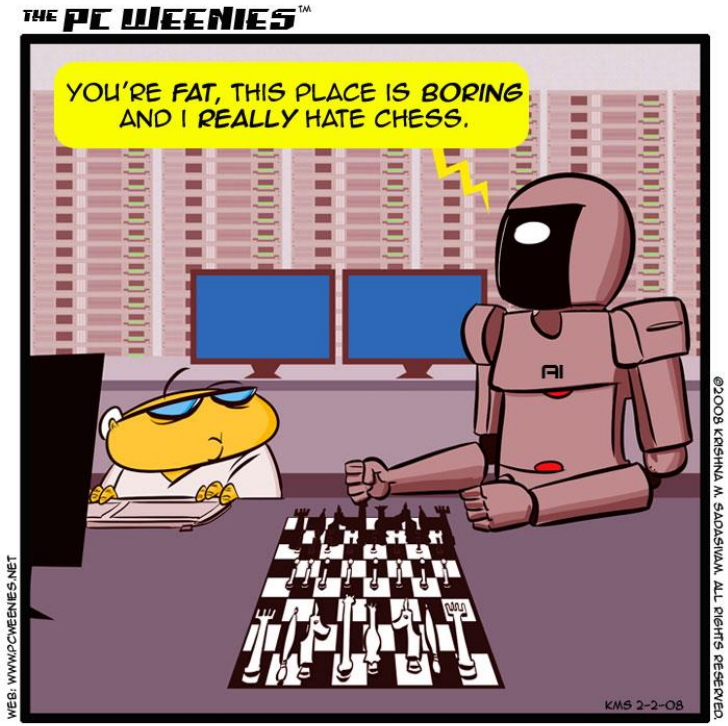
Programming Fundamentals

Lecture 09 – Introduction to Artificial Intelligence

Edirlei Soares de Lima
<edirlei.lima@universidadeeuropeia.pt>

What is Artificial Intelligence?

- Artificial intelligence is about making computers able to perform the **thinking tasks** that humans and animals are capable of.
  - Computers are very good at: arithmetic, sorting, searching, play some board games better than humans, ...
  - Computers are not very good at: recognizing familiar faces, speaking our own language, deciding what to do next, being creative, ...



HOW YOU'LL KNOW WHEN YOU'VE TRULY SUCCEEDED IN THE FIELD OF A.I. RESEARCH.

What is Artificial Intelligence?

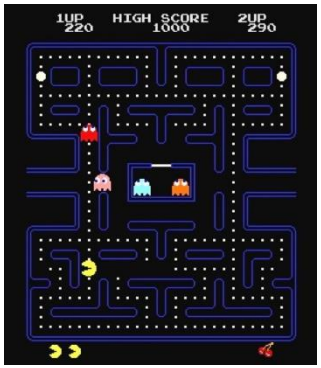
- AI researchers are motivated by:
  - **Philosophy:** understanding the nature of thought and the nature of intelligence and building software to model how thinking might work.
  - **Psychology:** understanding the mechanics of the human brain and mental processes.
  - **Engineering:** building algorithms to perform human-like tasks.

- Academic AI *vs* Game AI:
  - **Academic AI**: solve problems optimally, less emphasis on hardware or time limitations;
  - **Game AI**: entertain player, have to work with limited time and hardware resources.

Complexity Fallacy

- It is a common mistake to think that complex AI equals better character behavior.

- **When simple things look good**: Pac-Man
  - Semi-randomly decisions at junctions;



  - Player comments:
    - "To give the game some tension, some clever AI was programmed into the game. The ghosts would group up, attack the player, then disperse. Each ghost had its own AI."
    - "The four of them are programmed to set a trap, with Blinky leading the player into an ambush where the other three lie in wait."

Complexity Fallacy

- It is a common mistake to think that complex AI equals better character behavior.

- **When complex things look bad**: Black and White [2001]
  - Neural Networks and Decision Trees allowed creatures to learn.
  - When many people first play the game, they often end up inadvertently teaching the creature bad habits, and it ends up being unable to carry out even the most basic actions.

Perception Window

- Most players will only come across some characters and enemies for a short time, which might not be enough for the player to understand the AI.
  - Make sure that a character's AI matches its purpose in the game and the attention it will get from the player.
  - A change in behavior is far more noticeable than the behavior itself.

Illusion of Intelligence

- "If it looks like a fish and smells like a fish, it's probably a fish."
  - if the player believes an agent is intelligent, then it is intelligent.

- For game AI the nature of the human mind is not the key point.
  - The AI characters must look right and demonstrate intelligent behavior.

- Sometimes, simple solutions are enough to create a good illusion of intelligence.
  - Halo [2001] – increasing the number of hit points required to kill enemies made testers thought the AI was very intelligent.

Illusion of Intelligence

- Player's perception of intelligence can be enhanced by providing visual and/or auditory clues about what the agent is "thinking".

- Animation is an excellent way to create a good illusion of intelligence.
  - The Sims [2000] – although it uses a complex emotional model for characters, most part the characters' behaviors is communicated with animations.

  - Triggering animations at the right moment is the key point.

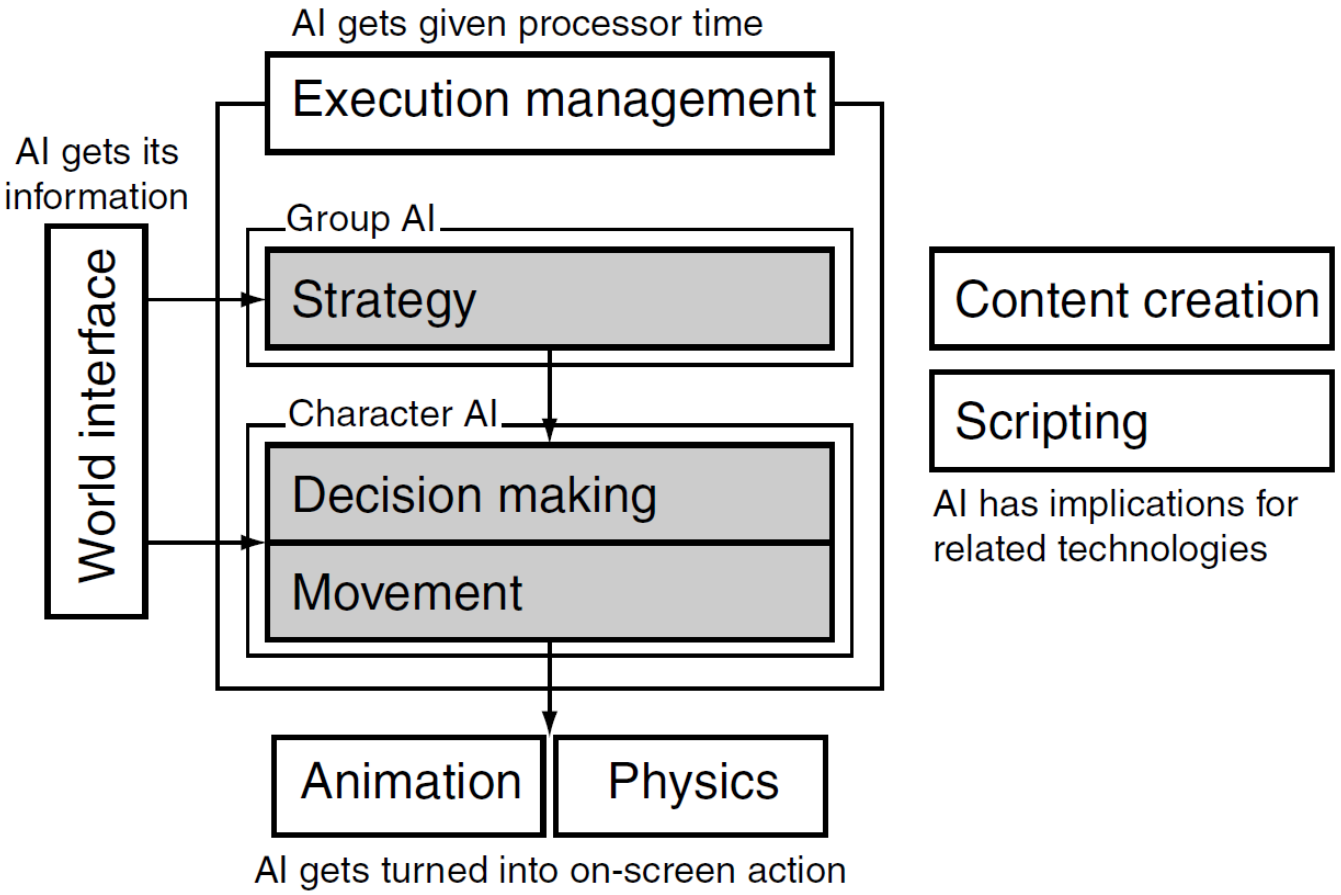
Illusion of Intelligence

- The goal of game developers is to design agents that provide the illusion of intelligence, nothing more.

- Game developers rarely create great new algorithms and then ask themselves, “So what can I do with this?”
  - Instead, they start with a design for a character and apply the most relevant tool to get the result.

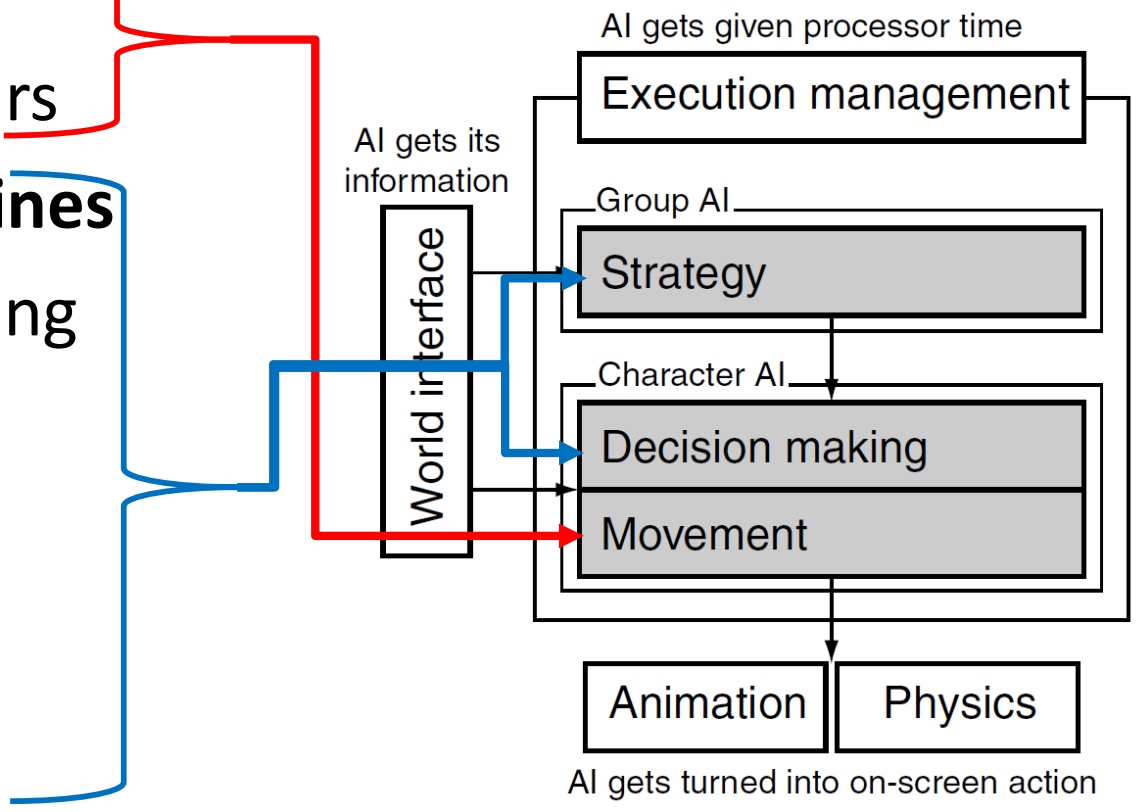
- Be careful to never break the illusion of intelligence:
  - Running into walls, getting stuck in corners, not reacting to obvious stimulus, seeing through walls, hearing a pin drop at 500 meters, …

Game AI – Model



Most Common Techniques

- **Pathfinding**
- Steering behaviours
- **Finite state machines**
- Automated planning
- Behaviour trees
- **Randomness**
- Sensor systems
- Machine learning



Randomness in Games

- Game programmers have a special relationship with random numbers. They can be used for several tasks:
  - Damage calculation;
  - Critical hits probability;
  - Item drop probability;
  - Reward probability;
  - Enemy stats;
  - Spawning enemies and items;
  - Shooting spread zones;
  - Decision making;
  - Procedural content generation;
  - …

```
decision = love.math.random(min, max)
```

Randomness and Probability

- Although most programming languages include functions to generate pseudo-random numbers, there are some situations where some control over the random numbers is extremely important.
  - **Gaussian Randomness**: normal distribution of random numbers.
  - **Filtered Randomness**: manipulation of random numbers so they appear more random to players over short time frames.
  - **Perlin Noise**: consecutive random numbers that are related to each other.

Gaussian Randomness

- Normal distributions (also known as Gaussian distributions) are all around us, hiding in the statistics of everyday life.

Height of Trees

Height of People

Gaussian Randomness

- Normal distributions (also known as Gaussian distributions) are all around us, hiding in the statistics of everyday life.

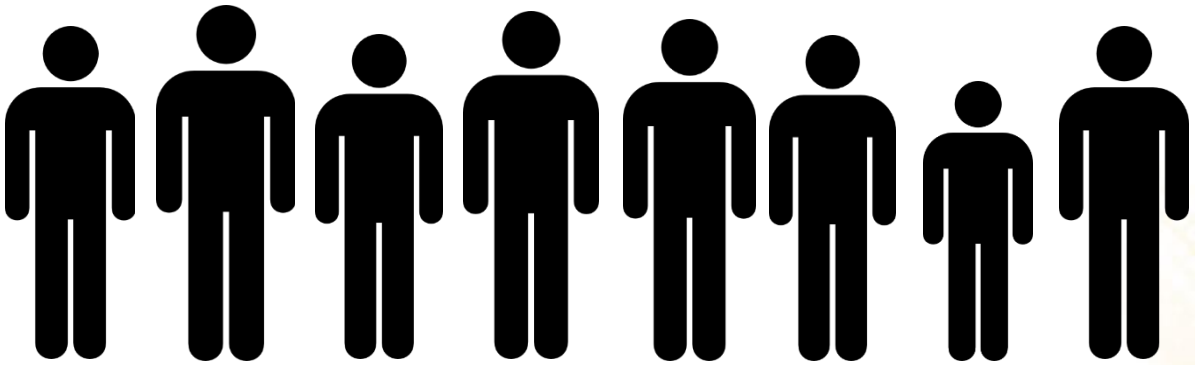
Speed of Runners in a Marathon

Speed of Cars on a Highway

Gaussian Randomness

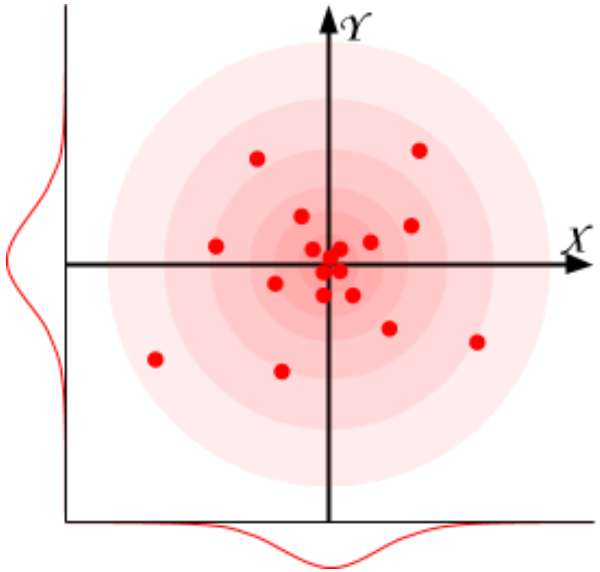
- There is randomness in previous examples, but they are not uniformly random.

- **Example:**
  - The chance of a man growing to be 170 cm tall is not the same as the chance of him growing to a final height of 150 cm tall or 210 cm tall.
  - We see a normal distribution with the height of men centered around 170 cm.

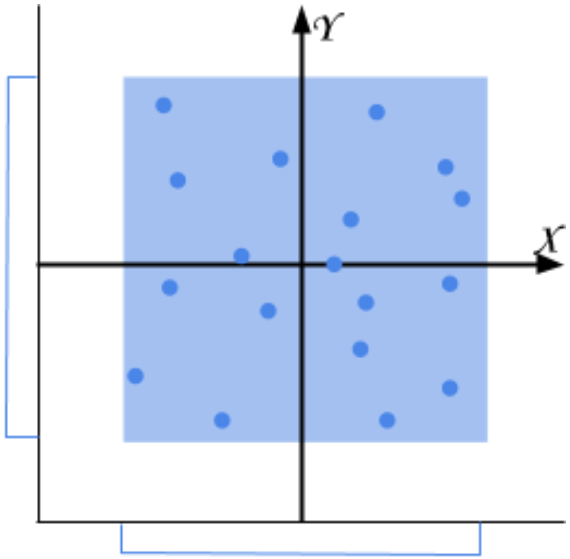


Gaussian Randomness

- Normal Distribution vs. Uniform Distribution:



Normal Distribution

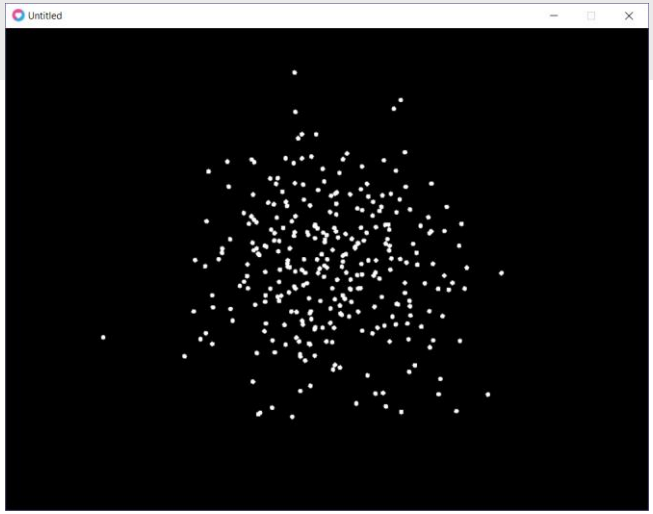


Uniform Distribution

Gaussian Randomness

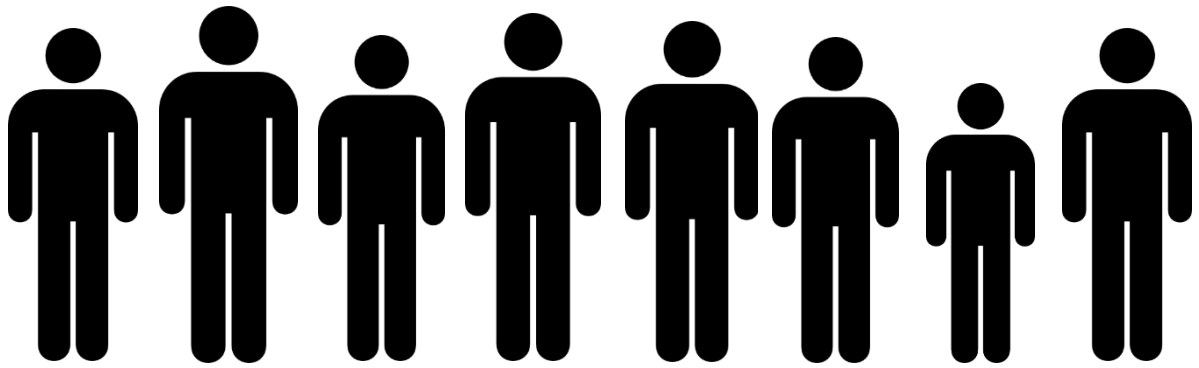
- How Gaussian randomness can be generated?
  - Löve function to generate Gaussian random numbers:

```
function love.draw()
  for x = 0, 300, 1 do
    love.graphics.circle("fill", love.math.randomNormal(80, 400),
                         love.math.randomNormal(80, 300), 3)
  end
end
```



Exercise 1

1) Create a random population of 50 characters whose height follow a normal distribution.
   - You must store the information of the characters in a array.
   - The characters can be visually represented as rectangles.



Finite State Machines

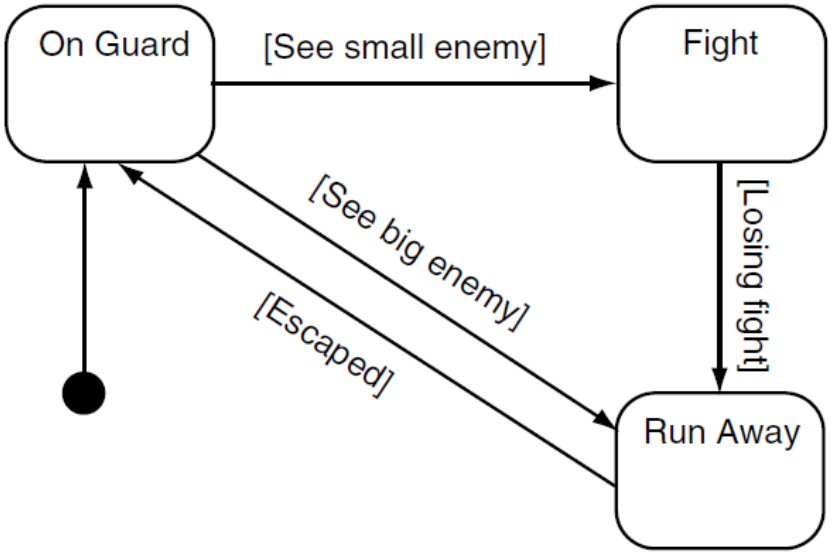
- Usually, game characters have a limited set of possible behaviors. They carry on doing the same thing until some event or influence makes them change.
  - Example: a guard will stand at its post until it notices the player, then it will switch into attack mode, taking cover and firing.

- State machines are the technique most often used for this kind of decision making process in games.

- What is a state machine?

Finite State Machines

- Actions or behaviors are associated with each state.
- Each transition leads from one state to another, and each has a set of associated conditions.
- When the conditions of a transition are met, then the character changes state to the transition's target state.
- Each character is controlled by one state machine and they have a current state.



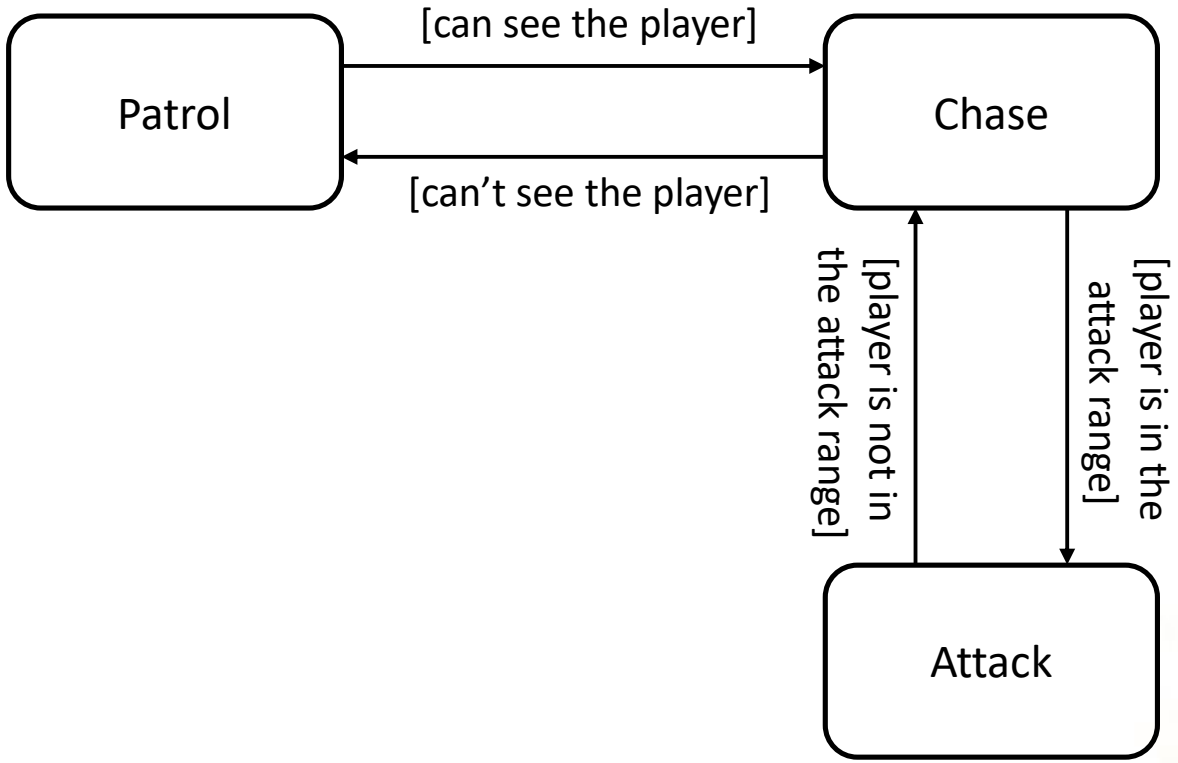
Hard-Coded Finite State Machines

```
local PATROL, DEFEND, SLEEP = 1, 2, 3
local state = PATROL

function UpdateState()
  if state == PATROL then
    if canSeePlayer() then
      state = DEFEND
      if tired() then
        state = SLEEP
      end
    end
  elseif state == DEFEND then
    if not canSeePlayer() then
      state = PATROL
    end
  elseif state == SLEEP then
    if not tired() then
      state = PATROL
    end
  end
end
```

Exercise 2

2) Implement a finite state machine to control an NPC based on the following diagram:



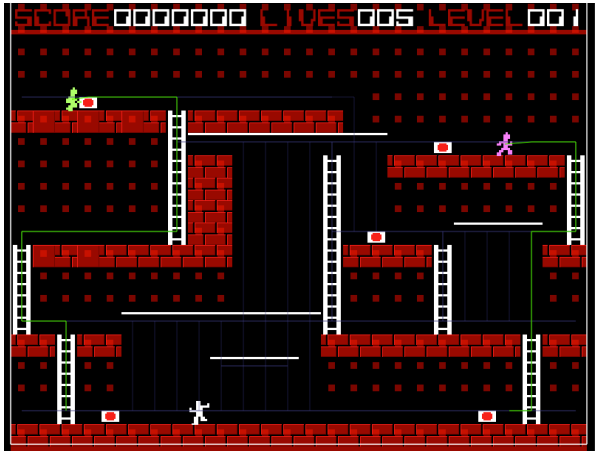
Pathfinding

- Game characters usually need to move around their level.

- While simple movements can be manually defined by game developers (patrol routes or wander regions), more complex movements must be computed during the game.

Pathfinding

- Finding a path seems obvious and natural in real life. But how a computer controlled character can do that?

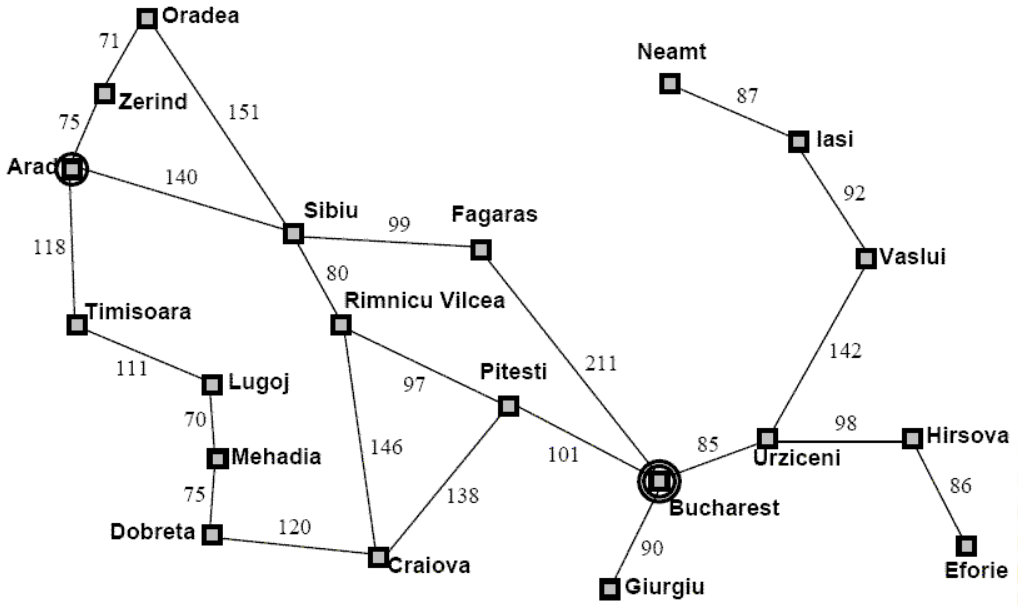


  - The computer needs to find the “best” path and do it in real-time.

Search Problem

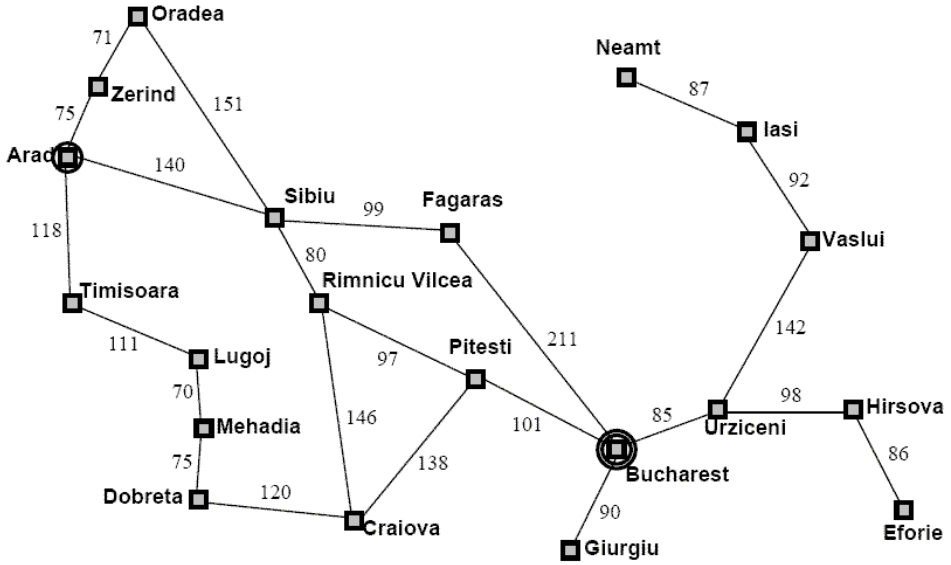
- **Pathfinding is a search problem:** find a sequence of actions from an initial state to an goal state.

- **Problem definition:**
  - Initial state
  - Goal state
  - State space
  - Set of actions
  - Cost functions



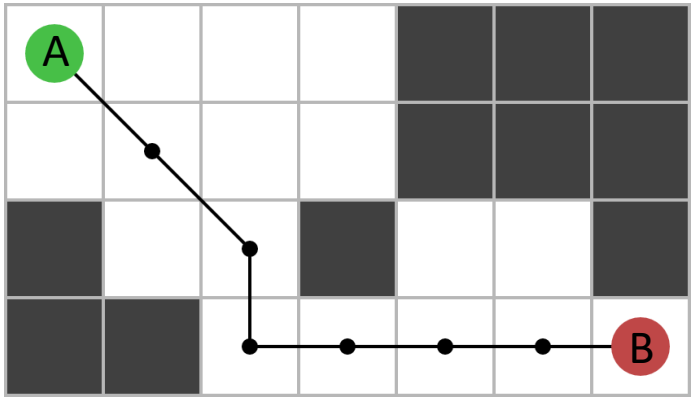
Example of Search Problem

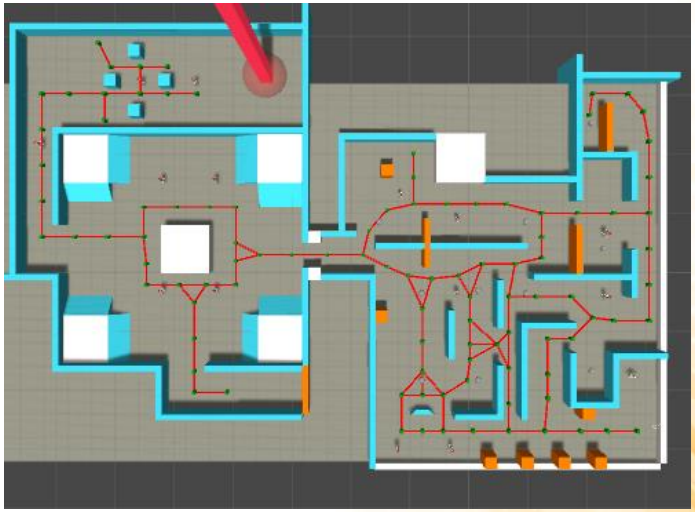
- **Route-finding:**
  - State space: map;
  - Initial state: current city;
  - Goal state: destination city;
  - Set of actions: go from one city to another (only possible if there is a path between the cities);
  - Action cost: distance between the cities;



General Pathfinding Problems

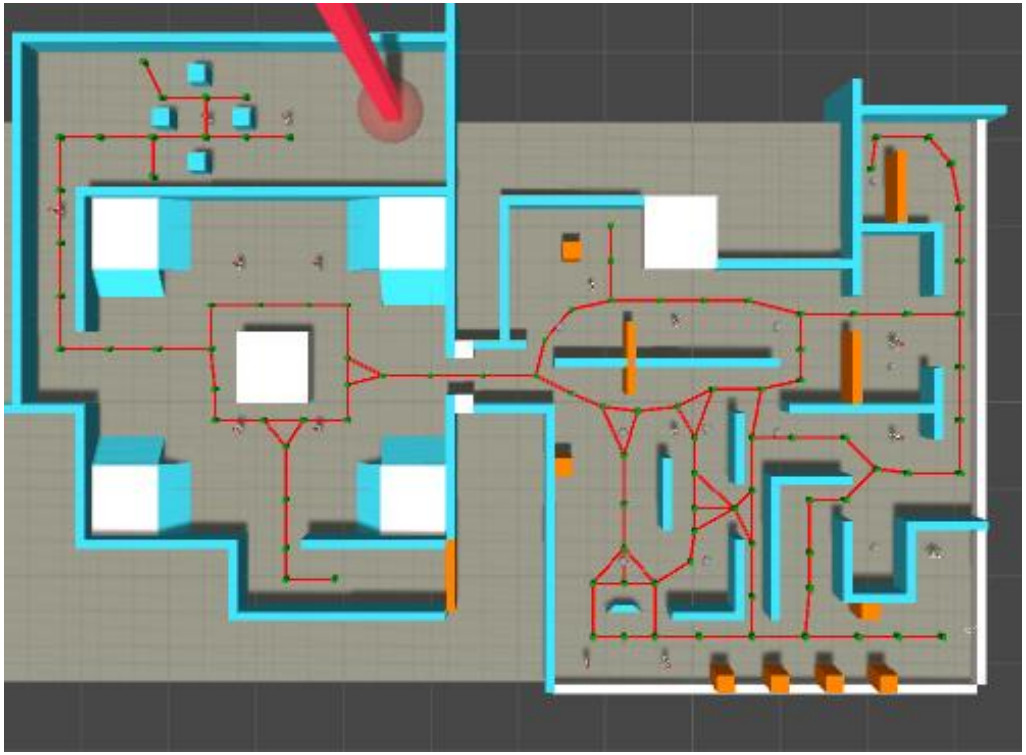
- State space: waypoint graphs or tiled-based maps;
- Initial state: current location (A);
- Goal state: destination location (B);
- Set of actions: movements;
- Action cost: distance or terrain difficulty;





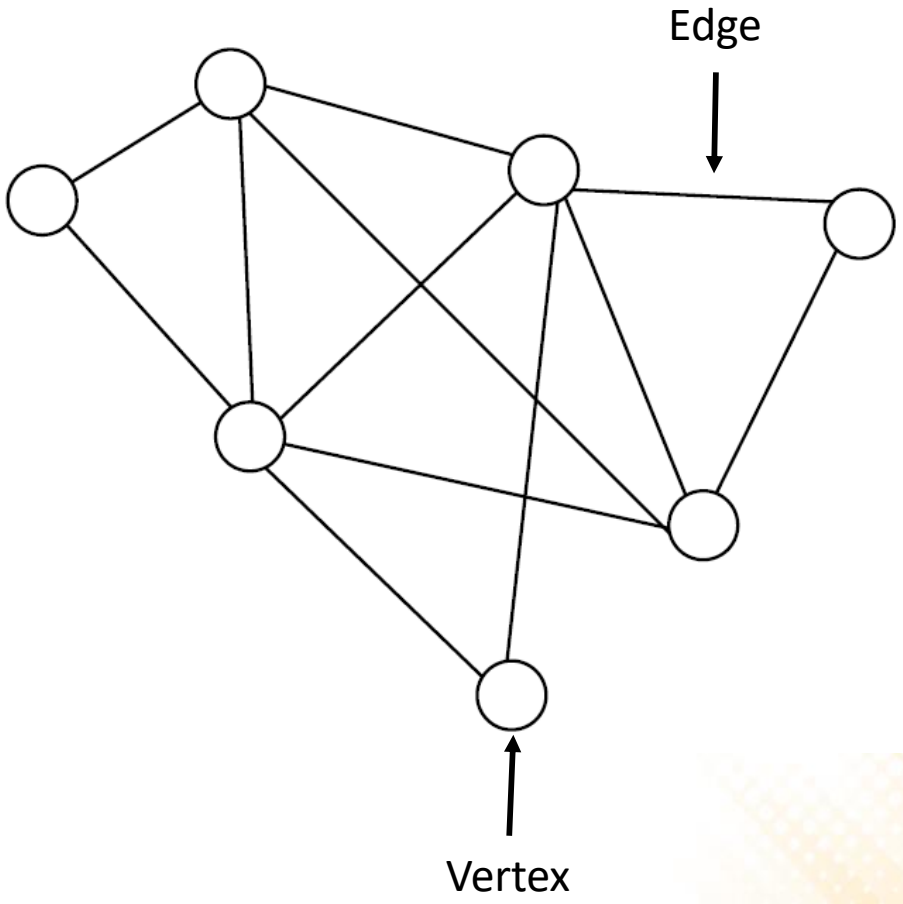
Navigation Graph

- Pathfinding algorithms can’t work directly on the level geometry. They rely on a simplified version of the level, usually represented in the form of a graph.



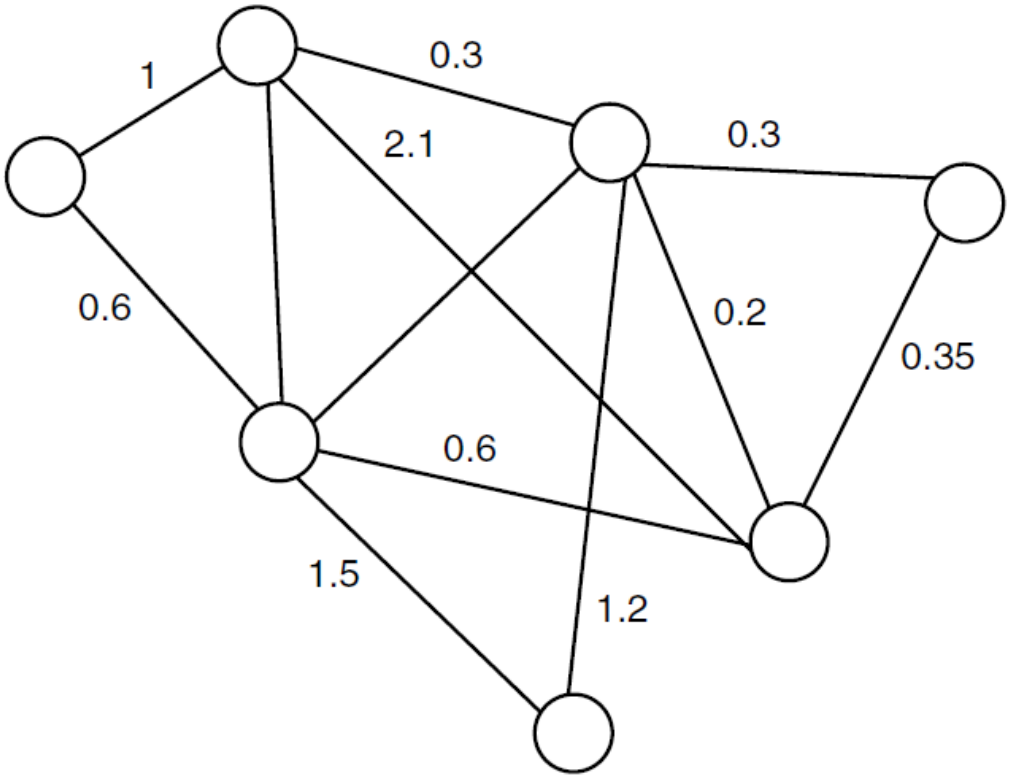
General Graph Structure

- **G = (V, E)**
  - G: graph;
  - V: set of vertices;
  - E: set of edges;



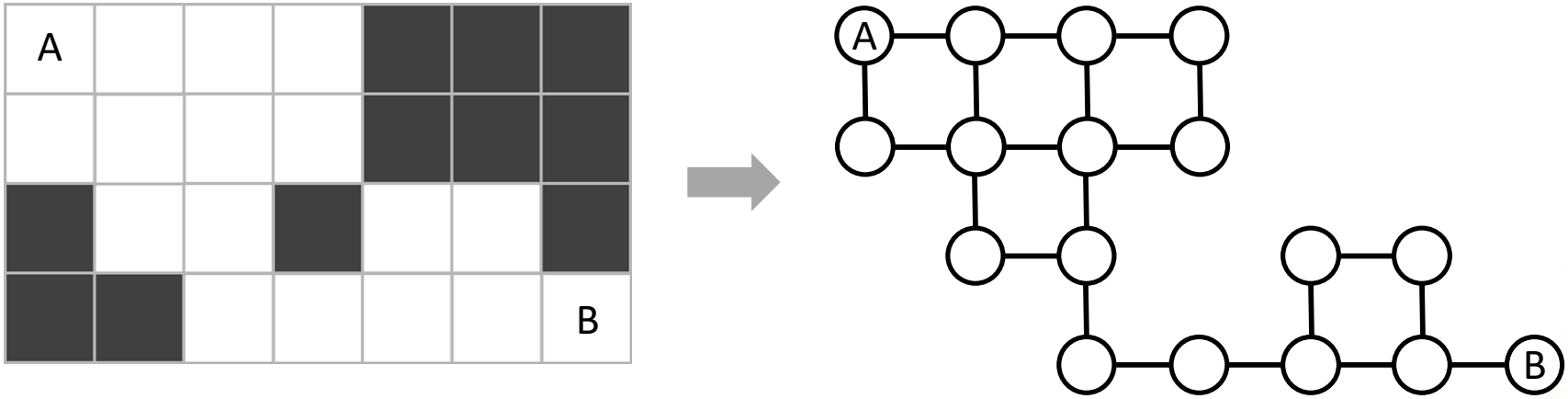
General Graph Structure

- **Weighted graph:** directed on undirected graph in which a number (the weight) is assigned to each edge.



Navigation Graph

- Tiled-based maps can also be seen as graphs:



memory data:

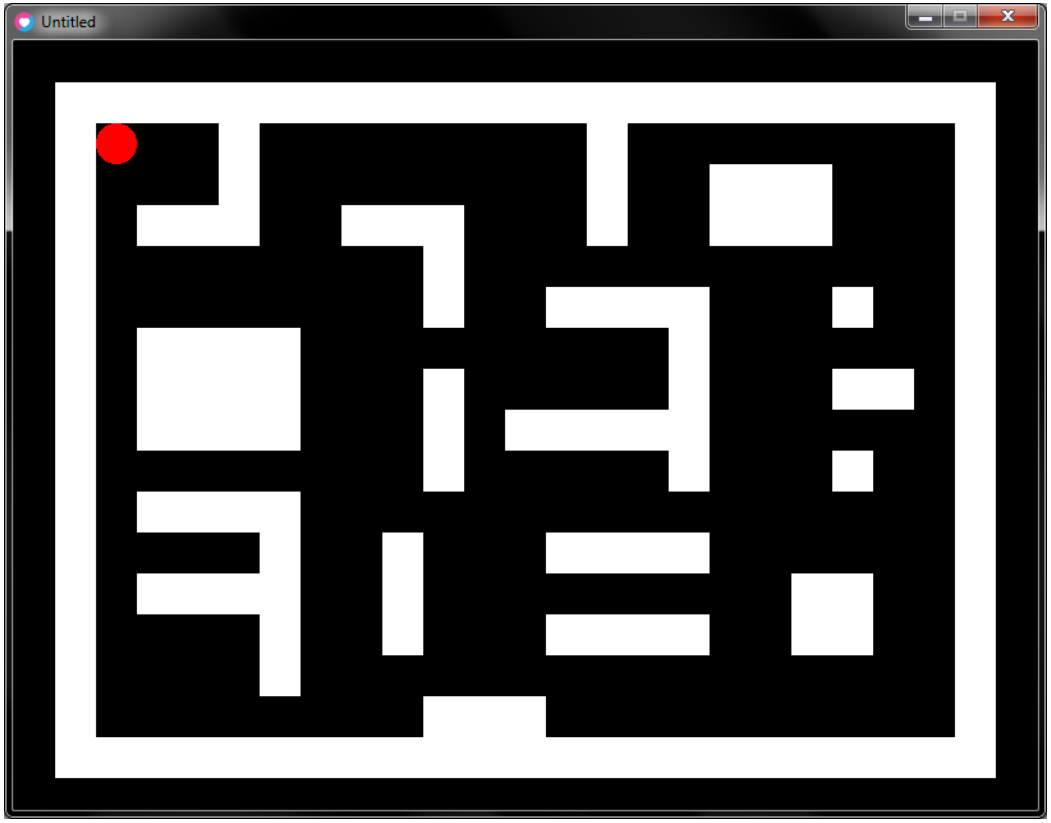
0 0 0 0 1 1 1
0 0 0 0 1 1 1
1 0 0 1 0 0 1
1 1 0 0 0 0 0

Pathfinding

- With the navigation graph in hands, how can we find the best path to go from one point to another?
  - Graph search algorithms!

- There are many graph search algorithms:
  - Breadth-first search (BFS)
  - Depth-first search (DFS)
  - Dijkstra algorithm
  - **A* algorithm**
  - …

Pathfinding / A* Algorithm – Example



http://www.inf.puc-rio.br/~elima/jogos/exemplo_pathfind.zip