German Research Center for Artificial Intelligence

BEHAVIOUR TREES AND PLAN EXECUTION

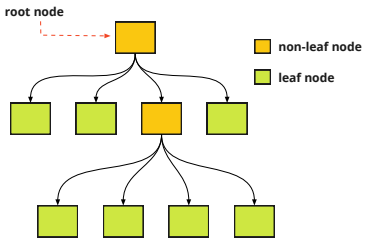
Dr. Teena Hassan
Robotics Innovation Center
DFKI Bremen

Prof. Dr. Dr. h.c. Frank Kirchner
Arbeitsgruppe Robotik, Universität Bremen
https://robotik.dfki-bremen.de/
robotik@dfki.de

11th January, 2022 – Bremen, Deutschland

German Research Center for Artificial Intelligence

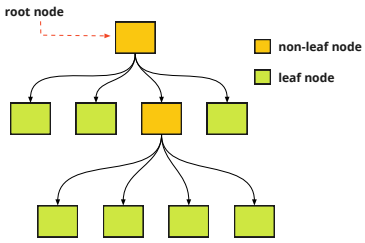
Concept of Behaviour Trees



A tree

What is a tree?

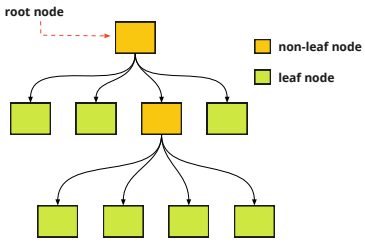
- A **hierarchical** structure of nodes.



A tree

What is a tree?

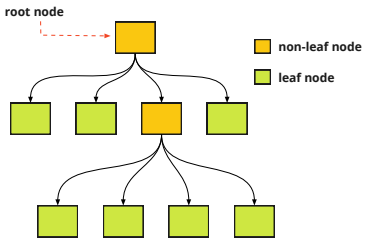
- A **hierarchical** structure of nodes.
- Node at topmost level is called the **root node**.



A tree

What is a tree?

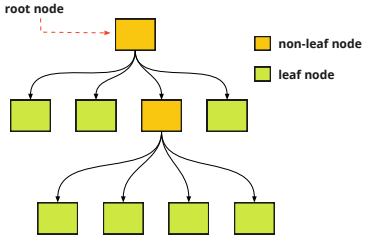
- A **hierarchical** structure of nodes.
- Node at topmost level is called the **root node**.
- Each node has zero or more successors, referred to as **child nodes**.



A tree

What is a tree?

- A **hierarchical** structure of nodes.
- Node at topmost level is called the **root node**.
- Each node has zero or more successors, referred to as **child nodes**.
- Nodes that have no children are called **leaf nodes**; all other nodes are called **non-leaf nodes**.

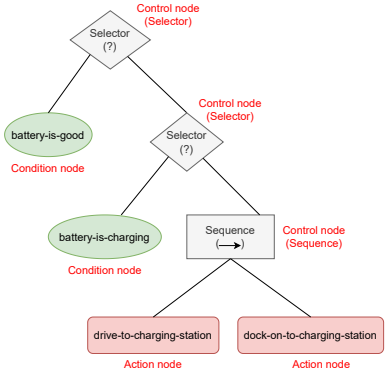


A tree

What is a tree?

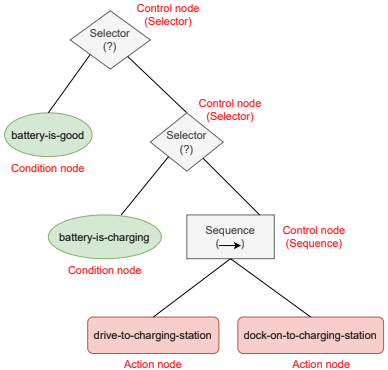
- A **hierarchical** structure of nodes.
- Node at topmost level is called the **root node**.
- Each node has zero or more successors, referred to as **child nodes**.
- Nodes that have no children are called **leaf nodes**; all other nodes are called **non-leaf nodes**.
- Each node has a predecessor called **parent node**.
  - Exception: The root node has no parent.

German Research Center for Artificial Intelligence



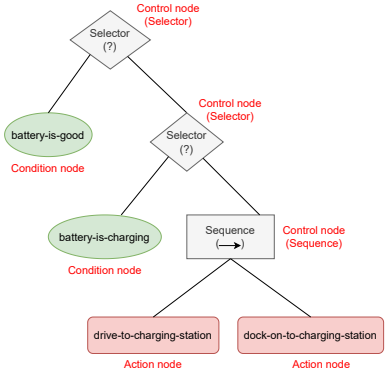
A behaviour tree: Charge robot's battery

- A **behaviour tree** is a hierarchical structure for controlling the execution of a set of actions.



A behaviour tree: Charge robot's battery

- A **behaviour tree** is a hierarchical structure for controlling the execution of a set of actions.
- It contains three types of nodes:
  - **Condition** nodes
  - **Action** nodes
  - **Control** nodes



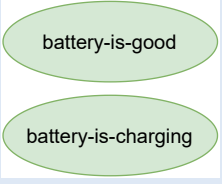
A behaviour tree: Charge robot's battery

- A **behaviour tree** is a hierarchical structure for controlling the execution of a set of actions.
- It contains three types of nodes:
  - **Condition** nodes
  - **Action** nodes
  - **Control** nodes
- Condition and action nodes are leaf nodes.
- Control nodes are non-leaf nodes.

German Research Center for Artificial Intelligence

Condition node:

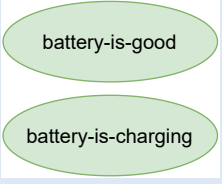
- Contains a Boolean-valued variable.
- Value of *True* indicates **success**.
- Value of *False* indicates **failure**.



German Research Center for Artificial Intelligence

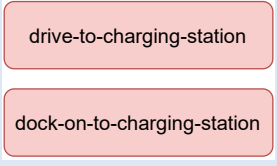
Condition node:

- Contains a Boolean-valued variable.
- Value of *True* indicates **success**.
- Value of *False* indicates **failure**.



Action node:

- Contains an action to be performed by the agent (e.g. robot).
- Action execution status: **running**, **success**, **failure**.



German Research Center for Artificial Intelligence

- Control nodes determine which child node should be executed next, i.e. how control should flow within the tree.
- Two commonly used control nodes are **selector** and **sequence**.

Selector

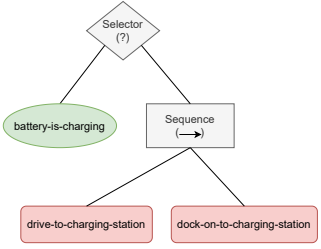
- Child nodes are executed one by one from left to right.
- The next child node is executed **if and only if** all the previous child nodes **failed**.

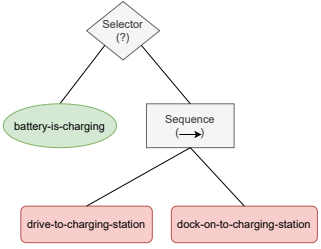
Sequence

- Child nodes are executed one by one from left to right.
- The next child node is executed **if and only if** all the previous child nodes were **successful**.

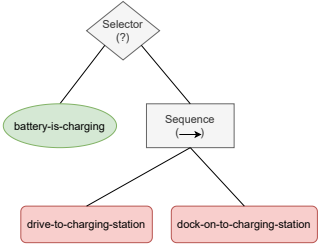
German Research Center for Artificial Intelligence

- Execution starts at root node.



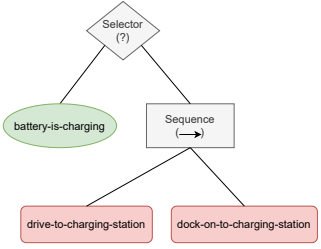


- Execution starts at root node.
- Root node here is a selector node.



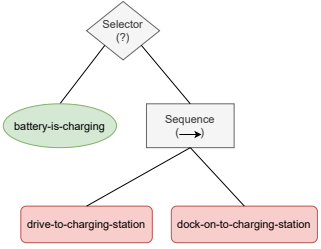
- Execution starts at root node.
- Root node here is a selector node.
  - First child node is *battery-is-charging*.

German Research Center for Artificial Intelligence

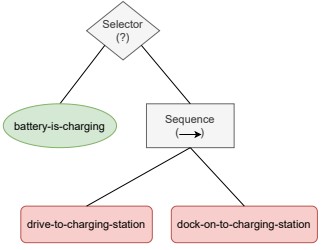


- Execution starts at root node.
- Root node here is a selector node.
  - First child node is *battery-is-charging*.
  - If battery is charging, then condition node is **successful**. The next child node is **not executed**.

German Research Center for Artificial Intelligence



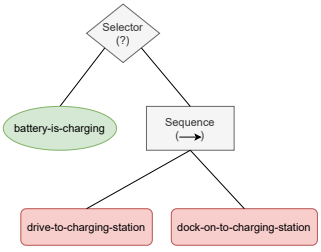
- Execution starts at root node.
- Root node here is a selector node.
  - First child node is *battery-is-charging*.
  - If battery is charging, then condition node is **successful**. The next child node is **not executed**.
  - If battery is **not** charging, then condition node **failed**. Control now flows to **next child node**.

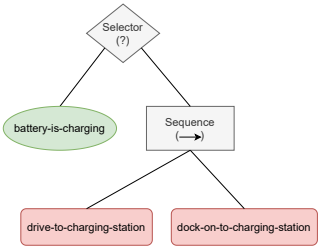


- Execution starts at root node.
- Root node here is a selector node.
  - First child node is *battery-is-charging*.
  - If battery is charging, then condition node is **successful**. The next child node is **not executed**.
  - If battery is **not** charging, then condition node **failed**. Control now flows to **next child node**.
  - Next child node is a sequence node.

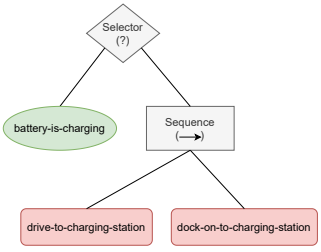
German Research Center for Artificial Intelligence

- Execution of sequence node:



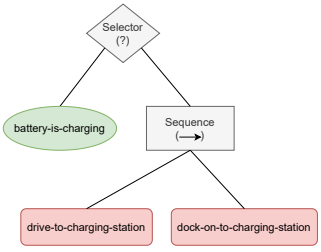


- Execution of sequence node:
  - First child node is the action *drive-to-charging-station*.



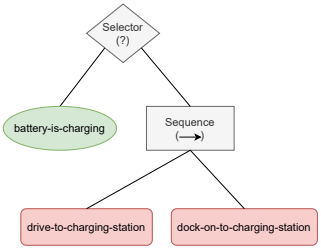
- Execution of sequence node:
  - First child node is the action *drive-to-charging-station*.
  - If this action **failed**, then the next child node is **not executed**.

German Research Center for Artificial Intelligence



- Execution of sequence node:
  - First child node is the action *drive-to-charging-station*.
  - If this action **failed**, then the next child node is **not executed**.
  - If this action is **successfully** completed, then control flows to **next child node**.

German Research Center for Artificial Intelligence

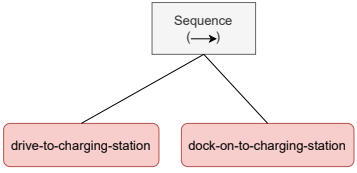


- Execution of sequence node:
  - First child node is the action *drive-to-charging-station*.
  - If this action **failed**, then the next child node is **not executed**.
  - If this action is **successfully** completed, then control flows to **next child node**.
  - Next child node is the action *dock-on-to-charging-station*.

German Research Center for Artificial Intelligence

Status of a sequence node:

- If **one** of the child nodes fails, then **failure**.
- If one of the child nodes is running, then **running**.
- If **all** child nodes are successful, then **success**.
- Similar to logical AND operation.



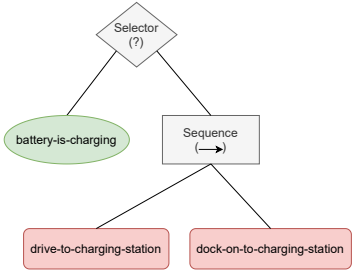
- Status of condition and action nodes: (See slide no. 6).

Universität Bremen

German Research Center for Artificial Intelligence

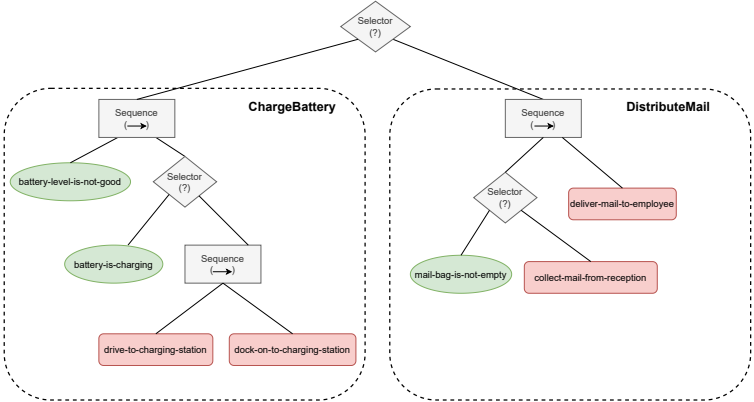
Status of a selector node:

- If **one** of the child nodes is successful, then **success**.
- If one of the child nodes is running, then **running**.
- If **all** child nodes failed, then **failure**.
- Similar to logical OR operation.



German Research Center for Artificial Intelligence

- Behaviour trees for simple tasks can be combined to model more complex tasks.
- Allows the reuse of subtrees in other tasks.
- If a task changes, then only the relevant subtree has to be updated.



German Research Center for Artificial Intelligence

- **Reactivity** refers to "the ability to quickly and efficiently react to changes"(Page 38 in [1]) in internal and external environments.

German Research Center for Artificial Intelligence

- **Reactivity** refers to "the ability to quickly and efficiently react to changes"(Page 38 in [1]) in internal and external environments.
- The status of a behaviour tree is checked at a specific frequency known as the **tick frequency**.

German Research Center for Artificial Intelligence

- **Reactivity** refers to "the ability to quickly and efficiently react to changes"(Page 38 in [1]) in internal and external environments.
- The status of a behaviour tree is checked at a specific frequency known as the **tick frequency**.
- Ticking begins at the root node and flows through the tree according to the logic of the control nodes.

German Research Center for Artificial Intelligence

- **Reactivity** refers to "the ability to quickly and efficiently react to changes"(Page 38 in [1]) in internal and external environments.
- The status of a behaviour tree is checked at a specific frequency known as the **tick frequency**.
- Ticking begins at the root node and flows through the tree according to the logic of the control nodes.
- If the status of any condition nodes or action nodes has changed, then it can be detected in the next 'tick'.

Behaviour Trees: Advantages
Reactivity

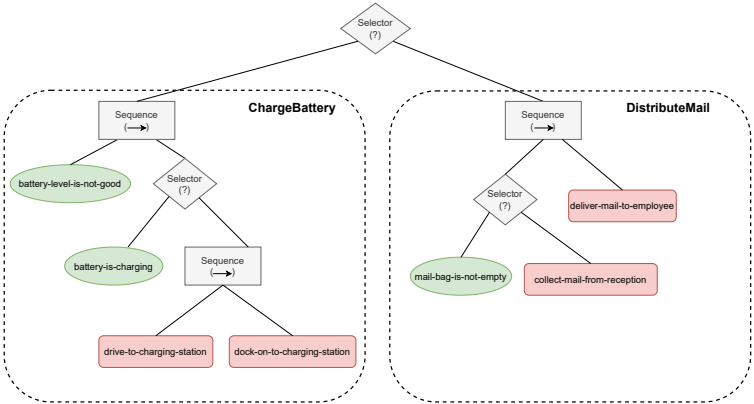
German Research Center for Artificial Intelligence

- **Reactivity** refers to "the ability to quickly and efficiently react to changes"(Page 38 in [1]) in internal and external environments.
- The status of a behaviour tree is checked at a specific frequency known as the **tick frequency**.
- Ticking begins at the root node and flows through the tree according to the logic of the control nodes.
- If the status of any condition nodes or action nodes has changed, then it can be detected in the next 'tick'.
- Any change will automatically cause the ongoing actions to be aborted or other actions to be started, depending on the control logic encoded in the tree.

[1] M. Colledanchise and P. Ögren, "Behavior Trees in Robotics and AI," Jul. 2018, doi: 10.1201/9780429489105.

German Research Center for Artificial Intelligence

- In the following tree, if the battery level goes low while the robot is distributing letters to employees, then the robot will abort this task and drive to the docking station.



- Introduced by the gaming industry to model the behaviour of non-player characters.
- Several software libraries are available.

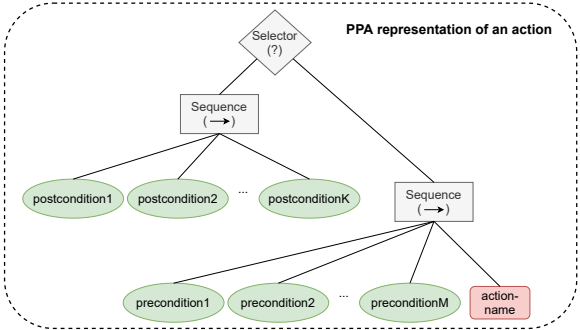
German Research Center for Artificial Intelligence

- Introduced by the gaming industry to model the behaviour of non-player characters.
- Several software libraries are available.
- The following libraries have been used in robotics:
  - BehaviorTree.CPP (written in C++)
    - Source code: `https://github.com/BehaviorTree/BehaviorTree.CPP`
    - Documentation: `https://www.behaviortree.dev/`
    - Used in the ROS 2 Navigation Stack (See here)
  - py_trees (wirtten in Python 3)
    - Source code: `https://github.com/splintered-reality/py_trees`
    - Documentation: `https://py-trees.readthedocs.io/en/devel/`

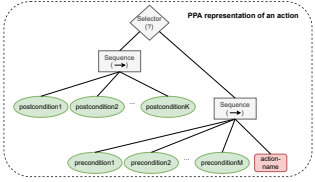
Converting Plans to Behaviour Trees

German Research Center for Artificial Intelligence

- An action has preconditions and effects (also known as postconditions).
- An action can be represented as a behaviour tree by applying the **Postcondition-Precondition-Action (PPA)** rule.

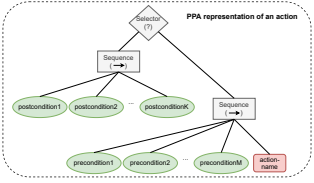


German Research Center for Artificial Intelligence



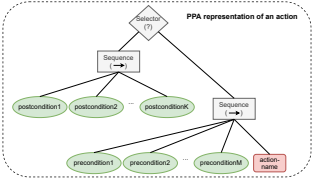
- For an action to be successful, all its postconditions should hold.
- Therefore, the postconditions are attached to a sequence node.

German Research Center for Artificial Intelligence



- If the current state of the world fulfils all postconditions of the action, then the action need not be executed or continued.
- Therefore, the postconditions and the action are attached to the left and right subtree, respectively, of a selector node.

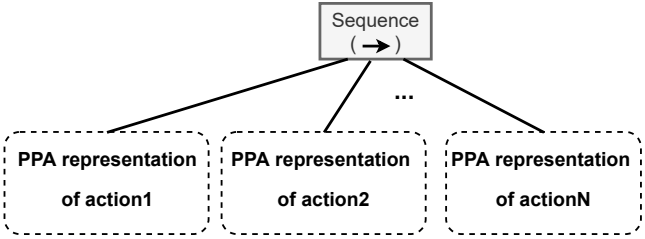
German Research Center for Artificial Intelligence



- An action can be executed only if its preconditions are satisfied.
- Therefore, the action and its preconditions are attached to the same sequence node.

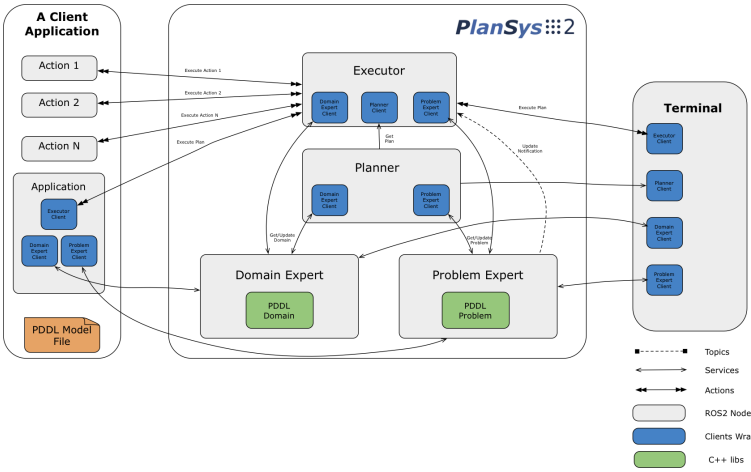
Note that, if preconditions or postconditions contain negations of predicates, these can be inverted before attaching to the behaviour tree.

Given a plan $\pi =<$ action1, action2, ..., actionN $>$, we attach the PPA representation of each action to a sequence node in the same order in which they appear in the plan.



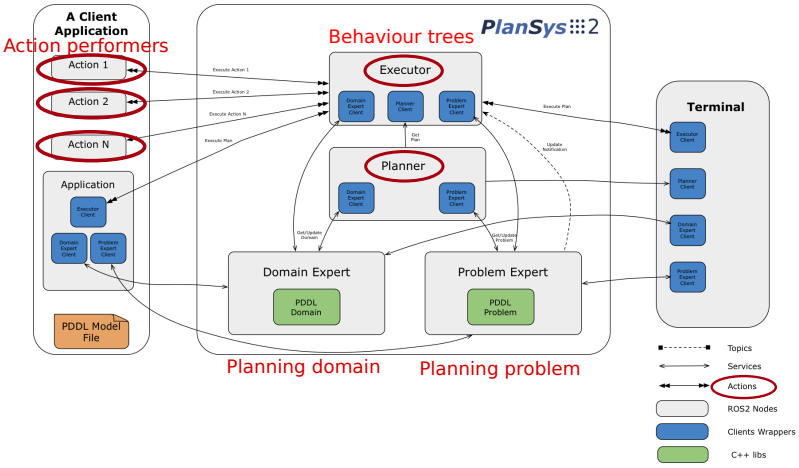
ROS 2 Planning System 2

German Research Center for Artificial Intelligence



Source: https://intelligentroboticslab.gsyc.urjc.es/ros2_planning_system.github.io/design/index.html

German Research Center for Artificial Intelligence



Source: `https://intelligentroboticslab.gsyc.urjc.es/ros2_planning_system.github.io/design/index.html`

Summary

German Research Center for Artificial Intelligence

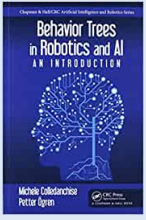
- Definition of behaviour trees.
- Types of nodes in a behaviour tree.
- Execution of nodes in a behaviour tree.
- Achieving modularity and reactivity using behaviour trees.
- Libraries for programming behaviour trees: BehaviorTree.CPP and py_trees.
- PPA rule to convert a linearly-ordered plan into a behaviour tree.
- A planning system in ROS 2.

References

German Research Center for Artificial Intelligence

Behavior Trees in Robotics and AI (English)

- Chapter 1.1: History and Motivation
- Chapter 1.2: Need for Reactiveness and Modularity
- Chapter 1.3: Classical Formulation
- Chapters 1.4, 1.5, 1.6: Examples
- Chapter 2.6: Advantages and Disadvantages



Available online:
`https://arxiv.org/pdf/1709.00084.pdf`

Thank You for Your Attention.