INTRO TO BEHAVIOR TREES

Damián Isla,
Moonshot Games
Let’s Situate Ourselves

**CLASSICAL AI**
- Search
- Planning
- Alpha-Beta
- Constraint Satisfaction
- Predicate Logic

**MACHINE LEARNING**
- Stats
- Regression
- Classification
- Bayes Nets
- NLP
- Etc.

**A-LIFE**
- Illusion of Life
- Intelligent Performance
- Flocking
3 Requirements for Game AI

- Coherence
  - the AI feels like a living, thinking creature

- Transparency
  - the AI’s behavior is explainable

- Workability
  - we can author, modify and fix them, with confidence and intentionality
Player Transparency

- Player can explain AI behavior
  - “He dove out of the way because I threw the grenade.”

- Player can predict certain AI behavior
  - “If I throw the grenade, I bet the AI will dive out of the way.”

- Player forms an ongoing narrative in his/her head
  - “I threw the grenade, so he dove out of the way, and then cursed at me, and threw a grenade back, but I shot it in the air and hurt him, so he went nuts and charged me …”

- The AI needs to facilitate / encourage that narrative
Designer Transparency

- Designer can explain AI behavior
  - “He dove out of the way because his danger_dive behavior is active.”

- Designer can predict certain AI behavior
  - “When I throw this grenade, there’s a 75% chance he’ll dive out of the way”

- Designer knows how to achieve different behavior
  - “Hmm ... he reacted to that grenade too quickly ... I think I’ll increase his projectile-acknowledgement delay from 0.5 to 0.7 seconds.”

- Design knows how to diagnose and fix MISbehavior
  - “WOAH ... why the hell did he do THAT?!”
Defining Characteristic of Game AI

The Role of the Author
(i.e. the game designer)

- The Input
- The Algorithm
- The Designer

Active areas of research: Intelligent designer tools
- Procedurally generated content (PCG)
- Smart content validation
- Design validation
Agenda

- Intro to Game AI
- The World’s Greatest Decision-Making Technology
- FSMs
- Why FSMs Suck
- Behavior Trees v0 – v2+
- Why Behavior Trees Suck
- Conclusions
Let's Situate Ourselves

SENSE

PERCEPTION

KR

PLAN

DECISION-MAKING

ACT

AIMING

ANIMATION

LOOKING

DIALOGUE
The Greatest Decision-Making Technology in the World

If X then Y else Z \( \times 10,000 \)

- Transparent
- Debuggable
- Easily authored

Only problem:
- Maintaining coherence, transparency and workability with 10,000 rules is hard.
Finite State Machines

- Easy to understand
- Easy to implement
- Easy to debug
- VERY flexible
Inside a State

- Where I go
- How I get there
- What I look/aim/shoot at
- Special Animations
- etc.

- Outward transitions: what should I do next?
Why FSMs Suck
Why FSMs Suck
Why FSMs Suck
Why FSMs Suck

So code complexity grows $\sim N^2$

This is where the code lives.
HFSMs

Still $N^2$, but at least $N$ is smaller
(also, God forbid you decide to restructure this)
The Scalability Problem

Every time I want to add a new state, I have to think

- Where can this state come from?
- Where can this state go to?
- What am I breaking?
- And btw, what does this state actually do?
- And then I have to maintain all of this
**The Scalability Problem**

“Do \( A \) whenever the player gets closer than 3 meters.”

- Duplicate this logic in each transition?
- What about when that logic changes?
- Every state needs to know a little about \( A \)
BEHAVIOR TREES V0
Warning

- “Behavior Trees” are a loose concept
- Terminology is far from standard
- Feature-set is far from standard
- This is going to be idiosyncratic

- Also, remember: Behavior Trees are Stupid
  - think of them as a convenient way to organize your thousands of “if-then-else” statements.
BTs Circa 1998

Children Compete!
- Child returns a floating-point desire-to-run
- Parent chooses highest-desire child
- + bonus for currently running child

Word of the Day: Hysteresis
Inside a Behavior

• How much do I desire to run?
• Where I should stand
• How I get there
• What I look/aim/shoot at
• Special Animations
• etc.

• Children

WE JUST KILLED N²!
Or Did We?
Or Did We?
Or Did We?

Charge:

if (target closer than 3)
   desire = 5.2
if (target facing me)
   desire += 1

Flee:

if (target closer than 4)
   desire = 3.3
if (target is player)
   desire += 2
Or Did We?

Charge:
  if (target shield down)
      return 1e5
  if (target closer than 3)
      desire= 5.5
  if (target facing me)
      desire += 1

Flee:
  if (target has rocket launcher)
      return 1e6
  if (target closer than 4)
      desire= 3.3
  if (target is player)
      desire += 2
Or Did We?

Floating-point desire-to-run implies some normalized scale of “desire” across all behaviors

- In practice, nearly impossible

- So we’re really back to $N^2$
  - for any scale > trivial

Note, if this were reinforcement learning, this would be $V^*(s)$ (total future discounted reward)
Insight #1

This is an instance of the Apples-to-Oranges problem.

How do you compare:

- desire to run away from a scary enemy
- desire to attack an enemy
- desire to take cover
- desire to jump in a friend’s vehicle
Insight #2

Designers (and players) think in terms of

- triggers
  - not “exit conditions” (like FSMs)
- priorities
  - not squishy analog desires
HALO 2

The Big Goal: Scalability

Scalability?

No! Scalability!
Binary Relevancy

It is way easier to write a binary relevancy function than a floating point desire-to-run function.

Decision Policies:
- prioritized list
- sequential
- sequential-looping
- random
- etc.
Inside a Behavior

- Do I want to run right now?
- If I am running, do I want to continue running?
- Where I should stand
- How I get there
- ...
- Children
Note, behaviors are often deactivated through interruption, rather than through explicit ending conditions.
Problem: Non-constant Priority

Unless player is in vehicle, in which case...

- Charge
- Fight
- Enter vehicle

- Enter vehicle
- Charge
- Fight
BTv1 Feature 2

“Impulses”

Alternative trigger conditions and priority for behaviors

(What have we lost here, compared to floating-point relevancy functions?)
Spore

Impulses became universal

- Enter player vehicle
- enemy nearby
- enemy present
- vehicle nearby

- Enter vehicle
- Charge
- Fight
- Enter vehicle
And lots more...

Tree masks

- super-fast condition bitvector for super common condition checks
And lots more...

Per-character tree customizations
And lots more...

Dynamic stimulus-driven branches

“GRENADE!”
BTv1 Summary

- Binary Relevancy
- Per-parent customized decision routine
- Impulse-triggers for non-constant priorities

On the debugging Front

- Pretty easy to expose why something happened
- Figuring out why something DIDN’T happen is fiendishly hard.
BEHAVIOR TREES V2+
Behavior Trees v2

- Behavior trees are now VERY widely used

- Lots of enthusiasm for BTs as a broadly expressive language for crafting behavior
Behavior Selection Tree Editor, CryEngine3
Behave 2 Unity Plug-in by Angry Ant
Behavior Trees as a Visual Language

- Just like, say, Unreal’s Kismet

- What is good about that?
- What is bad?
Behavior Trees as a Visual Language

Just like, say, Unreal's Kismet

What is good about that?

What is bad?
Considerations

- How often are decisions made?

- Pre-validation or not?
  - this starts to get really close to an HTN
  - though HTNs are generally looking for a single solution

- How granular is the tree?

- Who makes the trees?
  - this is always more complicated than it looks
WHY BEHAVIOR TREES SUCK
Search Fragment

Presearch

Search

Uncover

Suppress

Guard

Uncover

Search-sync

Approach

Pick next search point
void s_agent::behavior_update()
{
    if (omap.confusion < k_confusion_threshold)
    {
        s_pos2d pos = omap.get_target_position();
        move_to(pos);
    }
}

Lesson 1: Search is something that happens in our KR, not in our behavior.

Lesson 2: We lean on behavior too much.
Players understand TRIGGERS, not TRANSITIONS

- i.e. when the brute’s armor depletes, he charges, no matter WHAT he was doing at the time

Players understand priorities

Players understand discrete events

Players understand task-decomposition

→ The Tree is representative of the mapping we HOPE exists in the player’s head
AI in Halo 2 was known to be fun to play against and to make smart decisions, like flanking the player. Like in most games, this AI is fully scripted and contains no 'real' intelligence.
Thanks!

DAMIAN ISLA

naimad@moonshotgames.com

@DAMIAN_ISLA

HTTP://NAIMADGAMES.COM