**Knowledge Retrieval for Task Tree Extraction in Functional Object-Oriented Networks**

INTRODUCTION

In the realm of robotics and artificial intelligence (AI) research, significant efforts have been dedicated to creating intelligent agents capable of understanding human intentions and executing tasks in human-centered domains. These domains encompass a wide spectrum of applications, from assisting the elderly and disabled to food preparation and delivery. Developing AI systems for such applications is challenging due to the diversity of tasks and dynamic environments in which they operate.

In particular, the field of AI-driven cooking presents unique challenges. Ingredients and objects can vary greatly in form, shape, size, and state, and there are situations where an AI system may face difficulty executing a complete recipe due to the unavailability of specific ingredients or objects. This complexity is further heightened when a robot needs to adapt its knowledge to accommodate or exclude ingredients it might not have encountered previously.

Inspired by humans' ability to adapt creatively to novel situations, this research delves into the task of generating plans for unfamiliar scenarios based on the limited knowledge available to AI systems, represented through the Functional Object-Oriented Network (FOON).

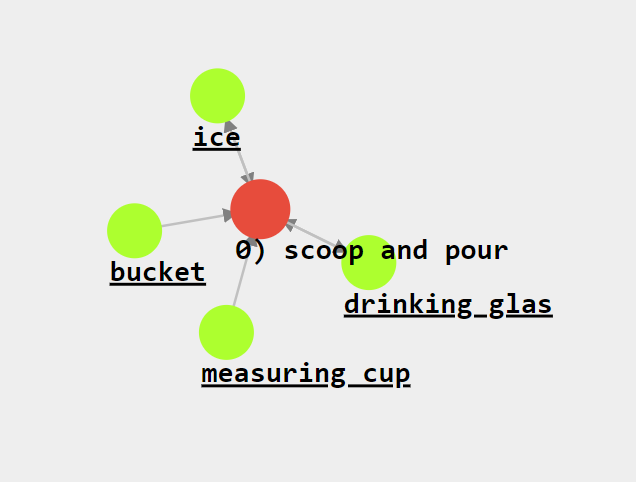


Fig. 1: A basic functional unit in FOON. The green nodes denote objects, while the red nodes denote motions.

FOON is a key knowledge representation in our AI system, enabling the creation of task trees for a wide range of recipes and ingredient variations. We aim to overcome the limitations by exploring ways to empower AI systems to handle novel recipes that may involve ingredient combinations not previously encountered within the network.

In our exploration, we will delve into the development of a recipe generation pipeline, the design of heuristic-based search algorithms, comparing the results to know which works better, and an evaluation of the generalizability of FOON for a wide array of recipes.

FUNCTIONAL OBJECT-ORIENTED NETWORK

Functional Object-Oriented Network (FOON), a graphical knowledge representation that captures high-level concepts related to human manipulation for service robots. FOON is a bipartite network comprising two fundamental types of nodes: object nodes and motion nodes. Object nodes represent physical entities, typically objects that are relevant to a particular task or action. These nodes can describe various attributes or states of objects, such as their size, shape, location, and condition. In our case of cooking they can represent vegetables, kitchen utensils, containers and their states such as vegetables being in cut state and etc. Motion nodes represent specific actions or motions that are part of a manipulation task. These nodes capture the movements, gestures, or manipulations required to interact with object nodes. For example, in the context of stirring tea, a motion node could represent the action of "stir". The connection between objects and actions is represented by edges, signifying both the relationship between objects and the order of actions within the network. A critical component of FOON is the concept of a "functional unit," where object nodes and motion nodes combine to describe a single action.

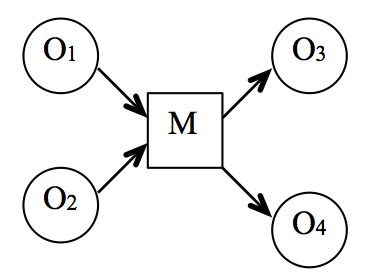


Fig. 2: A basic functional unit with two input nodes and two output nodes connected by an intermediary single motion node.

*A. Creating a FOON*

To create a FOON, we define a critical concept known as a "functional unit." A functional unit represents an individual action in an activity and serves as a unit of description for the state changes of objects before and after execution. Input object nodes detail the required states of objects essential for a specific task, while output object nodes depict the outcomes of these inputs. In some cases, the execution of an action may not lead to changes in all input object states, resulting in a lesser number of output object nodes than inputs.

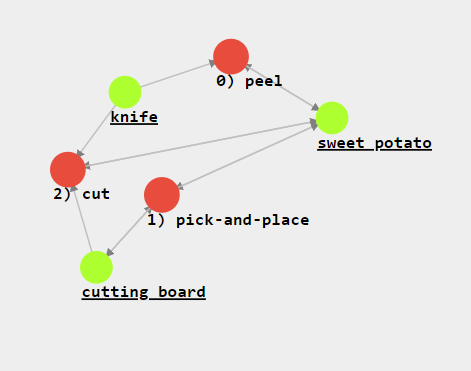


Fig. 3: A subgraph showing how to get sweet potato ready to use after going through the functional units which cut and peel the sweet potato.

The process of generating FOON involves annotating video demonstrations, converting them into the FOON graph structure, and documenting the actions, objects, and functional units leading to the creation of a particular meal or product. Each FOON that represents a single activity is referred to as a "subgraph." Subgraphs consist of functional units organized in a sequence to describe the states of objects before and after each action, the corresponding time-stamps in the source demonstration, and the manipulated objects. As of now, this annotation process is largely manual. However, on-going efforts aim to explore semi-automatic approaches [23]. To construct a "universal FOON," two or more subgraphs are merged by performing a union operation on all functional units. This process effectively eliminates duplicate functional units.

*B. Integrating Weights into FOON*

In previous iterations of FOON [5], [6], all motions were assigned equal weights within the network. This essentially implied that all manipulations could be executed by any robot without considering the robot's likelihood of successfully performing each action. However, this approach lacks realism, as different robots are designed with varying capabilities, precision, and dexterity.

To address this limitation, we introduce the concept of "weights" into FOON. These weights serve as indicators of the difficulty level associated with each manipulation, and they are based on the robot's success rate in executing a given action. Success rates are assigned as percentages to the motion nodes of each functional unit and consider factors like the robot's physical capabilities, past experiences, and ability to execute actions. Tools or objects used in manipulation also influence these values.

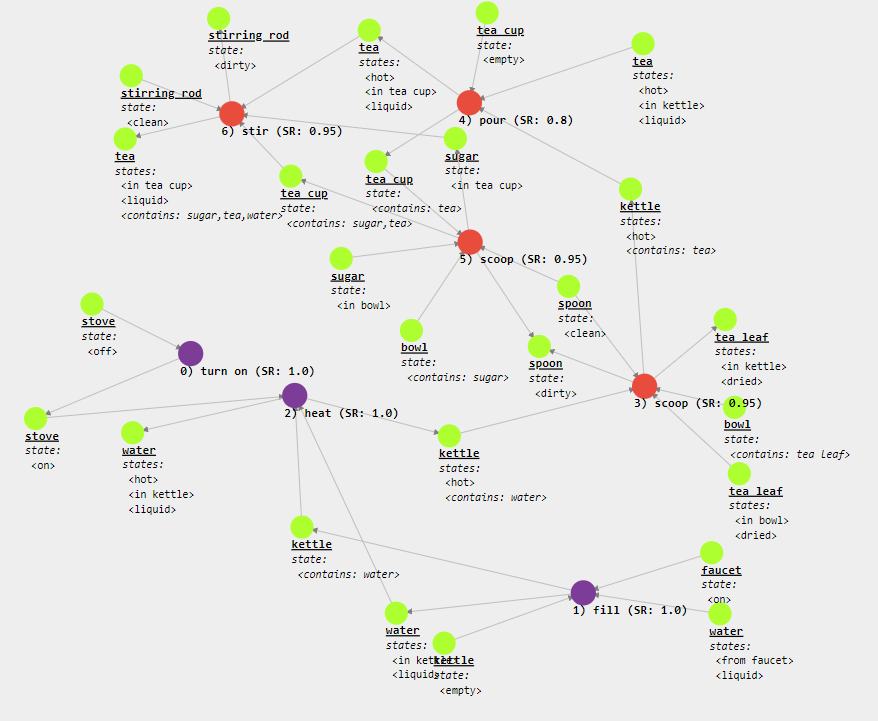


Fig. 4: A weighted FOON depicting the preparation of tea. Here the SR is the success rate of robot in performing each motion mentioned.

*C. Task Trees*

Task trees are a vital aspect of FOON and describe the sequential execution of actions within an activity. These trees are composed of functional units in sequence, each detailing the changes in object states before and after the action. The timestamps of action occurrences and the manipulated objects are also recorded within a subgraph. This knowledge structure simplifies the understanding of the workflow within a specific task. When combining multiple task trees, you can visualize the entire process for a complex task or activity.

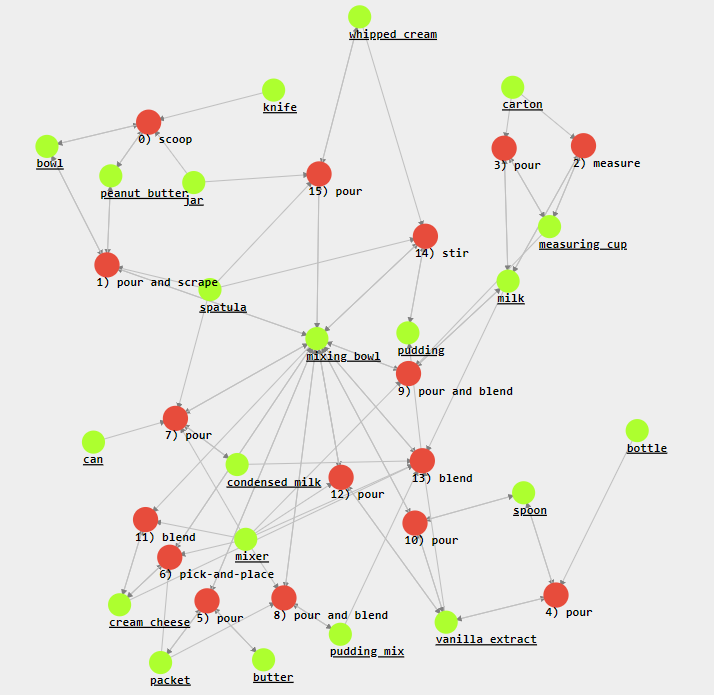


Fig. 5: An example of a Task Tree for getting whipped cream by using the input nodes which are readily available, using them and performing actions on them as functional units to create the final product.

FOON isn't just a knowledge representation; it's a tool for robots to solve problems. Imagine the robot has a goal. It uses a process called task tree retrieval to find a bunch of actions (functional units) it needs to reach that goal. This collection of actions is what we call a task tree. Unlike a standard collection of actions, a task tree doesn't stick to one example from a human - it learns from various sources to create a fresh plan.

To find the right actions, the robot needs a list of stuff in its environment (like what's in the kitchen). It checks which actions it can do based on what's available. This searching process is inspired by common graph-based methods, like the depth-first search (DFS) and breadth-first search (BFS). Starting from the goal, it looks for possible actions step by step.

Task Trees are retrieved using many methods and researchers have covered these methods in more detail in [5]. However, these methods don't include the new idea of using different search algorithms we are implementing on FOON in this work. So, we'll explore a different way to make task trees by considering how well actions work this way, we look at all the possible combinations of actions that could solve the problem.