Decision Trees

Simple decision making with tree graphs

Canberra Institute of Technology

aie
SPECIALIST EDUCATORS IN
GAMES, ANIMATION & FILM VFX

The Problem with Finite State Machines

- State Machines are great for simple behaviours

- But when we have lots of states we need to handle lots of transitions
  - The number of transitions vastly outnumber the number of states
  - If a transition is missed then the state machine can react badly or even not at all

- For complex decision making that needs to be quick, flexible and extendable, Finite State Machines might not be the ideal choice

Decision Trees

- Sometimes we want decisions to be made in a more free-form interruptible fashion
- Decision Trees are a decision making technique that allows for interruptible states
- A Decision Tree works by asking a series of quick questions to arrive at an “answer”
  - The tree branch nodes are the “questions”
  - The tree leaf nodes are the “answers”
  - The “answer” being the State or Action we desire

Decision Trees

- Decision Trees are made up of Decision nodes
  - Decisions can be questions or answers
  - Question nodes have branches to other Decisions

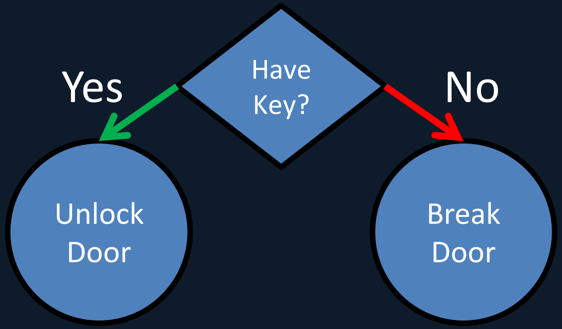
- When a Decision node is polled to make a decision:
  - A question determines which branch to poll next
  - An answer executes its decision
    - Such as “open door” for example

```
class Decision
   func makeDecision() = 0
```

```
class BooleanDecision : Decision
   boolean value
   Decision trueBranch
   Decision falseBranch

   func makeDecision()
      if value == true
         trueBranch.makeDecision()
      else
         falseBranch.makeDecision()
```

Example Question Node



Canberra Institute of Technology

aie
SPECIALIST EDUCATORS IN GAMES, ANIMATION & FILM VFX

Decision Trees “Answers”

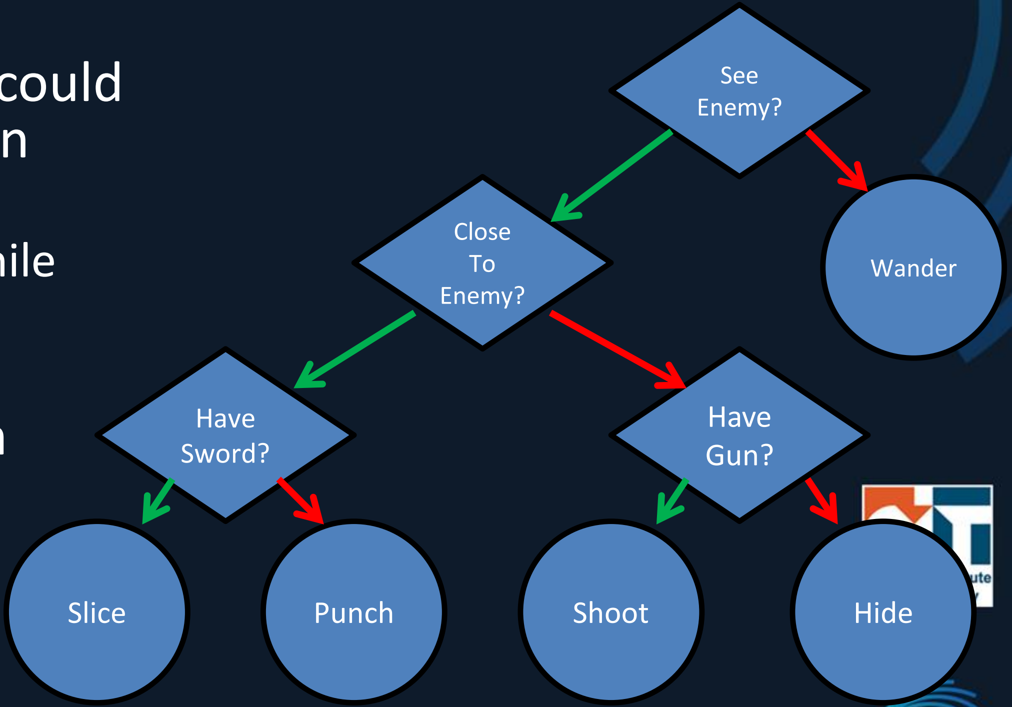
- Answer nodes are like States within a Finite State Machine
  - They perform the actions required of the A.I. once the decision has been made
    - For example, playing the correct animation and applying damage for an Attack decision

```
class AttackDecision : Decision
   integer damageToApply
   float range

   func makeDecision()
      for each enemy within range
         enemy.health -= damageToApply
```

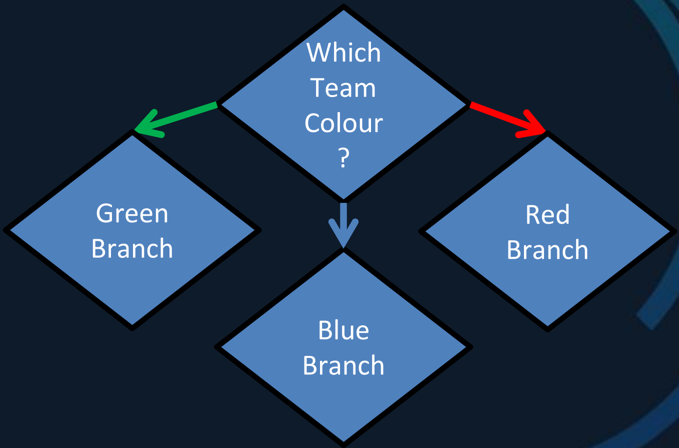
Decision Tree Example

- This example Decision Tree could represent the decisions of an agent in a combat game
  - Red lines represent “**No**” while green represent “**Yes**”
- The agent polls the Decision Tree from the tree’s root each time it needs to make a decision
  - For example, each frame within the agent’s update



Decision Tree “Questions”

- Question nodes don’t have to be a simple yes / no
  - Yes / No trees are binary trees
- Questions can act as a switch statement
  - Multiple branches
- Questions can also have timeouts, cooldowns and other modifiers
  - For example, a question may execute branch A the first time it is polled, and branch B all other times
  - A question may execute the previous branch it executed until a timer has run out



```
class BooleanDecision : Decision
   boolean value
   Decision trueBranch, falseBranch
   Decision lastDecision
   float timer, cooldownTime

   func makeDecision()
      timer -= deltaTime
      if timer < 0
         timer = cooldownTime
         if value == true
            lastDecision = trueBranch
         else
            lastDecision = falseBranch
      lastDecision.makeDecision()
```

Advantages

- Can respond to “interrupts”
  - As an example, “***while talking to agent B, agent A gets shot in the face***” and responds accordingly, without having to explicitly code a transition for “**Shot in the face while talking to a agent**”
- Can break down decisions into discrete code statements

Disadvantages

- Actions require lots of specific code
  - An action might need to move the agent as well as animate it and spawn particles
  - Duplicates code because a Flee action also requires moving the agent and playing an animation

Summary

- Finite State Machines aren't suitable for all situations
- Decision Trees are a way to arrange decision making into a series of questions within a tree graph
  - Branches are questions
  - Leaves are answers
- Creates a very modular and interruptible A.I. decision process