

演化计算在硬件自动设计中的应用

武汉中南民族学院计算机科学系(430074) 王江晴

摘要: 给出了基于演化计算的硬件自动设计方法的实现过程,分析了该方法的特点、存在的问题及解决方案,讨论了演化计算在该应用领域的发展方向。

关键词: 演化计算 可演化硬件(EHW) 可编程逻辑器件(PLD)

演化计算是一种通过模拟自然界的生物演化过程搜索最优解的方法,主要包括遗传算法(GA)、遗传程序设计(GP)、演化策略(ES)、演化规划(EP)等。演化计算具有自组织性、自适应性、自学习性等智能特性。由于这些智能特性,该方法已被成功地应用到那些难以用传统方法进行求解的复杂问题中^[1]。

演化计算对待求解问题本身一无所知,但只要给出了表示方案、适应函数、遗传算子、控制参数、终止准则和指定结果的方法等,演化算法就可以按不依赖于问题本身的方式对未知空间进行有效地搜索,最后找出问题的解。

1 演化计算概述

演化计算求解问题,不是从单个点,而是从一个点的群体开始搜索。它将问题的可行解进行编码,这些已编码的解作为群体中的个体(即搜索空间中的点);将问题的目标函数转换为个体对环境的适应性;模拟遗传学中的杂交、变异、复制来设计遗传算子;用优胜劣汰的自然选择法则来指导学习和确定搜索方向。

演化计算对由个体组成的群体进行演化,利用遗传算子产生具有更高平均适应值和更好个体的群体。在整个演化过程中,起关键作用的是个体的适应值,它驱使遗传算子创造出新的适应性更强的个体,从而推动整个群体的演化。经过若干代后,选出适应值最好的个体,它就是问题的最优解或近似最优解。

演化算法的基本结构如下:

```
{
  随机产生一初始群体,计算其中每个个体的适应值;
  repeat
    应用遗传操作(复制、杂交等)产生下一代群体;
    计算群体中每个个体的适应值;
  until 满足算法的终止准则;
  指定算法的执行结果
}
```

由此可知,演化计算对问题本身的领域并不了解,它所做的只是对算法产生的每个个体进行评价,并通过遗传操作产生新一代群体,使适应值好的个体比适应值差的个体有更多的繁殖机会。如此一代代演化下去,直到算法满足给定的终止准则。

2 演化计算在硬件自动设计中的应用

对于传统的硬件设计,必须提供详细的硬件功能规范说明,才能由设计人员按照这些规范说明进行相应的设计。随着硬件设计复杂度和密度的不断增加,这种按照蓝图设计方法进行的手工设计,极大地增加了设计者的工作量。硬件设计的自动化势在必行。那么,如何才能实现硬件设计的自动化呢?利用演化计算的智能特性和 PLD 的可重构特性即可完成这项任务,称之为可演化硬件 EHW(Evolvable Hardware)。

EHW 是一种可重构的硬件,它建立在 PLD 之上,每当环境发生变化,EHW 就自动地改变自身的硬件结构以适应所处的环境。进行 EHW 的设计不需要硬件功能的规范说明,它利用演化计算的自组织、自适应、自学习等智能特性,不断地重构自身的硬件结构,最终达到设计的要求。因此,EHW 特别适用于事先不知道硬件规范说明的场合。

EHW 是在 PLD 上实现的。PLD 也是一种硬件,其结构是可变的,由一个被称为结构位串的二进制位串来决定。改变结构位串就能够立即实现任何的硬件结构。也就是说,若需要 PLD 实现某种特定的硬件功能,只须寻找相应的结构位串即可。这样,硬件设计问题就转化为搜索问题,在结构位串空间搜索合适的结构位串。如果把结构位串当作演化算法中的个体,把对硬件功能的评价转换成适应函数。那么,通过演化计算,就能够找到最合适的结构位串并根据它来改变 EHW 自身的结构,以满足设计的要求。这样一来,硬件设计的任务也就自动地完成了。

3 应用过程中应解决的问题

将演化计算用于硬件的自动设计,是一种全新的设计方法,它提出了许多具有挑战性的问题。

3.1 群体的规模

演化计算是对自然界中生物演化过程的模拟,但由于群体的规模较大程度地影响着演化算法的模拟消耗,目前硬件演化的群体规模还远远小于生物演化规模。必须研制数量更加接近自然物种的群体以便能够得到更好的结果。

3.2 参数的选择

在进行硬件的演化之前,必须确定一些参数,如算法执行的最大代数、各种遗传操作的概率等,它们对算法的执行效率有很大的影响。但关于如何选择这些参数的知识还很不完整,有很强的经验性,需要做更进一步的研究。

3.3 结构位串的长度

EHW 的实质是一个算法问题,只是此算法与硬件联系在一起。在算法的设计过程中,一个关键性的问题就是结构位串的长度。对一个实际的硬件进行演化,其长度可达几十万位,甚至更多。对于这样的硬件是难以演化的。为了解决这个问题,可采用变长结构位串表示法、逻辑函数表示法等方法。这些方法可以有效地减少结构位串的长度,以便在较短的时间内生成较大规模的硬件。

3.4 执行的速度

一个较小的硬件设计,有时需几天的时间才能完成:要想使 EHW 达到实用的目的,就必须提高演化算法的执行速度。演化算法属于一种群体搜索算法,具有内在大规模并行性。如何充分发挥其并行性,是提高 EHW 的执行速度的关键所在。

3.5 算法的收敛

在硬件的演化过程中有两个重点:群体的多样性和选择性压力。这两个因素密切相关,增加选择性压力就会降低群体的多样性,导致算法的执行趋向于在找到最优解前过早收敛;反之则又会使搜索毫无效率。过早收敛是演化算法和其它优化算法共同存在的问题。

4 与其它实现方法的比较

为了完成硬件自动设计的任务,有很多的实现方法。与其它的方法相比,基于演化计算的实现方法有以下特点:

- (1) 执行速度非常快。EHW 自适应的结果是新的硬件结构自身,因此 EHW 与其它基于软件的自适应系统相比,能得到显著的加速。这种优点是实时应用所希望的。
- (2) 能够实现联机自适应。通常的硬件自适应是脱机的,这样的系统只有在自适应之后或学习阶段完成之后才能使用。而理想的自适应机应该能在实时应用中改变自己的结构(即联机自适应)。
- (3) 能够直接将学习的结果储存在它的硬件结构中。这就导致了一种与人工神经网络(ANN)或其它基于规则的系统完全不同的新的学习方法。
- (4) 学习结果的可理解性较好。ANN 的学习结果是用阈值与权系数来表示的。一旦系统出错,很难猜测出错的原因,不利于系统的维护。而 EHW 的学习结果是用可见的布尔函数表示的,它大大地改进了学习结果的可理解性。

5 实现实例

EHW 把 PLD 的结构位串当作群体中的个体,利用演化算法去寻找更好的结构位串(即更好的硬件结构)。一旦算法发现了好的结构位串,则将它下载到相应的 PLD 中²,如图 1 所示。

假设一个群体由 $\{\theta_1, \theta_2, \theta_3, \theta_4\}$ 4 个个体组成。若

演化算法选择 θ_1, θ_2 进行杂交, θ_3 进行变异, θ_4 进行复制,其演化过程如下:

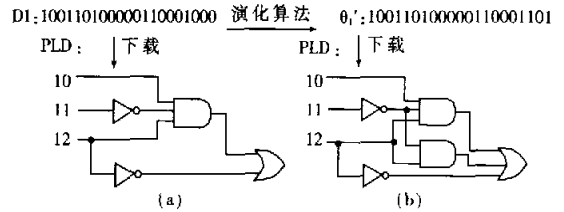


图 1 可演化硬件

(1) 杂交:该操作可以产生新的个体,从而检测搜索空间中新的点。若在演化算法中采用单点杂交,随机产生的杂交点为 18,则杂交过程为:

父串: $\theta_1=100110100000110001000$
 $\theta_2=1011101001001100011101$
 子串: $\theta_1'=1001101000001100011101$
 $\theta_2'=1011101001001100011000$

(2) 变异:该操作可增加群体的多样性,防止群体过早收敛。若随机产生的变异点为 3,变异过程为:

父串: $\theta_3=101110100100110001001$
 子串: $\theta_3'=100110100100110001001$

(3) 复制:该操作可提高群体的平均适应值。如:

父串: $\theta_4=100110100100100001001$
 子串: $\theta_4'=100110100100100001001$

这样,就得到了一个新的群体 $\{\theta_1', \theta_2', \theta_3', \theta_4'\}$ 。通过对新群体中每个结构位串进行评价,发现 θ_1' 的适应值更好,则将其下载到相应的 PLD 中。

基于演化计算的硬件设计方法是一种很有前途的硬件自动设计方法,该方法所涉及的研究领域比较广泛,在自适应控制、硬件容错、复杂电路的设计等方面都有应用。所使用的研究方法通常有两种:外部演化和内在演化。外部演化在演化过程中并不将算法所产生的结构位串下载到 PLD 中,而是用软件模拟的方式对每个结构位串进行评价,最后再将算法找到的适应性最好的结构位串下载到相应的 PLD 中去。这是目前常用的一种硬件演化方法。内在演化则将算法产生的每一个结构位串都下载到 PLD 中,通过 PLD 所实现的硬件功能对相应的结构位串进行评价。这种演化方法的演化速度很快,是 EHW 发展的主要方向。

参考文献

- 1 Back T, Hammel U, Schwefel H P. Evolutionary Computation:Comments on the History and Current State. IEEE Trans.on Evolutionary Computation,1997;1(1):3~17
- 2 Kajitani I, Hoshino T, Iwata M, Higuchi T.Variable length chromosome GA for evolvable hardware.In Proc. of the 1996 IEEE ICEC'96, 1996: 443~447

(收稿日期:2001-04-16)