COMP270: Mathematics for 3D Worlds & Simulations

2: Authored Behaviours I

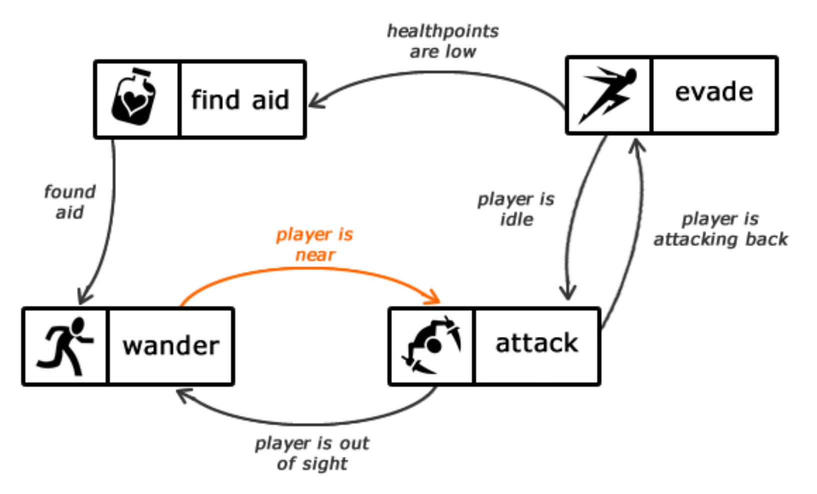
FALMOUTH UNIVERSITY

AI architectures

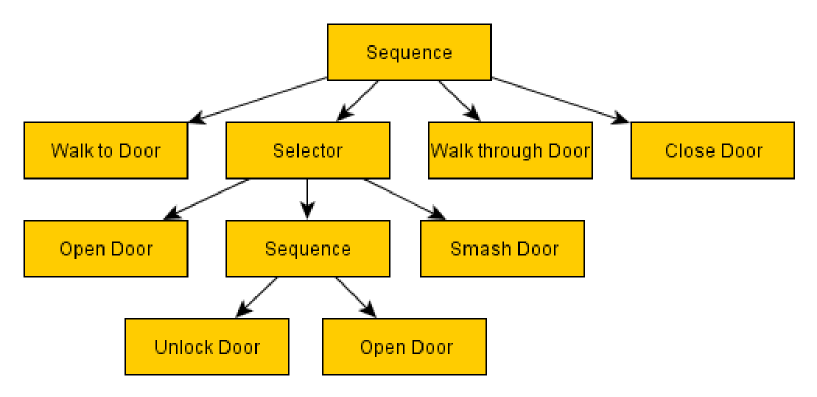
Rule-based AI

Generally implemented as `if` statements or event-based triggers

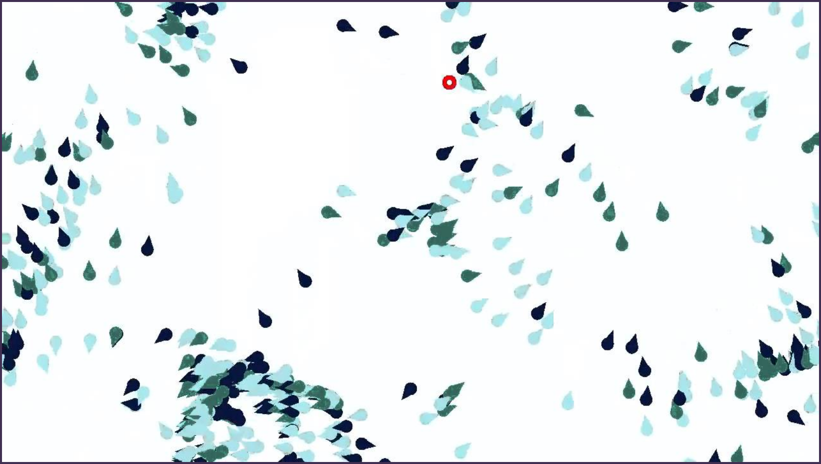
Finite state machines



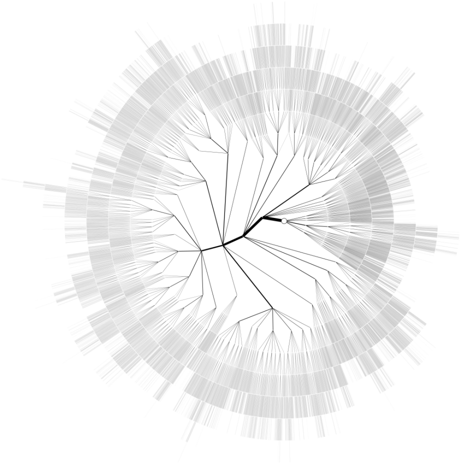
Behaviour trees



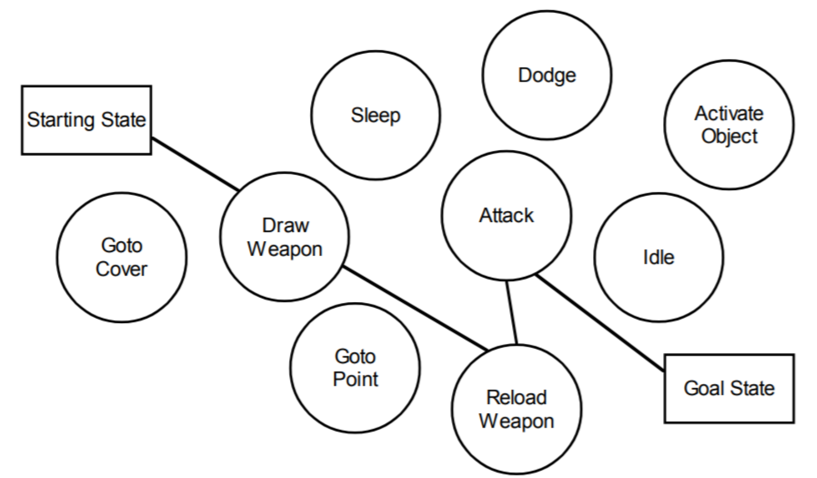
Multi-agent approaches (e.g. flocking)



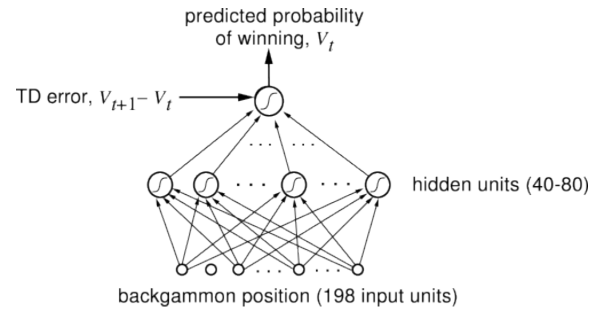
Game tree search



Planning


Starting State
Sleep
Dodge
Activate Object
Goto Cover
Draw Weapon
Attack
Idle
Goto Point
Reload Weapon
Goal State

Machine learning


predicted probability of winning, V_t
TD error, $V_{t+1} - V_t$
hidden units (40-80)
backgammon position (198 input units)

AI architectures

AI architectures

- Can roughly be divided into **hand-authored**...

AI architectures

- Can roughly be divided into **hand-authored**...
  - Rule-based, FSM, behaviour trees

AI architectures

- Can roughly be divided into **hand-authored**...
  - Rule-based, FSM, behaviour trees
- ... and **computational intelligence**

AI architectures

- Can roughly be divided into **hand-authored**...
  - Rule-based, FSM, behaviour trees
- ... and **computational intelligence**
  - Search, planning, machine learning

AI architectures

- Can roughly be divided into **hand-authored**...
  - Rule-based, FSM, behaviour trees
- ... and **computational intelligence**
  - Search, planning, machine learning
- Do you want to **design** the AI behaviours yourself, or do you want them to **emerge** from the system?

AI architectures

- Can roughly be divided into **hand-authored**...
    - Rule-based, FSM, behaviour trees
- ... and **computational intelligence**
    - Search, planning, machine learning
- Do you want to **design** the AI behaviours yourself, or do you want them to **emerge** from the system?
- Predictability and authorial control versus adaptability and novelty

AI architectures

- Can roughly be divided into **hand-authored**...
  - Rule-based, FSM, behaviour trees
- ... and **computational intelligence**
  - Search, planning, machine learning
- Do you want to **design** the AI behaviours yourself, or do you want them to **emerge** from the system?
- Predictability and authorial control versus adaptability and novelty
- Can also combine the two, e.g. use a rule-based system to constrain a CI system

Rule-based AI

Rule-based AI

Rule-based AI

- Generally **reactive** to the state of the world

Rule-based AI

- Generally **reactive** to the state of the world
- Based on **if-then** triggers, basic **calculations**, etc.

Rule-based AI

- Generally **reactive** to the state of the world
- Based on **if-then** triggers, basic **calculations**, etc.
- Generally hand-coded and only modifiable by a programmer

Case study: Ghosts in Pac-Man

Case study: Ghosts in Pac-Man

- Full details: `http://gameinternals.com/understanding-pac-man-ghost-behavior`

Case study: Ghosts in Pac-Man

- Full details: `http://gameinternals.com/understanding-pac-man-ghost-behavior`
- Each ghost has 3 states

Case study: Ghosts in Pac-Man

- Full details: `http://gameinternals.com/understanding-pac-man-ghost-behavior`
- Each ghost has 3 states
  - Chase: head for a specific position (see next slide)

Case study: Ghosts in Pac-Man

- Full details: `http://gameinternals.com/understanding-pac-man-ghost-behavior`
- Each ghost has 3 states
  - Chase: head for a specific position (see next slide)
  - Scatter: head for a specific corner of the level

Case study: Ghosts in Pac-Man

- Full details: `http://gameinternals.com/understanding-pac-man-ghost-behavior`
- Each ghost has 3 states
  - Chase: head for a specific position (see next slide)
  - Scatter: head for a specific corner of the level
  - Frightened: move randomly

Ghost “personalities”

Ghost “personalities”

- Red ghost: aim for Pac-Man

Ghost “personalities”

- Red ghost: aim for Pac-Man
- Pink ghost: aim for 2 spaces ahead of Pac-Man

Ghost “personalities”

- Red ghost: aim for Pac-Man
- Pink ghost: aim for 2 spaces ahead of Pac-Man
- Blue ghost: aim for position on the line between red ghost and 2 spaces ahead of Pac-Man

Ghost “personalities”

- Red ghost: aim for Pac-Man
- Pink ghost: aim for 2 spaces ahead of Pac-Man
- Blue ghost: aim for position on the line between red ghost and 2 spaces ahead of Pac-Man
- Orange ghost: aim for Pac-Man until 8 spaces away, then aim for corner

Ghost movement

Ghost movement

- No pathfinding — greedily move towards target

Ghost movement

- No pathfinding — greedily move towards target
- Can only change direction at an intersection

Ghost movement

- No pathfinding — greedily move towards target
- Can only change direction at an intersection
- Can't reverse or stay still

Ghost movement

- No pathfinding — greedily move towards target
- Can only change direction at an intersection
- Can't reverse or stay still
- Therefore can't get stuck, despite imperfect pathfinding

Ghost behaviour

Ghost behaviour

- Behaviour rules are very simple

Ghost behaviour

- Behaviour rules are very simple
- However, the combination of them leads to interesting gameplay and illusion of personality

Design lessons

Design lessons

- AI doesn't have to be complicated

Design lessons

- AI doesn't have to be complicated
- Simple AI, when interacting with a player and each other, can give engaging results

Design lessons

- AI doesn't have to be complicated
- Simple AI, when interacting with a player and each other, can give engaging results
- Bugs in AI don't always matter...

Finite state machines

Finite state machines

- A **finite state machine (FSM)** consists of:

Finite state machines

- A **finite state machine (FSM)** consists of:
    - A set of **states**; and

Finite state machines

- A **finite state machine (FSM)** consists of:
  - A set of **states**; and
  - **Transitions** between states

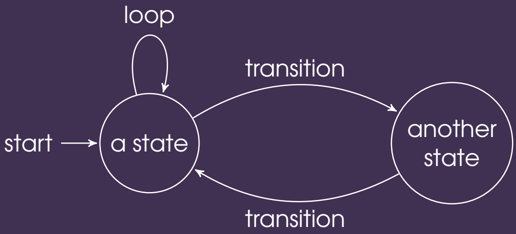
Finite state machines

- A **finite state machine (FSM)** consists of:
  - A set of **states**; and
  - **Transitions** between states
- At any given time, the FSM is in a **single state**

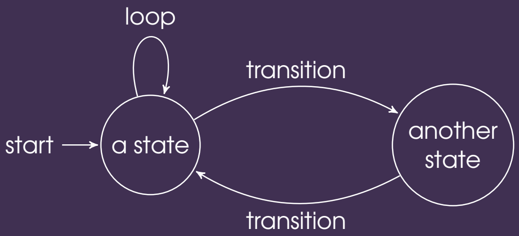
Finite state machines

- A **finite state machine (FSM)** consists of:
  - A set of **states**; and
  - **Transitions** between states
- At any given time, the FSM is in a **single state**
- **Inputs** or **events** can cause the FSM to transition to a different state

State transition diagrams

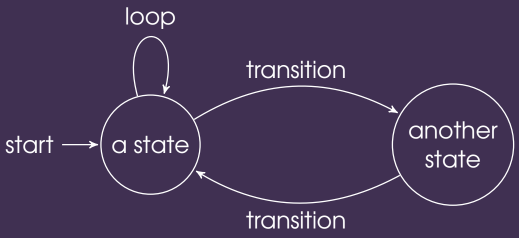


State transition diagrams



- FSMs are often drawn as **state transition diagrams**

State transition diagrams



- FSMs are often drawn as **state transition diagrams**
- Reminiscent of **flowcharts** and certain types of **UML diagram**

FSMs for AI behaviour

The next slide shows a simple FSM for the following AI behaviour, for an enemy NPC in a shooter game:

FSMs for AI behaviour

The next slide shows a simple FSM for the following AI behaviour, for an enemy NPC in a shooter game:

- By default, patrol (e.g. along a preset route)

FSMs for AI behaviour

The next slide shows a simple FSM for the following AI behaviour, for an enemy NPC in a shooter game:

- By default, patrol (e.g. along a preset route)
- If the player is spotted, attack them

FSMs for AI behaviour

The next slide shows a simple FSM for the following AI behaviour, for an enemy NPC in a shooter game:

- By default, patrol (e.g. along a preset route)
- If the player is spotted, attack them
- If the player is no longer visible, resume patrolling

FSMs for AI behaviour

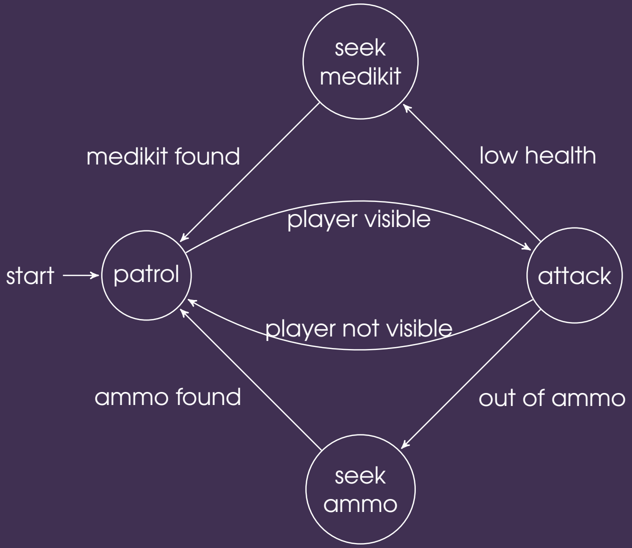
The next slide shows a simple FSM for the following AI behaviour, for an enemy NPC in a shooter game:

- By default, patrol (e.g. along a preset route)
- If the player is spotted, attack them
- If the player is no longer visible, resume patrolling
- If you are low on health, run away and find a medikit. Then resume patrolling

FSMs for AI behaviour

The next slide shows a simple FSM for the following AI behaviour, for an enemy NPC in a shooter game:

- By default, patrol (e.g. along a preset route)
- If the player is spotted, attack them
- If the player is no longer visible, resume patrolling
- If you are low on health, run away and find a medikit. Then resume patrolling
- If you are low on ammo, run away and find ammo. Then resume patrolling


seek medikit
medikit found
low health
player visible
start
patrol
attack
player not visible
ammo found
out of ammo
seek ammo

Other uses of FSMs

As well as AI behaviours, FSMs may also be used for:

Other uses of FSMs

As well as AI behaviours, FSMs may also be used for:

- Animation

Other uses of FSMs

As well as AI behaviours, FSMs may also be used for:

- Animation
- UI menu systems

Other uses of FSMs

As well as AI behaviours, FSMs may also be used for:

- Animation
- UI menu systems
- Dialogue trees

Other uses of FSMs

As well as AI behaviours, FSMs may also be used for:

- Animation
- UI menu systems
- Dialogue trees
- Token parsing

Other uses of FSMs

As well as AI behaviours, FSMs may also be used for:

- Animation
- UI menu systems
- Dialogue trees
- Token parsing
- ...

Beyond FSMs

Some topics for you to research, for when plain old FSMs aren't enough...

Beyond FSMs

Some topics for you to research, for when plain old FSMs aren't enough...

- Hierarchical FSMs
- Nested FSMs
- Stack-based FSMs
- Hierarchical task networks
- ...

FALMOUTH
UNIVERSITY

Behaviour Trees

Behaviour trees (BTs)

Behaviour trees (BTs)

- A **hierarchical** model of decision making

Behaviour trees (BTs)

- A **hierarchical** model of decision making
- Allow **complex behaviours** to be built up from **simple components**

Behaviour trees (BTs)

- A **hierarchical** model of decision making
- Allow **complex behaviours** to be built up from **simple components**
- Allow for **more complex** behaviours than FSMs

Behaviour trees (BTs)

- A **hierarchical** model of decision making
- Allow **complex behaviours** to be built up from **simple components**
- Allow for **more complex** behaviours than FSMs
- First used in Halo 2 (2005), now used extensively

Behaviour trees (BTs)

- A **hierarchical** model of decision making
- Allow **complex behaviours** to be built up from **simple components**
- Allow for **more complex** behaviours than FSMs
- First used in Halo 2 (2005), now used extensively
- Also used in robotics and other non-game AI applications

Using BTs

Using BTs

- Fairly easy to implement; plenty of resources online

Using BTs

- Fairly easy to implement; plenty of resources online
- **Unreal**: an advanced BT system is built in

Using BTs

- Fairly easy to implement; plenty of resources online
- **Unreal**: an advanced BT system is built in
- **Unity**: numerous free and paid options on the Asset Store e.g. Behavior Machine, Behavior Designer, Behave, RAIN

BT basics

BT basics

- A BT is a **tree** of **nodes**

BT basics

- A BT is a **tree** of **nodes**
- On each game update (i.e. each frame), the root node is **ticked**

BT basics

- A BT is a **tree** of **nodes**
- On each game update (i.e. each frame), the root node is **ticked**
  - When a node is ticked, it might cause some or all of its **children** to tick as well

BT basics

- A BT is a **tree** of **nodes**
- On each game update (i.e. each frame), the root node is **ticked**
    - When a node is ticked, it might cause some or all of its **children** to tick as well
    - So ticks propagate down the tree from the root

BT basics

- A BT is a **tree** of **nodes**
- On each game update (i.e. each frame), the root node is **ticked**
  - When a node is ticked, it might cause some or all of its **children** to tick as well
  - So ticks propagate down the tree from the root
- A ticked node returns one of three **statuses**:

BT basics

- A BT is a **tree** of **nodes**
- On each game update (i.e. each frame), the root node is **ticked**
  - When a node is ticked, it might cause some or all of its **children** to tick as well
  - So ticks propagate down the tree from the root
- A ticked node returns one of three **statuses**:
  - Success

BT basics

- A BT is a **tree** of **nodes**
- On each game update (i.e. each frame), the root node is **ticked**
  - When a node is ticked, it might cause some or all of its **children** to tick as well
  - So ticks propagate down the tree from the root
- A ticked node returns one of three **statuses**:
  - Success
  - Running

BT basics

- A BT is a **tree** of **nodes**
- On each game update (i.e. each frame), the root node is **ticked**
  - When a node is ticked, it might cause some or all of its **children** to tick as well
  - So ticks propagate down the tree from the root
- A ticked node returns one of three **statuses**:
  - Success
  - Running
  - Failure

BT basics

- A BT is a **tree** of **nodes**
- On each game update (i.e. each frame), the root node is **ticked**
  - When a node is ticked, it might cause some or all of its **children** to tick as well
  - So ticks propagate down the tree from the root
- A ticked node returns one of three **statuses**:
  - Success
  - Running
  - Failure
- “Running” status allows nodes to represent operations that **last multiple frames**

Blackboard

Blackboard

- It is often useful to **share** data between nodes

Blackboard

- It is often useful to **share** data between nodes
- A **blackboard** (sometimes called a **data context**) allows this

Blackboard

- It is often useful to **share** data between nodes
- A **blackboard** (sometimes called a **data context**) allows this
- Blackboard defines **variables**, which can be **read** and **written** by nodes

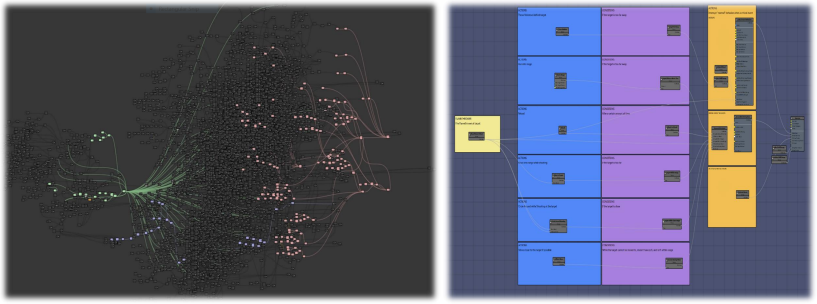
Blackboard

- It is often useful to **share** data between nodes
- A **blackboard** (sometimes called a **data context**) allows this
- Blackboard defines **variables**, which can be read and **written** by nodes
- Blackboard can be **local** to the AI agent, **shared** between several agents, or **global** to all agents

Blackboard

- It is often useful to **share** data between nodes
- A **blackboard** (sometimes called a **data context**) allows this
- Blackboard defines **variables**, which can be read and **written** by nodes
- Blackboard can be **local** to the AI agent, **shared** between several agents, or **global** to all agents
- (Shared blackboards mean that your AI has "telepathy" — this may or may not be desirable!)

BTs in The Division



http://www.gdcvault.com/play/1023382/
AI-Behavior-Editing-and-Debugging