**行为树（Behavior Tree）实践（1）– 基本概念**

2011年7月27日 | 标签: [Behavior](http://www.aisharing.com/archives/tag/behavior), [Behavior Tree](http://www.aisharing.com/archives/tag/behavior-tree), [Decision](http://www.aisharing.com/archives/tag/decision), [next-gen AI](http://www.aisharing.com/archives/tag/next-gen-ai), [行为](http://www.aisharing.com/archives/tag/%e8%a1%8c%e4%b8%ba), [行为树](http://www.aisharing.com/archives/tag/%e8%a1%8c%e4%b8%ba%e6%a0%91), [请求](http://www.aisharing.com/archives/tag/%e8%af%b7%e6%b1%82)

自从开博以来，每天都会关心一下博客的访问情况，看到一些朋友的订阅或者访问，不胜欣喜，也促使我去写一些更好的博文，来和大家分享和交流，从访问统计来看，有相当一部分是来自于搜索引擎的流量，关键字以“行为树”，或者“Behavior Tree”居首位，我想大家对此可能有些兴趣，加上，这几年反反复复一直在AI中研究和运用行为树，所以这次就来谈谈关于行为树（Behavior Tree）的一些东西，以前也写过一些文章（[1](http://www.aisharing.com/archives/22" \t "_blank)，[2](http://www.aisharing.com/archives/26" \t "_blank)，[3](http://www.aisharing.com/archives/29" \t "_blank)）来讨论行为树，不过已经是一两年前的事情了，较之以前，这次会更为系统，也会添加一些我新的思考和感悟。所谓行为树实践，其实在我脑海里就是Practice in Behavior Tree，没法子，受英文教材影响太多了:)

我想通过一个例子来介绍一下行为树的基本概念，会比较容易理解，看下图：

这是我们为一个士兵定义的一颗行为树（可以先不管这些绿圈和红圈是干吗的），首先，可以看到这是一个树形结构的图，有根节点，有分支，而且子节点个数可以任意，然后有三个分支，分别是巡逻（Patrol），攻击（Attack），逃跑（Retreat），这个三个分支可以看成是我们为这个士兵定义的三个大的行为（Behavior），当然，如果有更多的行为，我们可以继续在根节点中添加新的分支。当我们要决策当前这个士兵要做什么样的行为的时候，**我们就会自顶向下的，通过一些条件来搜索这颗树，最终确定需要做的行为（叶节点），并且执行它**，这就是行为树的基本原理。

值得注意的是，我们标识的三大行为其实并不是真正的决策的结果，它只是一个类型，来帮助我们了解这个分支的一些行为是属于这类的，真正的行为树的行为都是在叶节点上，一般称之为行为节点（Action Node），如下图红圈表示的

这些叶节点才是我们真正通过行为树决策出来的结果，如果用我以前提到的那个[层次化的AI结构](http://www.aisharing.com/archives/86)来描述的话，这些行为结果，相当于就是一个个定义好的“请求”（Request），比如移动（Move），无所事事（Idle），射击（Shoot）等等。所以行为树是一种决策树，来帮助我们搜寻到我们想要的某个行为。

行为节点是游戏相关的，因不同的游戏，我们需要定义不同的行为节点，但对于某个游戏来说，在行为树上行为节点是可以复用的，比如移动，在巡逻的分支上，需要用到，在逃跑分支上，也会用到，这种情况下，我们就可以复用这个节点。行为节点一般分为两种运行状态：

1. 运行中（Executing）：该行为还在处理中
2. 完成（Completed）：该行为处理完成，成功或者失败

除了行为节点，其余一般称之为控制节点（Control Node），用树的“学名”的话，就是那些父节点，如下图绿圈表示

控制节点其实是行为树的精髓所在，我们要搜索一个行为，如何搜索？其实就是通过这些控制节点来定义的，从控制节点上，我们就可以看出整个行为树的逻辑走向，所以，行为树的特点之一就是其**逻辑的可见性。**

我们可以为行为树定义各种各样的控制节点（这也是行为树有意思的地方之一），一般来说，常用的控制节点有以下三种

1. 选择（Selector）：选择其子节点的某一个执行
2. 序列（Sequence）：将其所有子节点依次执行，也就是说当前一个返回“完成”状态后，再运行先一个子节点
3. 并行（Parallel）：将其所有子节点都运行一遍

用图来表示的话，就是这样，依次为选择，序列和并行

可以看到，控制节点其实就是“控制”其子节点（子节点可以是叶节点，也可以是控制节点，所谓“执行控制节点”，就是执行其定义的控制逻辑）如何被执行，所以，我们可以扩展出很多其他的控制节点，比如循环（Loop）等，与行为节点不同的是，控制节点是与游戏无关的，因为他只负责行为树逻辑的控制，而不牵涉到任何的游戏代码。如果是作为一个行为树的库的话，其中就一定会包含定义好的控制节点库。

如果我们继续考察选择节点，会产生一个问题，如何从子节点中选择呢？选择的依据是什么呢？这里就要引入另一个概念，一般称之为前提（Precondition），每一个节点，不管是行为节点还是控制节点，都会包含一个前提的部分，如下图

前提就提供了“选择”的依据，它包含了进入，或者说选择这个节点的条件，当我们用到选择节点的时候，它就是去依次测试每一个子节点的前提，如果满足，则选择此节点。由于我们最终返回的是某个行为节点（叶节点），所以，当前行为的“总”前提就可以看成是：

当前行为节点的前提 **And** 父节点的前提 **And**父节点的父节点的前提 **And**….**And** 根节点的前提（一般是不设，直接返回True）

行为树就是通过行为节点，控制节点，以及每个节点上的前提，把整个AI的决策逻辑描述了出来，对于每次的Tick，可以用如下的流程来描述：

action = root.FindNextAction(input);  
if action is not empty then  
action.Execute(request,  input)  //request是输出的请求  
else  
print “no action is available”

从概念上来说，行为树还是比较简单的，但对AI程序员来说，却是充满了吸引力，它的一些特性，比如可视化的决策逻辑，可复用的控制节点，逻辑和实现的低耦合等，较之传统的状态机，都是可以大大帮助我们迅速而便捷的组织我们的行为决策。希望这次简单的介绍，对大家有所帮助，能力有限，不一定能表述的很清楚，有问题，或者有指教的，都请和我多多交流，最后，我对这个士兵的巡逻分支画了一个示意图，供大家参考：

**S — 选择节点   Se — 序列节点**

————————————————————————  
作者：Finney  
Blog：AI分享站(<http://www.aisharing.com/>)  
Email：finneytang@gmail.com  
本文欢迎转载和引用，请保留本说明并注明出处  
————————————————————————

[8](http://www.wumii.com/item/2nX4guvn)

**您可能也喜欢：**

* [共享节点型的行为树架构（1）](http://www.aisharing.com/archives/563)
* [从有限状态机（FSM）到行为树（Behavior Tree）（2）](http://www.aisharing.com/archives/452)
* [行为树（Behavior Tree）实践（2）– 进一步的讨论](http://www.aisharing.com/archives/99)
* [共享节点型的行为树架构（2）](http://www.aisharing.com/archives/572)
* [用800行代码做个行为树（Behavior Tree）的库（3）](http://www.aisharing.com/archives/530)

[无觅](http://www.wumii.com/widget/relatedItems)

 (2 投票, 平均: 5.00 / 5)

[发表评论](http://www.aisharing.com/archives/90#respond) | [Trackback](http://www.aisharing.com/archives/90/trackback) | 已被阅读3,041次

**一个诚实的人**

2013年8月16日20:05

[回复](http://www.aisharing.com/archives/90?replytocom=3309#respond) | [引用](javascript:void(0);) | [#1](http://www.aisharing.com/archives/90#comment-3309)

文章不错，如果能将reference写一下就更好了。

**Raven**

2012年2月16日16:41

[回复](http://www.aisharing.com/archives/90?replytocom=167#respond) | [引用](javascript:void(0);) | [#2](http://www.aisharing.com/archives/90#comment-167)

站长您好~~有没有什么好的书籍来学习行为树的知识（外文也行）或者网站~~？ 求推荐！~

[**Finney**](http://www.aisharing.com/)

2012年2月16日17:11

[回复](http://www.aisharing.com/archives/90?replytocom=168#respond) | [引用](javascript:void(0);) | [#3](http://www.aisharing.com/archives/90#comment-168)

有个网站很不错，英文的aigamedev.com，里面有一些behavior tree的资料

**小N\_大进行曲**

2011年11月26日00:11

[回复](http://www.aisharing.com/archives/90?replytocom=113#respond) | [引用](javascript:void(0);) | [#4](http://www.aisharing.com/archives/90#comment-113)

请问代码中的input指什么呢。

**小N\_大进行曲**

2011年11月26日00:06

[回复](http://www.aisharing.com/archives/90?replytocom=112#respond) | [引用](javascript:void(0);) | [#5](http://www.aisharing.com/archives/90#comment-112)

又看了一遍，略微懂了，但是请问Tick是什么意思，还有实例代码中的input指什么？  
实例代码是每帧都执行，还是事件驱动执行？

[**Finney**](http://www.aisharing.com/)

2011年11月26日12:16

[回复](http://www.aisharing.com/archives/90?replytocom=117#respond) | [引用](javascript:void(0);) | [#6](http://www.aisharing.com/archives/90#comment-117)

Tick就是游戏每一次循环，input可能名字不是很好，我这里指的是传入行为树的参数

**小N\_大进行曲**

2011年11月25日16:29

[回复](http://www.aisharing.com/archives/90?replytocom=111#respond) | [引用](javascript:void(0);) | [#7](http://www.aisharing.com/archives/90#comment-111)

请问 行为树 和 层次状态机 有什么区别啊。  
我了解过层次状态机，看这个文章觉得跟行为树没什么区别呀？

[**Finney**](http://www.aisharing.com/)

2011年11月26日12:15

[回复](http://www.aisharing.com/archives/90?replytocom=116#respond) | [引用](javascript:void(0);) | [#8](http://www.aisharing.com/archives/90#comment-116)

这个要说起来就多了，简单的说吧，行为树是树状结构，绝对没有“回路”，而状态机是图状的结构，就是有时会形成一个“回路”，而且，行为树的决策方式是自顶向下的，状态机是在每一个状态内部做决策的

**小N\_大进行曲**

2011年11月26日12:38

[回复](http://www.aisharing.com/archives/90?replytocom=119#respond) | [引用](javascript:void(0);) | [#9](http://www.aisharing.com/archives/90#comment-119)

对于每一帧的Update()函数来说：  
状态机，只需currentState.Update()，在其中遍历当前状态的所有Transition（转换条件）即可。  
而行为树，却需要从根部节点按照遍历规则，从头开始遍历。（比如说从最左边的节点开始）  
这样以来，是不是每一次Update，BT都要“统揽全树”，而FSM只需要“关心自己”，BT所做的事情就要比FSM多？效率会降低吗？

另外，今天看了一片英文文章，文章说，BT和FSM的区别在于，相对于FSM，BT把“转换条件”从“状态”中剥离出来，也就是说，FSM中的“状态”，在BT中演化成了“转换条件”+“行为”？

[**Finney**](http://www.aisharing.com/)

2011年11月26日12:49

[回复](http://www.aisharing.com/archives/90?replytocom=120#respond) | [引用](javascript:void(0);) | [#10](http://www.aisharing.com/archives/90#comment-120)

由于只是对转换条件的判断，所以效率并不低，而且从本质上来说，你在状态机中也是需要判断所有可能的转换条件的，顺便说一句，这篇文章对于区别说的比我好，呵呵

**坑王**

2012年2月4日11:40

[回复](http://www.aisharing.com/archives/90?replytocom=161#respond) | [引用](javascript:void(0);) | [#11](http://www.aisharing.com/archives/90#comment-161)

能请问一下这篇文章的名字吗？也想看一下