基于块同步模型的并行遗传程序设计方法①

安毅生，贺冰花

(长安大学 信息工程学院，西安 710064)

摘 要：采用并行计算方法可以有效避免遗传程序设计执行周期长的缺点。在分析遗传程序设计原理及块同步并行计算模型的基础上，以 Linux 多处理机系统为物理平台，实现了基于粗粒度并行模式的遗传程序设计方法，并对人工蚂蚁问题求解时处理机数与进化代数的关系进行统计和分析，实践表明采用并行计算模式可以更快的获得最优解。

关键词：块同步模型；遗传程序设计；遗传算子；人工蚂蚁问题；并行程序设计

Parallel Genetic Programming Method and its Application Based on BSP

AN Yi-Sheng, HE Bing-Hua

(School of Information Engineering, Chang'an University, Xi'an 710064, China)

Abstract: Parallel computing model can improve the performance of genetic programming and the actual speedup can be obtained. By analyzing the principle of Genetic Programming and Block Synchronous paralleled programming model, the relevant subjects discussed in the paper is a coarse-grained parallel model, and then the speedup of this model with different numbers of processor is also presented. At the end, instance drawn from practice is given for illustration.

Key words: block synchronous model; genetic programming; genetic operator; artificial ant problem; parallel programming

随计算机科学研究的重要课题之一是让计算机自动进行程序设计，即只要明确的给出计算机要解决的问题，而不需要告诉它如何去做。目前处于发展阶段的机器学习、自组织-自适应系统方法、神经网络方法、人工免疫方法等，正是试图寻找一种无需明确表达的求解程序来模拟复杂系统的结构和功能。遗传程序设计（Genetic Programming，GP）[1]通过引入自定义函数及动态程序复用方法，为解决这一问题提供了另一种可能的途径。

遗传程序设计是由 Stanford 大学的 J.R. Koza 教授于上世纪 90 年代提出，作为进化计算的一个新分支，它为自动开发计算机程序提供了一种新思路，可以用于解决采用传统方法不能很好实现或根本不能实现的组合优化问题。遗传程序设计是遗传算法中函数优化方法的扩展，这两种方法都是基于达尔文自然选择的进化过程，但是 GP 在求解程序时多采用树状结构表示个体的进化，更强调结构上的复杂性[2]。

尽管遗传程序设计有其优点，但较大的时间复杂度严重影响了它的执行效率[3]。因此，采用各种并行计算方法优化遗传程序设计的执行过程引起了很多学者的兴趣[4-6]。本文在分析遗传程序设计原理基础上，采用块同步并行计算模型，实现了粗粒度并行模式的遗传程序设计，并以人工蚂蚁问题的求解为例对该方法进行验证。

1 遗传程序设计的原理

遗传程序设计的基本思想是：随机产生一个适合于给定环境的初始计算机程序种群，即问题的搜索空间，通过反复的计算过程获得进化。与遗传算法类似，构成种群的个体都有一个适应度函数，依据适者生存

① 基金项目:陕西省自然科学基金(2009-JM8002-1),长安大学基础研究计划(CHD2011JC005)
收稿时间:2011-11-07;收到修改稿时间:2011-11-26

原则，针对每一个个体计算其适应度函数（求解问题的能力），采用的遗传算子如图 1 所示。在进化过程中，能够解决问题的程序片最有可能被选择参与到遗传操作中，因此高适应度个体保留到下一代种群，如此进化下去，给定问题的解或近似解将在某一代出现。遗传程序设计的完整过程如图 2 所示。

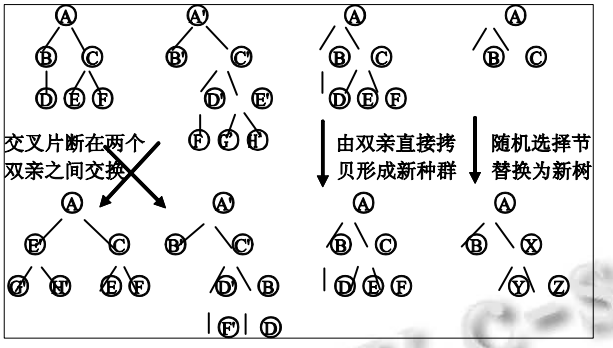

图 1 系统总体框图

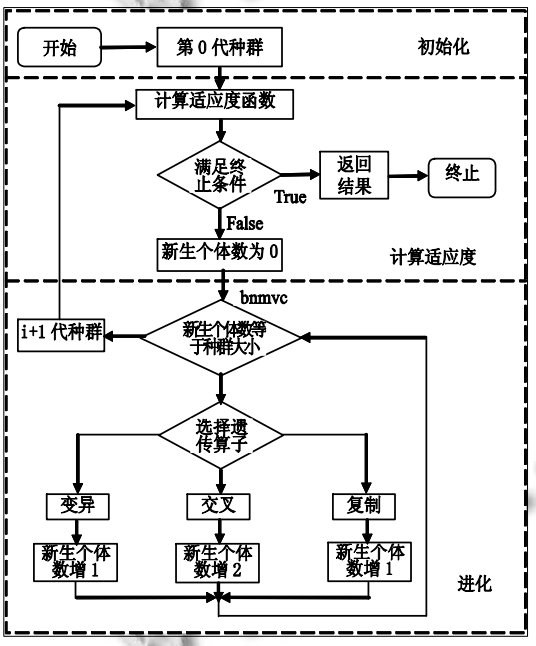

图 2 遗传程序设计的流程图[1]

2 块同步并行计算模型

块同步并行计算模型最早是由哈佛大学的 Valiant 教授等人提出[7]。BSP 模型提出了一组非常基本的操作，包括处理机的同步和处理机间的通信，既可以用专门的语言实现，也可以用函数库实现。

BSP 模型有一个垂直结构和一个水平结构。垂直结构表示随时间进行的计算过程，即一系列的全局“超步”，BSP 模型正是利用这种“超步”概念实现进程间同步。一个超步由一系列相互独立的本地计算及紧随其后的全局通信阶段和栅栏同步阶段组成，如图 3 所示。每一个超步可以分为三个有序的阶段，即：

1) 每个处理器上的本地计算。这个阶段中只用到存在本地存储器上的值。

2) 处理器间的通信，包括数据的传输。

3) 栅栏同步。标志超步的结束，而且保证在下一个超步中，传输的数据到达了目的处理器的内存中，目的处理器能够使用这些数据。

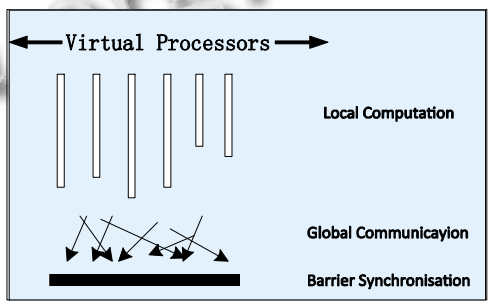

图 3 BSP 中的一个超步[7]

在每一个超步的结束阶段，BSP 过程等待所有处理机都完成了计算和通信任务，这样才能保证在下一个超步中，传输的数据到达目的处理机的内存，目的处理机能够使用这些数据。为了确保这种确定性的非局部存取，由一个特殊的超步启动的通信不一定会在同一个超步内完成。然而，发生在每一个超步结束阶段的栅栏同步可以确保每个超步中启动的非局部数据存取在后续的超步中使用。

BSP 模型的水平结构表示程序并发过程，由数量一定的进程组成。BSP 应用程序随机的将进程映射到处理器上，从而避免了具体分配进程的过程。因此水平结构的模式和目的处理器互连的拓扑结构所隐含的模式不一样，其所有通信操作的目的处理器接近于一种概率较大的处理器地址的排列。

BSP 程序高度结构化的特性使其能够预测全局通信的时间。这是很多现有的并行编程模型所不具备的，因为它们不能预估诸如拥塞等因素对各个通信过程影响的结果。而 BSP 模型全局性的处理通信模式，及随机性的进程分配，使得通信进程间相互作用的平均效果的计算成为可能，从而确定其总体通信时间的界限。

3 粗粒度并行实现模式

传统的 GP 并行实现模式是把串行实现中明显具有并行性的方面加以并行化。由于适应度函数的评价是一个非常耗时的过程，因此在 GP 实现过程中维持多个进程，其中主进程处理顺序的 GP 过程，完成程序片段的交叉、复制、编译操作，当需要计算适应度函数时，把种群划分为若干个子种群，为每个子种群分配一个从进程，计算结束后，再把每一个个体适应度函数返回主进程。

在全局并行实现方法中，进程是紧密耦合的，在每一代的适应度评价阶段都会在主-从进程之间传递大量的程序片段，这会增加进程通信的额外负载。为了有效的降低主从进程之间程序片段传输的数据量，本文采用基于精英程序片段迁徙的粗粒度并行实现方法。

GP 粗粒度并行实现方法的基本思想是将种群分成若干个子种群并分配给各自对应的处理器，每个处理器不仅独立计算适应度，而且独立进行选择，重组交叉和变异操作。为了确保某个处理器上最优种群能够促进全局最优解的产生，本文还定义了迁徙算子，以确保不同处理机上最大适应度程序片段在不同种群间的迁徙。

在粗粒度并行实现模式中，不存在主-从进程的模式，每一个进程可以被看成一个孤岛，定期地相互传送适应度最好的个体，凭借这种方法筛选出的个体在进程间传递，把遗传物质分布到所有的进程孤岛中。

4 应用实例

人工蚂蚁问题被认为是测试 GP 方法的经典实例[1]，具体描述如下：在长宽分别为 32 个单元的网格上设置 89 片食物，人工蚂蚁的初始位置为左上角(0,0)，沿着食物轨迹移动。蚂蚁具有如下的行为能力：a）前移动一个网格；b）左转 90 度，或右转 90 度。若食物在前方，蚂蚁既可以采取行为 a 也可以采取行为 b。

采用 GP 求解人工蚂蚁问题首先需要确定函数的终端节点；其次确定函数集合，其中，if 函数带有两个参数，如果蚂蚁的面前有食物，使用第一个参数，否则使用第二个参数。progn2 函数和 progn3 函数分别带有两个和三个参数。

求解人工蚂蚁问题的初始种群为 300。随机生成的个体包括{progn2(right, left), (if-food-ahead, right, left), progn2(move, move), progn3(left, progn2, progn2), (if-food-ahead, move, left), (if-food-ahead, move, right), …}。

串行处理求解时，第 0 代的计算机程序平均吃掉 3.5 片食物。进化到 27 代时出现能够把 89 片食物都搜索到的程序个体，即该个体是一个能够在尽量短的时间内遍历完所有食物节点的蚂蚁程序，如图 4 所示。

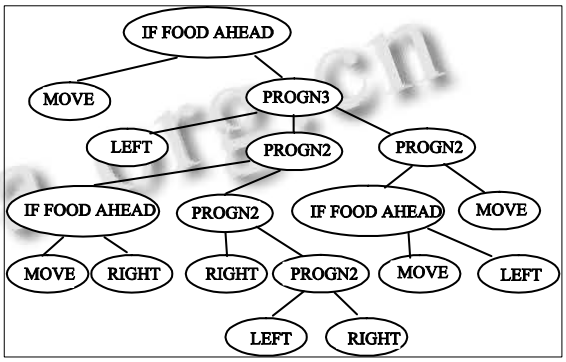

图 4　遗传程序片断的树型表示和 LISP 语言表示

采用 BSP 的粗粒度并行计算求解时选用 Linux 集群系统为硬件平台，处理器选用 Intel Xeon E3，处理器之间采用千兆以太网连接。

人工蚂蚁问题的粗粒度 BSP 求解伪代码如下所示：

```
generate_initial_population();  //生成初始种群；
bsp_begin();
for(int i=1;i<bsp_nprocs();i++)
bsp_store(local, i, sub_population(i), size of (sub_population(i)), num of (sub_population (i)));  //随即的在多个处理机上布置资种群；
bsp_fetch(remote, local, best_population, size, num);  //从其他进程上获得最优个体；
gp_process(sub_population(i)); //gp 的基本过程；
fitness_process(sub_population(i)); //适应度计算；
if (fitness<best) bsp_store(local, remote(local, remote, best_population, size, num)); bsp_sync();  //最优个体在种群间迁徙；
bsp_end();
```

伪码中的 BSP 函数包括：

```
bsp_begin();    // bsp 程序的开始;
bsp_end();// bsp 程序的结束;
bsp_initial();    // bsp 进程初始化;
```

bsp_nprocs(); // bsp 进程数查询;

bsp_sync(); // bsp 进程的栏栅同步;

bsp_store (to, from, data, size, num);

bsp_fetch (to, from, data, size, num); // bsp 进程之间最优种群的迁徙。

求解过程的参数统计如表 1 所示，从中可以看出随着处理器个数的增加，最优解出现的进化代数明显降低。

表 1 并行 GP 实现的参数统计

处理器数	最优解出现的进化代数	运行时间（秒）	加速比(%)
1	27	5437	0
2	24	4056	25.4
4	21	3121	42.6
8	23	2417	55.5
16	14	1879	65.4

5 结语

本文主要研究了基于 BSP 的并遗传程序设计问题。基于两点考虑，一是全面分析了基于 BSP 并行计算的粗粒度 GP 求解方法，二是遗传程序设计是遗传算法中的一个重要分支，国内在该领域的研究不够热烈，借此文抛砖引玉，希望能引起同行的兴趣。

参考文献

1 Koza JR. Genetic Programming: on the Programming of Computers by means of Natural Selection. MIT Press. 1996.

2 刘大有,卢奕南,王飞.遗传程序设计方法综述.计算机研究与发展,2001,38(2):213−222.

3 Alberto C, Amelia Z, Sebastián V. A parallel genetic programming algorithm for classification. Lecture Notes in Computer Science, 2011,6678:172−181.

4 Kenichi M, Takaya A, Takashi O. Autonomous acquisition of cooperative behavior based on a theory of mind using parallel genetic network programming. Artificial Life and Robotics, 2011,16(2):157−161.

5 Kent S. Speeding up Genetic Programming: A Parallel BSP Implementation. Genetic Programming 1996: Proc. of the First Annual Conference. 28−31 Jul. 1996.

6 Corne DW, Oates MJ, Kell DB. On fitness distributions and expected fitness gain of mutation rates in parallel evolutionary algorithms. Parallel Problem Solving from Nature - PPSN VII. 2002. 132−141.

7 Valiant LG. A bridging model for parallel computation. Communications of the Association for Computing Machinery, 1990,33(8):103−111.

(上接第 214 页)

5 结语

经教学实践检验，激发兴趣的教学模式效果十分显著。在发挥教师主导作用的前提下，采用各种方式激发学生的学习兴趣，让学生主动、愉快地学习，充分调动学生的积极性、主动性和创造性，引导学生融会贯通地掌握 C 语言，从而取得良好的教学效果。

参考文献

1 谭浩强.C 程序设计.第 3 版.北京:清华大学出版社,2005.

2 陈朔鹰,陈英.C 语言趣味程序百例精解.北京:北京理工大学出版社,1994.56−90.

3 盛雪丰.浅谈 C 语言程序设计教学的艺术化.科技信息,2009,(17):207.

4 葛丽萍.C 语言教学中的启发式教学应用研究.农业网络信息,2008,(6):130−131.

5 唐小健.浅谈比较教学法在《C 语言》教学中运用.职业教育研究,2010,(5):139.