计算机科学 1999Vol. 26№. 6

并行遗传算法研究进展

The Advance on Research of Parallel Genetic Algorithm

王洪燕　杨敬安
（合肥工业大学人工智能研究所　合肥 230009）

Abstract Parallel Genetic Algorithm is an important branch of Evolutionary Computing and more and more AI specialists pay attention to it because of its simplified topologies and robust search ability. In this paper, we discussed theoretic and application achievement of Parallel Genetic Algorithm, analyzed their advantage and disadvantage, and point out its future research direction.

Keywords Parallel GA, Global parallel GA, Coarse-grained parallel GA, Fine-grained parallel GA, Hybrid parallel GA

1 前言

遗传算法是一种模仿生物进化机制的自适应随机搜索方法，是由美国密执安大学的 Holland 教授在 70 年代提出来的。作为解决大规模优化问题的一种有力的工具，已成功地应用于商业，工程及科研领域。一般情况下，遗传算法能在一个合理的时限内求得问题的优化解，但求解大型的结构复杂的问题，对运算时间的需求就会迅速增加，因而为了减少运算时间，提高解的质量，对并行遗传算法的研究已成为当务之急。

本文主要综述几类较重要的并行遗传算法。首先简单论述了标准遗传算法的基本原理，指出算法中存在的基本困难；然后重点论述了并行遗传算法的基本原理，以及全局并行算法、粗粒度并行算法、细粒度并行算法和混合并行算法的发展历程、应用成果和研究进展，指出各种算法的优点和局限性，并强调漂移率和通讯拓扑是设计并行算法的关键。我们认为混合并行遗传算法最有发展潜力，也是并行遗传算法领域的研究方向及前沿。

2 标准遗传算法

Holland 的遗传算法通常称为标准遗传算法，它主要由选择、交叉和变异三个算子组成，分别模仿达尔文进化过程中的自然选择和群体遗传过程中发生的交配和突变等现象，构成了遗传算法的基本操作。遗传算法应用于实际问题时，要对优化问题进行编码，称为个体，个体的集合称为群体，每一个个体都表示问题的一个潜在解。遗传算法以随机产生的一群初始解（初始群体）为开始，通过使用遗传算子对初始群体进行操作，使初始群体一代一代向最优解进化。

选择机制基于适者生存理论，它应用在群体中的每一个个体按照其适应值进化到下一代群体的过程中。选择的具体方法可采取多种形式，一般总是保证当前群体中具有较高适应值的个体以较大的概率通过复制和繁殖生成新的群体。

标准遗传算法利用交叉和变异算子对解空间进行搜索。交叉算子是二者中的主要算子。交叉算子组合了父辈个体的特征，是产生新个体的主要途径，它和选择算子相结合，构成算法中交换信息的重要方法，同时也将强搜索能力赋予遗传算法。

已经证明遗传算法收敛到唯一解的时间和群体的规模有很大关系，Goldberg 和 Deb 实验了不同的选择方法，较快的算法的时间复杂度为 O(nlogn)，其中 n 为群体中个体的个数，但是较大规模群体能

王洪燕　博士生，主要研究兴趣为计算机应用、进化计算、人工智能。**杨敬安**　教授，博士生导师，IEEE 高级会员，纽约科学院 Fellow Academy Member，主要研究兴趣为图象理解、模式识别、计算机视觉、人工智能与机器人以及多传感器融合与集成。

增加遗传算法全局寻优能力和优化解的质量[16]。标准遗传算法中的这一对矛盾也是并行算法需要首先解决的问题。

3 并行遗传算法

并行算法的基本思想是将一个复杂的任务分解为多个较简单的子任务，然后将各子任务分别分配给多个处理器并行执行求解。这种分而治之的思想可以用不同的方法和途径实现，直接导致了各种不同类型的并行遗传算法。并行遗传算法可以分为四大类：全局并行遗传算法、粗粒度并行遗传算法、细粒度并行遗传算法和混合并行遗传算法。

全局并行遗传算法(GPGA)提出的时间最早，这类算法的群体只有一个，个体的适应值评估和遗传算子操作是并行执行的(见图1)。因为群体的个数只有一个，选择的范围是所有个体，并且个体间交叉也和串行算法一样是随机进行的，所以GPGA的本质没有太大改变。这种算法的特点是易于实现并在通讯开销不占主要地位的前提下可显著提高运算速度[6]。

粗粒度并行遗传算法(CPGA)中的群体被分解为多个子群体或群落(demes)，大部分时间子群体独立执行进化，子群体间在运行过程中可进行个体交换，这种现象又称为漂移(migration)。与标准遗传算法相比，CPGA在操作方法上有了根本性改变(见图2)。CPGA又称其为"孤岛"并行算法或分布式遗传算法。

第三种算法是细粒度并行算法(FPGA)，它的子群体的划分更精细，理想的情形是每个处理单元仅处理一个个体(如图3)，这种结构较适合于巨型机，但也可以在任何具有多处理单元的机器上实现。

将三种基本方法组合，就构成了混合并行遗传算法(HPGA)。HPGA的主要优点是可以选择利用各算法的优点，提高解的质量和收敛速度。

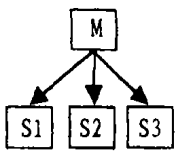


图1 GPGA结构图

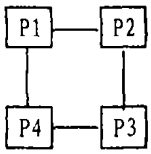


图2 CPGA结构图

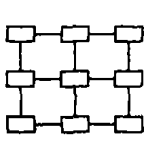


图3 FPGA结构图

Bethke于1976年对并行遗传算法进行了早期的研究工作[3]，他给出了传统遗传算法和基于代沟的遗传算法(generation gap based GA)的(全局)并行遗传算法，报告中指出并行算法对SIMD计算机处理器利用率是100%(忽略通讯部分)，他还指出了传统基于梯度的优化方法在并行化计算中处理器利用率方面存在的瓶颈问题。Grefenstette于1981年提出了四个并行GA模型，并讨论了有关"漂移"概率，拓扑结构，及其对避免早熟所起的作用。

3.1 全局并行遗传算法(GPGA)

GPGA模型中的群体不进行分割，而是作为一个整体进行进化计算，个体的适应值计算和遗传算子的操作是并行的；个体间体现出一种竞争，优的个体得以保留或遗传至下一代，劣的个体被淘汰，选择和交叉等遗传算子的应用范围是整个群体。

因为单个个体的适应值计算与其它个体无关，计算评估不需要个体间通讯，所以GPGA通常将个体的适应值计算并行化。GPGA又分为同步算法和异步算法。同步算法中代的进化需等待所有个体的适应值计算完毕，这种算法的运行模式和标准GA的基本运行模式较相似；异步算法同标准GA有较大差别，"主"处理器并不等待较慢处理器，代间分隔不明显，目前GPGA多采用同步模型。

GPGA对计算机的体系结构没有严格的要求，它可以运行在共享存储和分布存储的计算机上。共享存储计算器的各个处理器可以从共享存储区获取个体进行适应值计算，然后将结果写回存储区，处理过程中处理器间不会冲突(不需要进行同步)。在分布存储计算机上，由"从"处理器负责个体适应值计算，由"主"处理器负责个体发送，收集，然后应用遗传算子产生新一代群体。两种类型计算机通讯负载大体相当，对后者来说，随"从"处理器的增加，会相应增加通讯量，但也降低了每个"从"处理器的负载，Cantu-Paz的研究结果表明，存在一最佳"从"处理器数目[6]。

Fogarty和Huang于1991年在一台分布存储计算机上实现了一个CPGA算法来进化一组推理规则以进行问题求解[9]。他们实验了多种拓扑结构来减少通讯开销，得出的结论是拓扑结构对通讯开销影响不显著，随处理器个数增加会减少问题求解的时间，通讯开销所占比重也会增加。

在Hauser和Manner的实验中，使用了三种不同的计算机，但只有在NERV多处理器机上获得了较好的加速(6个处理器，加速倍数为5)，他们认为其原因是NERV机的通讯开销较小，而其他两种机型(SparcServer和KSRI)并未获得加速是因为其线程-处理器的调度控制功能弱、决策功能不完善[17]。

总之，GPGA易于实现并适用于适应值计算量

较大的应用场合;这种算法还保持了传统GA的搜索特性,许多传统GA的理论结果可直接推广到GPGA。

3.2 粗粒度并行遗传算法(CPGA)

CPGA算法的重要特征是具有较大的群落和漂移率,作为最流行的并行算法,CPGA的研究成果极为丰富。

最早对CPGA进行系统研究的是Grosso,他的研究目标是模拟进化群体的各并行子群体间的相互作用。Grosso使用了双倍体编码,进化群体分为5个群落,群落间以固定周期交换个体(漂移)。Grosso发现,将大群体划分为群落可快速提高群体的平均适应值,这也证明了群体遗传学的重要理论:较小体积群落中的个体间遗传传播速度比大体积群落中的个体传播要快。Grosso还发现,当各群落孤立时,解的适应值比非并行算法还差,漂移率较低时,解的质量接近孤立时的质量,而当漂移率增大到一定程度时,各群落的解的质量与非并行情形相当。这说明,存在一漂移率阈值a,漂移率低于此阈值时,解的质量等于孤立群落的解的质量,大于此阈值时,解的质量等于非并行情形下解的质量。

Tanese在他的算法中引入4维超立方体拓扑结构,沿超立方体的每一维,个体漂移周期恒定,漂移个体选自于群落中的最优个体,群落中的最差个体将被替换掉[25]。第一个实例中,漂移每五代发生一次,处理器个数和漂移个体所占总体比例是可变的,结论是存在两个百分比值,使得并行算法解的质量不低于串行算法。第二个实例中,每个群落中的变异和交叉概率动态改变用于寻求优化算法参数并平衡算法的"搜索"和"创新"能力。第三个实例的结论是漂移率太大或太小都会影响解的质量。

Tanese又于1989年更全面地用实验研究了飘移频率和漂移个体百分比问题。她比较了串行GA和并行GA(有通讯和无通讯情形)。所有算法的群体由256个个体组成,计算以500代为上限。无通讯CPGA(r个群落,每群落个体数为n)可以获得和串行GA(群体大小为rn)同等质量的最优解,但是最终群体的平均适应值较小。对有通讯的CPGA,最终解的平均质量超过了串行GA(可解释为串行GA发生了早熟现象)。

较近的研究成果是Belding完成的,在Belding的算法中,个体漂移的目标群落是随机的,所得结果却类似:漂移有助于获得全局最优解[1]。

综上所述,可以得出CPGA的如下特点:1)原理清晰,易于理解,可以看作是串行GA的直接扩展;2)对实验环境要求较低,可以用模拟并行机的方法进行实验;3)可以充分利用现有的关于串行GA的资源。

80年代末90年代初,并行GA的收敛速度和工作原理成为研究重点。也正是在这个时期,出现了首批并行GA的理论研究成果。以下将进一步讨论CPGA的理论研究状况,以及算法性能与漂移和拓扑结构的关系的研究,同时也涉及CPGA的应用问题。

应用大型、复杂的标准测试函数作为目标函数是CPGA趋于成熟的重要标志。Muhlenbein, Schomisch和Born实现了一个并行GA算法用于搜索几个标准测试函数的全局最优解,他们所使用的测试函数随后被其他学者应用于各自的研究工作中[13],Muhlenbein,Schomisch和Born在他们的算法中使用了局部优化策略,他们并没有讨论局部优化和并行化,其中哪一种算法对提高算法性能贡献更大些。

3.2.1 *理论成果* 前面部分所列举的研究成果都是基于实验得出的,事实上,为了充分挖掘CPGA的潜能,从理论的高度去研究它是必须的。

最初对并行GA的理论基础进行研究的是Petty和Leuze。他们得出了并行GA的模式定理,大部分适于标准GA的结论也适用于并行GA。

在理论方面,一个重要的问题是确定并行GA能否及如何获得比标准GA更优质的解。Whitley和Starkweather针对这两个问题得出两个重要结论:①孤立进化的群落可能生成各不相同的解,大部分情况下,带有漂移的并行GA算法会得到更为优质的解;②若部分解重组产生了大量适应值较低的个体,使用标准GA算法会更有效。

3.2.2 *漂移* 并行GA走向成熟的另一标志是研究工作越来越专注于算法的某一方面,许多文献专门就各种漂移模式进行研究来提高算法的效率。

CPGA的漂移可分为如下两类:同步模式,漂移总是按一固定周期发生;异步模式,仅当某一条件满足时,漂移才会触发。Grosso在其博士论文中给出了异步漂移的实例,其漂移条件是群体接近收敛。

为避免早熟现象,Braun给出的算法仅当群落完全收敛时才触发漂移[5],Munetomo,Takai和Sato也使用了类似的思想,而Cantu-Paz和Goldberg采用全联结拓扑建立了一个理论模型来预测

使用上述漂移模式的解的质量[7]。

选择合适的时机显然是漂移操作的一个关键问题，如漂移过早，质量差的漂移个体会影响整个群体的总体搜索方向，还会浪费通讯资源。

Marin，Trelles-Salazar 和 Sandoval 于 1994 年提出了一种中心控制模式的漂移算法[22]，在计算运行过程中，由从处理器周期性地将各自群落的最优解发给主处理器，再由主处理器选择最优个体并将其以广播的方式发往各从节点。试验结果显示用六个节点可获得线性加速，同时上述算法由于通讯开销不大，容易扩展应用到大型复杂的问题。

3.2.3 *通讯拓扑* 群落间的拓扑联结是 CPGA 研究领域较易被忽视的问题。事实上，联结拓扑是影响 CPGA 性能的重要因素，决定着群落间个体漂移效率。如果采用紧致联结方式(或联结半径较小)，优化个体就会快速传播并在整个群体中占主导地位；反之，如采用松散联结方式(或联结半径较大)，个体传播速度较慢，群落孤立会产生不同的优化解，这些解在后续运行过程中可能重组产生更优解。通讯拓扑还会影响漂移的开销，例如紧致的联结能提高个体的组合效率，也会增加通讯开销。

一般情况下，CPGA 使用静态的拓扑联结，其结构在进化过程中不会改变，常用结构为：超立方体和环状结构[13]。另一种联结方式称是动态拓扑：一个群落并不局限于和几个固定的群落进行通讯，其通讯对象是动态可变的，例如可选满足一定准则的群落作为目标群落。动态选择目标群落的动机是增强漂移效果，其准则包括：群体的离散程度和群落间的基因型距离(或个体代表间的距离)。

3.2.4 *粗粒度并行遗传算法的应用* CPGA 的代表性应用包括组合优化问题、负荷平衡问题和 VLSI 电路合成问题。CPGA 在图分割问题上的应用可见 Bessiere 和 Talbic[1991]，Cohoon，Martin 和 Richards[1991c]，Talbi 和 Bessiee[1991][2,8,24]，较突出的算法是 Levine 提出的[19]，所解决问题分别包含 36,699 和 43,749 个整型变量，算法的性能(解的质量和收敛迭代次数)随群落的增加而提高。

3.3 细粒度并行遗传算法(FPGA)

细粒度并行 GA 模型的群体被划分为许多小的群落，群落间大量频繁的通讯保证优化解传到群体的各部分，选择和交叉算子仅在群落内部执行。

Robertson 在 CM-I 上用 FPGA 实现了一个分类器系统。父个体的选择操作以及用于替换、交叉的分类器的选择操作都并行处理。算法的执行时间可以达到与分类器的个数无关(CM-I 的处理单元上限为 16K)。

ASPARAGOS 系统由 Gorges-Schleuter 和 Muhlenbein 引入来解决一些困难的组合优化问题。ASPARAGOS 采用了一种首尾相连的阶梯状的群体结构。Gorges-Schleuter 在随后的工作中又引入了一种线性群体结构并比较了不同的交叉策略。ASPARAGOS 使用了局部爬山算法来对个体进行求精。虽然很难确定群体的空间结构和爬山法间到底谁起了决定性的作用，但所有的实验结果显示出 ASPARAGOS 是一个非常有效的优化工具。

在 Manderick 和 Spiessens 提出的 CPGA 算法中，群体呈二维网格状分布[21]。选择和交叉操作在相邻个体间发生。实验结果显示算法性能随群落体积的增大而降低。极端情况下，若群落的体积充分大，算法等同于一个大的随机群体下执行标准 GA。在后来的工作中对上述算法进行了理论分析，对包含 s 个串长为 l 的群体，时间复杂度依选择的模式不同，分别为 $O(s+l)$ 或 $O(s(\log s)+l)$。

Anderson 和 Ferris 在两个环状、超立方体、网状和"孤岛"拓扑上实验了多种 FPGA 算法，群落间交迭个体仅有一个，对于他们的实例来说，环状和"孤岛"状结构是最优的。Baluja 分别比较了线状和二维网状结构，网状结构在所有被测问题上都优于线状结构。

Sarma 和 Jong 对选择操作并行化的邻域的大小与形状进行了研究，认为邻域半径与全网格半径之比是控制选择操作的关键参数，优化解的传播速度随这一比值变化而波动[23]。

FPGA 也已有许多成功的应用实例，如二维装箱(bin packing)问题和满意求解问题[18]。Job shop Scheduling 问题也是 FPGA 应用活跃的领域。

大量文献对 CPGA 和 FPGA 的算法性能进行了比较[13]。根据判断方法和算法环境的不同，CPGA 和 FPGA 各有所长。Gordon，Whitley 和 Bohm 认为 FPGA 算法的关键路径较 CPGA 的要短，这表明在超级计算机上，如有足够的处理器，就有可能做到算法速度与群体大小无关！这只是一个理论上的结论，没有考虑到通讯带宽和内存需求。Gordon 又于 1994 年研究了各种并行 GA 模型内存引用的空间结构，所得结论有助于为各种模型选择合适的应用平台[12]。

3.4 混合并行遗传算法(HPGA)

若将 CPGA 和 FPGA 相结合便得到一种混合

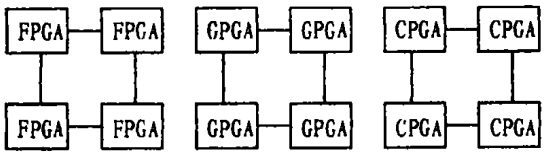
并行 GA 算法，这种算法要解决的首要问题是降低其复杂度。一般情况下，HPGA 算法拓扑呈层次结构(图 4)，以 CPGA 为顶层，以 FPGA 为底层(见图 4(a))，如 Gruau 提出的算法：每个群落放置在一个二维的网格上，群落间以二维的环状结构连接，群落间个体漂移以固定周期进行[15]。对神经网络训练问题，实验报告取得了理想结果。

Gorges-Schleuter 于 1997 年改进了 ASPARAGOS 系统，新系统使用环状结构代替了原系统中的梯状结构，各子群体在运行将收敛时可接收其它子群体中的优化个体，当全部收敛时，每个子群体贡献出最优个体和次优个体做最后一轮进化[14]。

Lin，Goodman 和 Punch 使用了同上述算法类似的环状拓扑来联结各子群体，每个子群体都呈环面状拓扑[20]。他们把这一混合算法同一个 FPGA 算法和三个 CPGA 算法进行了比较，其中两个 CPGA 算法使用了环状拓扑结构(一个算法的群落个数为 5，群落所含个体个数可变；另一算法群落个数可变，各群落恒包括 50 个个体)。以加工车间调度问题做为测试实例，他们得出的结论是 FPGA 算法的性能排在最后，对于 CPGA 算法，增加群落的数目要比增加群体的体积更有效地提高算法的性能，HPGA 算法无论从哪个方面看都具有最优性能。

第二种混合并行化 GA 算法的途径是以 CPGA 作为顶层拓扑而以 GPGA 来进化各子群落(见图 4(b))，群落间的漂移同 CPGA 算法，个体的适应值计算则是并行的。这种算法较适用于目标函数计算复杂费时的应用问题。实例可参考 Bianchini 和 Brown[1993][4]，解的质量超过 GPGA 算法，进化时间优于 CPGA 算法。

第三种方法是在顶层和底层都使用 CPGA 算法(见图 4(c))，其特点是底层拓扑联结更紧密，漂移更频繁[11]。



(a) 底层为FPGA　(b) 底层为CPGA　(c) 底层为CPGA

图 4 混合算法结构图

结论　并行 GA 是一类较标准 GA 更为复杂的算法，至今存在许多尚未解决的问题：①如何确定漂移概率，提高算法的性能；②如何降低通讯开销并提高拓扑紧致性来改进解的质量；③如何确定群落的个数，提高算法的稳定性。

在并行 GA 研究领域，占主导地位的是 CPGA 算法，这也是本文的中心议题。由于算法的复杂性，许多文献都选择了算法的一个独立因素进行研究的策略。本文也综述了全局并行算法、粗粒度并行算法、细粒度并行算法和混合并行算法的发展历程、应用成果和研究进展，指出各算法的优点和局限性，并指出 HPGA 是提高算法性能的最有效途径。

从众多的实例分析来看，并行 GA 算法在解决大型复杂问题方面，不仅可以提高解的质量，同时也可以提高进化速度，克服 GA 算法的速度瓶颈。

参 考 文 献

1 Belding T C. The distributed genetic algorithm revisited. In: Eschelman L. ed. Proc. of the Sixth Intl. Conf. on Genetic Algorithms. San Francisco, CA: Morgan Kaufmann, 1995. 114～121

2 Bessiere P, Talbi E-G. A parallel genetic algorithm for the graph par titioning problem. In: ACM Int. Conf. on Supercomputing ICS91. Cologne, Germany, 1991

3 Bethke A D. Comparison of genetic algorithms and gradient-based optimizers on parallel processors: Efficiency of use of processing capacity: [Tech. Rep. No. 197]. Ann Arbor, MI: University of Michigan, Logic of Computers Group, 1976

4 Bianchini R, Brown C M. Parallel genetic algorithms on distributed-memory architectures. In: Atkins S, Wagner A S. eds. Transputer Research and Applications 6. Amsterdam: IOS Press, 1993. 67～82

5 Braun H C. On solving travelling salesman problems by genetic algorithms. In: Schwefel H-P, Monner R. eds. Parallel Problem Solving from Nature. Berlin: Springer-Verlag, 1990. 129～133

6 Cantu-Paz E. Designing efficient master-slave parallel genetic algorithms: [IlliGAL Report No. 97004]. Urbana, IL: University of Illinois at Urbana-Champaign, 1997

7 Cantu-Paz E, Goldberg D E. Modeling idealized bounding cases of parallel genetic algorithms. In: Koza J, et al. eds. Genetic Programming 1997: Proc. of the Second Annual Conf. same to[1], 1997a. 353～361

8 Cohoon J P, et al. A multi-population genetic algorithm for solving the K-partition problem on hyper-cubes. In: Belew R K, Booker L B. eds. Proc. of the 4th Intl. Conf. on Genetic Algorithms. same to[1], 1991c

9 Fogarty T C, Huang R. Implementing the genetic algo-

rithm on transputer based parallel processing systems. Parallel Problem Solving from Nature, 1991. 145～149

10 Goldberg D E. Genetic and evolutionary algorithms come of age. CACM, 1994, 37(3): 113～119

11 Goldberg D E. Personal communication. June 1996

12 Gordon V S. Locality in genetic algorithms. In: Schaffer J D, et al. eds. Proc. of the First IEEE Conf. on Evolutionary Computation, Volume 1 Piscataway, NJ: IEEE Service Center, 1991. 428～432

13 Gordon V S, Whitley D. Serial and parallel genetic algorithm as function optimizers. In: Forrest S. ed. Proc. of the 5th Intl. Conf. on Genetic Algorithms. same to [1], 1993. 177～183

14 Gorges-Schleuter M. Asparagos96 and the traveling salesman problem. In: Back, T. ed. Proc. of the 4th Intl. Conf. on Evolutionary Computation. New York: IEEE Press, 1997

15 Gruau F. Neural network synthesis using cellular encoding and the genetic algorithm. Unpublished doctoral dissertation, L'Universite Claude Bernard-Lyon I, 1994

16 Harik G, et al. The gambler's ruin problem, genetic algorithms, and the sizing of populations. In: Back, T. ed. same to [14], 1997. 7～12

17 Hauser R, Manner R. Implementation of standard genetic algorithm on MIMD machines. In: Davidor Y, et al. eds. Parallel Problem Solving from Nature, PPSN Ⅱ. Berlin: Springer-Verlag, 1994. 504～513

18 Kroger B, et al. Parallel genetic packing on transputers. In: Stender J. ed. Parallel Genetic Algorithms: Theory and Applications. Amsterdam: IOS Press, 1993. 151～185

19 Levine D. A parallel genetic algorithm for the set partitioning problem: [Tech. Rep. No. ANL-94/23]. Argonne, IL: Argonne National Laboratory, Mathematics and Computer Science Division, 1994

20 Lin S-H, et al. Investigating parallel genetic algorithms on job shop scheduling problem. In: Angeline P, et al. eds. 6th Intl. Conf. on Evolutionary Programming. Berlin: Springer Verlag, 1997. 383～393

21 Manderick B, Spiessens P. Fine-grained parallel genetic algorithms. In: Schaffer J D. ed. Proc. of the 3rd Intl. Conf. on Genetic Algorithms. San Mateo, CA: Morgan Kaufmann, 1989. 428～433

22 Marin F J, et al. Genetic algorithms on LAN-message passing architectures using PVM: Application to the routing problem. In: Davidor Y, et al. eds. Parallel Problem Solving from Nature, PPSN Ⅱ. Berlin: Springer-Verlag, 1994. 534～543

23 Sarma J, Jong K D. An analysis of the effects of neighborhood size and shape on local selection algorithms. In: Parallel Problem Solving from Nature IV. Berlin: Springer-Verlag, 1996. 236～244

24 Talbi E-G, Bessi'ere P. A parallel genetic algorithm for the graph partitioning problem. In: Proc. of the Intl Conf. on Supercomputing. Cologne, 1991

25 Tanese R. Parallel genetic algorithm for a hypercube. In: Grefenstette J J. ed. Proceedings of the 2nd Intl. Conf. on Genetic Algorithms. Hillsdale, NJ: Lawrence Erlbaum Associates, 1987. 177～183

(上接第 56 页)

沿用上述的推导过程，也可得出十进制实数编码的模式定理，但总感觉不很完善，尚有待进一步深入研究。

参 考 文 献

1 席裕庚，等．遗传算法综述．控制理论与应用，1996，13(6)：697～708

2 刘勇，康立山，陈毓屏．非数值并行算法—遗传算法．北京：科学出版社，1995

3 Srinivas M, Patnaik L M. Genetic Algorithms: A Survey

4 Vose M D. Generalizing the Notion of Schema in Genetic Algorithms. Artificial Intelligence, 1991, 50: 385～396

5 Goldberg D E. Real-Coded Genetic Algorithm. Virtual Alphabets and Blocking. Complex Systems, 1991, 5: 139～167

6 Wright A H. Genetic Algorithms for Real Parameter Optimization. In: Rawlins G J E, ed. Foundations of Genetic Algorithms. San Mateo, CA: Morgan Kaufmann, 1991. 205～218

7 Michalewicz Z, et al. A Modified Genetic Algorithm for Optimal Control Problems. Computers Math. Applic., 1992, 23(12): 83～94