



游戏中的人工智能技术

浙江大学计算机学院

学习内容和目标

- 游戏**AI**的基本概念
- 游戏中简单的**AI**模式
- 游戏中常用的**AI**技术
 - 有限状态机
 - A^* , 模糊逻辑等
- 实现**AI**引擎的要点

GAME AI技术简介 (I)

- **GAME AI的描述**

- 使得游戏表现出与人的智能行为/ 活动相类似，或者与玩家的思维/感知相符合的特性。

- **GAME AI的实现**

- 技术实现
 - 利用充分的领域知识和常识
 - 客观世界的运动规律(game physics)
 - 利用已有的AI技术
- 融合娱乐性

GAME AI技术简介 (2)

- 游戏中涉及的AI技术
 - 专家系统
 - 用知识表示专家的经验，并在此基础上作自动推理
 - 案例式推理
 - 将输入与数据库中已有的案例进行比较，选取最为相近的案例，其已有的解决方法即为输出
 - 有限状态机
 - 基于规则的系统，有限个状态连接成一有向图，每一条边称为一个转移

GAME AI技术简介（3）

- 游戏中涉及的AI技术（续）
 - 产生式系统
 - 包含多个产生式，每一条产生式由条件和动作两部分组成，当产生式的条件满足时，系统就执行相应的动作
 - 决策树
 - 给定输入，从树的根部开始，将输入与当前结点相比较，选择当前结点的某一个子结点作为下一次比较的对象。当到达树的叶子时，则给出相应的决策
 - 搜索方法
 - 找到一系列动作（或状态转移），使得最终的结果满足某一特定目标

GAME AI技术简介（4）

- 游戏中涉及的AI技术（续）

- 规划系统

- 给定世界的初始状态，以及下一步可能采取的动作的精确定义，找到完成某个特定目标的最优路径

- 一阶谓词逻辑

- 谓词逻辑通过定义“物体”、“属性”、“关系”等对当前场景的状态进行推理

- 情景演算

- 用一阶逻辑计算在给定情景下AI生命的反应

GAME AI技术简介（5）

- 游戏中涉及的AI技术（续）
 - 多Agent
 - 研究在多个相互竞争相互合作的智能体之间所产生的交互智能行为
 - 人工生命
 - 多agent系统一种，试图将生命系统中一些普遍规律应用到虚拟世界的人工智能体上
 - 群组行为(Flocking)
 - 人工生命的一类，研究协同移动技术，例如人工智能体如何在大量的羊群中移动

GAME AI技术简介（6）

- 游戏中涉及的AI技术（续）
 - Robotics
 - 让机器在自然环境下交互的工作
 - 遗传算法
 - 直接模拟生物进化过程，通过随机选择、杂交和突变等对程序、算法或者一系列参数进行操作
 - 神经网络
 - 模拟动物神经系统功能的机器学习方法
 - 通过反复调节系统内部中各个神经元之间的连接参数，使得训练得到的系统在大多数情况下作出优或者近似优的反应

GAME AI技术简介（7）

- 游戏中涉及的AI技术（续）

- 模糊逻辑

- 与传统二值（对-错）逻辑不同，模糊逻辑用实数表示物体隶属于某一类的可能性

- 置信网络

- 提供建立不同现象之间内在因果关系的工具，并利用概率理论处理未知的和不完全的知识
 - 对当前状态作出判断，并决定下一步可能的动作以及其带来的后果

GAME AI技术简介 (8)

- **GAME AI技术的分类**
 - 确定型
 - 基于领域固定领域知识，模拟简单的固定行为
 - 行为型
 - 基于行为模式来模拟智能行为
 - 战术型
 - 策略模拟
 - **RTS** (real time strategy)
 - 其他

确定型AI算法

- 确定性算法指预先编入代码当中的可预测的行为
 - 从最简单的算法开始
- 例如，系统中有一颗小行星，以某一速度作匀速直线运动，它在任意时刻的位置由下列公式决定：
- 某种程度上，它们是智能的，但是这种智能是确定的，可预测的

$$x = x_0 + v_x \times t$$

$$y = y_0 + v_y \times t$$

Tracking/Chasing AI

- 当智能体找到目标后，一心一意向其移动，而不考虑任何其他因素，例如障碍物、另外的目标等
- 非常机械化
- 在每一帧中，智能体计算其到目标的前进方向，并根据其速度，前进一段距离



Tracking/Chasing算法

- Tracking算法还可以做的更为真实一点，就像红外导弹一样：
 - 在每一帧中，智能体仍然首先计算其到目标的前进方向
 - 这时，智能体的速度允许发生变化，并根据更新后的速度，计算下一帧的位置
 - 速度有一个上限，超过这个上限，智能体的速度将减慢，直到重新加速为止



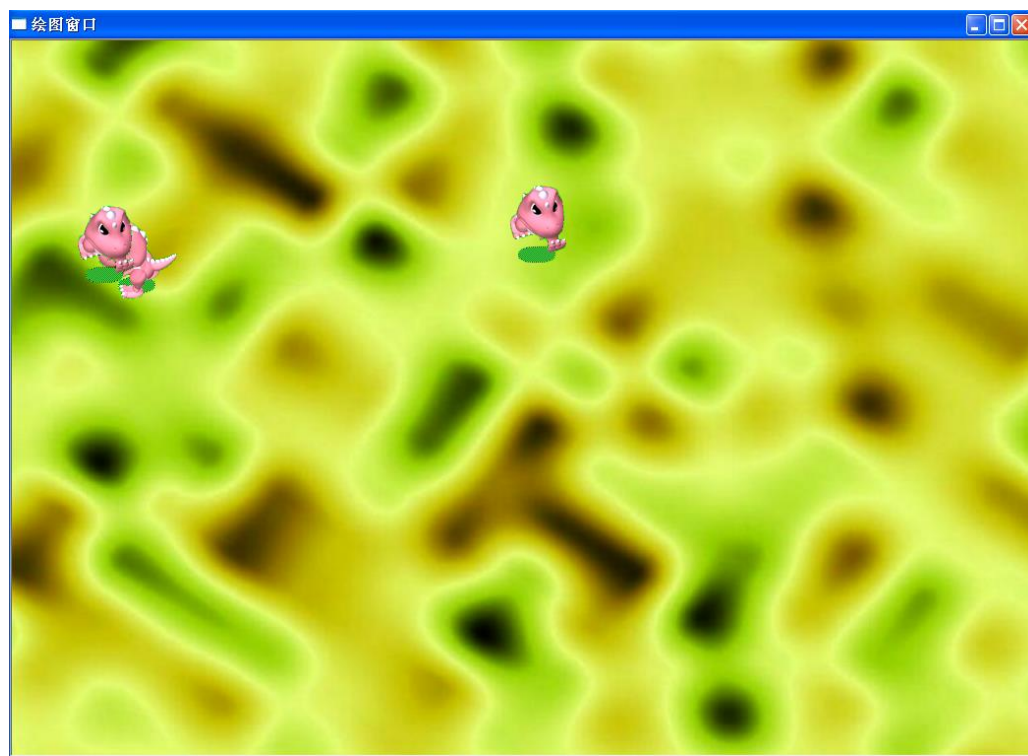
Evading算法

- 与前面的**chasing**算法基本相同，唯一区别是智能体沿着远离物体的方向移动



追逐行为的模拟示例

- 你追我赶 Game AI/chasing and avoiding demo



基于行为模式的AI

- 在任一时间点，每一个智能体都按照预先设定的某种模式运动
- 决策系统根据系统当前的状态，为每一个智能体从模式集合中选择适当的模式
- 模式描述了智能体将在下面几帧中所采取的一系列动作
- 特例：**scripted AI**，当系统到达某一特定状态（例如，每个回合的结束），系统运行的一段程序（用脚本写），决定系统下一步的动作

典型的行为模式

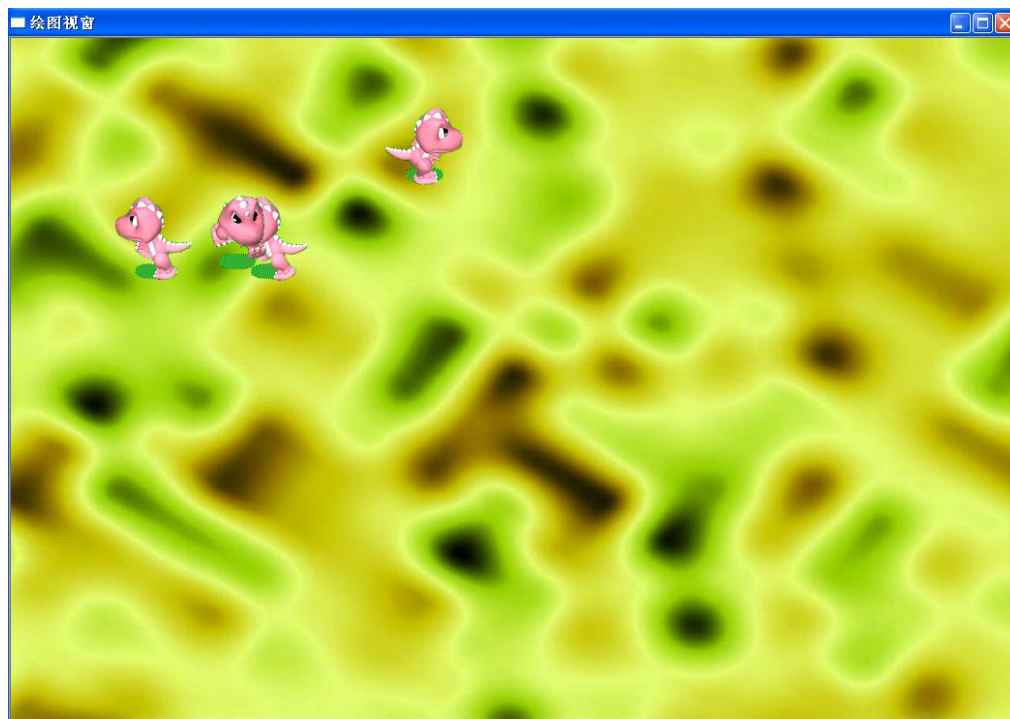
- 基本模式
 - 用一段指令定义模式
 - 写一个解释器解释这段指令，并用于控制智能体的行为
- 条件逻辑模式
 - 更为灵活的控制
 - 可以通过条件逻辑选择模式
 - 也可以选择本身带有条件逻辑转移的模式

编程技巧

- 非常直观
 - **Pattern**是一列数组
 - 数组的每一项定义智能体在该帧的速度（方向+大小）
 - 在模拟过程中，智能体就按照预先设定的参数在每一帧之间运动
 - 当移动到数组末尾时，重新选择一个新的模式

行为型的AI技术示例

- Chasing behavior AI demo



怒

休息中

小補血

中補血

大補血

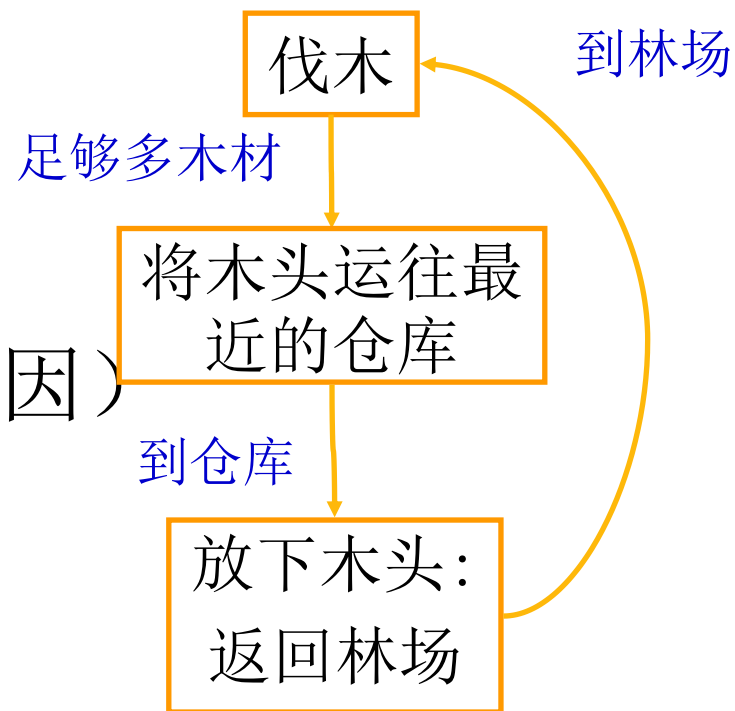
敵人全滅!!

策略性AI与通用问题求解

- AI的研究人员试图寻找一个通用的计算模型和方法，解决所有的问题
 - 感知输入系统
 - 有一个记忆模拟存储系统
 - 推理机
 - 行为输出系统
- 博弈问题
 - 有限状态机（FSM）
 - 规划和搜索

有限状态机

- 状态（要采取的行为）
 - 追击
 - 随机走动
 - 巡逻
 - 吃
- 转移（发生转移的原因）
 - 时间片结束
 - 发生某个时间
 - 完成某个行为



有限状态机

- 机器
 - 所有部件的总称
- 状态
 - 对于层次有限状态机而言，状态包括各种子状态
- 转移
 - 系统从当前活动状态出发，判断下一个活动状态，改变系统当前的格局，并执行相应的操作



有限状态机

- 条件
 - 定义发生转移的先决条件
- 输入和事件
 - 允许状态机对环境变化作出反应
- 动作
 - 作为状态的一部分，或者伴随转移出现



状态空间图

- 有向图
- 每个结点表示系统状态模型，每条弧表示状态转移所伴随的动作行为
- 结点可以是无穷多个
- 有些结点之间可能没有弧相连接

特定状态的查找

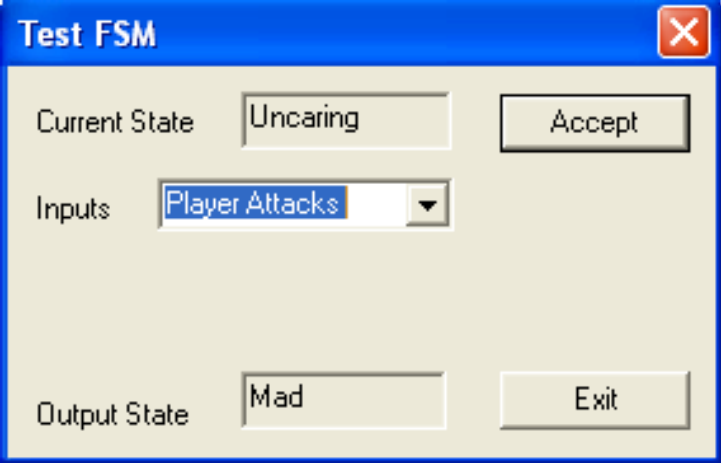
- 结点 — 包含查找目标
- 终点 — 搜索路径的结束
- 查找空间 — 所有结点的集合
- 目标 — 所要到达的结点
- 经验 — 在一定程度上提示下一步搜索的方向
- 解答路径 — 从起始结点开始，到目标的一条有向路径

模糊的有限状态机


- 将模糊逻辑和有限状态机结合
 - 状态之间的迁移不再是确定的
 - 同时有多个状态

有限状态机和模糊的有限状态机示例

- FSM/FuFSM
- 代码示例
 - 隶属度
- 演示



A screenshot of a software window titled "Test FSM". It features a blue title bar with a close button. The window has a light beige background. It contains four main input/output fields: "Current State" with the text "Uncaring", "Inputs" with a dropdown menu showing "Player Attacks", "Output State" with the text "Mad", and an "Accept" button. There is also an "Exit" button at the bottom right.



A screenshot of a software window titled "Test Fuzzy State Machine". It features a blue title bar with a close button. The window has a light beige background. It contains a "Select Input" dropdown menu showing "Player Attacks" and an "Exit" button. Below these, there is a section titled "Degree of Membership" and "Current States" with a text area containing the following text:

Degree of Membership	Current States
10 ->	is Sort of Mad
100 ->	is Totally Annoyed

规划

Part of intelligence is the ability to plan
- Move to a goal State

- 将系统表示成一系列状态的集合
- 通过操作（Operator）改变状态



路径规划

- 状态
 - 智能体在空间的位置
 - 其他离散空间
 - 体素
 - 室内位置
 - 局部区块 (**tile**)
- 操作
 - 从一个位置移动到其他位置



路径规划算法

- 必须对状态空间进行搜索，才能转移至目标状态
- 完全性
 - 如果目标状态存在，算法是否能够将其找到？
- 时间复杂度
- 空间复杂度
- 能够找到最优解



搜索策略

- 如何评价搜索算法
 - 时间：多长时间能够找到解
 - 找到的解是最优、次优还是其他
- 盲目搜索
 - 没有先验知识
 - 仅仅知道目标状态是什么
- 经验搜索
 - 用经验公式表示拥有的先验知识
 - “经验”只能作相对简单、低级的决策

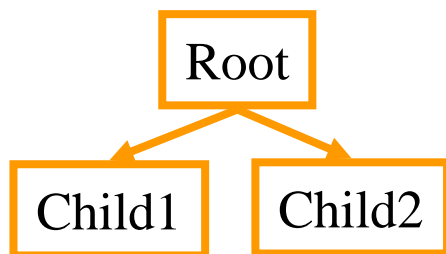


广度优先搜索

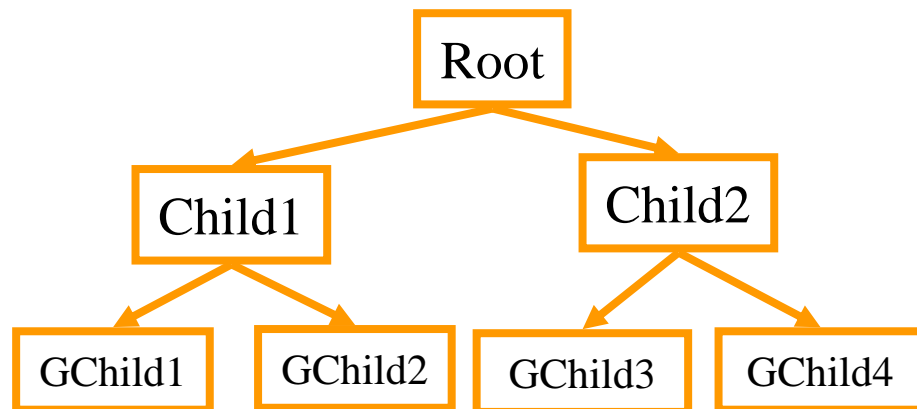
- 根结点->儿子结点->孙子结点
- 缺点：内存消耗大



(1)



(2)

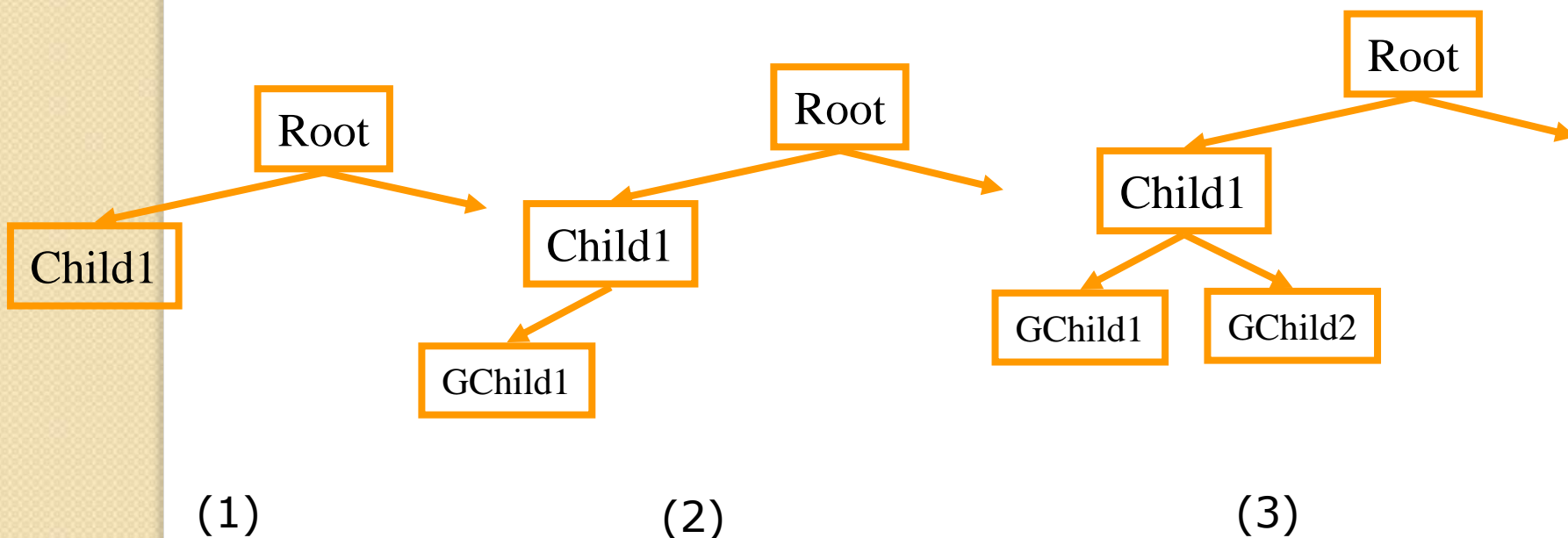


(3)



深度优先搜索

- 先儿子结点，后兄弟



双向搜索

- 同时产生两棵搜索树
 - 一棵从起点出发
 - 一棵从目标出发



启发式搜索

- 定义目标函数，反映拥有的先验知识
 - 估计离目标的距离
 - 估计到达目标的花费
- 用上述估计指导路径的搜索，加快搜索过程



贪婪搜索法

- 永远沿着具有最小目标函数值的路径进行搜索
- 不一定能够找到目标
- 可能得到局部最优解，而不是全局最优



A*启发搜索

- 考虑到贪婪搜索法不能保证找到最优解
- 改进 — 目标函数由两个部分组成
 - 从当前状态到目标状态的“花费”（估计）
 - 从初始状态到当前状态的“花费”



基本想法

- 贪婪搜索法

- 对可能的后继状态 n' ，计算其到目标状态的“花费” $h(n')$ ，并置于一个优先队列中

- A^*

- 对可能的后继状态 n' ，计算其目标函数 $f(n')$ ，并置于优先队列中
- $f(n') = g(n') + h(n')$ ，其中 $g(n')$ 是从初始状态到 n' 的“花费”



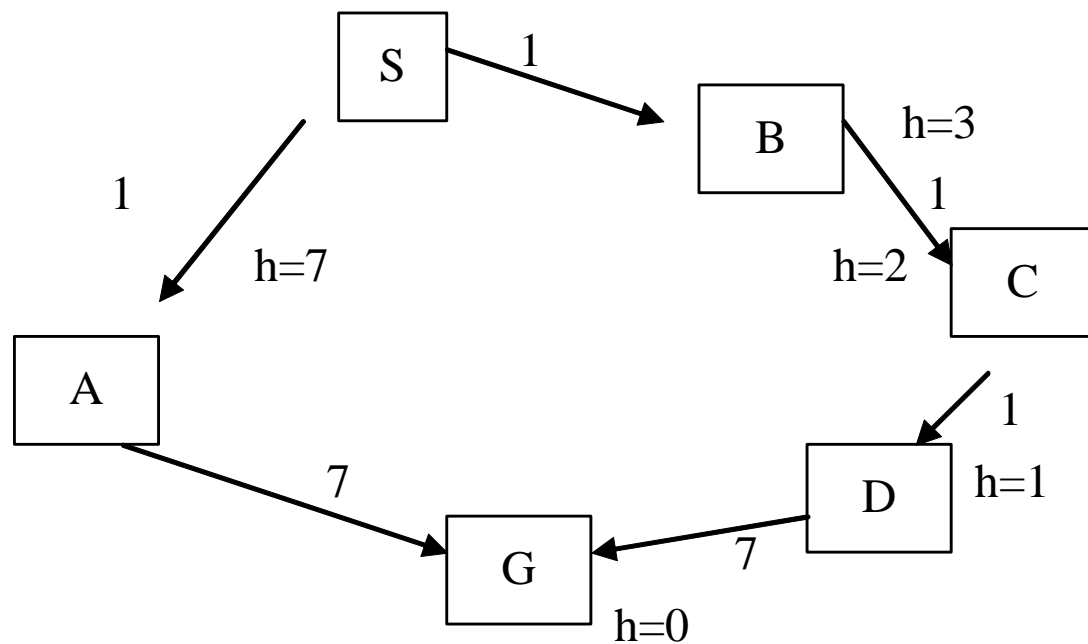
基本想法

- 选择下一步状态 n ，使得 $f(n)$ 是队列中最小的
- 如果 $h(n)$ 估计准确的话，方法是可行的



结束条件

- **A***算法结束条件是：当且仅当目标状态被从优先队列中挑选出来



A*算法

- 优先队列PQ — 初始为空
- V（一系列三元组(状态, f, 回溯指针)集合, 表示访问过的结点） — 初始为空
- 将初始结点S置于PQ中, V中放入(S, f(s), NULL)
- 算法:
 - 如果V为空, 退出程序, 没有解
 - 否则, 从PQ中取出第一项, 记为n
 - 如果n就是目标结点, 则搜索结束
 - 否则, 产生n的后继结点



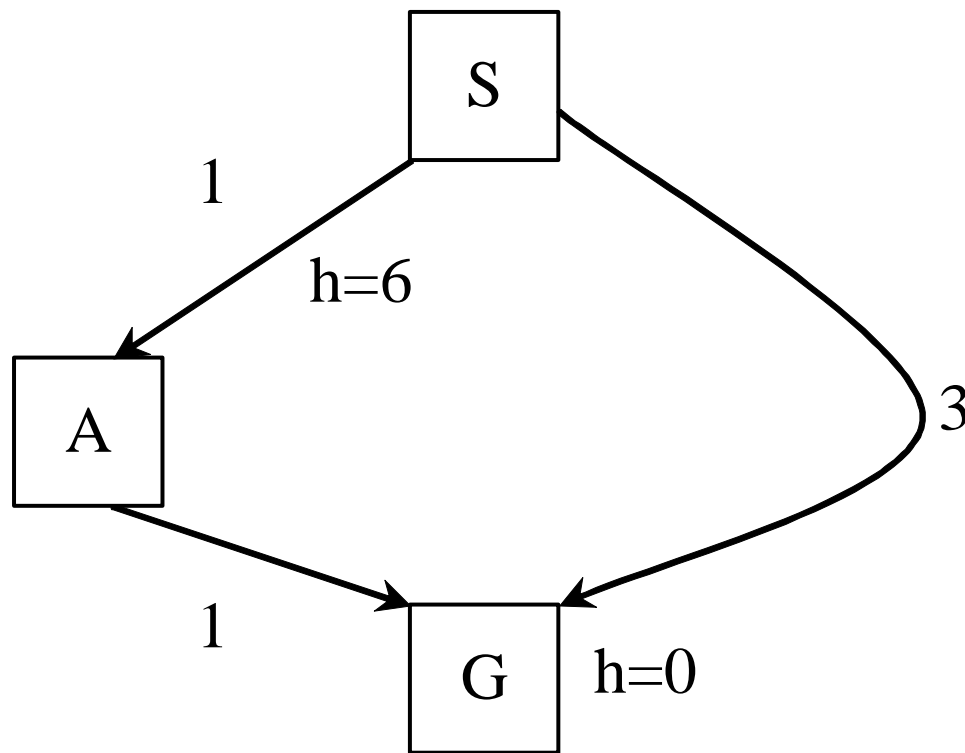
A*算法

- 对 n 的每一个后继结点 n'
 - 计算 $f' = g(n') + h(n') = g(n) + \text{cost}(n, n') + h(n')$
 - 如果 n' 未被访问过，或者 n' 曾经被访问过，但是记录的 $f(n') > f'$ ，或者 n' 已经在PQ队列中，但是记录的 $f(n') > f'$
 - 放置/更新 n' 于优先队列中，使其对应的目标函数值为 f'
 - 添加 (n', f', n) 至 V 当中
 - 否则忽略 n'



A*算法能否找到最优路径

• 否

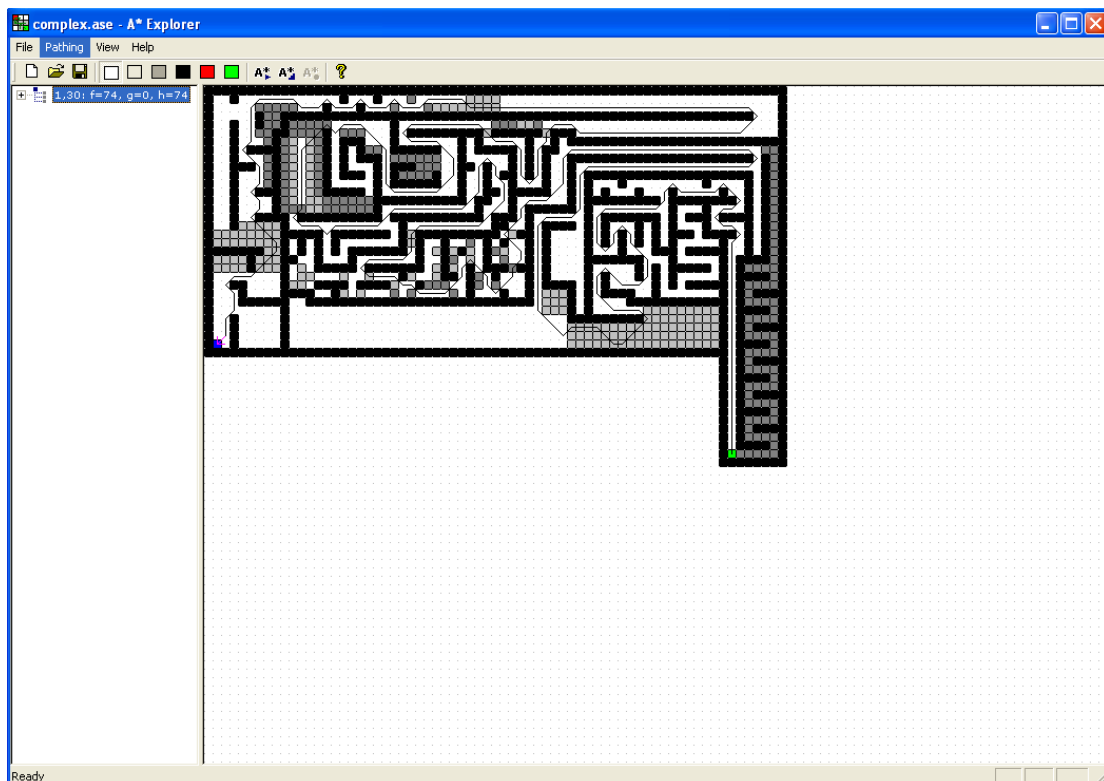
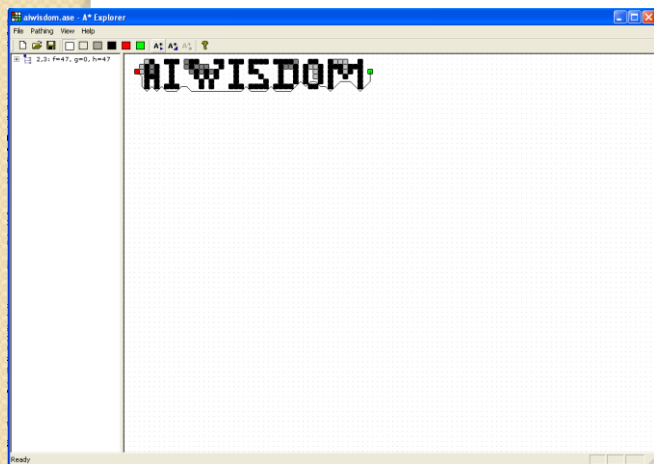


A*算法性质

- 令 $h^*(n)$ =从目标到 n 最小花费真实值.
- 经验 h 称为可行的当且仅当对所有的状态 n ,
 $h(n) \leq h^*(n)$.
- 可行经验确保永不过估计结点到目标的花费
- 具有可行经验的A*算法一定收敛到最优解
- 比较费内存
- 当不存在解时，算法失败
 - 避免对全空间进行搜索
 - 作双向搜索

路径的规划和寻找演示

- A star demo
 - 最短路径
 - 战术最短路径
 - 暴露时间
 - 有效火力
 - 视野



群体行为的模拟（I）

- 物群的行为

- 物群聚集在一起飞行，遇到另一物群时，他们将避开和分散，必要时分成多群
- 分开后，将寻找伙伴，形成新的物群，并最终恢复原来的物群
- 物群能够对付突发行为，能否对不断变化的环境做出实时的反应，并作为一个整体行动
- ○ ○ ○ ○ ○

群体行为的模拟（2）

- 物群模拟的简单规则
 - 分离（**separation**）：同物群中的其他成员若即若离。
 - 列队(**alignment**)：与物群中的其他成员保持相同的航向
 - 内聚(**cohesion**)：不掉队
 - 避开（**avoidance**）：避开障碍物和天敌
 - 生存（**survival**）:必要时进行捕食或者逃脱被吃
 -

群体行为的模拟（3）

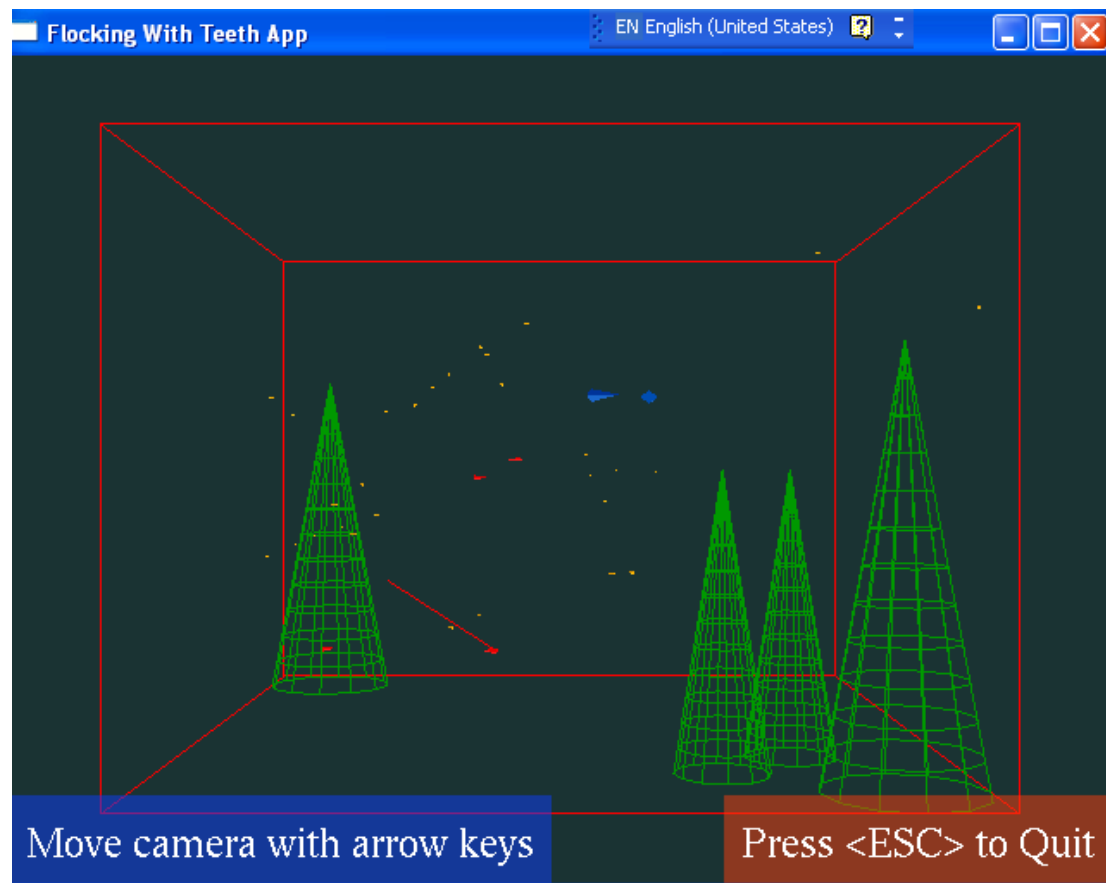
- 游戏中的物群行为
 - **RTS**游戏的部队的编队模拟
 - **RPG**游戏中的群体行为模拟
- 行为模拟的实现
 - 无状态
 - 不纪录任何信息
 - 每次将重新评估其环境

群体行为的模拟（4）

- 示例
 - 前进方向不确定，但整体行动
 - 避开障碍物
 - 飞行动物
 - 老鹰：飞行速度快，视野广，吃麻雀
 - 麻雀：飞行速度一般，视野一般，吃昆虫
 - 昆虫：飞行速度慢，视野小，不捕食，能繁殖
 - 物群的喂养
 - 饿→吃→试图接近猎物
 - 昆虫不能灭绝

群体行为的模拟（4）

- 演示: flocking demo



模糊逻辑

- 传统逻辑把思维过程绝对化，从而达到精确、严格的目的
- 举例：一个被讨论的对象 X ，要么属于某一个集合 A ，要么不属于该集合，两者比居其一，而且两者仅居其一，决不模棱两可
- 对于命题：张三的性格稳重，如何判断这一命题的真假？

模糊逻辑

- 对于上述的例子，模糊逻辑允许我们用 一个 $[0,1]$ 的实数表示 X 属于 A 的隶属程度。传统逻辑即隶属程度只能从0和1之间选择的情况
- 对于“性格稳重”这个模糊概念，我们能够用“一点而也不稳重”、“不太稳重”、“不好说”、“有点稳重”、“挺稳重”、“很稳重”等没有明确界限的词语形容

模糊逻辑的应用

- 将重心转移至物体属于某个集合的隶属程度上
- 在**AI**领域的主要应用为
 - 决策
 - 行为选择
 - 输入、输出过滤

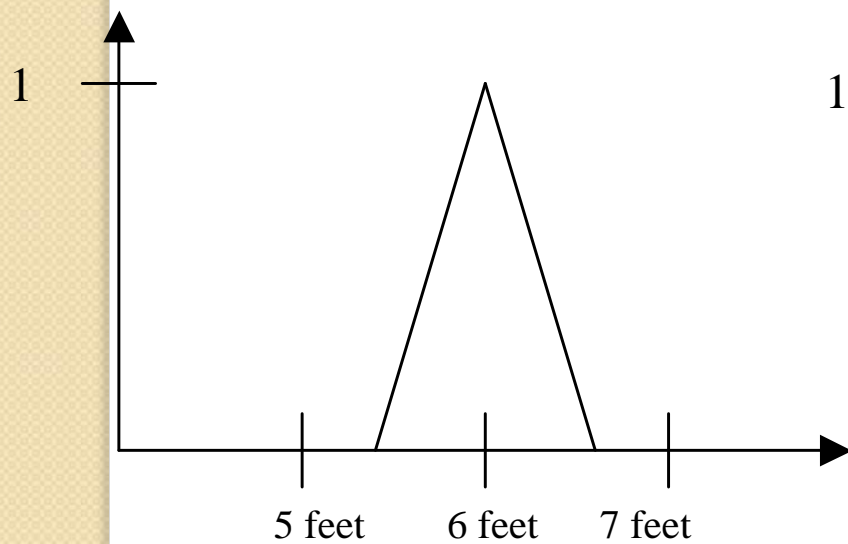


符合逻辑操作

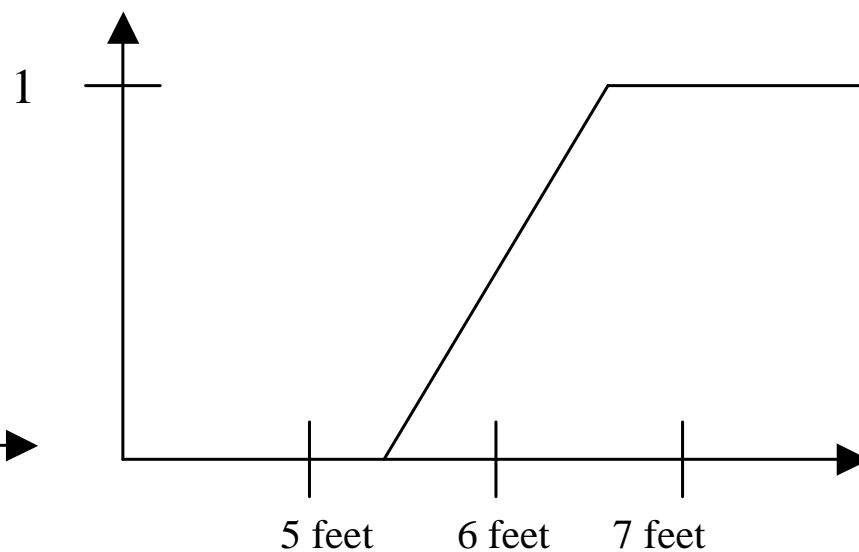
- 设A,B,C均为U中的模糊集
 - 模糊并
 - 若对 $x \in U$, 均有 $\mu_C = \max(\mu_A(x), \mu_B(x))$, 则称C为A与B的模糊并
 - 模糊交
 - 若对 $x \in U$, 均有 $\mu_C = \min(\mu_A(x), \mu_B(x))$, 则称C为A与B的模糊交



例子



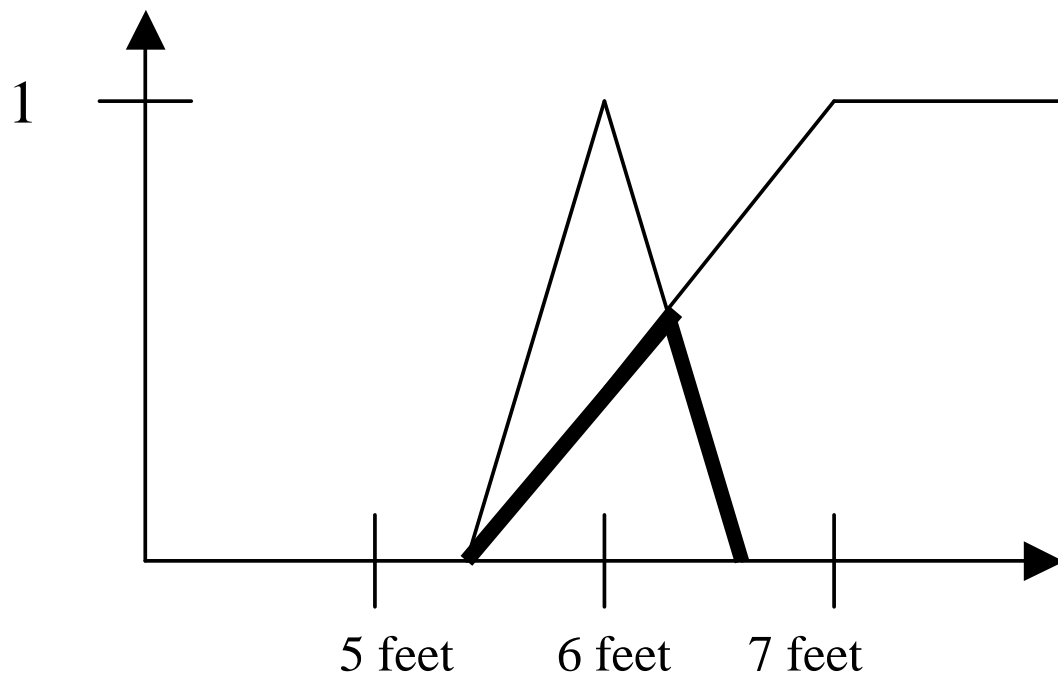
大约6英尺



长的高的人

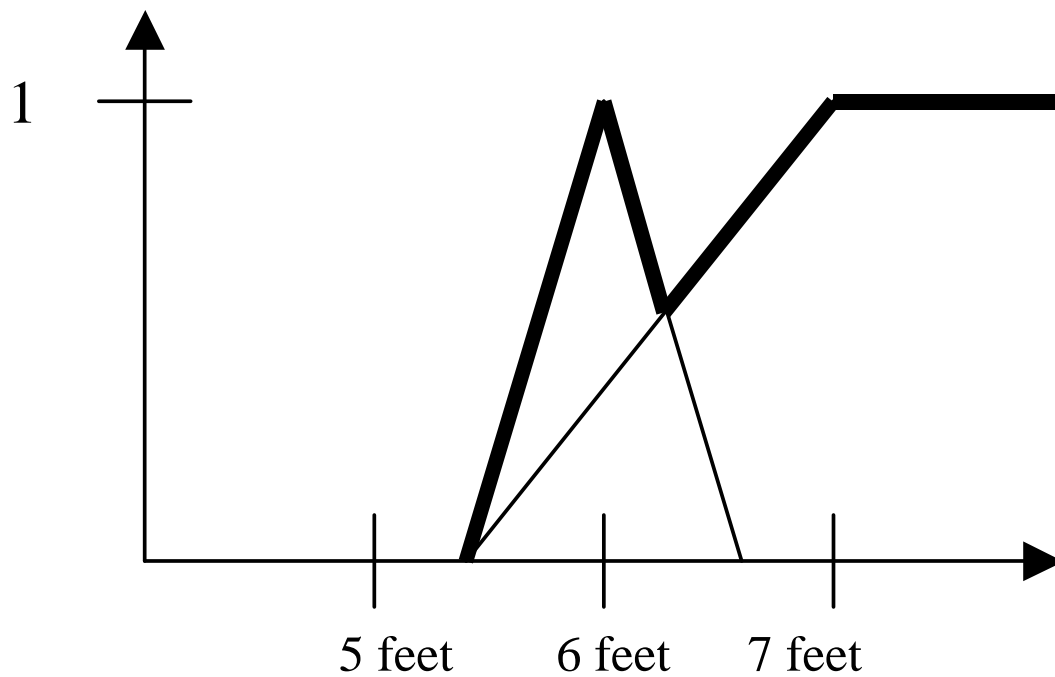


与



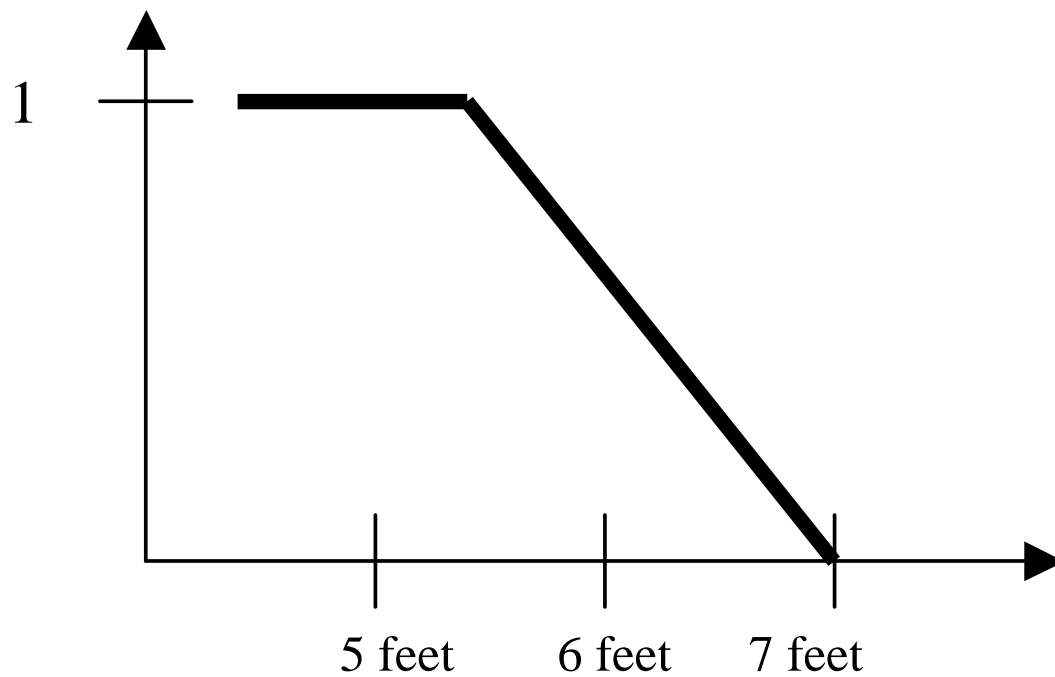
浙江大学CAD&CG 国家重点实验室

或



浙江大学CAD&CG 国家重点实验室

非



浙江大学CAD&CG 国家重点实验室

模糊控制

- 举例： 车辆驾驶
- 前提： 两辆车之间不能相撞
- 在模糊逻辑中的实现：
 - 用两个变量描述每一辆车
 - 当前时刻，车与前面一辆车之间的距离 d
 - 当前时刻与前一时刻距离的差 δd



模糊控制

- If $\delta d=0$ 且 d =两个车位长, 保持现有速度
- If $\delta d<0$ 且 $d<$ 两个车位长, 减慢速度
- If $\delta d>0$ 且 $d>$ 两个车位长, 加快速度



小结

- 模糊逻辑和模糊控制被广泛用于游戏当中
 - 当你想模拟人的思维模式时
- 模糊逻辑同样能够用于表示无生命时间
 - 给定风速和方向，问云如何移动



小结

- 在游戏中，模糊逻辑还能够用于
 - 对抗敌人的人工智能
 - 非玩家的角色（描述某个贩卖情报的人对你的信任程度）
 - **Flocking**算法



神经网络

- 简化的人脑模型
 - 人脑大概有 10^{12} 个神经元
 - 每一个神经元都能够处理和发送信息
- 神经元的三个主要组成部分：
 - 细胞体，神经元新陈代谢的中心
 - 树突，接收来自其他神经元的信号
 - 轴突，向其他神经元发送信号



神经网络

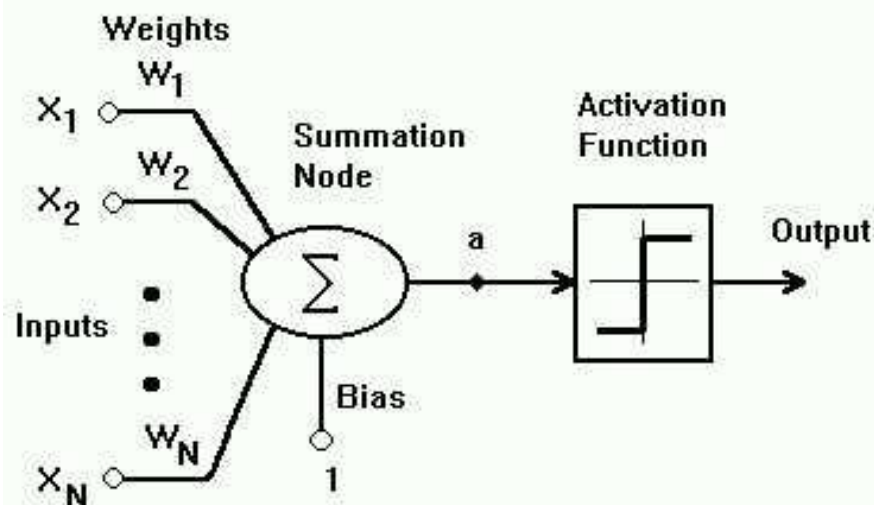
- 生物学发现
 - 神经元是人脑的基本组成部分
 - 如果将神经元看作结点，它们之间的连接看作弧，则这些神经元组成一个稠密连接的图
- 虽然单个神经元的工作过程较简单，当大量神经元连成一个网络并动态运行时，系统是非常复杂的



人工神经元模型

- 是人类大脑神经元的简化

- N个输入
- 1个输出
- 作用函数



$$a = W_1 X_1 + W_2 X_2 + \dots + W_N X_N + \text{Bias}$$

$$\text{output} = \text{Threshold}[a]$$

$$\text{where } \text{Threshold}[a] = \begin{cases} -1, & \text{for all } a \leq 0 \\ 1, & \text{for all } a > 0 \end{cases}$$



人工神经网络基础

- McCulloch and Pitts与1943年第一次提出人工神经网络概念
- 一个处理单元将接收的信息 x_0, x_1, \dots, x_{n-1} 通过用 w_0, w_1, \dots, w_{n-1} 表示互联强度，以点积的形式合成自己的输入，并将输入与以某种方式设定的阈值 θ 相比较，再经某种形式的作用函数 f 的转换，得到该单元的输出 y
- f 可以是阶梯函数、线性或者是指数形式的函数



神经计算

- 神经网络是基于人脑的平行体系结构
- 与多处理器计算机相类似
 - 独立处理单元
 - 高度互联
 - 简单消息传递
 - 适应性交互



训练神经网络

- 初始化：随机设定各条边的 W 值
- 给定一对（输入，输出），已有神经网络根据输入计算输出，将其与预计输出相比较，并根据两者之间的差值调整各条边的 W 值
- 神经网络也可以自动学习，但是相比训练，收敛速度要慢很多

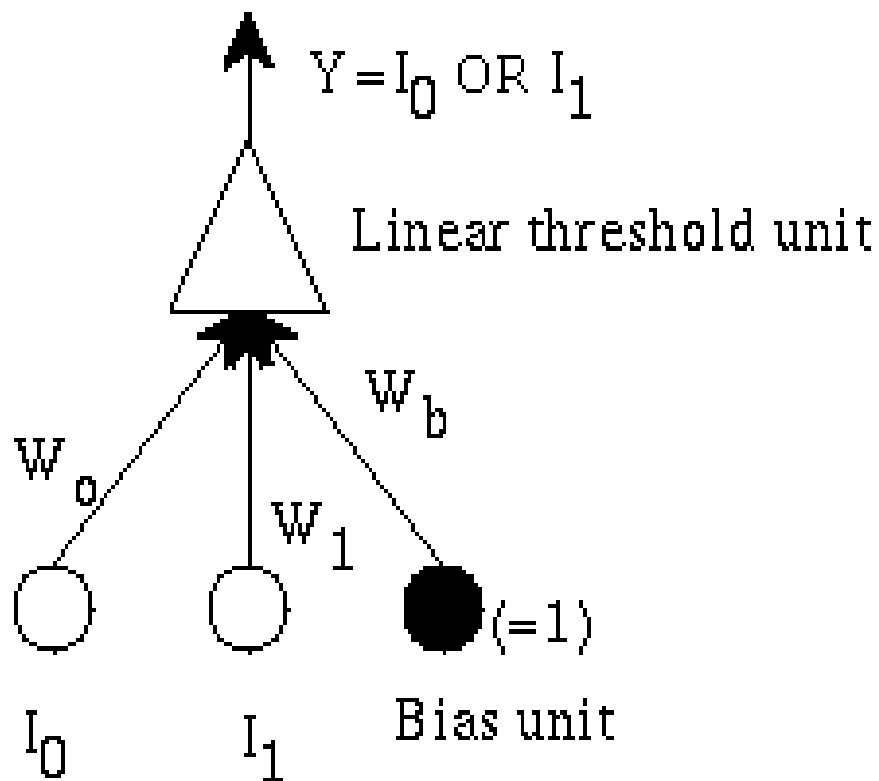


训练神经网络

- 对已知样本分类的正确率
- 对未知样本分类的正确率
- 过训练



对逻辑关系“或”的学习



网络有两个输入，一个输出，都是二元变量

输出为1如果 $W_0 \times I_0 + W_1 \times I_1 + W_b > 0$

输出为0如果 $W_0 \times I_0 + W_1 \times I_1 + W_b \leq 0$



调整权重

- 权重的修改与期望输出和实际输出之差成正比

η 是学习率， d 是期望输出， y 是实际输出， x_i 是输入

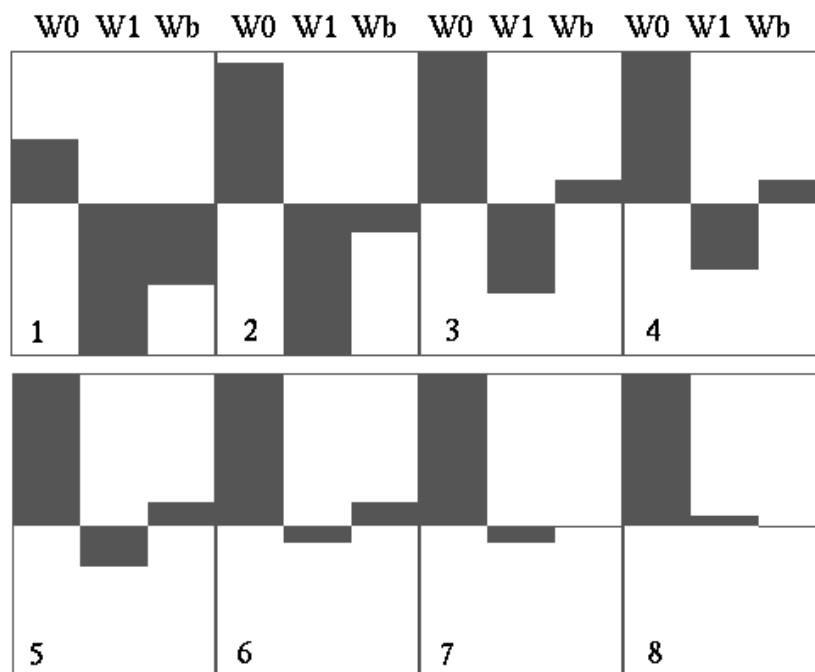
$$\delta W_i = \eta[d - y] \times x_i$$

I_0	I_1	Desired output
0	0	0
0	1	1
1	0	1
1	1	1



示例

- 当第8步时, $(d-y)=0$, 因此 $\delta W=0$, 则训练结束



多层感知器

- 反向传播网络
- 径向基函数网络
- 能够学习任意复杂的模式
- 输入、输出均可以为实数

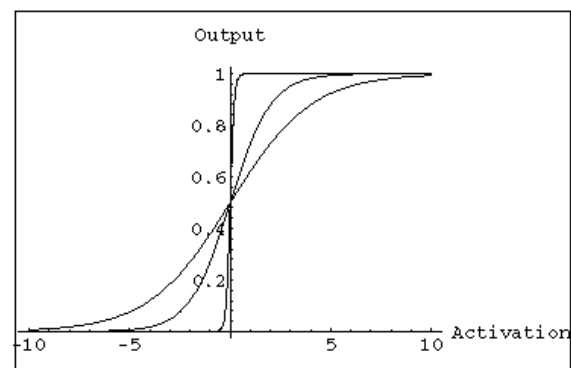
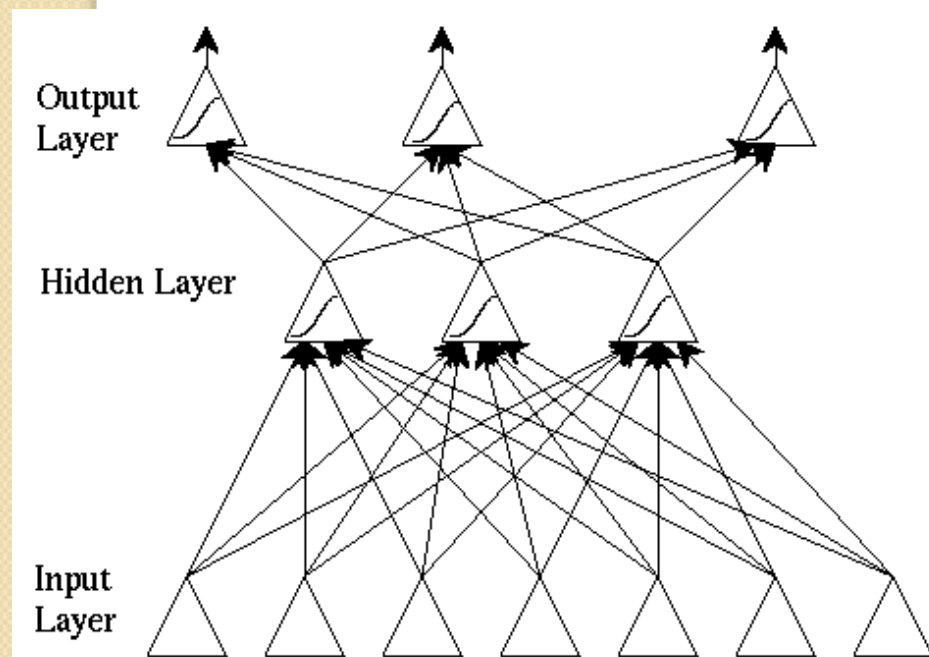


反向传播网络

- 三层：输入层、隐含层、输出层，前一层的输出是后一层的输入
- 是一种前馈网，不形成回路
- 可以有多个隐含层
- 三层结点已经能够产生任意复杂的映射



典型BP网络结构



作用函数:

$$Y = \frac{1}{1 + \exp(-k(\sum W_{in} \times X_{in}))}$$



浙江大学CAD&CG 国家重点实验室

BP学习算法

1. 将全部权值与结点的阈值设置为一个小的随机值
2. 加在输入与输出
3. 计算实际输出
4. 修正权值
 1. 从输出结点开始，反向的向第一隐含层（即存在多层隐含层时最接近输入层的隐含层）传播由总误差诱发的权值修正
5. 在到达预定误差精度和循环次数后退出，否则转步骤2重复



径向基函数网络

- 前馈网络，只有一个隐含层
- 能够表示任意复杂的映射
- 隐含层的作用函数称为径向基函数，在某一点函数有最大值，而离开该点一定距离的值被映射为0
- 一般的，取径向基函数为高斯函数



训练RBF网络

- 需要决定
 - 隐含层包含多少个结点
 - 每个结点作用函数
- 训练过程
 - 首先通过观察训练样本，决定作用函数的形状
 - 用前面的**delta**规则修正权重
- 应用
 - 物体分类



小结

- **BP**和**RBF**网络是两个常用的神经网络模型
- 当系统遇到新的未知样本，**RBF**可以通过添加隐含层结点加强系统的判断能力
- 两者都能处理动态时序数据



神经网络应用

- 当我们没办法明确给出一个算法解时
- 当我们有充足的样本时
- 当我们需要从数据中获得一点什么时

我们可以使用神经网络



神经网络应用

- 对于那些传统计算解决不了的问题，神经网络也无法解决
- 神经网络可以简化某些特定问题的解答，例如，从数据中提炼一个模型
- 对于数据形成过程未知或者复杂的问题而言，神经网络能够帮助我们从一定程度上理解内在的规律



神经网络应用

- 投资分析
- 笔迹分析
- 过程控制
- 市场调查
- 状态监控



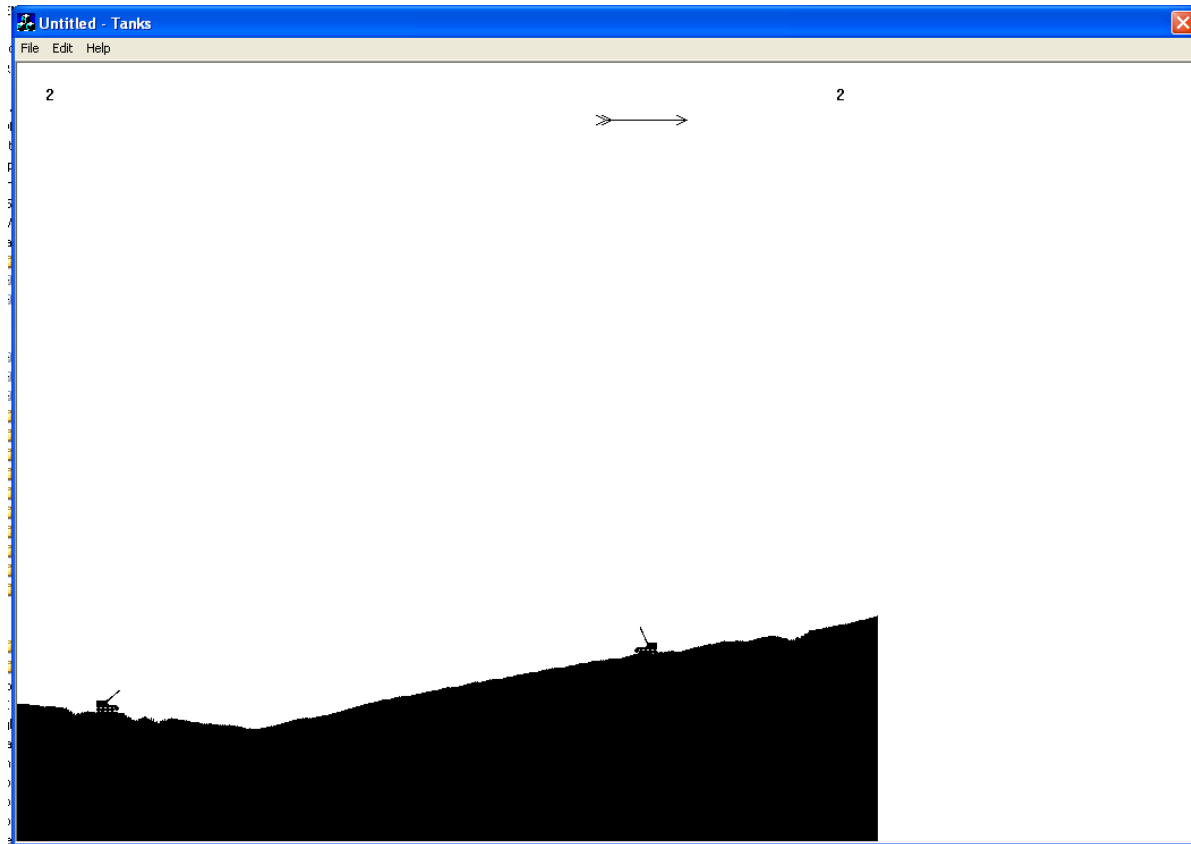
神经网络与游戏

- 判断所处的环境
- 决定下一步的动作
- 用于表示积累的经验



神经元网络

- 坦克的射击训练示例
 - Neuro network demo



遗传算法

- 遗传算法的基本思想是基于**Darwin**进化论和**Mendel**的遗传学说的
 - 适者生存原理
 - 基因遗传原理（基因突变和基因杂交）
- 遗传算法一般用于在难以预测其中各个因素之间相互作用的大型系统上作非线性优化



遗传算法工作原理

- 选择初始群体
- 观察每个个体对环境的适应能量
- 选择
- 重复
 1. 杂交
 2. 变异
 3. 观察每个个体对环境的适应能量
 4. 选择
- 直到满足某些结束条件



进化和遗传学的概念

- 串 (**string**)
 - 它是个体(**Individual**)的形式，在算法中为二进制串，并且对应于遗传学中的染色体(**Chromosome**)
- 群体(**Population**)
 - 个体的集合称为群体，串是群体的元素
- 基因(**Gene**)
 - 基因是串中的元素，基因用于表示个体的特征。例如有一个串 $S = 1011$ ，则其中的1，0，1，1这4个元素分别称为基因。它们的值称为等位基因(**Alletes**)
- 基因位置(**Gene Position**)
 - 一个基因在串中的位置称为基因位置，有时也简称基因位。基因位置由串的左向右计算，例如在串 $S = 1101$ 中，0的基因位置是3。基因位置对应于遗传学中的地点(**Locus**)

进化和遗传学的概念

- 基因特征值(**Gene Feature**)
 - 在用串表示整数时，基因的特征值与二进制数的权一致；例如在串**S=1011**中，基因位置**3**中的**1**，它的基因特征值为**2**；基因位置**1**中的**1**，它的基因特征值为**8**
- 非线性
 - 它对应遗传学中的异位显性(**Epistasis**)
- 适应度(**Fitness**)
 - 表示某一个体对于环境的适应程度

选择

- 这是从群体中选择出较适应环境的个体。这些选中的个体用于繁殖下一代。故有时也称这一操作为再生(**Reproduction**)。由于在选择用于繁殖下一代的个体时，是根据个体对环境的适应度而决定其繁殖量的，故而有时也称为非均匀再生(**differential reproduction**)



选择

- 根据适者生存原则选择下一代的个体。在选择时，以适应度为选择原则。适应度准则体现了适者生存，不适应者淘汰的自然法则
- 给出目标函数 f ，则 $f(b_i)$ 称为个体 b_i 的适应度
- $$P(\text{select } b_i) = \frac{f(b_i)}{\sum f(b_j)} \times n$$
为选中 b_i 为下一代个体的次数



选择

- 性质：
 1. 适应度较高的个体，繁殖下一代的数目较多。
 2. 适应度较小的个体，繁殖下一代的数目较少；甚至被淘汰。
- 选择产生对环境适应能力较强的后代。对于问题求解角度来讲，就是选择出和最优解较接近的中间解。



交叉

- 对于选中用于繁殖下一代的个体，随机地选择两个个体的相同位置，按交叉概率 P ，在选中的位置实行交换。这个过程反映了随机信息交换；目的在于产生新的基因组合，也即产生新的个体。交叉时，可实行单点交叉或多点交叉



交叉

例如有个体

$S1=100101$

$S2=010111$

选择它们的左边**3**位进行交叉操作，则有

$S1=010101$

$S2=100111$

一般而言，交叉概率**P**的取值为**0.25-0.75**



变异

- 根据生物遗传中基因变异的原理，以变异概率 P_m 对某些个体的某些位执行变异。在变异时，对执行变异的串的对应位求反，即把1变为0，把0变为1。变异概率 P_m 与生物变异极小的情况一致，所以， P_m 的取值较小，一般取0.01-0.2
 - 例如有个体 $S=101011$ ，对其的第1、4位置的基因进行变异，则有 $S'=001111$
- 单靠变异不能在求解中得到好处。但是，它能保证算法过程不会产生无法进化的单一群体。因为在所有的个体一样时，交叉是无法产生新的个体的，这时只能靠变异产生新的个体。也就是说，变异增加了全局优化的特质。



组合

- 选择 + 杂交 = 进化
 - 选择使得适者生存
 - 杂交将不同个体中优良的基因保存下来，创造新的具有各方面优势的品种
- 选择 + 变异 = 在优化中加入随机扰动
 - 遗传算法是采用随机方法进行最优解搜索，选择体现了向最优解逼近，变异体现了全局最优解的复盖
 - 坏的变异将最终被选择出去



组合

- 选择 + 杂交 + 突变 = 遗传算法的力量



遗传算法

- $P :=$ 以随机方式产生串的集合
- 如果最优个体的适应度还未达到给定的阈值，或者最优个体的适应度和群体适应度仍然在上升
 1. 令 $f_i = \text{Fitness}(p_i)$, $i = 1 \dots n$
 2. 令 $P' = \text{SelectionNewPopulation}(p, f)$
 3. 随机两两组合 P' 中的个体
 4. 对每一对个体，以概率 C 进行杂交
 5. 对 P' 中的每一个个体，以概率 M 进行编译
 6. 令 $P = P'$



结束条件

- 最优个体的适应度达到给定的阈值
- 最优个体的适应度和群体适应度不再上升
- 达到预先设定的最大循环数（繁衍代数）
- 群体中的所有个体具有相同的属性



遗传算法参数

- 群体大小 n
- 交叉概率 P_c
- 变异概率 P_m
- 繁衍代数
- 其他： 取决与具体的操作和结束条件



编码方式

- 除二进制编码外，问题的各种参数可以用实数向量构成子串
- 选择：与串类似
- 变异：将按照高斯概率分布的随机变量 g 加到某个参数上



遗传规划

- 遗传算法的一个分支，由**Koza**提出，与遗传算法用串的形式表示所不同的是，遗传规划的表示是计算机程序
- 它是一种自动编程技术
 - 终结符集合：变量、常数
 - 函数集合：程序中的函数
- 用分析树的形式表示中间产生的程序

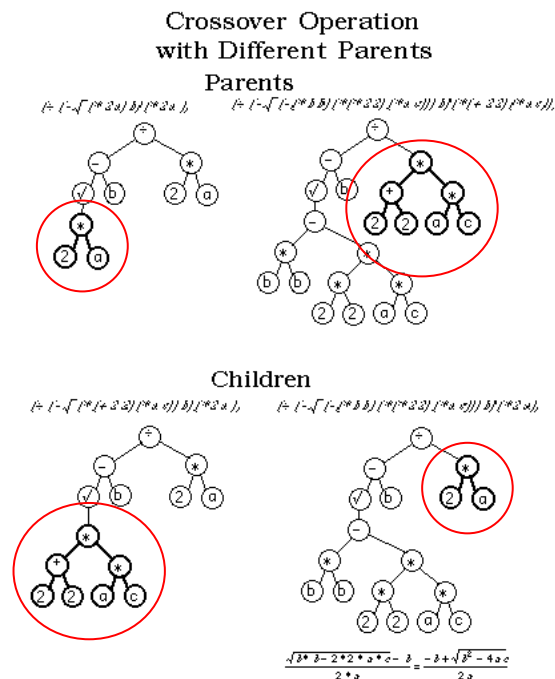


初始化

- 先确定终结符集合和函数集合
- 随机产生初始群体
- 每一个个体（分析树）按照下列方式产生：
 - 令 T =空的分析树
 - 令 C =随机的终结符或者函数
 - 将 C 加入 T 中
 - 如果 C 位于预先设定的树的最大深度，则从终结符集合中任意选择 C 的儿子结点，并添加到 T 中
 - 否则，对 C 的儿子重复上述过程

杂交

- 从待杂交的两棵树中任意选择一个结点，并交换以该结点为根结点的两棵子树



变异

- 选择某一个结点，将函数替换成另一个函数，将终结符替换成另一个终结符
- 选择某一个结点，删除以其为根的整棵子树，再以随机方式生成一棵子树



遗传规划算法

1. 令 P = 随机初始群体
2. 循环直到满足某个结束条件
 1. 下一代 $R\%$ 个体与上一代完全一致
 2. 下一代 $M\%$ 个体是从上一代变异而来
 3. 下一代 $C\%$ 个体是从上一代杂交而来
 4. $(R+M+C=100)$ 允许重复选择，所有选择都是基于适应度函数
3. 遗传规划的群体个数一般都比较大大



注意事项

- 编码（表示）的选择：字符串并不是唯一的表示方法
- 遗传操作的选择
- 适应度函数的选择
- 参数的取值



演示

- GA demo



浙江大学CAD&CG 国家重点实验室

AI引擎设计

- AI引擎可以帮助解决很多问题
 - 使得游戏角色之间交流更为容易
 - 提供实现AI行为的解决方法
 - 协助保留每一个错误报告

技巧

- 对于行为简单的物体，使用简单的确定性AI技术
- 对于不是主要角色，但是需要一点智能行为的物体，可以对其设定几种模式，并加上一点随机的因素扰动即可
- 对于比较重要的角色，可以使用有限状态机技术，加上另外一些辅助技巧
- 对于最最重要的角色，你需要利用一切可能利用的技术
 - 状态驱动，使用条件逻辑、概率、以及经历过的状态等控制状态转移
 - 如果状态转移条件满足，物体要能够发生强制性状态转移动作

什么是好的AI

- 用户觉得游戏角色挺聪明的
- 感觉到游戏角色的确随着经历而在成长
- 一层层的揭开面纱，知道最后才恍然大悟

一些资源

- <http://www.gameai.com>
- Herbert Schildt, “Artificial Intelligence Using C”, McGraw-Hill, 1987
- Kim W. Tracy, Peter Bouthoorn, “Object-oriented Artificial Intelligence Using C++”, Computer Science Press, 1997
- AI game development
- AI game programming wisdoms
- AI game programming wisdoms 2