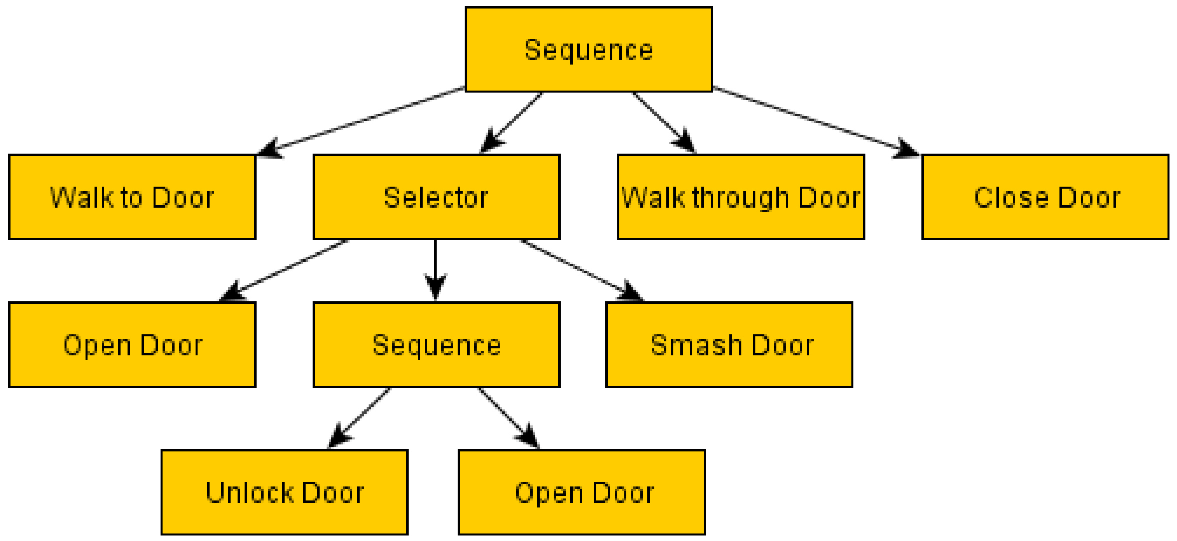
Procedural Content Generation using Behavior Trees (PCGBT)

Anurag Sarkar and **Seth Cooper**

Northeastern University

Motivation

- Behavior trees (BTs) – commonly used technique for modeling NPC/enemy AI behaviors in games
- Enable designing behaviors in modular, reactive manner
    - Modular – simpler behaviors combined into more complex behaviors
    - Reactive – different behaviors can be selected for execution based on runtime conditions
    - Human-readable



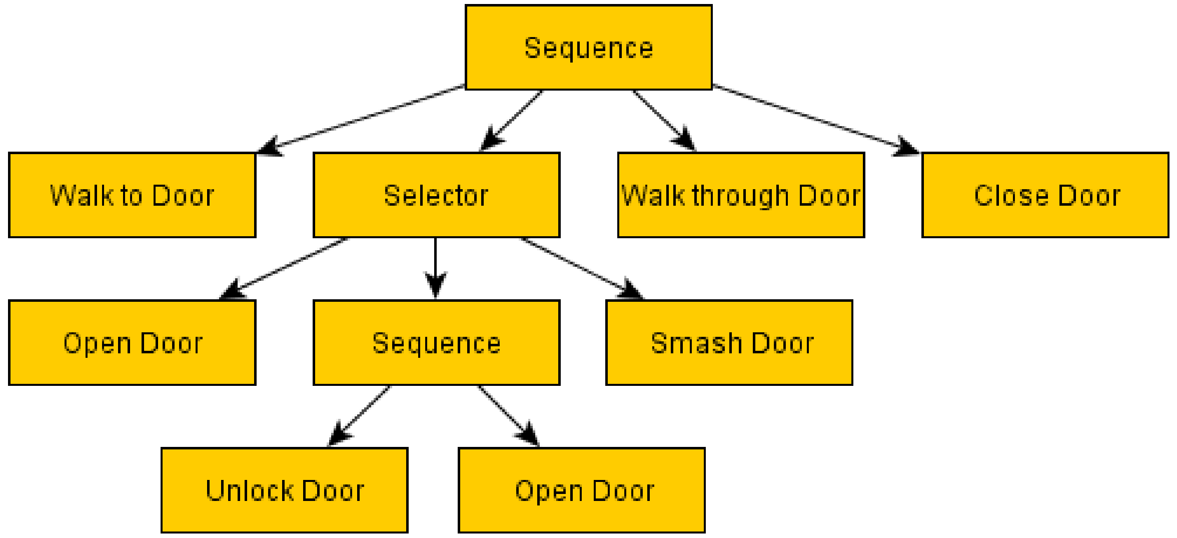
Source: https://www.gamedeveloper.com/programming/behavior-trees-for-ai-how-they-work

Motivation

- Behavior trees (BTs) – commonly used technique for modeling NPC/enemy AI behaviors in games

- Enable designing behaviors in modular, reactive manner
  - Modular – simpler behaviors combined into more complex behaviors
  - Reactive – different behaviors can be selected for execution based on runtime conditions
  - Human-readable

- Desirable qualities for procedural content generators!



Source: https://www.gamedeveloper.com/programming/behavior-trees-for-ai-how-they-work

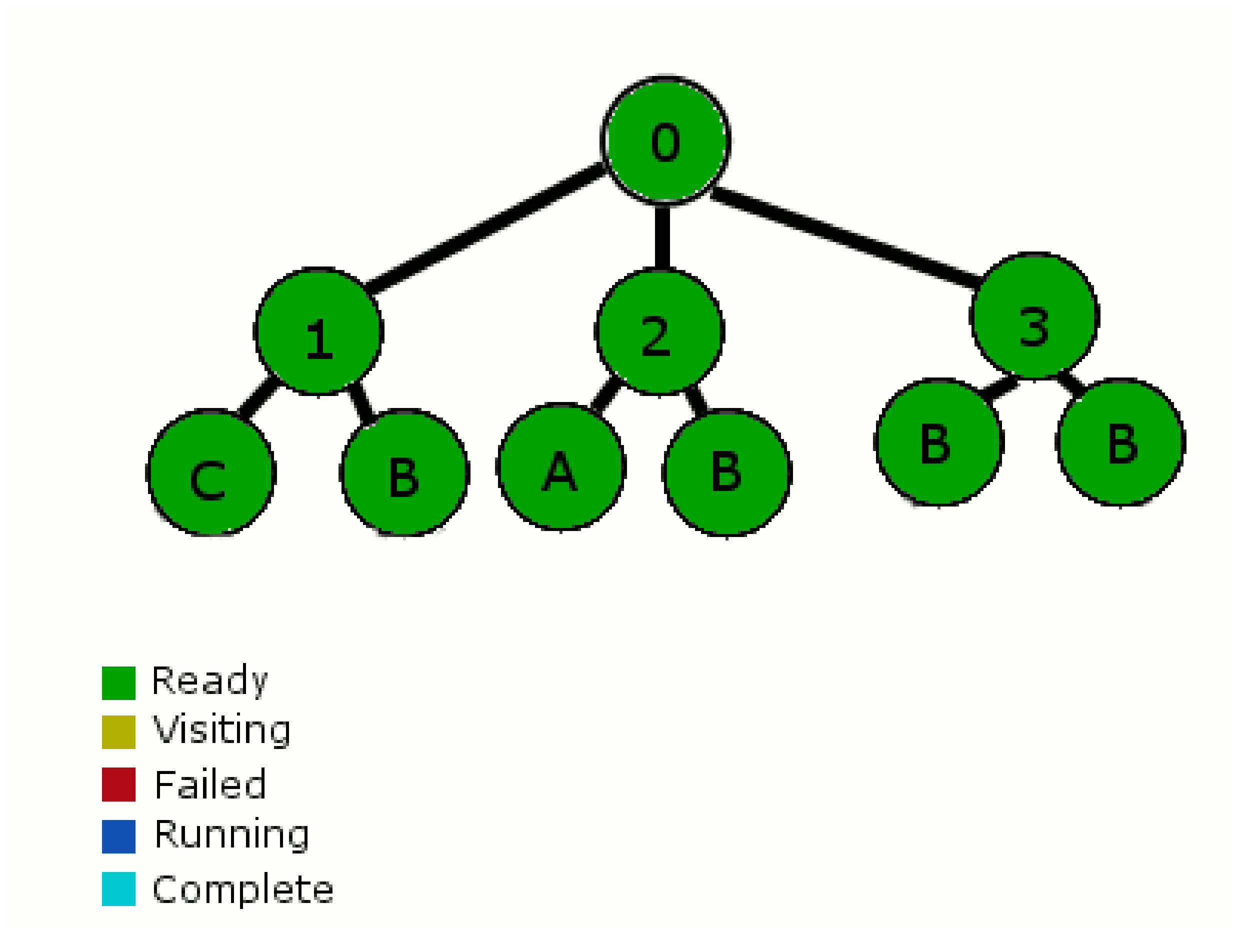
Overview

- Procedural Content Generation using Behavior Trees (PCGBT)
  - Repurpose behavior trees to model game design agents rather than game playing agents
  - NPC Actions (e.g. Find Cover) → Design Actions (e.g. Place Room)
  - Simple behaviors → Sections of levels
  - Complex behaviors → Entire levels

- Applications for level generation
  - Super Mario Bros. / Mega Man
  - Dungeons / Metroid
  - Generic
  - Blending

Behavior Trees

- Directed trees consisting of
    - Root
    - Control flow (internal) nodes
    - Leaf nodes

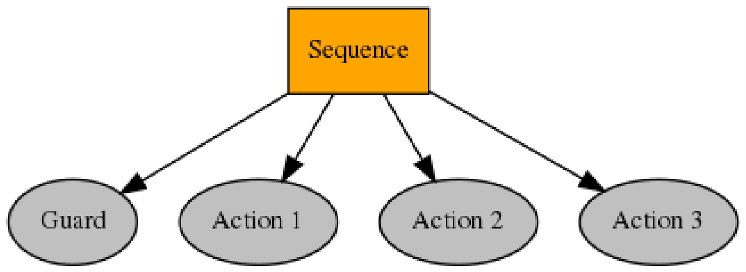


Source: https://gamedev.stackexchange.com/questions/51693/

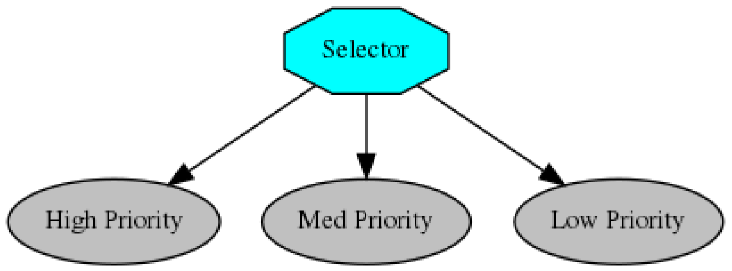
Behavior Trees

- Directed trees consisting of
  - Root
  - Control flow (internal) nodes
  - Leaf nodes

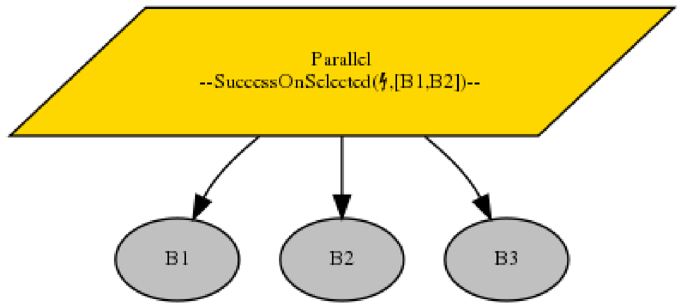
- Control flow nodes
  - Sequence
  - Selector
  - Parallel
  - Decorator



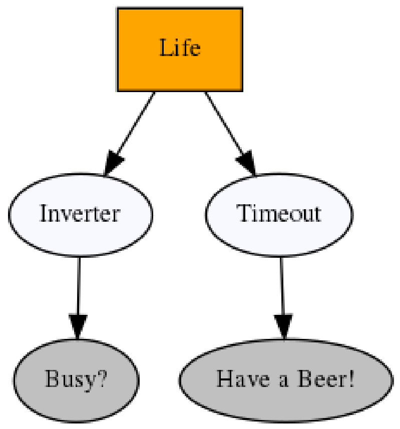
Sequence



Selector



Parallel



Decorator

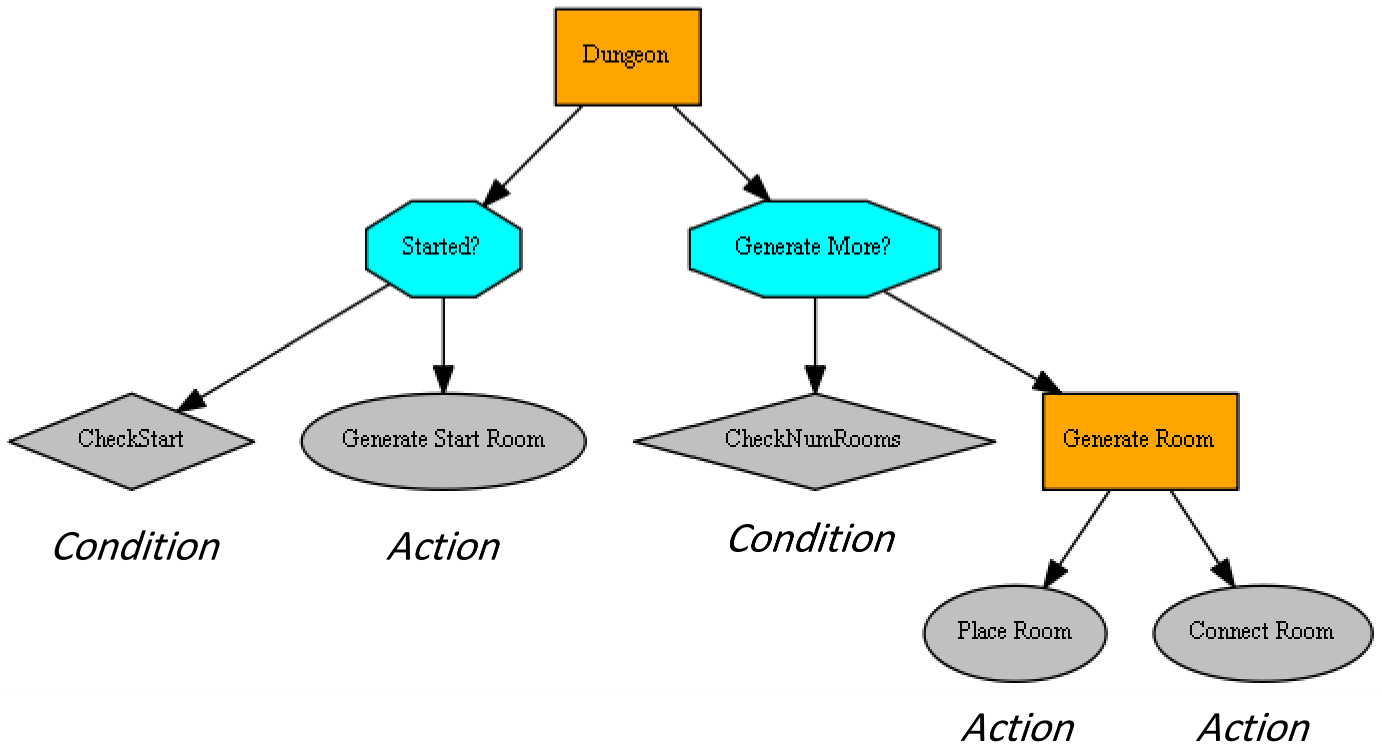
Source: py-trees.readthedocs.io

Behavior Trees

- Directed trees consisting of
  - Root
  - Control flow (internal) nodes
  - Leaf nodes

- Control flow nodes
  - Sequence
  - Selector
  - Parallel
  - Decorator

- Leaf nodes
  - Action
  - Condition



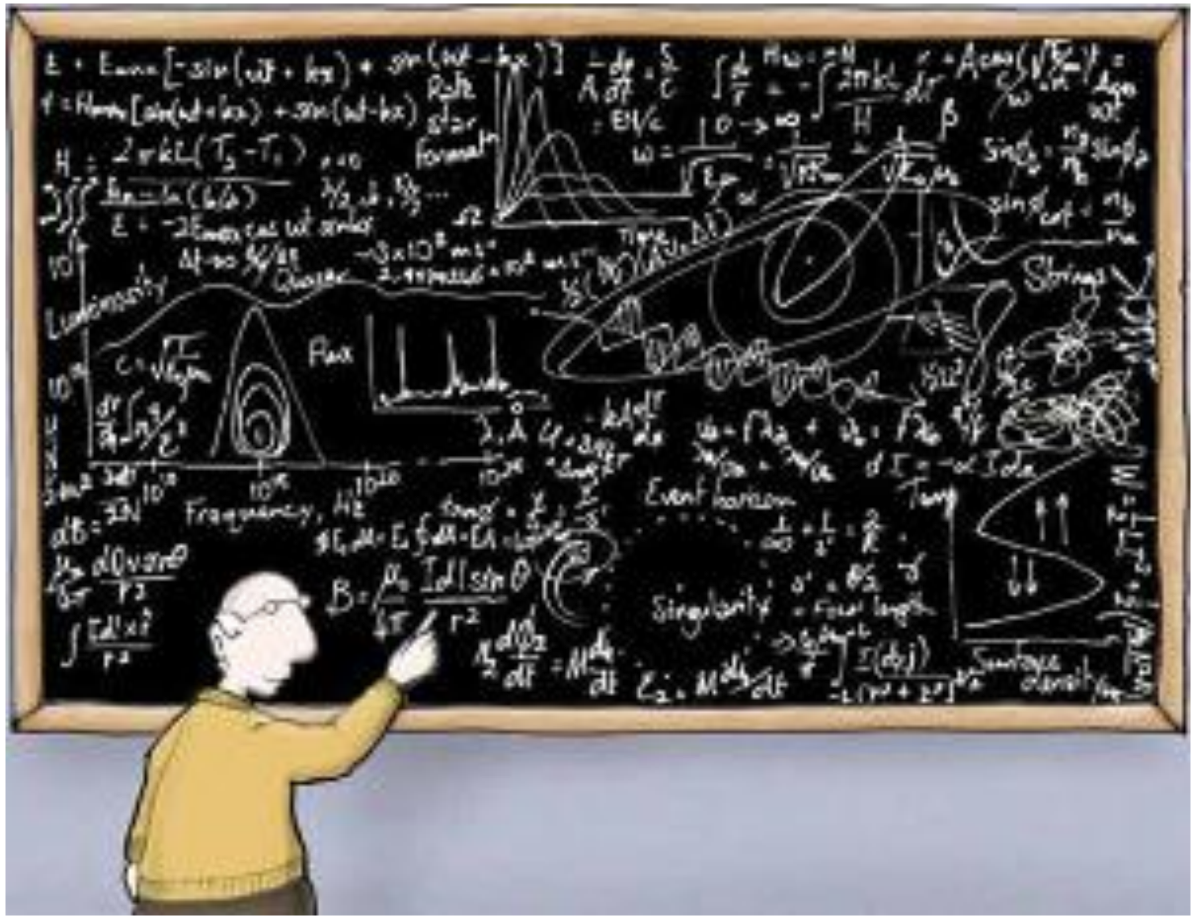
Behavior Trees

- Directed trees consisting of
  - Root
  - Control flow (internal) nodes
  - Leaf nodes

- Control flow nodes
  - Sequence
  - Selector
  - Parallel
  - Decorator

- Leaf nodes
  - Action
  - Condition

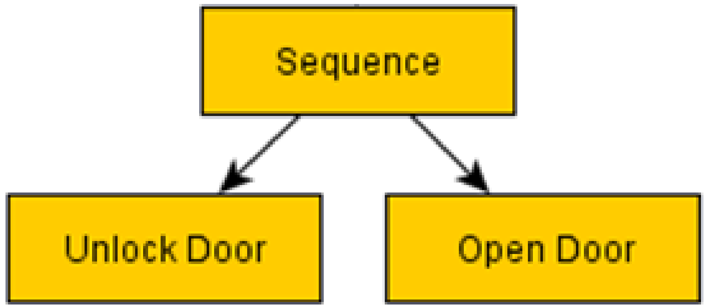
- Blackboard



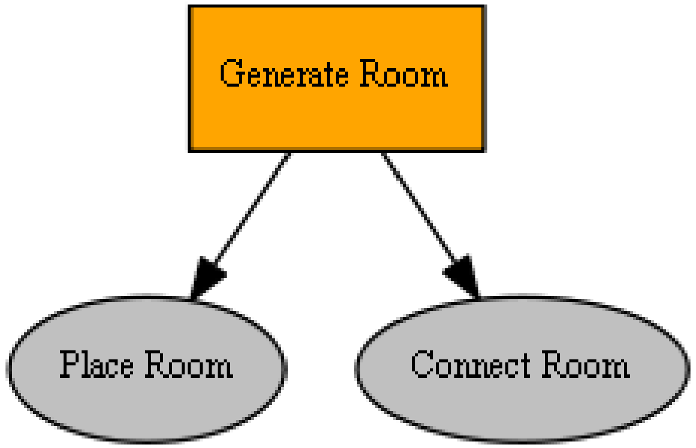
Source: py-trees.readthedocs.io

BT ---> PCGBT

- Action (leaf) nodes
  - BT - execute NPC/enemy behaviors
    --- work with an underlying library of scripted actions and behaviors
  - PCGBT – perform level design tasks e.g. generate a section of a level
    --- work with an underlying library of level segments and/or generative algorithms



BT



PCGBT

PCGBT

- Content library which defines what the BT works with
  --- e.g. level segments (this work), generative procedures

- Action node behavior
  --- e.g. sampling segments satisfying certain constraints (this work), run specific generative algorithm

- Blackboard for storing globally accessible information useful for level generation
  --- location where content needs to be generated, player info, designer prefs, game state etc.

PCGBT

- Content library which defines what the BT works with
  --- e.g. level segments (this work), generative procedures

- Action node behavior
  --- e.g. sampling segments satisfying certain constraints (this work), run specific generative algorithm

- Blackboard for storing globally accessible information useful for level generation
  --- location where content needs to be placed/generated, player info, designer preferences etc.

- NOTE: Exploratory work
  --- Only used sequence and selector nodes in a non-dynamic setting (but parallel and decorator nodes could also be used)
  --- All branching decisions made at random (but could use designer preferences, player behavior, dynamic difficulty adjustment etc.)

Super Mario Bros.

- Content library

  --- 15x16 level segments extracted from VGLC levels

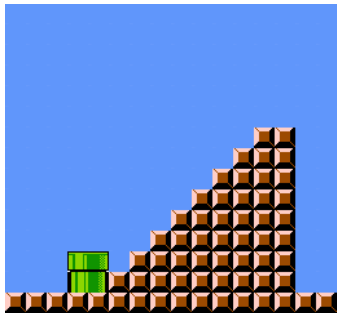
  --- segments manually categorized based on design patterns (Dahlskog and Togelius 2012) contained within them

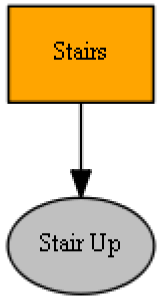
- Action node behavior

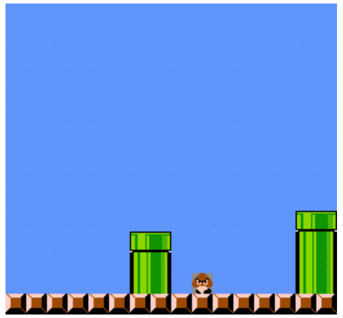
  --- Nodes take one or more patterns as parameters, sample a segment from the set of segments containing at least 1 of these patterns

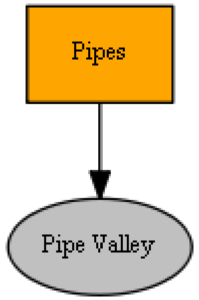
  --- Place segment at current location; increment current x on blackboard by 1

Super Mario Bros.

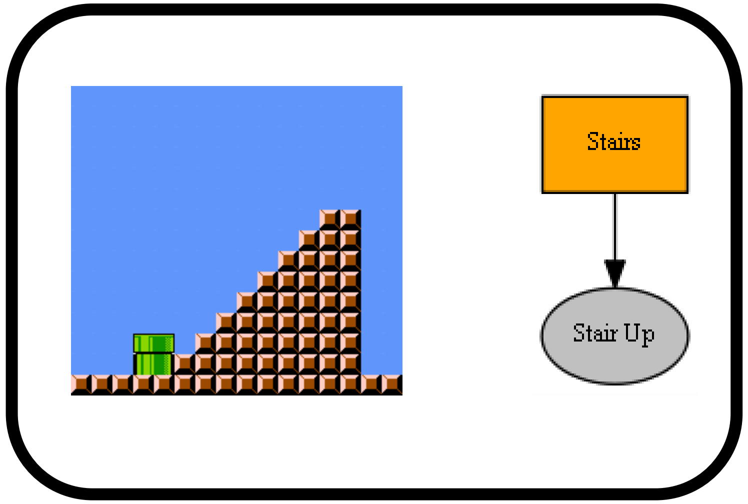


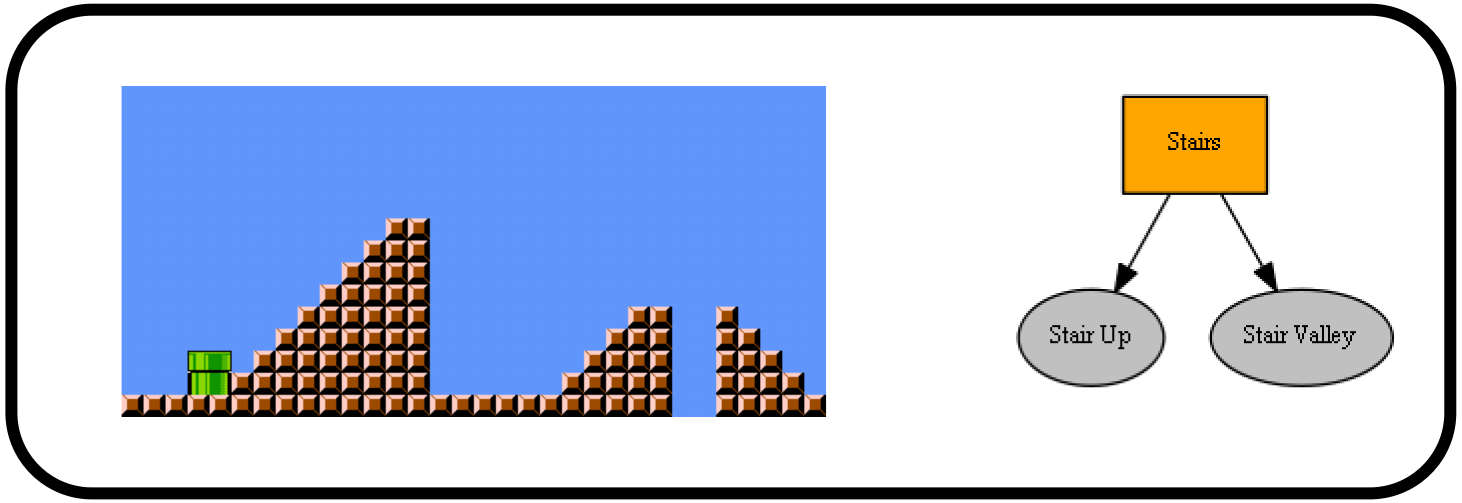


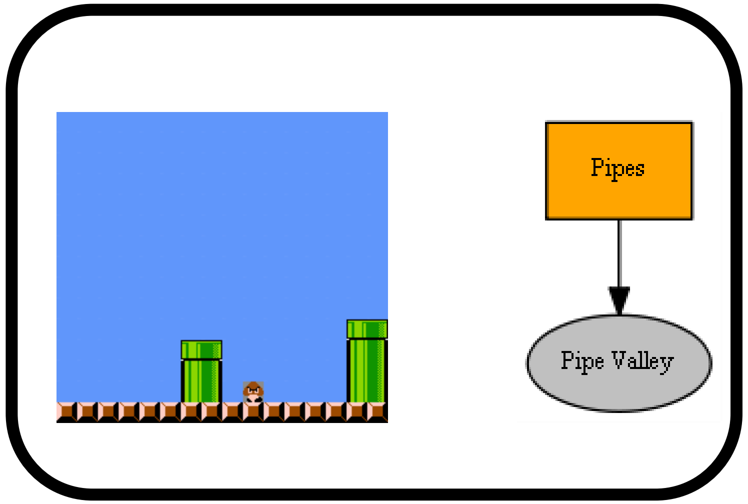


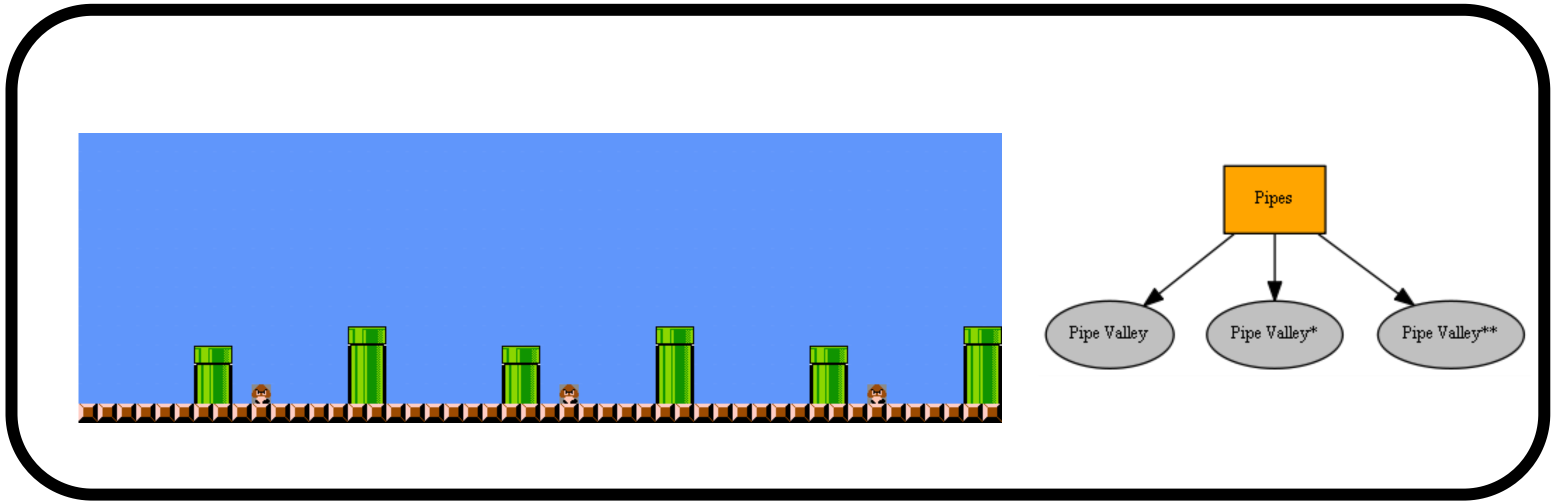


Super Mario Bros.

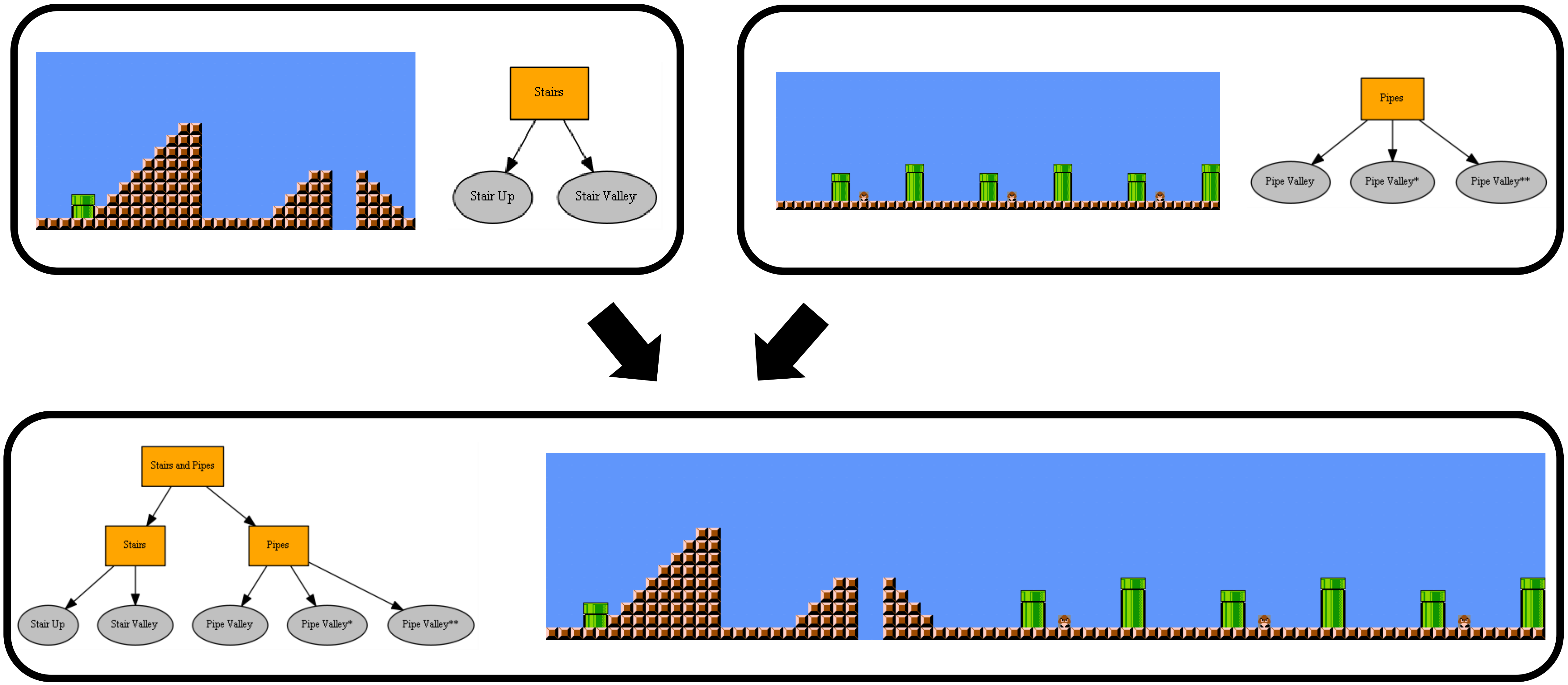




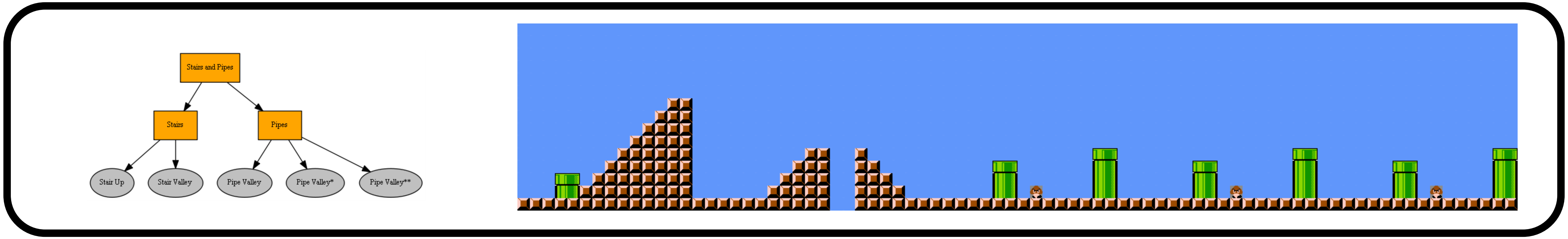


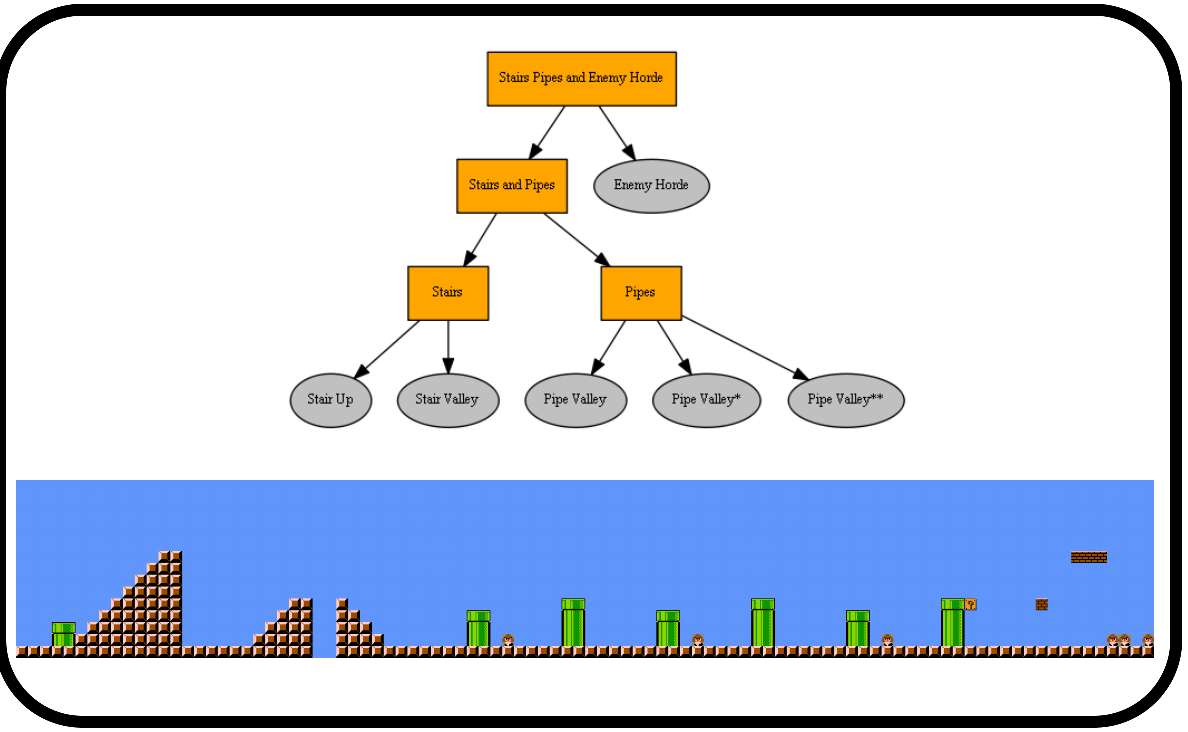


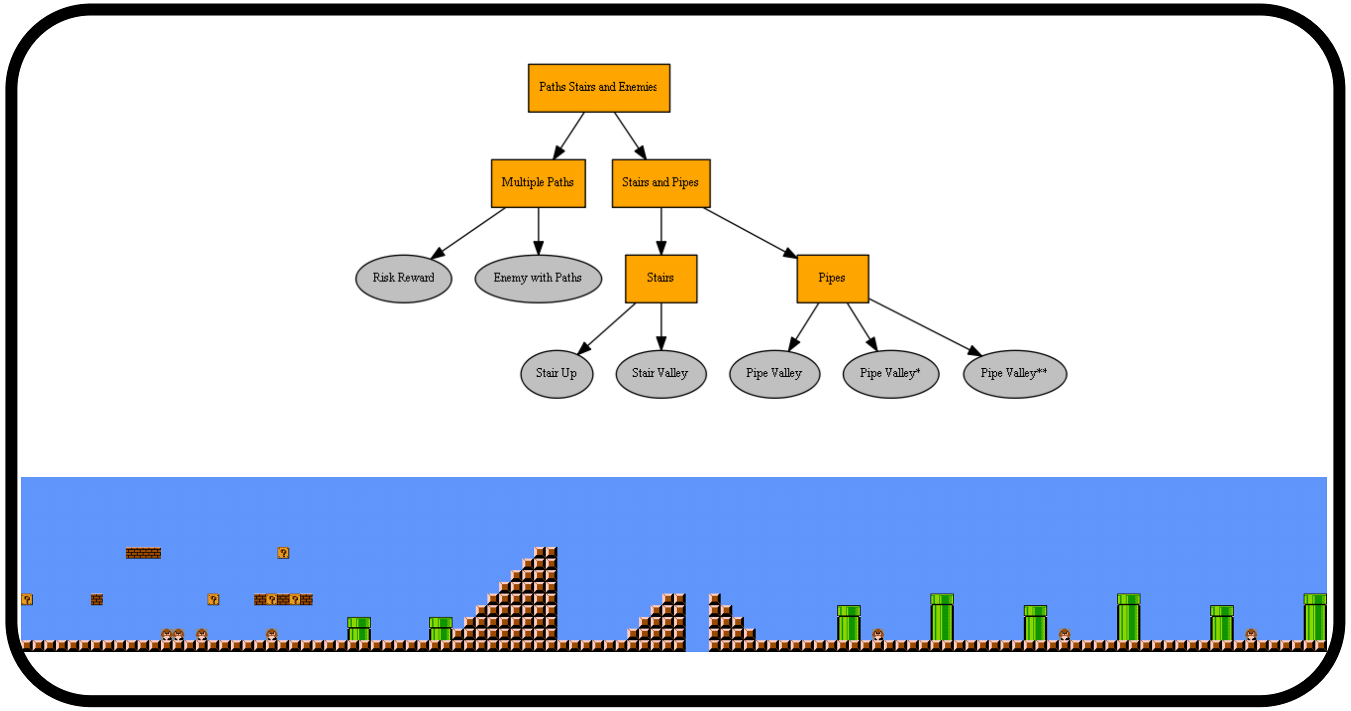
Super Mario Bros.



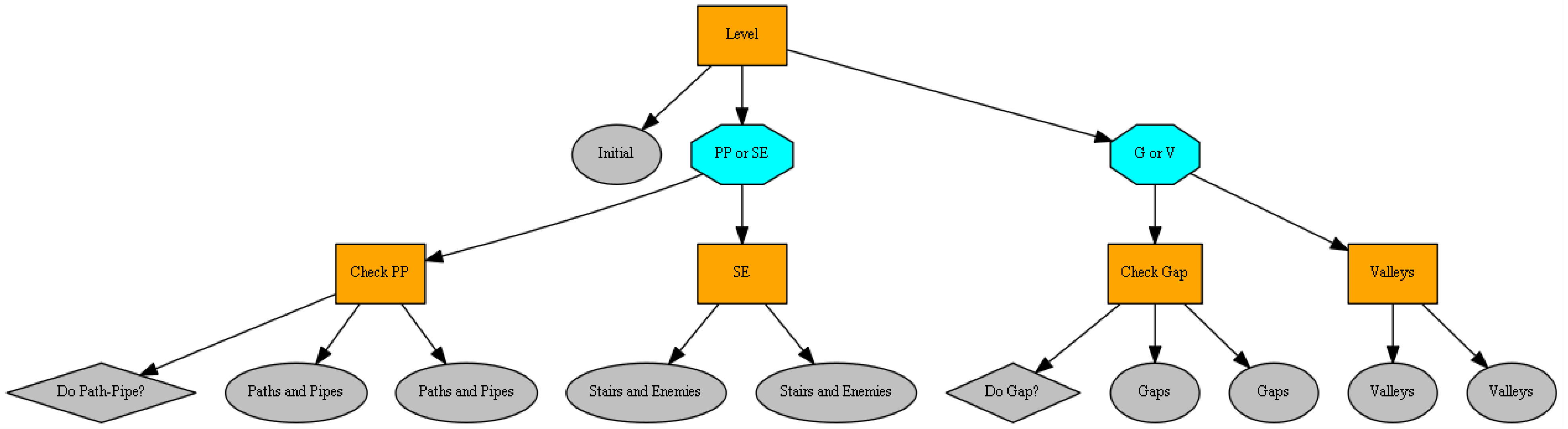
Super Mario Bros.

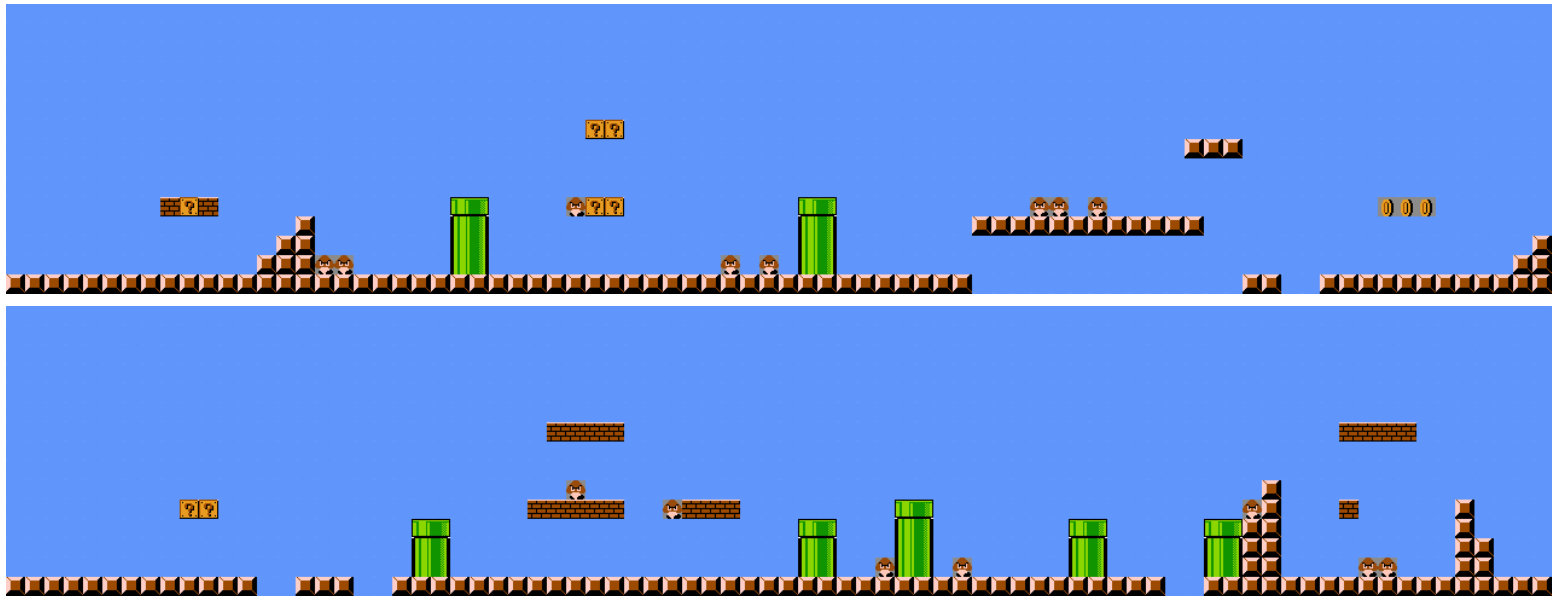






Super Mario Bros.



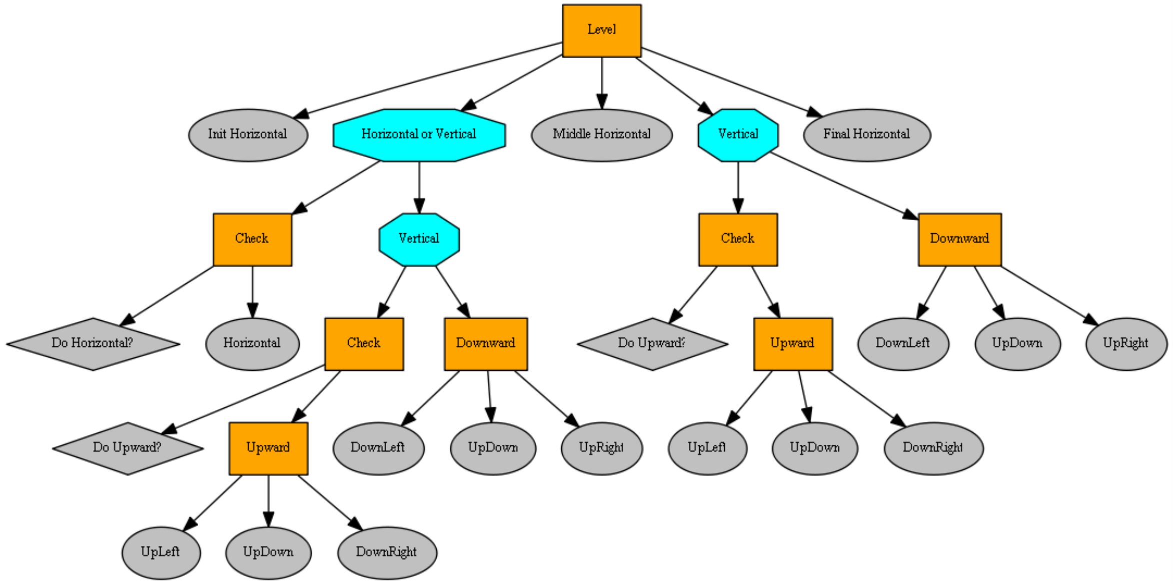


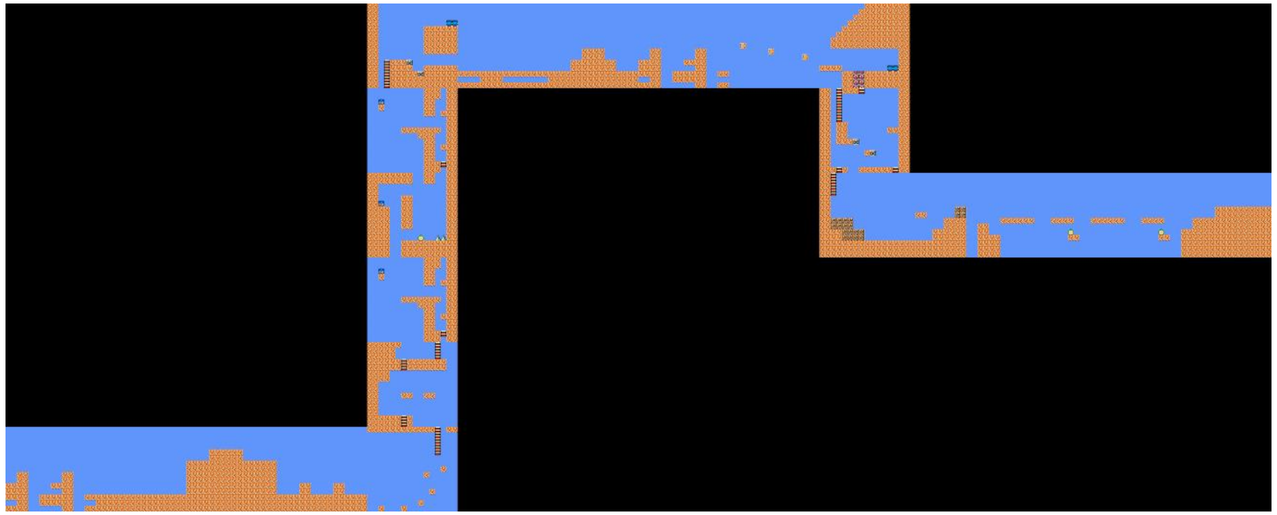
Mega Man

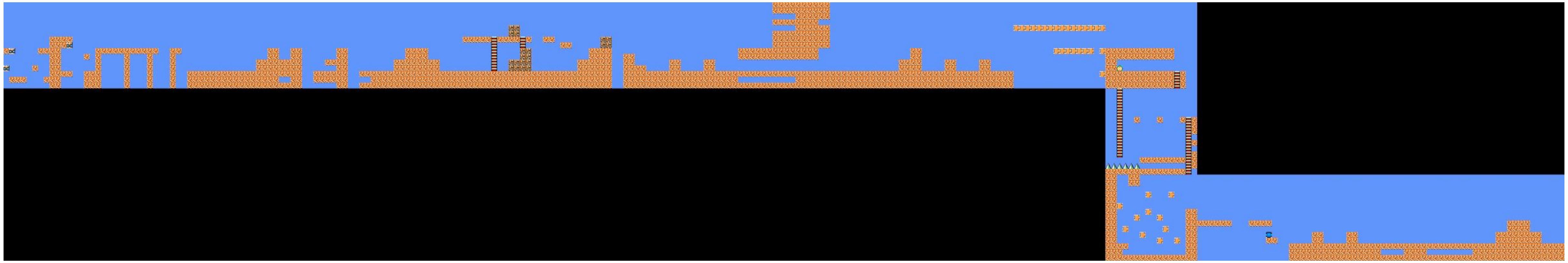
- Content library
  --- 15x16 level segments extracted from VGLC levels
  --- Segments grouped based on open directions --- each tagged with U, D, L and/or R

- Action node behavior
  --- Nodes take one or more direction as parameters, sample a segment from the set of segments open in those directions
  --- Check if opening exists between sampled segment and prior segment, else re-sample
  --- Adjust current x, y on blackboard accordingly

Mega Man


Level
Init Horizontal
Horizontal or Vertical
Middle Horizontal
Vertical
Final Horizontal
Check
Vertical
Check
Downward
Do Horizontal?
Horizontal
Check
Downward
Do Upward?
Upward
DownLeft
UpDown
UpRight
Do Upward?
Upward
DownLeft
UpDown
UpRight
UpLeft
UpDown
DownRight
UpLeft
UpDown
DownRight





Dungeons

- BTs for generating dungeons by modeling simple layout generation algorithm to test approach:
  --- on genres outside platformers
  --- using multiple ticks of the root node
  --- for non-linear level generation

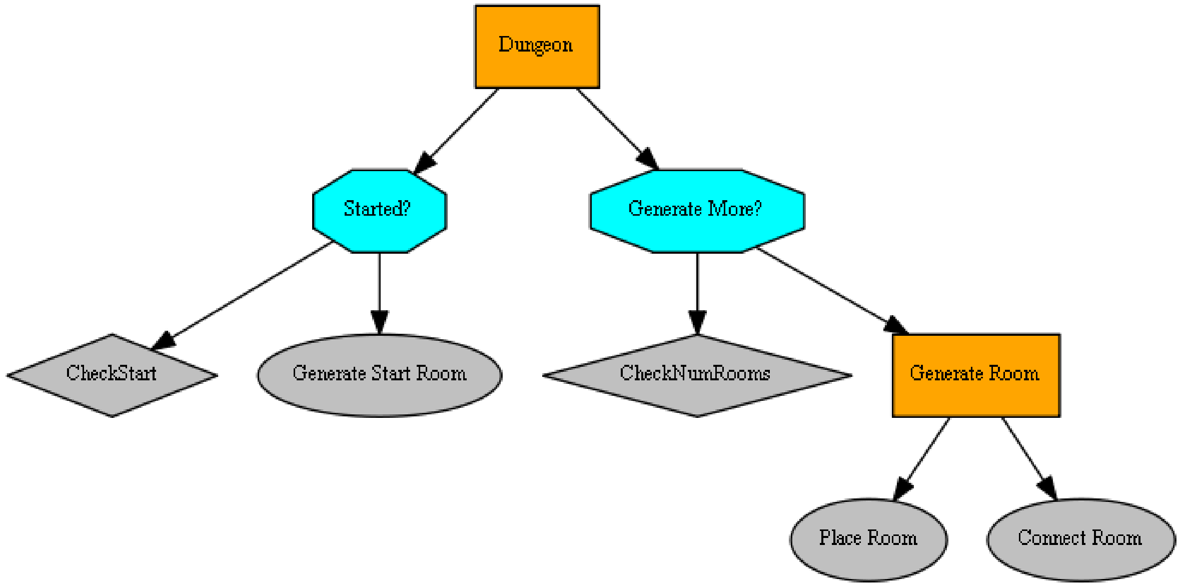
Dungeons

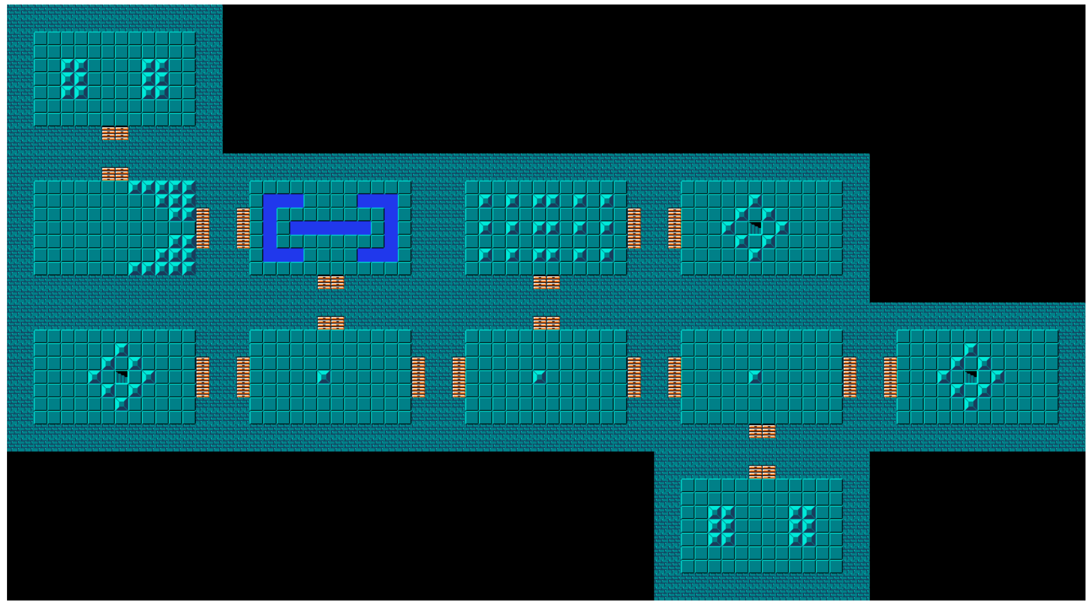
- BTs for generating dungeons by modeling simple layout generation algorithm to test approach:
  --- on genres outside platformers
  --- using multiple ticks of the root node
  --- for non-linear level generation

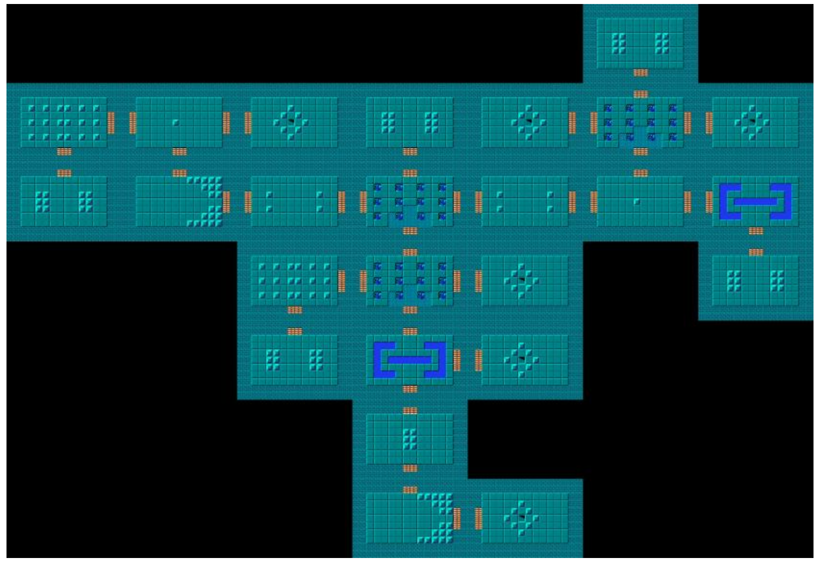
- Content library
  --- 11x16 rooms from Zelda levels in the VGLC
  --- Each room tagged with direction(s) containing doors

- Action node behavior
  --- Nodes sample from rooms with desired doors determined by layout algorithm

Dungeons


Dungeon
Started?
Generate More?
CheckStart
Generate Start Room
CheckNumRooms
Generate Room
Place Room
Connect Room





Generic BTs

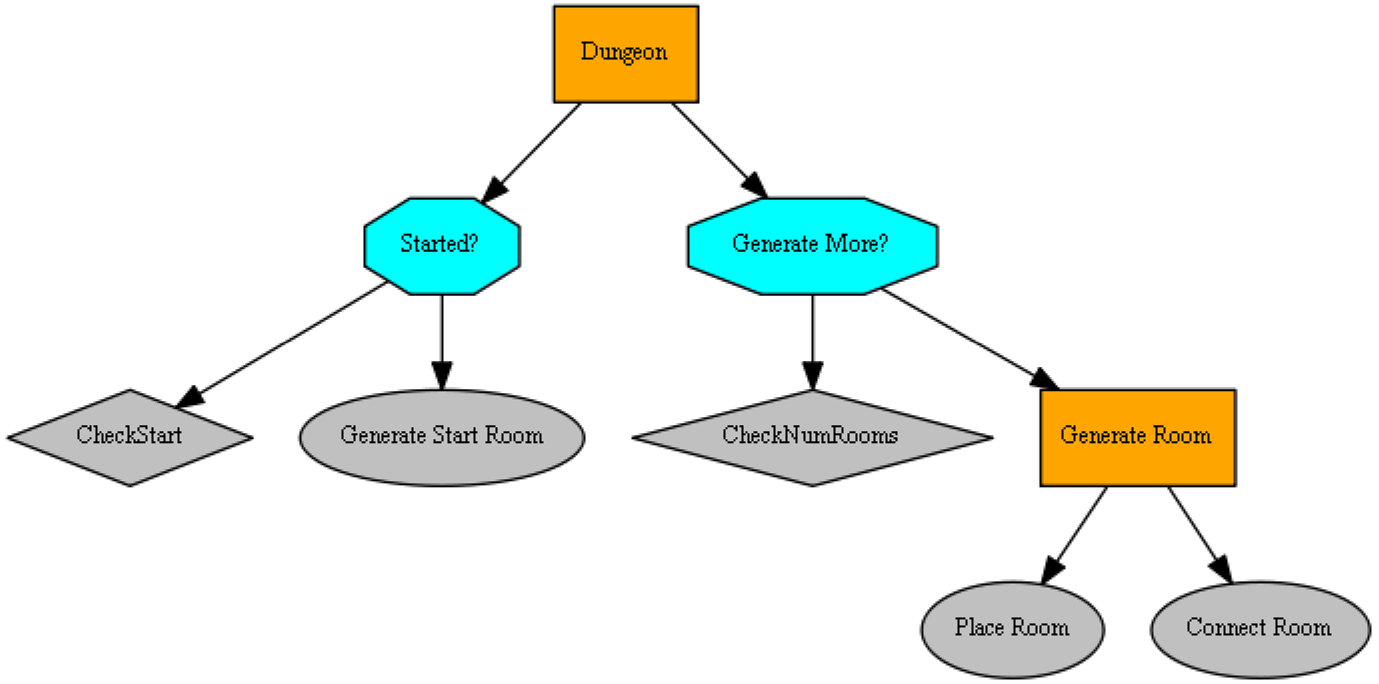
- Game-agnostic/generic BTs that can generating levels for multiple games
  - Requirement: section generated by control flow nodes and segments produced by action nodes are compatible across multiple games

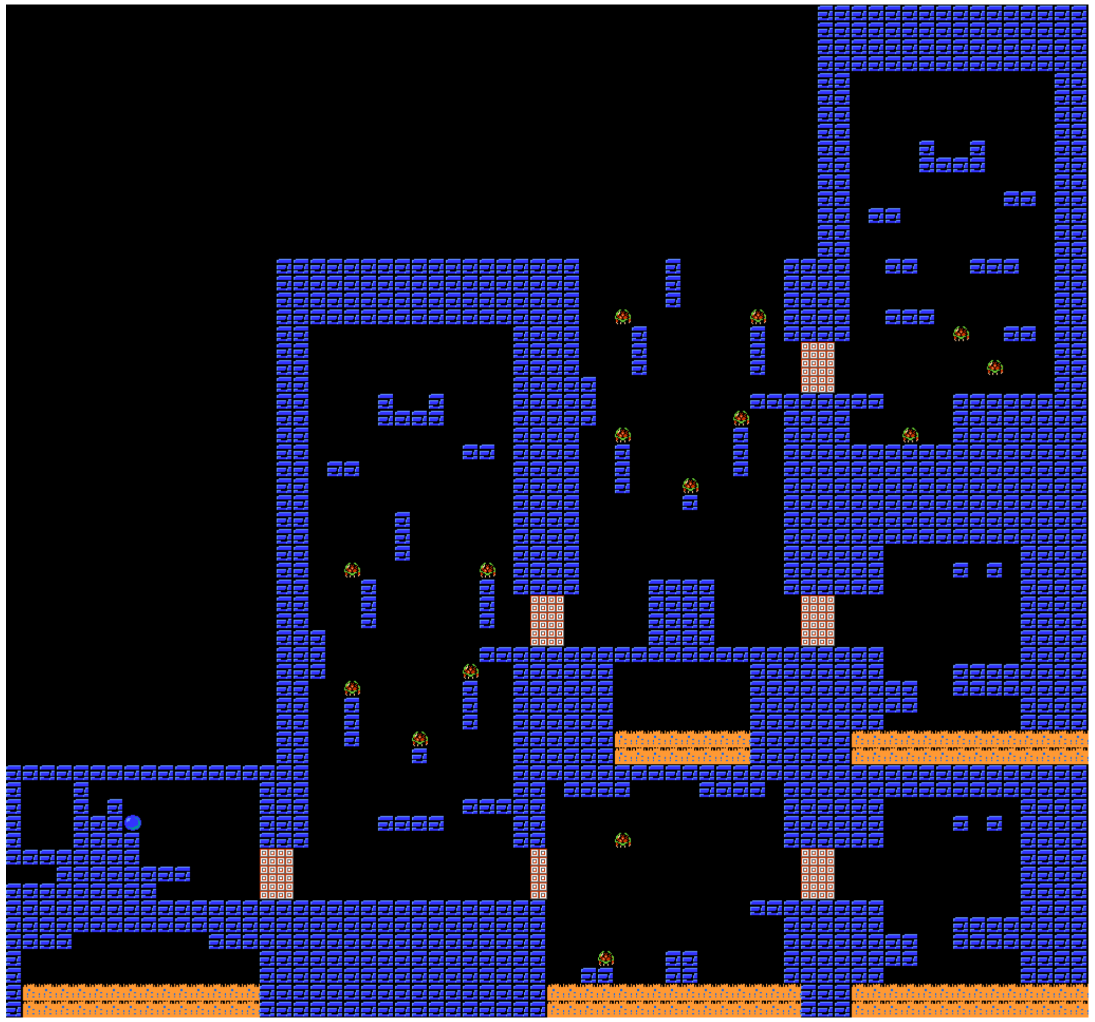
- Non-generic
  - SMB BT since it utilizes Mario-specific design patterns

- Generic
  - MM BT could generate levels for any platformer consisting of vertical and horizontal sections and segments with openings in 4 directions
  - Dungeon BT could be used for any game with interconnected segments
  - *Metroid* --- platformer with a sprawling, interconnected game-world

Generic BTs

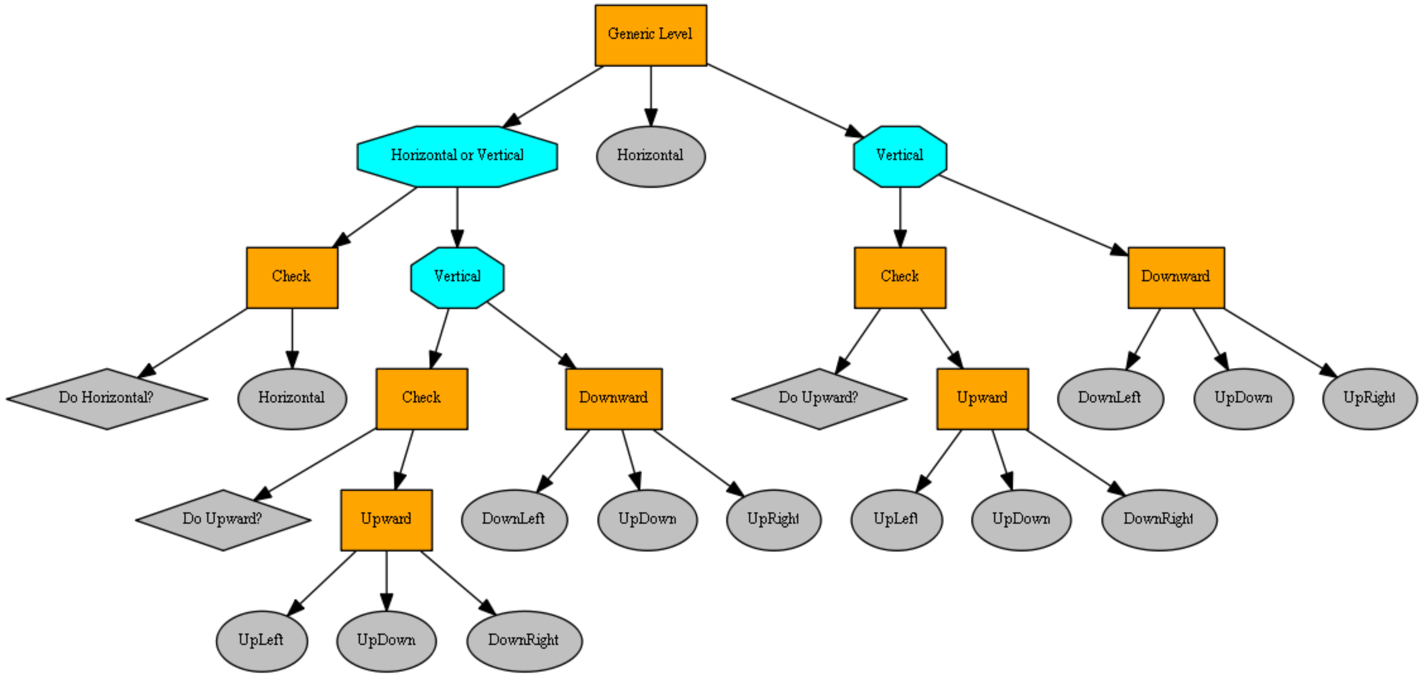
- Same tree as dungeon layout BT, but action nodes here work with 15x16 Metroid segments rather than 11x16 Zelda rooms

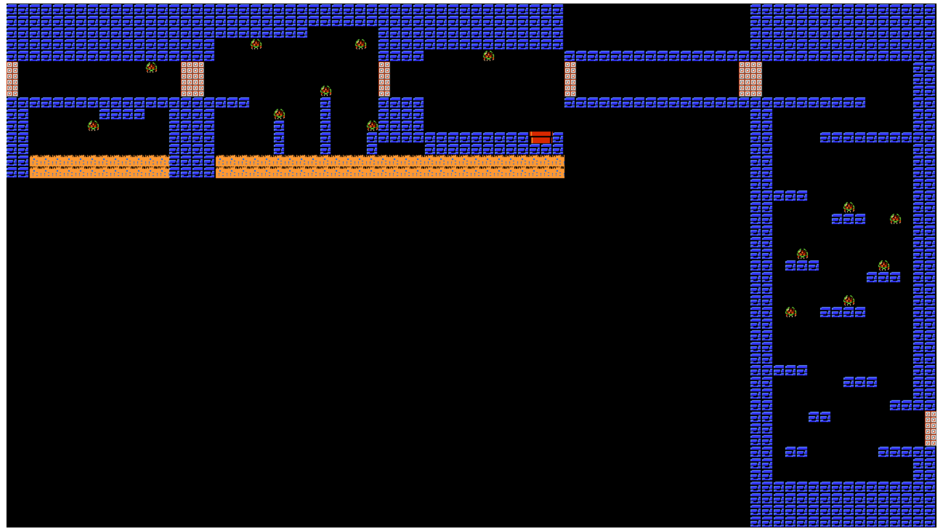


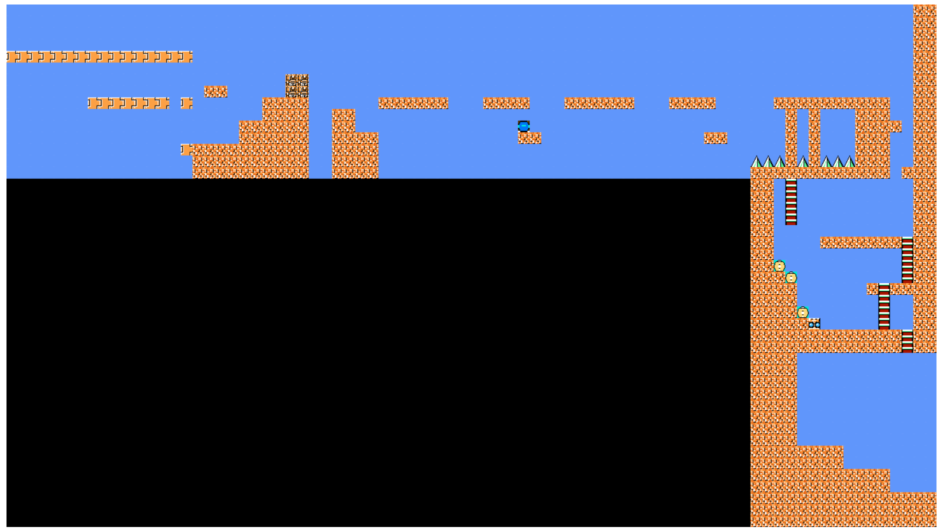


Generic BTs

- Action nodes sample 15x16 MM segments or 15x16 Metroid segments



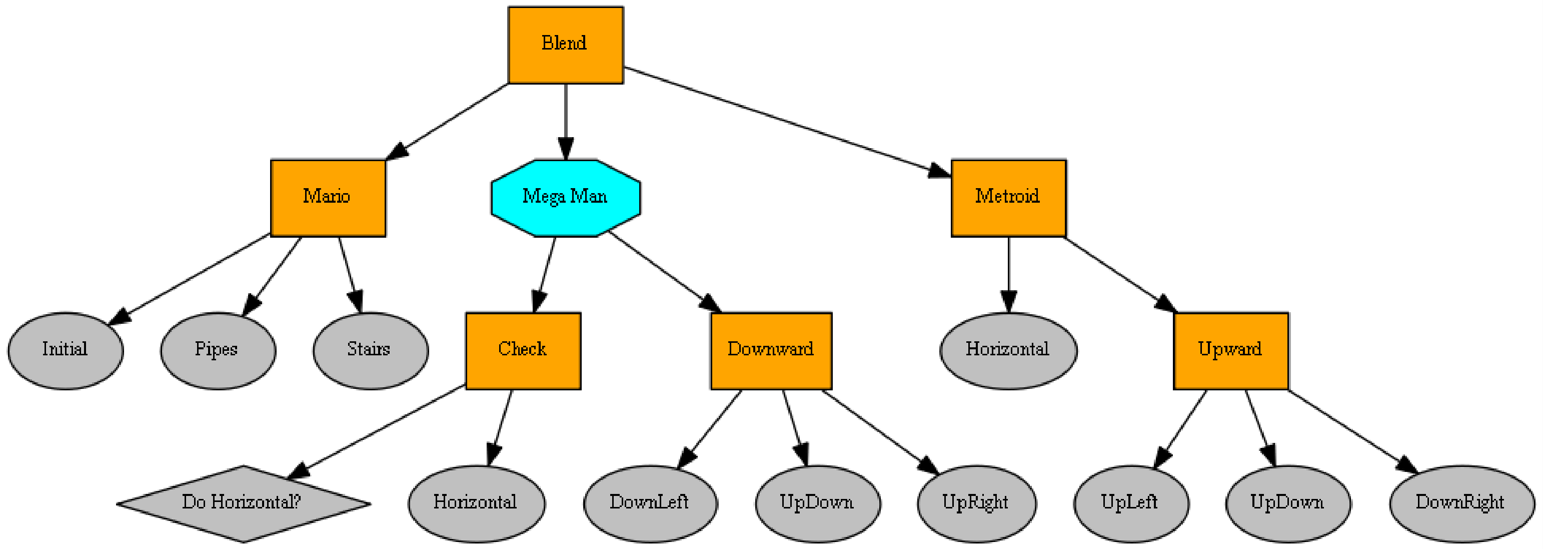

Metroid Level

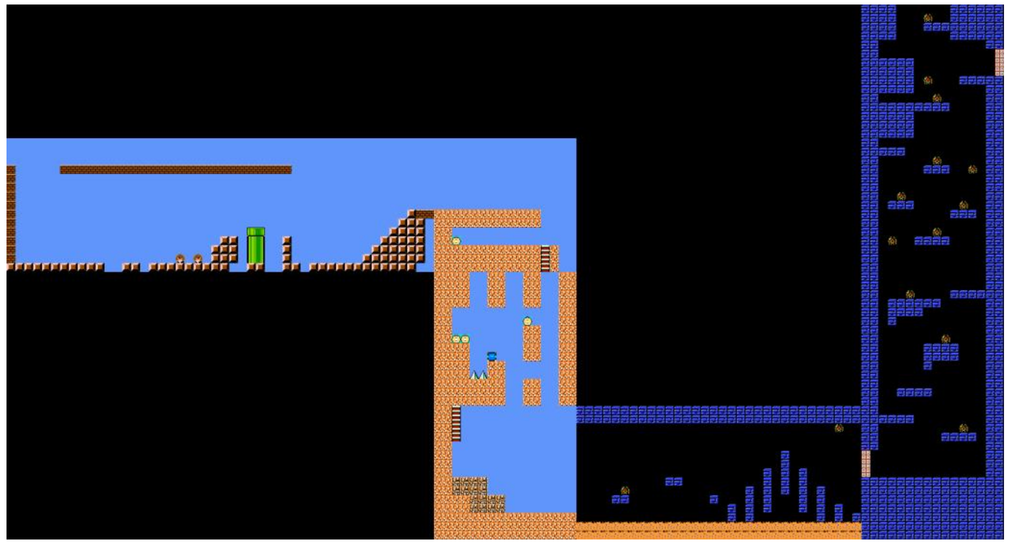

Mega Man Level

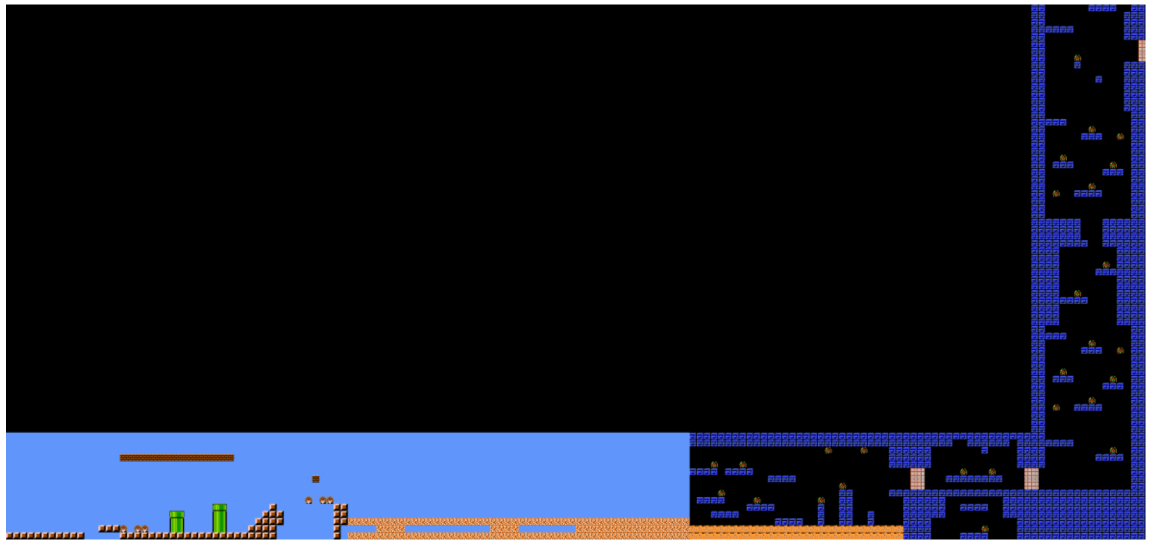
Blend BTs

- Combine BTs for generating level sections into a single tree
  → Generate whole levels

- Combine BTs for generating levels for different games into a single tree
  → Generate whole blended levels







Takeaways

- BTs can be repurposed for modeling design agents and generate levels for several games

- Procedural Content Generation using Behavior Trees (PCGBT)
  - Use of BTs for modeling procedural level generators
  - NOT a specific algorithm/BT implementation

    --- condition/action node implementations are agnostic to the framework

    --- decoupling the framework from implementation helps generalize to multiple design styles

- Primary utility

  --- allow designers to combine handmade/generated content into whole levels in a modular, explainable (and potentially dynamic) manner

Research Directions

- Dynamic Level Generation
    - Generate level sections based on runtime conditions
    - Tailor generation towards different player types, perform DDA

Research Directions

- Dynamic Level Generation
  - Generate level sections based on runtime conditions
  - Tailor generation towards different player types, perform DDA

- Hybrid BTs
  - Combining PCGBTs with traditional BTs for NPC behavior
  - E.g. PCGBT generates level section, traditional BT checks playability
  - Co-evolve game-playing (i.e. BTs) and game-design (i.e. PCGBTs) agents?

Research Directions

- Dynamic Level Generation
  - Generate level sections based on runtime conditions
  - Tailor generation towards different player types, perform DDA

- Hybrid BTs
  - Combining PCGBTs with traditional BTs for NPC behavior
  - E.g. PCGBT generates level section, traditional BT checks playability
  - Co-evolve game-playing (i.e. BTs) and game-design (i.e. PCGBTs) agents?

- RL and Evolution
  - Prior work has used RL and evolution to generate traditional BTs
  - For PCGBTs, RL/evolution could infer/evolve BT structures from a set of exemplar levels

Research Directions

- Dynamic Level Generation
  - Generate level sections based on runtime conditions
  - Tailor generation towards different player types, perform DDA

- Hybrid BTs
  - Combining PCGBTs with traditional BTs for NPC behavior
  - E.g. PCGBT generates level section, traditional BT checks playability
  - Co-evolve game-playing (i.e. BTs) and game-design (i.e. PCGBTs) agents?

- RL and Evolution
  - Prior work has used RL and evolution to generate traditional BTs
  - For PCGBTs, RL/evolution could infer/evolve BT structures from a set of exemplar levels

- General Game Design
  - Dynamically generate different games at runtime using generic BTs?
  - Dynamically switch different games in and out during gameplay using blend BTs?

Future Work

- Playability evaluations / expressive range analyses
- Generate levels at run-time / generate segments from scratch
- GUI/interactive application to enable designers to create custom PCG-BTs
- User study to evaluate PCGBT-generated levels

Future Work

- Playability evaluations / expressive range analyses
- Generate levels at run-time / generate segments from scratch
- GUI/interactive application to enable designers to create custom PCG-BTs
- User study to evaluate PCGBT-generated levels

Contact
Anurag Sarkar
Northeastern University
sarkar.an@northeastern.edu