

基于大值堆的自调整粗粒度并行遗传算法模型

滕 腾, 李龙澍

(安徽大学 计算机科学与技术学院, 安徽 合肥 230039;
安徽大学 计算智能与信号处理教育部重点实验室, 安徽 合肥 230039)

摘要:一般粗粒度并行遗传算法(CGGA)的性能受诸多因素的影响表现不尽如人意。以降低通信代价为主要目标,受物种金字塔模型的启发,设计了一种双阈值限制下的自调整堆结构,并对其堆调整具体操作进行了改进,以期改进后算法中种群间的通信代价大幅度降低,优化收敛速度,提高算法效率。通过对遗传算法的几个典型测试函数通信量的分析和实验表明,基于该模型的并行遗传算法在降低通信代价、提高收敛速度、优化最终解方面收效明显。

关键词:并行遗传算法;CGGA;通信代价;堆模型

中图分类号:TP18

文献标识码:A

文章编号:1673-629X(2007)10-0105-04

A Self-Adjust CGGA Model Based on Max-Heap

TENG Teng, LI Long-shu

(School of Computer Science and Technology, Anhui University, Hefei 230039, China;
Ministry of Education Key Lab. of IC & SP at Anhui University, Hefei 230039, China)

Abstract: Common coarse-grained genetic algorithm(CGGA) has been criticized for many reasons. In this paper, focus on communication costs, gain the idea from creature species pyramid structure and suggest a heap model limited under two valves expect to significantly reduce the communication costs between two groups. The expectation of migration costs and experiment on typical GA test functions in the last part of this essay all verify that this model could greatly decrease the cost of communication and accelerate the convergence speed.

Key words: parallel genetic algorithm; CGGA; communication cost; heap model

0 引言

遗传算法(Genetic Algorithm, GA)是1975年由美国Michigan大学教授J. H. Holland提出的借鉴大自然物竞天演、优胜劣汰的自然选择和遗传机理的人工智能技术^[1],其本质是一种求解问题的高效并行全局搜索方法。它能在搜索过程中自动获取和积累有关搜索空间的知识,并自适应地控制搜索过程以求得最优解。然而,随着问题规模与复杂度的不断提升,GA在应用上仍有许多问题有待研究,一个突出的问题是收敛速度与收敛性之间的矛盾。因此,人们提出并行GA的模型以望改进算法。

众所周知,在自然进化过程的任何时刻,总是同时有大量的物种在彼此独立地向前进化,显然,自然界进

化过程本身就是一个并行过程,而遗传算法是人们对于自然进化过程的机器模拟,其本质上就继承了自然进化所固有的并行性。

目前典型的并行遗传算法(PGA)主要有:(1)全局单群体主从式PGA(见图1);(2)全局单群体细粒度PGA(见图2);(3)多群体粗粒度PGA(见图3)。

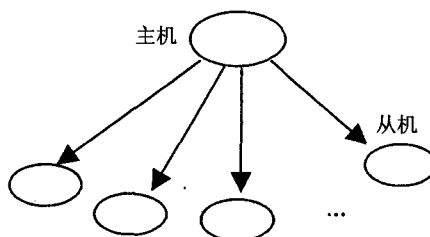


图1 MS-PGA的拓扑模型

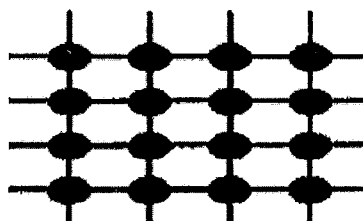


图2 FG-PGA的拓扑结构

收稿日期:2006-12-04

基金项目:国家自然科学基金项目(60273043);安徽省自然科学基金(050420204);安徽省高校拔尖人才基金(05025102);安徽省教育厅自然科学基金项目(2006KJ098B)

作者简介:滕 腾(1986-),男,安徽合肥人;李龙澍,教授,博士生导师,主要研究方向为智能软件和知识工程。

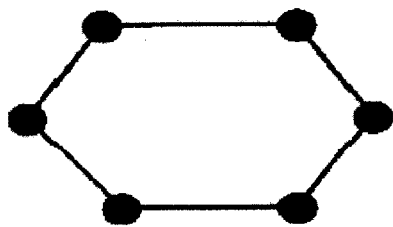


图 3 粗粒度拓扑结构

1 并行遗传算法目前存在的问题

尽管遗传算法无论是从直观上还是理论上都具备并行性,然而,标准的遗传算法除适应度计算外,几乎所有的遗传操作都是建立在全局统计的基础上,这意味着在整个进化过程中,这种通信开销不可忽视,实际上通信代价已成为目前 PGA 发展的首要问题^[2,3],现今也有许多文章就降低通信代价方面提出了多种改进模型^[4,5]。

在主从式 PGA 模型中,服务器是通信瓶颈且不能克服过早收敛的问题,单群体细粒度 PGA 实现代价较高,且通信量大,故笔者从多群体粗粒度 PGA 模型入手,提出一种改进方案。

介于对解空间搜索能力的考虑,希望群体具有较高的变异率和较高的交叉率,然而,介于对算法收敛性的考虑,又不希望看到最优解因高变异环境而丢失, Srinvas 等^[6]提出了自适应遗传算法,将算法中的交叉率和变异率进行线性自适应调整,从而较好地提高了算法效率,然而,该策略在演化初期存在停滞现象,不利于增强遗传算法的鲁棒性和维持较佳的种群平均适应度,文中提出了一种模型,在使用固定变异率与交叉率的前提下同时获得高搜索能力与收敛性。

另外,文献^[7]提出了配对选择在遗传操作中的作用,并提出了基于适应度相似性的配对方案,于是希望设计的模型在某种程度上可以从该角度优化算法。

2 自调整堆模型的设计与分析

2.1 模型的结构

最大值堆是一种可以不断进行自调整以始终保证堆中任意元素的值均比其低层的两个子元素的值要大的数据结构,文中的模型正是基于最大值堆设计的双阈值多群体粗粒度 PGA 模型,堆中各元素为独立的子种群,见图 4。

关于该模型的一些说明如下:

(1)初始化各子种群后根据各子群体的平均适应度进行堆排序,平均适应度最高的子群体位于堆顶,发生种群迁移时子种群只能与其父种群发生通信。

(2)底层群体的任务是对解空间进行搜索,因此具

有高变异率和高交叉率,上层群体的主要目标是保留最优解,因此变异率自底向上递减,最顶层种群交叉率和变异率均为 0,只用于不断刷新和保存整个算法的最优解集。

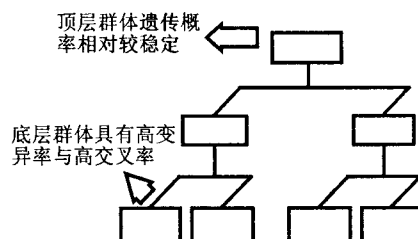


图 4 基于最大值堆的双阈值多群体 PGA

(3)底层(形象地称其为产生池)个体情况很不稳定,对其采用精英选择策略以提高搜索质量,各层间通信频率自底向上递减。

(4)通信的判断条件显然不能仅仅由子种群的平均适应度决定,例如最低两层的某两个欲通信的子群体 i, j, j 比 i 低一层,由于 j 中具有较高的变异率和交叉率,因而 j 中个体适应度差异很大,例如某时刻 j 中某新个体适应度很高(几近于某个最优解),而 j 中其他个体适应度不高,从而 j 的平均适应度不高,则此时 j 无法将该最优解迁出。于是想到了同时采用平均适应度和适应度方差两个指标的乘积来衡量迁移条件,

设迁移决定量 $W_j = k * \sum_{i=1}^{n_j} (fit_i - \bar{fit}_j)^2 * \bar{fit}_j$, 其中 k 为决定量修正函数,是一个随时间 t 递减的函数, $k > 1$ (关于 k 函数的说明可见 2.3 节)。

(5)设定双阈值 ϕ 和 θ ($\phi < \theta$),将每次待通信时的情况分为三类:

①若通信双方迁移决定量差值小于 ϕ ,说明当前此两群体对解空间的搜索情况差不多,正如 Grosso 所发现的,此时过早进行通信只能传递一些低阶的模式,对两群体的搜索影响较小,反而浪费通信时间,故此情况下不进行迁移操作;

②若此时较低层的迁移决定量明显高于较高层群体的迁移决定量,即二者之差大于 θ ,则为了降低通信代价直接进行宏观的堆调整;

③最后一种情况,若两群体迁移决定量差异度大于 ϕ 而小于 θ ,则进行个体迁移。

(6)借鉴文献^[4],迁移率应由通信双方的迁移决定量差异决定,迁移决定量差异越大迁移率应该越大,所以不妨设迁移率:

$$\lambda = \max \left\{ \frac{|\Delta W_{i,j}|}{\max\{W_i, W_j\}}, 1 \right\}$$

2.2 优化的说明

显然,该模型在如下方面对 PGA 做出了改进:

(1)降低通信代价。因为只有适应度差异介于 ϕ 和 θ 之间才进行个体迁移动作。

(2)在配对选择环节上达到了文献[7]所提到的效果,因为该模型在整体上起着将总种群中个体按适应度分级的作用,而随着进化过程的进行,总种群中适应度相近的个体将几乎都在同一层的各个群体中(当然,可能除了底层生产池)。

(3)较低层种群以其高变异率、高交叉率而得到了不错的解空间搜索能力,而高层群体变异率、交叉率相对较低,具有相对的稳定性,可以很好地保留优良个体。

2.3 改进模型的算法设计

设 W_i 、 W_j 为 i 、 j 两子群体的迁移决定量, i 群体在 j 群体下层, $\Delta W_{i,j} = |W_i - W_j|$, k 为决定量修正函数,因为不希望看到由于两种群间迁移决定量之差随进化的推移而迅速扩大从而使该阈值界定失效, k 可以设为时间 t 的递减函数: $k = 1 + a/t^b$, a 、 b 为常数,函数图像见图 5。

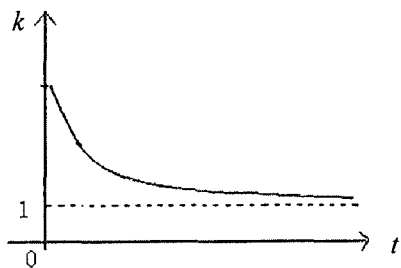


图 5 修正函数 k 的时间变化函数图

而随后的迁移操作判断如下:

$$\text{if } (W_i \geq W_j) \begin{cases} \Delta W_{i,j} \geq \theta & \text{adjust heap} \\ \phi < \Delta W_{i,j} < \theta & \text{migrate from } i \text{ to } j \\ \Delta W_{i,j} \leq \phi & \text{no operation} \end{cases}$$

else do no operation

算法流程:

(1)初始化总种群,并分割为 $2^h - 1$ 个规模相等的子群体 A_i , $i = 1, 2, \dots, 2^h - 1$,其实 h 即为堆的层数。

(2)计算各子群体的迁移决定量 W_i ,根据 W_i 进行最大值堆排序,设定各层变异率 P_{mutation_i} 、交叉率 P_{across_i} , $i = 1, \dots, h$,设定阈值 ϕ 、 θ ,设定决定量修正函数 $k(t)$,设定各层间的通信周期 T_i , $i = 1, 2, \dots, h - 1$,表示每隔 T_i 代第 i 层与第 $i + 1$ 层进行通信判断。

(3)若到达算法中止条件(如最大进化代数)则结束,否则各子群体并行进化。

(4)若到达某 T_i 周期,则计算堆中的第 i 层与第 $i + 1$ 层的每个群体的迁移决定量,对堆中的第 i 层与第 $i + 1$ 层的每两对可进行通信的子群体进行通信条件判断:若无迁移动作,则转(3),若发生堆调整则转(6)。

(5)发生迁移,使用一定的策略将较低层的群体中的较优的 N_λ (N_i 为第 i 个子群体的个体数量)个体迁入较高层群体并替换其中的较差的 N_λ 个体,转(3)。

(6)对堆进行堆调整,转(3)。

3 模型理论与通信代价分析

3.1 模型理论

由于模式理论的种种缺陷,研究者开始尝试利用有限状态马尔可夫链模型研究遗传算法的运行过程。对于遗传算法可以解决的优化问题,问题的可行域都是由有限个点组成的,即便是参数可以连续取值的问题,实际上搜索空间也是以要求精度为单位的离散空间,因此遗传算法的实际运行过程可以用有限状态马尔可夫链的状态转移过程建模和描述。对于有 m 个可行解的目标函数和种群规模为 N 的遗传算法, N 个个体共有 $\binom{N+m-1}{m-1}$ 种组合^[8],相应的马尔可夫模型也有 $\binom{N+m-1}{m-1}$ 个状态。遗传算法优化的过程,可以看作算法在循环过程中不断对可行域进行随机抽样,利用前面抽样的结果对目标点的概率分布进行估计,然后根据估计出的分布推算下一次的抽样点。马尔可夫模型认为遗传算法是通过对搜索空间不同区域的抽样,来估计不同区域的适应度,进而估计最优解存在于不同区域的概率,以调整算法对不同区域的抽样密度和搜索力度,进而不断提高对最优解估计的准确程度。可见,以邻域结构为依据划分等价类的马尔可夫模型更符合实际,对问题的抽象更能体现优化问题的本质。而遗传算法的马尔可夫链模型文献^[8]和^[9]已经有介绍,文中将不再赘述。

其实,粗粒度并行遗传算法遗传搜索过程中,其各个子种群一个新的解群体的产生仅依赖于当前群体,因此从一个给定群体状态达到特定群体状态的搜索过程的条件概率,在任何时刻不受前一代进化结果的影响,从而遗传搜索过程满足马尔科夫性,由文献^[3]可知,基于经典遗传算法的各个子种群的搜索过程是一个齐次遍历马尔科夫链,粗粒度并行遗传算法利用迁移算子基于某种机制:每一代将源种群的最优个体替代目标种群的最差个体,基于该算法的搜索过程是齐次遍历马尔科夫链。

3.2 通信代价分析

当发生堆调整时,由于此调整的时间代价 $O(\log n)$ 与迁移通信代价相比微不足道,故可忽略。迁移率即迁移个体数占总群体个体数的比率。由于个体数 m

以及群体状态数 n 是有限的,当 ϕ, θ 的值确定时,满足 $\phi < |\Delta W_{i,j}| < \theta$ 的状态对也是确定的,设为: $\langle a_1, b_1 \rangle, \langle a_2, b_2 \rangle, \dots, \langle a_t, b_t \rangle$ 。

则表示第 k 代发生迁移的概率 $P(k)$ 为:

$$P(k) = \sum_{i=1}^t P(a_i@k)P(b_i@k)$$

$$\text{设分段函数 } R(x) = \begin{cases} 1 & x = 0 \\ 0 & x \neq 0 \end{cases}$$

则第 k 代迁移率期望为:

$$E(\lambda_k) = R(k \bmod T_i)[0 \times (1 - P(k)) +$$

$$\sum_{i=1}^t P(a_i@k)P(b_i@k) \times \max\left\{\frac{|\Delta W_{i,j}|}{\max\{W_i, W_j\}}, 1\right\}]$$

而第 k 代通信量期望是: $E(M_k) = N \times E(\lambda_k)$ 。

显然该迁移策略降低了无效通信,提高了算法的效率。

4 实例分析

实验参数为:子群体数为 7(3 层),子群体规模为 20,各层(从下至上)交叉概率为 0.9、0.6、0,变异概率为 0.2、0.1、0,初始迁移代频为 7、3,传统 PGA 迁移率为 0.1,使用图 3 的拓扑结构, $n = 10$ 。实验均是使用 VC++6.0 单机模拟并行环境。测试函数 $f_1 \sim f_3$ 见表 1~3。

测试函数 1:

$$\text{Max} f_1(x, y) = 0.5 - \frac{\sin^2 \sqrt{x^2 + y^2} - 0.5}{(1 + 0.001(x^2 + y^2))^2}$$

$$- 10 < x, y < 10$$

测试函数 2:

$$\text{Min} f_2(x) = \sum_{i=1}^n x_i^2 / 1000 - \prod_{i=1}^n \cos(x_i / \sqrt{i}) + 1$$

$$- 500 < x_i < 500$$

测试函数 3:

$$\text{Min} f_3(x) = 1 + 10n + \sum_{i=2}^n [x_i^2 - 10 \cos(4\pi x_i)]$$

$$x_i \in [-5, 5]$$

表 1 函数 f_1 的实验结果

实验序号	终止函数值	传统 PGA 迁移个体数	文中改进算法迁移个体数
1	1.000000	44323012	29512
2	1.000000	45122184	32028
3	1.000000	38137600	25256

表 2 函数 f_2 的实验结果

实验序号	终止函数值	传统 PGA 迁移个体数	文中改进算法迁移个体数
1	0.000000	26754328	9940
2	0.000000	30012148	11304
3	0.000000	25467824	9628

表 3 函数 f_3 的实验结果

实验序号	终止函数值	传统 PGA 迁移个体数	文中改进算法迁移个体数
1	1.000000	1210608	10188
2	1.000000	1114832	8540
3	1.000000	1060320	8044

函数 f_1 有无数个局部极大值点,但只有一个全局最大点(0,0),最大值为 1;函数 f_2 的全局最小为 $x_i = 0, i = 1, 2, \dots, n$,其局部最小为 $x_i \approx k \cdot \pi \cdot \sqrt{i}, i = 1, 2, \dots, n; k = 1, 2, \dots, n; f_3$ 为多峰函数,其全局极小为 $x_i = 0, i = 1, 2, \dots, n$,在定义域内大约有 $10n$ 个局部极小点。

通过实验可以看出,文中算法在降低并行遗传算法通信代价方面效果显著。

5 结束语

基于粗粒度多种群并行遗传算法提出了一种降低通信代价的优化策略,理论分析和实验均表明该模型有效减少了无效模式的迁移,提高了算法时间性能。当然,在对 CCGA 模型的算法行为方面,笔者的认识还很粗略,文中阈值 ϕ 和 θ 的设定也尚需讨论,所以,文中模型各细节对整体算法的影响将是下一步将探讨的问题。

参考文献:

- [1] Holland J H. Adaptation in natural and artificial systems[M]. Ann Arbor, MI: The University of Michigan Press, 1975.
- [2] Pettey C B, Leuze M R, Grefenstette J J. A parallel genetic algorithm[C]//In: Proceedings of the 2nd International Conference on Genetic Algorithms (ICGA - 2). Hillsdale, NJ: [s. n.], 1987: 155 - 167.
- [3] Rudolph G. Convergence Analysis of Canonical Genetic Algorithms[J]. IEEE Trans. on Neural Network, 1994, 5(1): 96 - 101.
- [4] Lai Xin - Sheng, Zhang Ming - Yi. Parallel Genetic Algorithms with Migration Scheme Based on Penetration Theory [J]. Chinese Journal of Computers, 2005, 28 (7): 1146 - 1152.
- [5] Guan Yu, Xu Bao - wen. Parallel Genetic Algorithms with Schema Migration[J]. Chinese Journal of Computers, 2003, 26(3): 294 - 301.
- [6] Srinvas M, Patnaik L M. Adaptive Probabilities of Crossover and Mutation in Genetic Algorithm[J]. IEEE Trans on Systems, Man and Cybernetics, 1994, 24(4): 656 - 667.
- [7] Xie Zhi - wen, Yin Jun - xun, Jin Jing. Similarity matching selection of genetic algorithms[J]. Computer Applications, 2005, 25(11): 2665 - 2667.

(下转第 112 页)

当前的状态从而得知 LRNG 的先前状态。这样一来, LRNG 的前向安全性便遭到了破坏。所以 LRNG 的设计者必须调整“更新熵池”和“读取熵池”之间的顺序,即先计算熵池的输出然后再更新熵池的状态。

在对熵池更新进行了较为详尽的分析之后,仍然有一点需要说明,那就是在现有的设计下,即使黑客不知道生成器的先前状态,他仍然可以用攻击去影响 LRNG 的输出。因为在 LRNG 接受输入的过程中,当 primary pool 已满,从事件源处收集到的随机事件会被直接加入 secondary pool 中,而 secondary pool 正是/dev/random 读取输入的来源(这样看来,/dev/random 也不是那样地安全)。黑客正可以利用上面的这种情况人为地制造随机事件(称这种人为制造的事件为噪声“noise”)去影响 LRNG 的输出。所以笔者建议:总是要将新进的熵放入 primary pool。如果 primary pool 已满,就阻塞,直到 primary pool 又有了新的空间为止。

上面这个例子分析了/dev/random 设备存在的某种隐患。接下来,笔者还将给出两种通过/dev/random 设备发起攻击的方法。这种攻击叫做拒绝服务攻击。详述如下:

在任一单位时间里,用户都可以不受限制地从/dev/random 和/dev/urandom 这两个设备上读取任意数量的随机数。可是当熵池内熵的估计值小于某个特定阈值的时候,/dev/random 的输出就会被自动阻塞,直到有新的事件被添加进熵池中为止。以上的特性为两种拒绝服务攻击打开了方便之门,这两种攻击都试图阻止用户从/dev/random 中读取随机数。

第一种攻击操作起来很简单,只需要用 dd if=/dev/random 这个命令就可以实现。这种攻击的基本原理就是:不停地从/dev/random 设备上读取随机数。因为这个读取随机数的操作,在数量上没有限制,在权限上也不存在要求,所以就会导致其它用户的合理要求被阻塞(有可能是长时间地被阻塞)。

更进一步,还可以通过调用 get_random_bytes(*buf, nbytes)来实现远程的拒绝服务攻击。因为非阻塞的 urandom pool 是从 primary pool 中获得输入,一旦攻击发起,primary pool 和 secondary pool 必然要陷入瘫痪。实现这种远程攻击的一个最简单的方法就是不断地发送 TCP 连接请求。对于每一个连接请求,都会产

生一个 TCP-syn-cookie,这个 TCP-syn-cookie 会要求从 urandom pool 中得到 128 个字节的随机数,因此造成了熵估计值的下降。

为了解决以上这个问题,有必要为每个用户增加一个权限设置。这个权限代表了用户的级别,不同的级别用户在对/dev/urandom 设备的使用权限,以及单位时间内从/dev/urandom 设备中读取随机数的数量上都有不同。最高级别的用户可以从/dev/urandom 上读取任意数量的随机数;最低级别的用户没有使用/dev/urandom 的权限;中间级别的用户可以使用/dev/urandom,但在读取随机数的数量上又依据级别存在不同的限制。

4 结 论

LRNG 是 Linux 内核以及 Linux 操作系统的重要组成部分,它在系统加密等方面发挥着极其重要的作用。分析了 Linux 随机数生成器的工作原理和设计思路,并指出了设计上存在的缺陷以及由此产生的风险,最后针对各种风险给出了安全建议。由于 LRNG 到目前为止都缺乏最新的相关文档以及分析说明,所以以上的工作也正是文中目的所在。

参考文献:

- [1] Castejon - Amenado J, McCue R, Simov B. Extracting randomness from external interrupts[C]//In The IASTED International Conference on Communication, Network, and Information Security. USA: [s. n.], 2003:141 - 146.
- [2] Kelsey J, Schneier B, Wagner D, et al. Cryptanalytic attacks on pseudorandom number generators[J]. In Fast Software Encryption, 1998, 1372:168 - 188.
- [3] de Raadt T, Hallqvist N, Grabowski A, et al. Cryptography in openBSD: An overview[C]//In USENIX Annual Technical Conference, FREENIX Track, USENIX. [s. l.]: [s. n.], 1999:93 - 101.
- [4] Kelsey J, Schneier B, Ferguson N. Yarrow - 160: Notes on the design and analysis of the yarrow cryptographic pseudorandom number generator[J]. In Selected Areas in Cryptography, 1999, 1758:13 - 33.
- [5] Murray M R V. An implementation of the Yarrow PRNG for FreeBSD[C]//In Leffler S J. BSDCon, USENIX. [s. l.]: [s. n.], 2002:47 - 53.

(上接第 108 页)

- [8] Li Min - Quan, Kou Ji - Song, Lin Dan, et al. Primary Theory and Application of Genetic - Algorithms[M]. Beijing: Science Press, 2002.
- [9] Dai Xiao - ming, Chen Chang - ling, Shao Hui - he, et al. Con-

vergence Analysis of Coarse - Gains Parallel Genetic Algorithm and Its Application to Optimization[J]. Journal of Shanghai Jiaotong University, 2003, 37(4):499 - 502.