学校编码：10384　　分类号_____密级 _____

学　　号：200228041　　UDC_____

厦 门 大 学

硕 士 学 位 论 文

并行演化算法及其在神经网络中的应用研究

Parallel Evolutionary Algorithm and Its Application on Neural Network

许有准

指导教师姓名：曾 文 华 教授

专 业 名 称：计算机应用技术

论文提交日期：2005 年 5 月

论文答辩日期：2005 年　　月

学位授予日期：2005 年　　月

答辩委员会主席：_____

评　　阅　　人：_____

2005 年 5 月

厦门大学学位论文原创性声明

兹呈交的学位论文，是本人在导师指导下独立完成的研究成果。本人在论文写作中参考的其他个人或集体的研究成果，均在文中以明确方式标明。本人依法享有和承担由此论文而产生的权利和责任。

声明人（签名）：许有雄

2005 年 5 月 8 日

摘　要

演化计算(Evolutionary Computation, EC)源于生物进化启示，是一类仿生的启发式搜索方法。通过使用简单的编码技术来表达复杂的问题现象，并使用多个个体构成的种群对解空间并行地搜索，使得在算法的一次运行之后，能够找到Pareto最优集的几个成员。演化算法能够在不要求函数连续、可微、单峰等性质的情况下，实用地找到问题的近似全局最优解。演化计算的这些优点，使其成为求解优化问题的主要工具，而且广泛地应用于：函数优化、机器学习、约束满足问题、组合优化，尤其是具有大量 参数的繁杂优化问题，其解析解难以获得。演化算法包括遗传算法、演化 规划、演化策略和遗传程序设计。

在研究当前为避免早熟而采用的多样性保持方法，以及改善演化计算效力策略的基础上。本文提出了基于区域分解的异步并行细胞状演化算法和逼近模糊交叉算子。算法性能的理论分析表明，其具有高可扩展性。对复杂问题的优化结果表明，结合异步并行细胞状模型和逼近模糊交叉算子的演化算法，能够有效平衡探-搜索比例，以更低的数值代价快速地收敛到全局最优解，且具有高可靠性和精确度，显著地提高演化算法的收敛速度及效力。

在研究如何更有效地使各子种群相互协作地进行搜索的基础上，本文提出了基于聚类异步孤岛的并行演化算法，以及聚类局部搜索算子和岛内 适应值函数。该算法可以通过调控局部适应值景象，减少重叠搜寻、增加协作，以提高搜索效率。与区域分裂模型比较，基于聚类异步孤岛的并行演化算法具有更好的可扩展性，以及易于并行实施。

本文还提出了两层并行演化神经网络，并将该网络用于建立航煤干点软测量模型，同时对现场工业数据进行应用仿真研究。结果表明，该两层 并行神经网络具有较强的通用性以及良好的泛化能力。

关键词：演化计算；并行计算；效力；数值代价

Abstract

Evolutionary computation is inspired from biology evolution. It's a class of heuristic search technology. By using simple coding technology to represent phenomena of complex problem, the individuals of population parallel explore and exploit the search space. After the algorithm run for one time, it can find several members of the Pareto optimum set. Evolutionary algorithm can find the near global optimum solution which is not continuous, differentiable and unimodal. Thus, evolutionary algorithm becomes the major tool for solving optimization problem. It is widely applied to function optimization, machine learning, satisfactory problem, combinatorial optimization and especially complex optimization problem with lots of parameters, whose analytic solution is difficult to obtain. Evolutionary algorithms include genetic algorithms, evolution programming, evolutionary strategies and genetic programming.

Base on study of various heuristic schemes for keeping the diversity of evolutionary populations, alleviating premature convergence and enhancing the search efficacy of evolutionary computation, a kind of asynchronous parallel cellular evolutionary algorithms (APCEA) that is based on the area partition and an approach fuzzy recombination operator (AFR) are proposed. A theoretical performance analysis reveals the high scalability of APCEA. The experimental results optimizing various classes of test functions indicate that APCEA+AFR can efficiently balance the exploring and exploiting, and local the global optimum solution quickly and accurately. It is a much more competent canonical evolutionary algorithms.

Base on study of various heuristic strategies for keeping subpopulation cooperate efficiently with each other in searching, a kind of asynchronous island parallel evolutionary algorithms (AIPEA) based on the clustering, a cluster local search operator (CLS) and an fitness function of island are proposed. The algorithm can alleviate overlapping search by tuning the fitness landscape to

improve the search efficiency. It is more scalable and implementation-friendly than the partition model.

Finally, two layer parallel evolutionary neural network model is proposed. And a soft-sensor model is built up based on the parallel evolutionary neural network model. The experimental results optimizing large industrial data indicate that the two layer parallel neural network has good generality.

Keywords: Evolutionary Computation; Parallel Computation; Efficacy; Numeric Effort.

目 录

第一章 绪 论

1.1 演化计算的产生及发展

演化计算(Evolutionary Computation, EC)是基于生物进化思想而发展起来的一种通用问题求解方法[1]，其本质上是一类仿生的启发式搜索方法。通过使用简单的编码技术来表达复杂的问题现象，并使用多个个体构成的种群对解空间并行地搜索，使得在算法的一次运行之后，能够找到Pareto 最优集的几个成员。演化算法能够在不要求函数连续、可微、单峰等性质的情况下，实用地找到问题的近似全局最优解。演化计算的这些优点，使其成为求解优化问题的主要工具，而且广泛地应用于：函数优化、机器学习、约束满足问题、组合优化。尤其是具有大量参数的繁杂优化问题，其解析解难以获得。

1975 年，美国 Michigan 大学的 John Holland 及其学生提出了简单遗传算法(Simple Genetic Algorithm, SGA)的框架[2]。然而，传统演化计算的起源可以追溯到 20 世纪 50 年代后期，其中有影响的工作是 Bremermann，Friedberg，Box 等[3]。演化计算最初具有三大分支：遗传算法(Genetic Algorithms, GA)[4][5]，演化规划(Evolutionary Programming, EP)[6][7][8]和演化策略(Evolution Strategies, ES)[9][10]。后来，Koza[11]在遗传算法的基础上又开创性地发展了演化计算的另一个分支：遗传程序设计。演化规划的方法最初是由美国的 Fogel L. J.[12]等作为产生人工智能的一种尝试而提出的。Rechenberg 和 Schwefel[9]为求解主要由试验得来的困难的离散和连续多参数优化问题提出了演化策略。

演化计算是一个新兴学科，近年来发展迅速。至少有两个国际期刊致力于此：《IEEE Transactions on Evolutionary Computation》和《Evolutionary Computation》(MIT Press)。各种AI类杂志不断刊登EC方面的文章，如，《IEEE Transactions on Systems, Man, and Cybernetics》和《BioSystems》(Elsevier)。另外，以演化计算为主题的多个国际会议在世界各地定期召开，如IEEE International Conference on Evolutionary Computation和International Conference on Evolutionary Computation。

1.2 并行演化计算研究现状

1.2.1 理论研究

当前对演化计算所进行的理论研究工作主要包括如下几类：表示理论、操作子理论、结构化算法、收敛性理论、适应值景象理论、一致性理论、工作模型理论和物种形成理论与小生境。其中，结构化算法、一致性理论与工作模型理论和并行演化计算直接相关[13]。

定义表示理论包括所有引导对一个给定遗传型－显型行为进行理解的一切正规解释。模式理论[4][5]便是其中一种。模式理论包括模式定理、隐并行性原理和基因块假说三部分。由Holland建立的遗传算法隐并行性原理[4]认为遗传算法有效处理的模式总数正比于群体规模N的立方。张钹等[14]证明其证明过程存在纰漏，并提出理想浓度模型。文献[15]将模式定理扩展到分布式并行遗传算法领域。

定理（模式定理）[4] 设$P(t)=\{S_1(t), S_2(t), \ldots, S_N(t)\}$表示SGA在第t代时的群体，$P_c$和$P_m$分别为其杂交概率和变异概率，$\delta(H)$、$o(H)$和$\Phi(H, t)$分别表示模式的定义长度、阶数和适应值，$\xi(H, t)$表示第t代群体$P(t)$中包含模式H的个体个数，则有

$$\xi(H,t+1) \geq \xi(H,t) \cdot \frac{\Phi(H,t)}{\overline{\Phi(t)}} \cdot [1 - P_c \cdot \frac{\delta(H)}{L-1} - o(H) \cdot P_m] \quad (1\text{-}1)$$

其中 L 为个体的编码长度。

模式定理给出模式在下一代群体中实例个数的下确界。这对于预测一个给定模式在下一代中的实例个数是相当有用的。但是，模式定理只对二进制编码才适用，对其它编码尚无相应结论。而且，由于模式定理只是给出了下一代群体中模式 H 实例个数的下确界，无法获取遗传算法的复杂性及推断算法收敛与否。

1.2.2 有争议的性能分析

衡量并行演化算法的性能参数包括：运行时间、寻优代价、加速比及评估函数运行速率等，其中尤以加速比最常用。超线性加速比是有争议的，特别是传统研究范围，一些非正规研究导致这一结果。加速比广泛用来衡量并行算法的效率，但其定义却多种多样。传统的加速比[16]定义为 $S_n=T_1/T_n$，其中 T_1 为最好串行算法的最差执行时间，T_n 为并行算法在 n 个处理器上的最差执行时间。由于并行演化算法为非确定性算法，对于这个传统的加速比定义，首要修改的便是要使用按比率考虑的平均时间[16]，即

$$S_n = \overline{T_1} / \overline{T_n} \quad (1\text{-}2)$$

这便是强加速比定义[17]，即加速比采用并行演化算法运行时间与最好的串行算法进行比较。由于难以确定一个串行演化算法是否为最好的算法，以及在分析演化算法过程中，经常需研究大量的问题，此时找出被研究问题的最快算法便是不现实的。故多数研究者通常不采用强加速比定义，而采用弱加速比定义[13]，即采用并行算法运行时间与其相应串行算法运行时间进行比较。弱加速比定义的一个要点为停机准则。对于将停机准则设为预定义的全体搜索个体总数的弱加速比定义，称为预定义代价的弱加速比定义[17]。而对于将解质量纳入停机准则（相同质量的解被求得）的弱加速

比定义，称为基于解的弱加速比定义[17]。Cantú-Paz[16]提倡采用此种定义，它包括两类：正规弱加速比定义和较串行弱加速比定义[18]。较串行弱加速比定义，即加速比采用运行于 n 个处理机的并行演化算法与运行于单一处理器的串行演化算法进行比较。正规弱加速比定义，其比较对象为同一并行演化算法分别在单一处理机上与 n 个处理机上的运行时间。出于公平与有意义的考虑，通常采用正规弱加速比定义。

相应于加速比的一个百分化指标，并行效率定义为（>100%意味着超线性加速比）：

$$\eta_n = (S_n/n)*100\% \tag{1-3}$$

Karp 和 Flatt[17]发明了一个衡量任何并行算法性能的更有效尺度，即算法的串行成份：

$$\Gamma_n = \frac{(1/S_n)-(1/n)}{1-(1/n)} \tag{1-4}$$

较只使用加速比而言其更加敏锐。理想情况下，算法的并行成份应为一常数。当加速比很小时，若对于不同的 n，Γ_n保持为一常数，仍然说结果是好的，因为失去的并行效率归因于问题本身有限的并行性。

事实上，对于并行演化计算来讲，超线性加速比是可能的[17][19]。加速比与处理机个数之间存在一个指数关系[20]，即

$$S_n = n \cdot a^{n-1} \tag{1-5}$$

其中，a 为加速因子（a>1 意味着加速比为超线性），它可以解释超线性加速比。

超线性加速比的获取来源如下：

(1)由于并行演化算法具有内在的启发式多点特性，通过使用更多处理器，改变空间搜索次序，有更高的机会找到更优解；

(2)随着更多处理器的使用，其它资源（如内存、cache 等）相应增加；

(3)遗传算子并行地工作于更多更小的数据结构，其数据结构更适合于

存放在 cache 中，较使用单一主存具有更高的效率。

Donaldson 等人的研究表明，在异构系统中加速比没有理论上限，虽然理论上是可行的，但对异构系统所进行的测试至今未能取得超线性加速比[17]。文献[19]发现在异构系统中，求解问题的数值代价较同构系统大大减少。在繁杂与耗时的优化问题上使用并行演化计算的重要性，不仅仅在于它节省优化时间，还在于它能够找到更好的优化结果[21]。

1.2.3 分类

根据 Cantú-Paz[16]的归纳总结，并行演化算法可分：主从式并行演化算法、粗粒度并行演化算法、细粒度并行演化算法和分层演化算法。但是从当前众多的研究，我们发现并行演化算法又产生了一个新类，即改进的并行演化算法。

主从式模型中，单一大种群中的个体虽并行演化，但选择操作只在主处理器上直接运行[13]。其行为类似于串行演化算法，其探索空间与串行演化算法一样[22]。其并行化包括评估操作，甚至杂交和变异操作，但是选择操作只能串行化[15]。具有非常易于实现、特别适合目标函数需耗大量处理机时间问题的特点。

粗粒度模型[23]又称多种群模型[16]、孤岛模型、分布式模型[15]或迁移模型[24]。细粒度模型又称扩散模型[24]、大规模并行模型。二者同属于结构化种群模型。一个分布式并行演化算法有大量的子种群，细粒度并行演化算法每个子群体只有一个个体。分布式并行演化算法子群体间连接较松散，而细粒度并行演化算法子群体间连接却较紧密。另外，分布式并行演化算法只有为数不多的子算法，而细粒度并行演化算法却有大量这类的子算法。如图 1.1 所示。

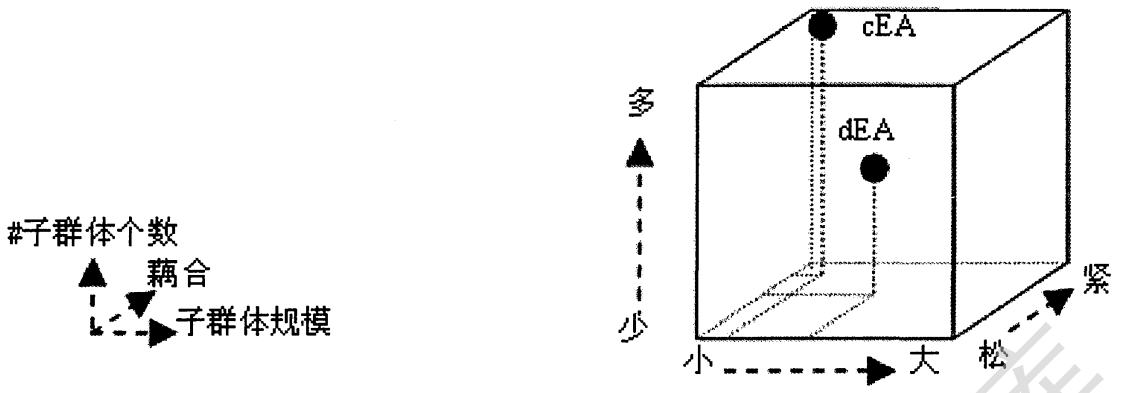

图 1.1 种群结构化演化算法立方

1.2.3.1 粗粒度并行演化算法

粗粒度模型是一个相隔一个纪元（epoch）便进行个体交换的多种群模型.易于在分布式 MIMD 并行计算机上实现。由于通信开销小，非常适合于在集群系统上运行。有许多参数对于指导粗粒度模型搜索相当重要。

(一) 迁移策略

迁移策略决定孤岛间的藕合情况，并很大程度决定并行算法的运行效率。此处，将迁移策略定义成 5 元组[25]：

$$M=(\gamma,\tau,\omega_S,\omega_R,sync) \tag{1-6}$$

其中，

(1) γ 为迁移率，$\gamma\in\{0,1,\ldots\infty\}$。它可以用子群体的百分比来衡量。在实际应用中，迁移率受子种群规模所限，而在理论上，它却是不受限的。

(2) τ 为迁移频率（由评估次数来衡量），$\tau\in\{0,1,\ldots\infty\}$。根据这个定义，如果并行算法在 e 次评估之后结束，那么任何大于此值的迁移频率（即 τ>e）意味着子群体孤立演化。

(3) ω_s 为迁移选择策略。此算子决定拓扑结构及迁移个体。ω_s 涉及两个子群体，并决定任何两个子群体之间的共享个体集。迁移选择可以使用

已有选择算子。

(4) ω_R为替换策略。即如何将迁移个体融入目标群体。

(5) sync 为同步标志。标示着采用何种通信方式。

由于需比较具有不同基本步的模型，故用评估次数来衡量迁移频率，而非演化代数。将迁移频率作为整个群体规模 μ 的函数来研究，即如，0.25μ、0.5μ、1μ、2μ 等。而 τ=0 表示种群之间无连接。

这样，一个迁移算子被用在一个并行演化算法Δ_{par}的通信阶段，表示一个子群体与另一个子群体间的藕合关系。选择算子$\omega_s(\Delta_i, \Delta_j)$决定邻居子算法$\Delta_i$与$\Delta_j$间的共享结构集：

$$\omega_{M\Theta_M}(\Delta_j) = \omega_R \circ \omega_S(\Delta_i, \Delta_j) \left| \forall \Delta_i, \Delta_j \in \Delta_{par} \right. \tag{1-7}$$

迁移率即为共享结构数：

$$\gamma = \left| \omega_S(\Delta_i, \Delta_j) \right| \tag{1-8}$$

应用迁移算子的可能性为：

$$\zeta_M = 1/\tau, \quad \tau > 0 \tag{1-9}$$

许多研究者强烈建议采用异步通信[20][26]。这可通过采用一旦迁移个体到达，便将其融入目标群体的做法实现，避免阻塞 τ 步。

控制迁移算子ω_M的参数集Θ_M由迁移策略 M 所定义。

并行分布式演化算法的再现循环便为孤岛的再现循环与迁移算子的组合，即

$$\omega_d = \omega_M \circ \omega_{island} \tag{1-10}$$

Alba 和 Troya 通过大量的实验发现：

(1) 过度频繁的通信会导致种群多样性被破坏，致使局部收敛，从而降低并行算法的性能[25]。

(2) 选择随机串进行迁移较选择最优个体进行迁移更合适。最优个体往往导致超级个体，破坏种群多样性[18]。随机串的迁移可以避免对中小规模目标子群体的征服性影响[25]。

(3) 子群体间松散的藕合可以使并行算法具有更高的效力[18]。藕合度越高，选择压力越大[20]。

Fernández、Tomassini 和 Vanneschi[27]通过大量实验及前人经验，指出为了取得最好结果，迁移率 γ 应为种群的 10%左右，而迁移频率 τ 应取 5μ 至 10μ 之间。

(二) 连接拓扑

拓扑结构决定优秀个体的传播速度。由于并行演化算法的演化过程通常超过两个纪元，故若子种群规模确定，仍然面临拓扑结构的挑选。各子种群间的连接拓扑包括完全隔离、单向环、双向环[28]、网格、超立方体和圆锥体等[29]。为了取得相当的算法效力，子种群越小需要的连接度越高[16]。研究表明，对于度相等的拓扑结构，可以取得几乎相同的解。文献[30]的研究成果表明，不同的拓扑结构将直接影响并行算法的效率；而个体的交换能够给种群带来新的基因，这对于提高算法的性能很有帮助。度高的拓扑结构具有较高的求解效率，但却增加了通信开销。因而，需要平衡计算与通信开销。文献[31]亦有此结论。为了研究拓扑结构及个体迁移参数对 PADGP 问题（Parallel and Distributed Genetic Programming，使用 GP 的典型基准测试问题）求解效率的影响，文献[27]对环、网状及随机拓扑结构进行比较。发现拓扑结构对结果没有显著的影响，对于随机拓扑结构，被确定为至少有像环和网状一样的实施优势。

目前，多数的研究都集中于静态拓扑结构，即子群体之间的连接拓扑在算法运行前便已确定。大多数静态拓扑结构的并行演化算法的拓扑结构都是根据研究者所能利用的计算机连接拓扑确定的。拓扑结构的另一类选择便是动态拓扑结构，即子群体之间的连接拓扑在算法运行前不确定。而是根据某个动态连接基准进行连接。其背后动机便是判定移民迁往哪些子群体能够产生作用[23]。

(三) 同步与异步

同步与异步并行演化算法运行相同的算法。当算法运行在同步模式下时，子群体阻塞起来等待迁入者的到来。而在异步模式下，则是当迁入者到达时，便将其插入目标子群体。在一个同构的并行硬件平台上运行，同步与异步模型具有相似的数值性能。文献[20]采用一整套实验比较dGA(distributed Genetic Algorithm)的同步与异步模型，且在相同的问题上，采用相同的参数运行同步与异步模型。该实验还瞄准了不同的运行平台。结果显示，在实时方面，异步模型性能的超越了其相应的同步模型。

当大量处理器被使用时，异步实施并无助于降低通信负载[26]。在异步并行模式下，如果子算法运行在非常不相同的处理器上，个体交换（迁出与迁入同一个体）可以发生在演化的非常不同阶段。这时，易于产生有名的无效问题（迁入个体不适合目标子群体）或超级个体问题（迁入者大大优于目标子群体中的个体）[15]。

Alba 和 Tomassini[18]经过大量的研究发现：

(1) 异步模型需要较低的执行时间，受迁移频率的影响较少；

(2) 异步模型更适合于容易及复杂领域，在分布式并行遗传算法应用上，性能优于其相应的同步模型；

(3) 对于所测试的算法，异步模型无论从运行时间还是加速比上都优于同步模型。

1.2.3.2 细粒度并行演化算法

一向以来，细粒度并行演化计算没有受到像其它并行演化计算一样的重视，导致没有太多的成果可借鉴。究其原因应该是它与其运行机器具有强相关性，且需要特殊硬件如连接机（Connection Machine）和细胞状自动机(cellular Automata Machine)。由于从个体到处理器的映射相当直接，