本蚁群为到达邻近的某个本线网引脚,只剩下了一条通路,遇到这些情况,蚁群将派出蚂蚁,不作等待地,走多步直达本线网的这些引脚.蚂蚁经过的网格点,须标上本线网的标号,以供其它蚁群识别;而蚂蚁到达了某个引脚之后,就将该引脚从本线网的引脚集中删去,以免本蚁群将其认作未布通的引脚.

STEP 3:决定停等或继续前进 蚁群行进时(除了作直达处理),一次走一步,也即在网格中挪动一格,到了新位置,就要重新搜集信息(1)、(2)、(3).此外,考虑到要各个蚁群都要布通各自线网,开关盒的布线任务才算完成.因此,每个蚁群在行进时,不能单只自顾自去完成本群落的布线任务,也还要照顾到其它蚁群去完成它们各自的任务,也就是用协同工作的方式来共同完成任务.出于这种考虑,我们设计了如下的基于协同工作原则的停等机制:某蚁群到了一个新位置后,将检测其正前方是否有属于其它线网的且还未连结的引脚E,若没有则蚁群将在下述引力作用下继续前进;若有则该蚁群通常应停下来等待一个时间τ_w,以给其它蚁群布通E的机会,而不致阻塞引脚E的出路.为此,算法引进了一个表征概率大小的停等因子W_0,按概率来决定是停下来等待还是继续前进.W_0可取0.9,那就是说,停等的概率为90%,继续前进的概率为10%.此时可产生一个随机数$W\in(0,1)$,当$W_0\geqslant W$,蚁群将等待τ_w后再转到STEP1,继续行进;若$W_0<W$,则蚁群不作等待而继续前进.采用停等机制后,可有效地避免不同线网争用布线区域而引起的冲突.

STEP 4：计算引力　每个蚁群都须定量计算出各目标对群落本身吸引力的大小. 可以设想本线网的各个引脚都是本蚁群须获得的食物，食物的气味对蚁群构成吸引力(Attraction)，食物愈多愈近，气味愈浓，引力也愈大. 目前我们的算法须分别计算蚁群左侧、右侧、左前方和右前方这四个方向上的引力，且定义引力函数为

$$F(\mathrm{cp},\mathrm{p}) = \begin{cases} G, & \text{当 } |\mathrm{cp.x} - \mathrm{p.x}| = 1 \text{ or } |\mathrm{cp.y} - \mathrm{p.y}| = 1 \\ 1, & \text{其它} \end{cases} \tag{2}$$

其中　$G \gg 1$ 是一正的引力常数，$|\mathrm{cp.x}-\mathrm{p.x}|=1$ or $|\mathrm{cp.y}-\mathrm{p.y}|=1$ 表明蚁群与引脚 p 的横坐标或纵坐标相差为 1，蚁群有可能在同一层上拐弯而不会违犯布线规则.

STEP 5：选择行进方向　蚁群只能向前、向左、向右或向下(穿越过孔)行进，不能后退. 根据由公式(2)计算得到的上述四个方向上引力大小，蚁群作综合考虑后，将在引力的牵引下朝可继续前进的方向行进. 如图 3 所示，若蚁群 k 从北边南下，到达位置 A，在其右侧和右前方各有一个引脚，分别为 t 和 p 计算引力之后，这两个方向上的引力为 1，其余为 0. 显然，蚁群网左拐弯(向东)将增加线网线长，向南或向西前进都不会增加线长，但由于蚁群由北南下，向西走将增加过孔数目. 所以，在这时，蚁群 k 查看能否经过 A_1，若能，则将选择南方为前进方向；若不能经过 A_1，蚁群再查看能否朝西前进，等等. 更特别的问题是蚁群在某个位置前行遇到障碍时，需左转或右转，若根据引力计算，左前方和右前方都有本线网未连结的引脚，为不违犯布线行进规则，这时就须采用分兵策略，即蚁群分为两个子蚁群，分别左行和右行. 子蚁群的蚂蚁数和将要去连结的引脚集，可根据侦测到左前方和右前方各有本线网未布通的引脚多少来决定. 前进到一个新的网格点上后，蚁群将标上本线网的标号. 若该网格点是本线网的一个引脚，就在这个引脚上标记已连结，并把该引脚从本线网的未连结引脚集(V_k)中删去(这操作就如同蚁群派遣一个蚂蚁去占据了这个引脚，当本蚁群的每个蚂蚁都占据了一个引脚时，该线网布线的任务也就完成了). 之后转到 STEP 1.

蚁群行进过程的概略算法如图 4 所示.

```
Algorithm
  // Vk is the collection of unconnected pins
  while (Vk ≠ Φ){
    // path condition detecting
    for each p in Vk {
      if(ant colony and p are on a line){
        try to connect p by this line;
      }
      if(ant colony is near p){
       if(there isn't path to p){
        routing fails;
        stop routing;
       }
       if(there is only one path to p ){
        // advance multisteps at once
        connect p through this path;
       }
      } // if (ant colony is near p)
    } // for each p in Vk
    if(connected pins in above checking){
      delete those pins from Vk;
      continue;
    }
    // determine to wait or continue
    if(determine to wait){
      wait;
      continue;
    }
    calculating attraction;
    select direction for next step;
    if(need generate more ant colony){
      generate new ant colony;
    }
    move one step;
    // cp is ant colony's current position
    if(cp ∈ Vk)
      delete cp from Vk;
    }
  } // end of while (Vk ≠ Φ)
end
```

图 4　蚁群行进过程的概略算法

3 IACR-S1 算法的编程实现

3.1 蚁群行进机制

在IACR-S1算法编程实现中，每个蚁群实际上是有一个线程来实现的. 考虑到算法执行一般采用单CPU计算机，因此各蚁群的行进机制采用异步方式. 各蚁群均选定了其初始位置后，按一随机次序先后出发(不是同时出发)；每个线网的蚁群均启动之后，蚁群主要受停等机制和操作系统的线程调度策略影响，异步前进.

3.2 编程语言的选择

蚁群布线系统特别适合于用JAVA语言实现. 将各蚁群均视为对象(线程)，它们异步行进时可采用多线程机制；对每个线程来讲，蚁群在每个位置检测到的信息和位置、引力计算结果所形成的复杂数据需安全地封装；所有这些功能都是JAVA所具有的. 利用JAVA的这些特点，使编程实践非常接近人脑思维，有利于编程效率的提高. 此外，JAVA独具的内存自动管理功能，使编程人员能从繁琐的内存管理工作解脱出来；结合JAVA的例外处理机制，可使程序调试和运行的稳健性大为提高. 事实上，在程序反复多次修改、运行的过程中，我们从未遇到系统崩溃的情况. 最后，JAVA语言的跨平台特征使我们的程序易于在不同的软硬件平台上运行，便于与各种基准程序布线结果相比较.

4 开关盒布线算例

因篇幅所限，这里只举两个应用IACR-S1布线的实例：其一是高密度开关盒Augmented Dense Switchbox例；其二是开关盒样本Sample Switchbox例. 算例的布线结果分别见图5和图6，同其他基准程序(Benchmark)的比较结果列于表1和表2.

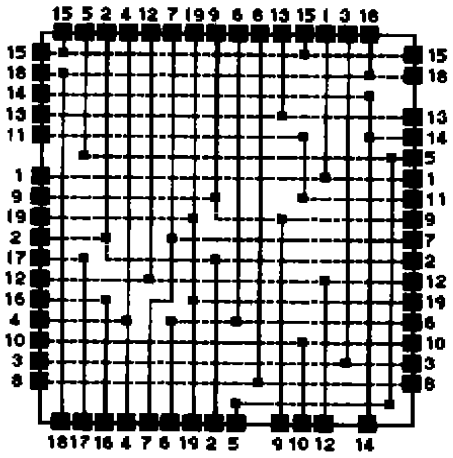

图5 IACR-S1对高密度开关盒的布线图

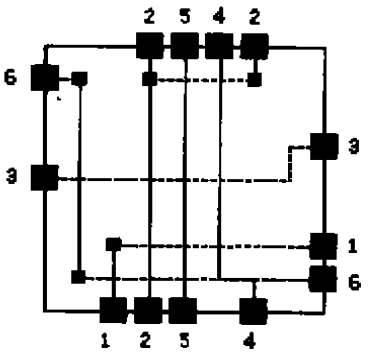

图6 IACR-S1对开关盒样本的布线图

表 1　各种布线算法对高密度开关盒的比较

布线算法	过孔	线长	时间/s
GSR[6]	36	529	?
MIGHTY[4]	32	530	?
BEAVER[5]	27	529	1
GAP[3]	29	529	2281
IACR-S1	31	529	3

表 2　各种布线算法对开关盒样本的比较

布线算法	过孔	线长	时间/s
MIGHTY	5	60	?
WEAVER[7]	4	60	73
BEAVER	3	60	1
IACR-S1	5	60	1

由该二表可见：在总线长和过孔数大体相近的情况下，IACR-S1 的计算时间是最短的. 理由有二：首先，其他基准用的是 C 语言，而用 JAVA 语言时，因现时装载 JAVA 解释程序的芯片硬件尚未普及，它运行起来要比 C 语言慢得多，一般认为运行时间要长一、二十倍；其次，其它基准运行时间多半是在 SUN 工作站上测得的，而我们的程序是在微机(Pentium 166)上运行的，速度自然要打一个折扣. 这样，若牺牲一点运行时间，就可以进一步运用一些优化策略，获得更好的布线结果. 在该二算例中，我们选停等因子 $W_0=0.9$，τ_w 取为 20ms，引力常数 G 取为 500；这些参数均未作过优化处理，故获得更优结果是有潜力的.

5　结论及进一步的工作

本文建议的智能仿生算法 IACS 被证实可用于开关盒布线. 通过算例结果可知其计算复杂度较低，因此今后还可融入针对具体问题的约束优化条件，借以发展出各类性能驱动的通道布线算法，目前我们正在这个方向上作努力，待有结果后再作报道.

参　考　文　献

[1] M. Dorigo, Optimization, Learning and Natural Algorithms, Ph. D. dissertation, DEI, Politecnico di Milano, Italy, 1992 (in Italian).

[2] M. Dorigo and L. M. Gambardella, IEEE Trans. Evolutionary Computation, 1997, 1(1): 53～66.

[3] J. Lienig, IEEE Trans. Evolutionary Computation, 1997, I(I): 29～39.

[4] Y. Shin and A. Sangiovanni-Vincentelli, IEEE Trans. Computer Aided Design, 1987, CAD-6(6): 942～955.

[5] James P. Cohoon and Patrick L. Heck, IEEE Trans. Computer Aided Design, 1988, 7(6): 684～697.

[6] W. K. Luk, the *VLSI Journal*, *1985*, 3, *129*～149.

[7] R. Joobbani and D. P. Siewiorek, IEEE Design & Test, 1986, 3(I): 12～33.

Ant Colony Switchbox Router Based on Coordination Mechanism

Zhuang Changwen, Fan Mingyu, Li Chunhui, Yu Juebang

(*Department of Optoelectronic Technology, University of Electronic Science and Technology of China, Chengdu* 610054)

Received 18 March 1998, revised manuscript received 11 May 1998

Abstract An ACS (Ant Colony System) algorithm based on cooperative learning mechanism has been successfully used to solve the TSP probem. An Intensified ACS (IACS) algorithm is proposed in this paper based on the ant colony's cooperative working mechanism. A switchbox router (IACR-S1) is designed based on the IACS Algorithm. The router implemented by JAVA language is shown to be of low computation complexity as compared to other benchmark examples such as MIGHTY , WEAVER , GAP , etc..

EEACC: 1130B, 2210B