A.I. in Game Development

Catalin Zima-Zegreanu
Game Designer
Academia vs Practice

- Academic research is always a few steps ahead

- Games use techniques which are known in academia for a while, but can’t be integrated for various reasons
  - Resources, Feasibility, Game Design

- Gradually new techniques get adopted
  - And games using new techniques get lots of recognition
Using AI Techniques in games

- Maze Generation
- Navigation
- Goal Selection
- Tactical AI
- Evolution of Intelligent Agents
Maze Generation

- Why?
  - Procedural Generated Content
  - Randomized level each time you play
  - Infinite replayability

- How?
  - Randomized Depth First Search
The Algorithm

- Start with a large grid of cells, each cell has 4 walls.
- Select starting cell

Repeat

- Select random unvisited neighbor
- Remove the ‘wall’ between them
  - Recurse with the new cell
- If no unvisited neighbors remain, return

The maze is generated
Maze Generation
Maze Generation - in Minecraft
Navigation
Navigation & Pathfinding

How do creatures move in the world?
  - Unit movement in Strategy games
  - Enemy movement in First Person Shooters
  - Any kind of entity moving in almost any kind of game

A* most popular pathfinding algorithm
  - Represent the map of the world as a graph
  - For now, assume we have a grid

Case Study: Starcraft
A* Pathfinding
A* Pathfinding

INTERACTIVE DEMO
(if internet works)

http://www.redblobgames.com/pathfinding/a-star/introduction.html
A* Pathfinding

“*But wait, grids? That’s so ‘80s!!!*”
Extending A*

- Navigation Meshes
- Hierarchical A*
A* Navigation Meshes
A* Navigation Meshes
A* Navigation Meshes
A* Navigation Meshes
A* Navigation Meshes
Hierarchical A* - Starcraft
Hierarchical A* - Starcraft

- 1,048,576 pathfinding cells
  - Each cell is 8 pixels wide
- 200 units per player
- 4 players
- LOTS of time and computations
Hierarchical A* - Starcraft

Solution

- Split region in big cells
- Compute adjacency data
- Run A* on big cells

For individual units
- Run A* only inside the small cell
Starcraft Pathfinding Video
Bonus: Starcraft Anecdote

- We don’t always want perfect A.I.
- Harvesters using AI were getting stuck in each other

- Solution: Temporary disable pathfinding and collisions
Bonus: A* vs BFS

- Most advanced algorithm is not always the best

- Tower Defense Games
  - Multiple enemies are moving around an environment towards your base
  - They must avoid turrets and defenses you’ve built

- Simple solution: Apply A* on all units
  - For each unit, search a path towards player’s base
  - Expensive, inefficient

- Elegant Solution: Breadth First Search
Action Selection
Action Selection

- How does an entity decide what to do next?
  - Any simulation game
- Case Study: The Sims
  - Autonomous agents simulating humans
  - Lots of objects to interact with in the environment
The Sims
The Sims

- A sim is modeled after real humans
  - Trait
  - Needs
  - Emotions
- A sim must do various activities
  - Eat, sleep, cook, work, learn, watch boring AI presentation, play
- A sim must interact with lots of objects and people
  - Bed, fridge, stove, car, books
  - Thief, policeman, teacher, mother, boyfriend, children
- How does a sim, left on his own decide what to do?
  - Greedy Selection
The Data

The Sim

- **Traits**
  - shy, playful, serious, romantic

- **Needs**
  - hunger, hygiene, social, fun
  - vary in time based on what happens

- **Others**
  - Skill Level, Emotion, etc

Objects

- **Actions**
- **Needs Satisfied**
- **Traits that influence**
Smart Objects

- An object contains all information about it
  - what interactions are available
  - What needs they can satisfy
  - What animations to use

- Example: A Toilet
  - Use Toilet: +20 hygiene, +20 comfort,
  - Clean Toiler: +20 environment, -30 fun, -10 social
Making a decision

- Look at all the objects around, and analyse
  - The **actions** available on each object.
  - The **reward** provided by each action.
  - The **distance** to the object.
  - The **level of need** felt by the actor at the time.

- **Greedy**
  - order actions by the value they bring
  - from top N actions, choose one randomly
Example: “Happyscape”
Example: “Happyscape”

Eat from Fridge
  + 30 hunger

Play Pinball
  + 40 fun, -10 social
  +10 if ‘playful’

Play Chess
  +30 fun
  +30 if ‘serious’

Talk with Ana
  + 30 social, +10 fun
  +5 if ‘romantic’

Clean Fridge
  +10 hygiene, -40 fun, -20 social
  +50 if ‘clean obsession’
  - 40 if ‘lazy’

Johnny
  playful, romantic
  hunger: 90
  social: 60
  boredom: 50
  hygiene: 10
Example: “Happyscape”

Value of an action:

$$\text{sum(\text{benefit} \times \text{level of need})}$$

Johnny

- playful, romantic
- hunger: 90
- social: 60
- boredom: 50
- hygiene: 10
Example: “Happyscape”

Eat from Fridge = 2700
  + 30 hunger * 90

Play Pinball
  + 40 fun, -10 social
  +10 if ‘playful’

Play Chess
  +30 fun
  +30 if ‘serious’

Talk with Ana
  + 30 social, +10 fun
  +5 if ‘romantic’

Clean Fridge
  +10 hygiene, -40 fun, -20 social
  +50 if ‘clean obsession’
  - 40 if ‘lazy’

Johnny
  playful, romantic
  hunger: 90
  social: 60
  boredom: 50
  hygiene: 10
Example: “Happyscape”

Eat from Fridge
  + 30 hunger

Play Pinball = 1900
  + 40 fun * 50, -10 social * 60
  +10 if ‘playful’ * 50

Play Chess
  +30 fun
  +30 if ‘serious’

Talk with Ana
  + 30 social, +10 fun
  +5 if ‘romantic’

Clean Fridge
  +10 hygiene, -40 fun, -20 social
  +50 if ‘clean obsession’
  - 40 if ‘lazy’

Johnny
  playful, romantic
  hunger: 90
  social: 60
  boredom: 50
  hygiene: 10
Example: “Happyscape”

Eat from Fridge
   + 30 hunger

Play Pinball
   + 40 fun, -10 social
   +10 if ‘playful’

Play Chess = 1500
   +30 fun * 50
   +30 if ‘serious’

Talk with Ana
   + 30 social, +10 fun
   +5 if ‘romantic’

Clean Fridge
   +10 hygiene, -40 fun, -20 social
   +50 if ‘clean obsession’
   - 40 if ‘lazy’

Johnny
   playful, romantic
   hunger: 90
   social: 60
   boredom: 50
   hygiene: 10
**Example: “Happyscape”**

Eat from Fridge
- +30 hunger

Play Pinball
- +40 fun, -10 social
- +10 if ‘playful’

Play Chess
- +30 fun
- +30 if ‘serious’
- +30 if ‘playful’

Talk with Ana = 2600
- +30 social * 60, +10 fun * 50
- +5 if ‘romantic’ * 60

Clean Fridge
- +10 hygiene, -40 fun, -20 social
- +50 if ‘clean obsession’
- -40 if ‘lazy’

Johnny
- playful, romantic
- hunger: 90
- social: 60
- boredom: 50
- hygiene: 10
Example: “Happyscape”

Eat from Fridge
   + 30 hunger

Play Pinball
   + 40 fun, -10 social
   +10 if ‘playful’

Play Chess
   +30 fun
   +30 if ‘serious’

Talk with Ana
   + 30 social, +10 fun
   +5 if ‘romantic’

Clean Fridge = -3100
   +10 hygiene x 10, -40 fun x 50, -20 social x 60
   +50 if ‘clean obsession’
   - 40 if ‘lazy’

Johnny
   playful, romantic
   hunger: 90
   social: 60
   boredom: 50
   hygiene: 10
Example: “Happyscape”

Value of an action:
\[ \text{sum(benefit} \times \text{level of need)} \]

Eat from Fridge  \[ 30 \times 90 = 2700 \]
Play Pinball 60  \[ (40+10) \times 50 - 10 \times 60 = 1900 \]
Play Chess  \[ 30 \times 50 = 1500 \]
Talk with Ana \[ (30+5) \times 60 + 10 \times 50 = 2600 \]
Clean Fridge 60  \[ 10 \times 10 - 40 \times 50 - 20 \times 60 = -3100 \]

Johnny
playful, romantic
hunger: 90
social: 60
boredom: 50
hygiene: 10
Example: “Happyscape”

Value of an action:

\[
\text{sum}(\text{benefit} \times \text{level of need})
\]

<table>
<thead>
<tr>
<th>Action</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eat from Fridge</td>
<td>2700</td>
</tr>
<tr>
<td>Talk with Ana</td>
<td>2600</td>
</tr>
<tr>
<td>Play Pinball</td>
<td>1900</td>
</tr>
<tr>
<td>Play Chess</td>
<td>1500</td>
</tr>
<tr>
<td>Clean Fridge</td>
<td>-3100</td>
</tr>
</tbody>
</table>

Johnny

playful, romantic
hunger: 90
social: 60
boredom: 50
hygiene: 10
Example: “Happyscape”
“Happyscape”
“Happyscape”
Tactical AI
Tactical AI

- What’s the best way to compete with an enemy?
  - Strategy games, RPGs, competitive games

- Case Study: Total War
  - Strategy Game knows for great AI
Total War
Video of a Battle
Total War Series

A.I. happens at different levels

- Unit Level
- Squad / Battle Level
- Strategy Level
Unit Level
Unit Level

- Manages behavior of individual unit
- What to do? Who to attack? Where to go?
- Simplified ‘neural network’
- A mini-network for each action
  - attack, defend, flee
  - Output is ‘Yes’ or ‘No’ based on several inputs
Total War

Enemy type
Distance to enemy
Weapon range
Number of enemies
Number of allies

Should I attack?
Yes
No

Should I flee?
Yes
No
Squad Level
Total War

Squad Level

- Tactics and Reactions
- If-Then rule-based system
  - Rules taken out of *Sun Tzu’s The Art of War*
    - If you outnumber the enemy 10:1 then split and surround them.
    - If you outnumber the enemy 5:1 then attack them directly.
    - If you outnumber the enemy 2:1 then divide them up.

- Group Pathfinding and Formations
  - A* used for squads
  - Individual units don’t search for paths, but stay at a certain offset from squad centre
  - Individual logic only for local avoidance
Total War

Emergent Behavior

- Rules at individual and squad levels interact and lead to patterns
- Example: Incoming Cavalry attack a squad of Archers
  - Individuals neural networks tell it to scared and flee (edge troops)
  - Squad size decreases -> Squad Logic must react
  - Squad might get split in two based on rules reasoning
- Whole armies can be split and scared by weaker opponents that flank them
Strategy Level
Total War

Strategy Level

- Turn based top-level strategy
- Genetic Algorithms used to ‘evolve’ the virtual strategists
- Several temperaments and skills modelled based on a set of traits
- Variety in strategy used by opponents
- Evolve over time
  - ‘Natural Selection’ also results from interacting with the Player
Total War - Genetic Algorithm

Chromosome
composed of personality traits of a strategist

<table>
<thead>
<tr>
<th>Aggressive</th>
<th>Anger</th>
<th>Ambush</th>
<th>Attacker</th>
<th>Risky Attacker</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defender</td>
<td>Diplomat</td>
<td>Builder</td>
<td>Engineering</td>
<td>Espionage</td>
</tr>
<tr>
<td>Taxman</td>
<td>Trader</td>
<td>Corrupt</td>
<td>Justice</td>
<td>etc.</td>
</tr>
</tbody>
</table>

Mutations
New generation of strategists derived from previous successful ones, with small changes to the personalities

Fitness Function
Success in battle / The territories acquired / richness of the territories / not being killed by the human player
Bonus: Indirect Control

- Players never control individual units, just indirect control through orders
  - Unit emotions may override orders
  - Challenges players to understand how units function
  - Shows off the A.I.
- Makes opponent A.I. seem more intelligent, since you’re both using the same tools
The evolution of intelligent agents
Intelligent Agents

- In-game entities that oppose the player
- Must move and make decisions in real-time
- Navigate environment, attack player, hide, take cover, cooperate with other agents
- Must be smart enough to pose a challenge
- Must be dumb enough to be fun

- Arcade, Action, First Person Shooters, etc
PAC-MAN (1980)
PAC-MAN (1980)

- Very simple **State Machine**
  - Scatter, Chase, Frightened, Respawn
- Each ghost has specific behavior rules
  - Red: Aggressive, target PC tile, variable speed
  - Pink: target tile in front of PC, can be tricked
  - Blue: more complex target tile, unpredictable
  - Orange: aggressive if PC is close, otherwise wandering

- Simple Rules lead to complex behaviors
- Clear association between color and behavior -> strategic play
- No randomness -> predictable -> exploitable
Thief (1998)
Thief (1998)

- Was considered revolutionary
- Guard AI was a State Machine, focused on SENSORS

- They See, Hear, and even Smell (as ‘fake sounds’)
  - Vision sensors depend on light, size, distance
  - Sound sensors depend on intensity, distance, walls
  - Context Sensitive Alertness
Thief State Machine Example

- **Patrol**
  - X seconds pass without detecting anything
  - See Player
  - Hear Sound
  - See dead body
  - Smell scent

- **Chase**
  - In Range
  - Player Disappeared

- **Investigate**

- **Attack**
Thief (1998)

- Need to be kept a bit ‘unintelligent’, to avoid frustration
  - Short memory
  - Speaking to themselves
  - Predictable + small random variations
  - => Easier to fool (which is fun)

- Still a relatively simple FSM, but addition of sensors was a huge evolution
F.E.A.R

- Milestone in enemy A.I.
- Agents make plans to attack and subdue the player
- Agents use the environment
- Agents collaborate between them
Agents make plans to attack and subdue the player

- First use of a Task Planner in games
  - Combination between STRIPS and A* for searching a plan
  - Start with a set of all possible actions
  - Computes a sequence of actions to reach a goal
    - Example: Dodge -> Move Closer -> Take Cover -> Ambush -> Kill Enemy
  - Different enemy types have different action sets
    - soldier, assassin, medic
- Huge improvement over FSM, quickly adopted by most games
F.E.A.R

Agents use the environments

- Environment is filled with Smart Objects
- Smart Objects inform the agent about things he can do
  - Hide, take cover
  - Smash doors, windows
  - Shoot a barrel to make it explode
  - Use a ladder to get to higher position
Agents collaborate between them

- **Dynamic Blackboard** technique
  - used to store **facts about the world**
    - *I saw the Player there*
    - *There's a grenade in the next room*
  - allows sharing of information between agents

- **Squad Tactics**
  - High-level AI logic coordinates multiple individuals
  - Agents communicate and exchange facts about the world
  - Communication is also vocalized for the player to hear
Other notable techniques

- Decision Tree Learning
- Belief-Decision-Intention System
- Partial-Order Planning
- N-Gram statistical Prediction
- Neural Networks
- Behavior Trees
- Machine Learning
Conclusions

- Lots of cool usages for AI
- Even algorithms that seem insignificant have great uses
- Games that implemented new ways to use AI almost always got recognition
  - Black & White, Fighting Games, Outcast, Fable, Left 4 Dead, Bioshock Infinite
Extra: Machine Learning
Deep Learning

- Machine Learning method based on artificial neural networks
- Challenge: Teach a ML machine to play a game as if it were a human
  - No access to game’s code
  - Same knowledge and input as a human
- Challenge: Make it beat other humans
- DeepMind, by Alphabet Inc (Google)
- OpenAI (Elon Musk)
DeepMind: Starcraft 2
Questions?