遗传算法在游戏开发中的应用①

Application of Genetic Algorithm In Game Development

杨科选　梁昔明　(中南大学 信息科学与工程学院 湖南 长沙 410083)

摘　要： 本文简要介绍了游戏人工智能的发展现状，遗传算法的实现原理及过程，分析了基于规则的游戏人工智能存在的不足之处，重点介绍了遗传算法在解决这些问题时的优势以及具体的处理过程。

关键词： 游戏编程 人工智能 遗传算法 基于规则 游戏

现代的计算机游戏通过综合图形的、物理的和人工智能的方法来达到游戏的真实感。真实感游戏体验是很难确切定义其内涵的，但一般来说，这通常指游戏的沉浸感以及游戏中出现的非玩家角色的智能性。过去几年中，游戏中的图形技术和物理特性模拟技术都取得了很大的进展。实际上，图形开发包，如 Maya 和 3Ds Max 带来的令人瞠目的图形已经是一件普通的事了。物理特性模拟开发包，如 Havok 可以使开发人员创造出完全真实的世界。但是，对图形技术和物理特性模拟技术的应用已经不足使一个游戏具有独特性了。所以，游戏开发人员需要寻找使自己游戏进一步异化的创新。因为游戏人工智能没有像图形技术和物理模拟技术那样取得巨大的发展，所以它提供了一个游戏创新和异化的空间。

1　引言

1.1 游戏人工智能的发展现状

目前，游戏人工智能(artificial intelligence,AI)主要是基于规则的，规则的形式是条件→行为，当条件被满足时，行为就被执行。

判断规则条件是否符合的方法可以有很大的不同。可以是如图 1 所示的决策树形式，这棵决策树包含了条件和与之对应的一组操作。树的非叶子结点对应着当前情况下的一个判断，叶子节点表示行为。例如。根结点判断非玩家角色是否有武器，如果有，就接着判断非玩家角色是否靠近敌人，如果靠近，那么非玩家角色就发起攻击。另一条发起攻击的路径是在非玩家角色没有武器的情况下，判断敌人是不是比非玩家角色小，如果是，就发起攻击。

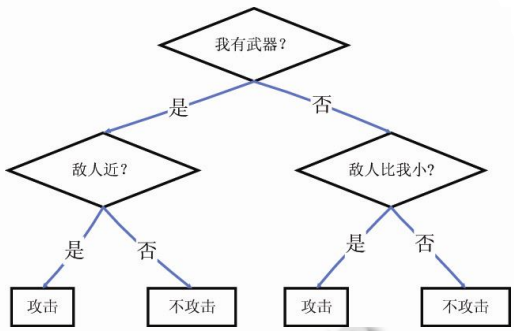

图 1　一个简单的决策树

另一个编码一组规则的方法是有限状态机。一个有限状态机包含一组开发人员定义的状态，这些状态对应着特定的行动，状态间的连线表示情况判断，并且使得状态从一个过渡到另一个。例如：在如图 2 的有限自动机中，最初的状态可能是空闲状态，非玩家角色没有行动直到发现了敌人。当发现敌人时，行动时攻击。当达到攻击的状态，如果非玩家角色又找不到敌人了，那么就会激发一个搜索行为。

与决策树相比，有限状态机相当于一组带标记的规则。标记和状态相对应，标记设置在一个规则的行动部分，也执行相关的行动。尽管有限状态机决策能力不比标准规则高，但在一定程度上更容易维护，因为它不需要显式的维护标记和其对应的相关状态。还

① 基金项目:国家自然科学基金项目(60874070);高校博士点基金项目(20070533131)
收稿时间:2008-10-05

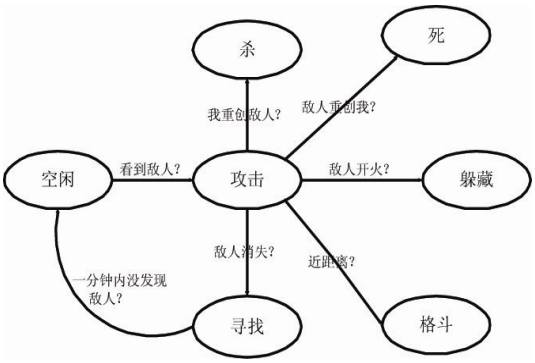

图 2　一个典型的有限状态机

有一些方法可以编码规则：模糊逻辑可以用来判断条件满足的概率，专家系统包含一系列规则和一个内置的规则编辑器……总之，规则可以表现为不同的形式，其中一些形式比另一些形式要容易修改、维护和理解，但是所有的实现都等价于条件→行动。

1.2 规则的问题

规则的开发费时、成本高。这是因为游戏开发人员要透彻的了解游戏，这样才能找出合适的规则。

基于规则的行为受到游戏开发人员智力的限制。不通过游戏开发人员的艰苦努力，就不会有很好的游戏智能。在紧张的开发时间表下，希望游戏开发人员在有限的时间内开发出足够的专门技术来使所有的非玩家角色具有智能经常是过分的要求。而且，好的程序员不一定能够制作出好的游戏角色。

基于规则的行为会使玩家产生怀疑，因而破坏玩游戏的沉浸性。一个游戏是否成功的重要影响因素是玩家在玩游戏时不产生怀疑。非玩家角色的非理性行为会使玩家产生不信任感。玩家会时时感到有一只开发者的手在背后操纵游戏，从而破坏了游戏的真实感。

规则是脆弱的，当情况稍稍超出规则定义的范围，规则就失效了。这是因为规则难以处理以下的一些情况：不确切或并行处理的行动；与时间直接相关的行动；位置的不确定；漏掉的、不精确的或不确定的观察资料。

总之，规则的制定需要设计者对游戏有足够的了解，要考虑到多个方面的情况，这对于开发者来说是一个几乎不可能完成的任务。而且，一系列的规则又会带来上述的弊端，影响游戏的可玩性。替玩家做出有挑战性的游戏环境是游戏设计师的职责。事实上，游戏在开发时大部分都是平衡这个游戏世界。游戏必须让玩家觉得有足够的难度，既能调动玩家的兴趣又不使玩家受挫，有时候玩家会发现一些漏洞或窍门，也就是作弊。替不同技巧水平的玩家做出一个真正平衡又有挑战性的游戏，那是一项艰难的任务。所幸，遗传算法可以助我们一臂之力。

2　遗传算法

2.1 遗传算法的基本思想

遗传算法是近年来发展起来的一种崭新的全局优化算法。它借用了仿真生物遗传学和自然选择机理，通过自然选择、遗传、变异等机制，实现各个个体的适应性的提高。从某种程度上说遗传算法是对生物进化过程进行的数学方式仿真。

遗传算法对求解问题的本身一无所知，它所需要的仅是对算法所产生的每个染色体进行评价，把问题的解表示成染色体，并基于适应值来选择染色体，使适应性好的染色体有更多的繁殖机会。在执行遗传算法之前，给出一群染色体，也即是假设解，然后，把这些假设解置于问题的“环境”中，也即一个适应度函数中来评价。并按适者生存的原则，从中选择出较适应的染色体进行复制，淘汰低适应度的个体，再通过交叉，变异过程产生更适应环境的新一代染色群体，再对这个新种群进行下一轮进化，直到最适合环境的值。

2.2 遗传算法通用实现过程

(1)对待解决问题进行编码；

(2)随机初始化群体；

(3)计算群体上每个个体的适应度值；

(4)评估适度，对当前群体中每个个体计算其适应度；

(5)按预定的选择算子产生后代；

(6)对后代进行交叉操作；

(7)对后代进行变异操作；

(8)判断是否满足停止条件，满足转第 3 步，否则进入 9；

(9)输出种群中适应度值最优的个体。

3　遗传算法解决游戏开发不可预测性问题

游戏设计时，游戏环境中最不可预测的因素就是玩家。就某种程度而言，游戏设计师必须要预测玩家的行为，以建立有挑战性的敌人。可惜的是，实际很难预测并响应每个可能的玩家的行为，遗传算法可以帮我们很好的解决问题。本质上，我们让游戏世界有好多种解法，然后找出那些表现的最好的解法。当然，解法并非对每位玩家都是一成不变的。这也正是遗传算法的优势所在，它使得游戏软件 AI 能去适应各个玩家。

考虑一个假想的多人角色扮演游戏，用遗传算法来解决我们遇到的问题。在此游戏中，玩家可以从多种角色类型和能力中选择一种角色。也就是说，计算机控制的角色，必须对各种玩家控制的角色展现游戏的挑战性。玩家可能选择当战士，主要以暴力对打。当这类玩家角色出现时，他们可能使用剑、斧或其他任何武器进行攻击。另一方面，玩家可能选择另一种完全不同的角色类型。当法师的玩家，其行为跟战士又会全然不同。角色类型和武器种类的各种组合，会让游戏设计师难以建立单一计算机控制的个体，让每种玩家控制角色都感受到挑战性。

3.1 数据编码

我们搜寻某种计算机控制角色，让某位玩家或一群玩家感受挑战性，但这种搜寻无法事先计算。每位玩家或者一群玩家的行为都不同，必须确认某些情况或者响应，会增加或减少族群中的响应度。例如，某一玩家以法器攻击非玩家角色，对于玩家的行动可以建立好几种可能的响应。我们可以用攻击玩家、试图逃跑或试图隐藏作为响应，并把这种情况和响应赋给某个染色体。如果染色体被设定成在此情况下做攻击响应，则玩家使用法器时将被攻击。对于不同的场景，族群的成员会有不同的行为。所以，我们以不同的染色体存储每种场景下的响应。在计算机中实现，我们只需建立一个类，包含一个染色体数组。这个数组的每个元素都代表计算机控制角色的一项特征或行为。先定义每个可能行为，然后把每个染色体和每个行为连接起来。

3.2 初始化族群

定义了各种可能场景以及每个场景相关的可能行为，下一步需要对族群进行初始化。我们知道，如果族群不够多样化，找出最佳解决方案的可行性会降低。而建立多样化族群的最佳方式，就是以随机方式给行为赋值。我们可以利用 switch 语句以及随机函数来给每个染色体赋一个随机行为。这样就可以演化出在某一场景下非玩家角色最佳的响应。

3.3 适应度分等

到底怎么样的行为才算是最佳响应，理想化的情况就是非玩家角色能够根据玩家的行为做出合适的反应，让玩家觉得有挑战性，而又不觉得游戏设计师在背后操纵。如果计算机控制角色总是被打败呢？就此而言，计算机控制角色会被视为不适应，因此，不可能把其特征传递给下一代。要早出族群成员中对玩家最有挑战性的成员，必须用某种方式量化和评定挑战性的等级，也就是建立一个好的适应度函数。角色扮演游戏，通常会把容许击中次数赋给每个角色。当角色在战斗中受伤，容许击中次数就会降低，当容许击中次数为零时，角色就会死亡。所以，我们可以用计算机控制角色对玩家造成损害的击中次数来评价其适应度。族群中的每位成员，都要记录其给予玩家损害的总击中次数。相反的，也要记下玩家对族群中成员所造成损害的次数。我们可以定义两个变量，第一个每一次计算机控制角色对玩家造成损害时就会递增；另一个当计算机控制角色被玩家损伤时就会递增。然后，计算造成的损害和受到损害的比值，以此来确定族员适应度。

3.4 交叉和变异

有了适应度函数，我们可以计算出各个个体的适应度，并可以通过排序函数对它们进行排序，可以按最优到最差的次序排列族群。然后我们就可以通过调用排序函数，找出适应度最好的个体，并用这些个体特征通过交叉过程产生下一代。为了得到更佳的适存个体，我们可以在上一代的基础上引入随机突变。引入一个随机突变函数，重新随机给染色体赋值。例如每个特征有 5%的随机突变概率，我们可以利用每行条件语句 if(Rnd(1,20)==1)来实现。

经过几代之后，相信非玩家角色就会演化成为对玩家而言比较有挑战性的对手了。

(下转第 143 页)

(上接第 130 页)

4 结论

本文简要介绍了游戏人工智能中规则的应用及其所带来的问题，重点介绍了遗传算法的原理以及在游戏人工智能某些领域能够更好的解决规则难以解决的问题，是对游戏 AI 开发的一个很好的补充。除这里讨论的问题以外，还有很多其他游戏问题，可以有效地使用遗传算法。对游戏而言，遗传算法只是一种寻找问题的最佳解决方案的方法。游戏软件 AI 中有很多问题要解决，但并非所有问题都适用遗传算法。例如，寻找路径也能用遗传算法解决，然而这种问题通常用诸如 A*算法的方法解决更合适。当问题的因素有某些不可预测性时，遗传算法最合适不过了。

随着游戏产业的蓬勃发展，如何设计出具有高智能娱乐游戏将是决定一款游戏是否有竞争力的重要因素。直接模拟人类自身进化过程的遗传算法在提高游戏中非玩家角色的智能性，提升游戏的趣味性将会有更大的发展空间。因此，对遗传算法在游戏中的应用研究具有非常重要的实际意义。

参考文献

1 Michael Dickheiser.游戏编程精粹.北京:人民邮电出版社, 2007:176－255.

2 李敏强,寇纪淞,林丹,李书全.遗传算法的基本理论与应用.北京:科学出版社, 2003.

3 王小平,曹立明.遗传算法—理论.应用与软件实现.西安:西安交通大学出版社, 2002.

4 Bourg DM. Seemann G.游戏开发中的人工智能.南京:东南大学出版社, 2006.

5 Buckland M.游戏编程中的人工智能技术.北京:清华大学出版社, 2006.