Task tree retrieval using Iterative deepening and Heuristic search algorithms

sreenija sunkireddy

ABSTRACT

This paper gives a brief about FOON (Functional Object-Oriented Network) which is a graph structure used for knowledge retrieval for the sake of robots to do the tasks. Knowledge retrieval is a crucial part of human robot communication as it serves as a map/guide for the robot to do the tasks in a specified manner just as humans have the manual. This paper also explains the video annotations and different search algorithms to retrieve the task tree. The mentioned algorithms are Iterative deepening algorithm and Heuristic algorithm in which Heuristic has two types of candidate item selection based on success rate of the motion and no of input objects including no of ingredients. This paper also discusses the time and space complexities of each algorithm.

1.INTRODUCTION

While assigning tasks like cooking, robots have the problem of identifying objects of different shapes and sizes and while making a recipe it has the difficulty in executing the functional units in an orderly manner[4] and to solve these issues robots need a step-by-step instruction manual just like humans follow to do a certain task. Here the manual for robots is a task tree in which is a connected graph that guides the robot how to perform the tasks.

FOON (Functional Object-Oriented Network) is a knowledge representation for robots[1][2]. This representation is a graph that contains information about how objects can be used in certain tasks or manipulations to do things. A robot can do the tasks by following a symbolic plan and that plan is obtained from a task tree through a knowledge retrieval process from the FOON. To evaluate the correctness of the task tree, we compare it with the conventional task tree forms such as manuals and recipes.

2.VIDEO ANNOTATION AND FOON CREATION

A Functional unit is simply put as one single action performed. Functional unit is represented using nodes. A node is a data connecting point in a communicating network. There are two types of nodes:

1.Object node

2.Motion node

Object node:

Object node is any item that is used in the cooking/manipulation procedure. An object has a state type and state label to describe the state or condition it is observed in. Object node has two types of nodes a) input node – have outgoing edge b) output node – have incoming edge.

Motion node:

Motion node is a point where any type of action happens whether it is manipulation or non-manipulation.

A functional unit mainly has 3 parts: input nodes, motion node, output nodes and their states. And these nodes are represented within //. You need to list input objects BEFORE the motion line and output objects AFTER the motion line. Each component is tab-separated (i.e., \t). If the object name has multiple parts, separate it using space.

Format of a functional unit is given as:

//
Input node 1
State of input node 1
Input node 2
State of input node 2
Motion node 1
Output node 1
State of output node 1
Output node 2
State of output node 2
//

- The object line is formatted as:

O\t<NAME OF OBJECT>\t<0 or 1, describing if this object is moving or not>

- The state line is formatted as:

S\t<OBJECT'S STATE>\t {LIST OF INGREDIENTS, COMMA SEPARATED}

The state is identified by the phrase or word within the parentheses. Certain items can be containers. Therefore, we need to list the items contained within them by identifying them in curly brackets

.

- The motion line is formatted as:

M\t<NAME OF MOTION>\t<STARTING TIME>\t<ENDING TIME>

We add time frames while mentioning a motion object to know the corresponding time the motion has happened.

A recipe has multiple functional units and this collection of functional units is called a sub-graph. Merging two or more subgraphs together creates a universal foon and the duplicate functional units are automatically removed during the creation of universal foon.

Example of a functional unit is:

//
O onion 0
S whole
S on {cutting board}
O knife 1
S clean
M chop <assumed> <assumed>
O onion 0
S chopped

S on {cutting board}
O knife 1
S dirty
//

Example of a functional unit which depicts a motion node connected to input ,output nodes:

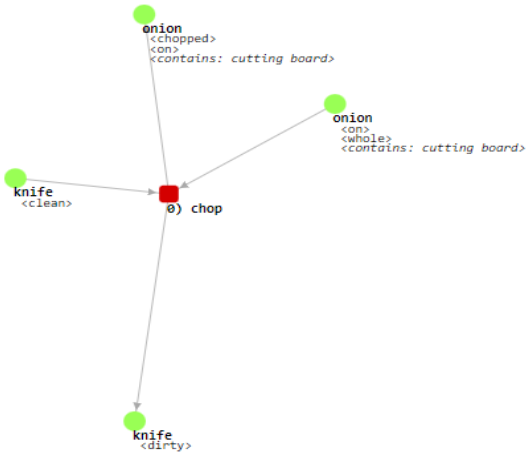

Figure 0: This figure represents a functional unit where onion is chopped on a chopping board.[8]

Green nodes represent object nodes
Input nodes have outgoing edges and output nodes have incoming edges.
Red nodes represent motion nodes

Example of a subgraph:

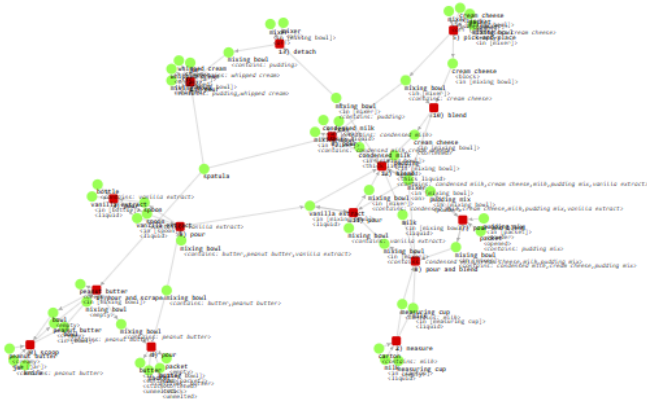

Figure 1: This figure represents the graph of a whole recipe[6] of whipped cream and we call it a sub graph.

To perform these actions a robot needs to retrieve a task tree and A task tree is nothing but a sequence of actions or tasks to achieve a goal and the procedure to obtain a task sequence for execution is called task tree retrieval (in other words we can also say that its knowledge retrieval). Task tree is a task plan that comes from multiple recipes i.e., it is extracted from universal foon. Input of the task tree retrieval is a list of items in the environment (example: kitchen) and a goal node, output of the task tree retrieval is a list of functional units (task tree) that arrives at goal nodes.

Example of universal foon:

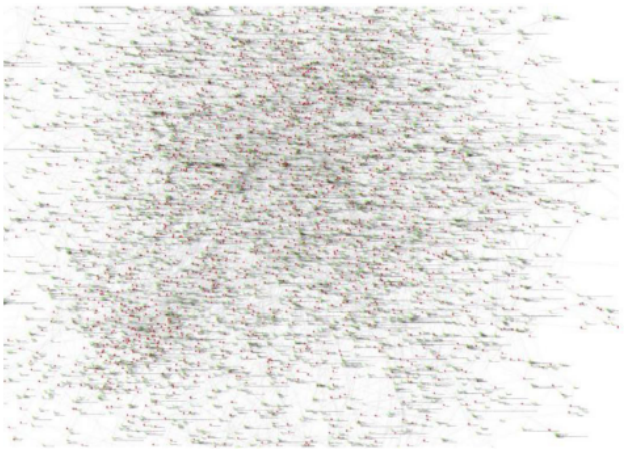

Figure 3:This figure shows an example of Universal foon

3.METHODOLOGY

Our algorithms take input-nodes and goal-nodes as inputs and start the search process and the process ends only after all the nodes are being visited. We create a universal foon by merging all graphs i.e., foon.txt. and

- A goal_node.json file in which each of the goal nodes will have its own task tree.
- Kitchen.json file for what already exists in the kitchen.
- Motion.txt file which indicates success rate for every motion.
- Utensils.txt file which shows all available utensils in the kitchen
- FOON_class,Object_class and motion_class are some of the other classes.

Here we have implemented two kinds of algorithms which are iterative deepening search algorithm and heuristic algorithm.

Iterative deepening search algorithm:

In iterative deepening search we define a depth bound and iterate depth bound and perform DFS (depth first search) within the depth bound.
We define depth limit and the search starts from depth limit 0 and perform DFS for the limit 0 and we iterate from depth limit 0 to max depth limit and perform DFS for each depth limit and we stop when the goal node is found.
Depth of the solution d
Branching factor at each non-leaf node b
Completeness yes
Optimality/Admissibility - no if weights are different

Time complexity - O(b^d) which is a little worse
Space complexity - O(b*d)
IDS has both advantages of BFS and DFS.

Example of a IDS:

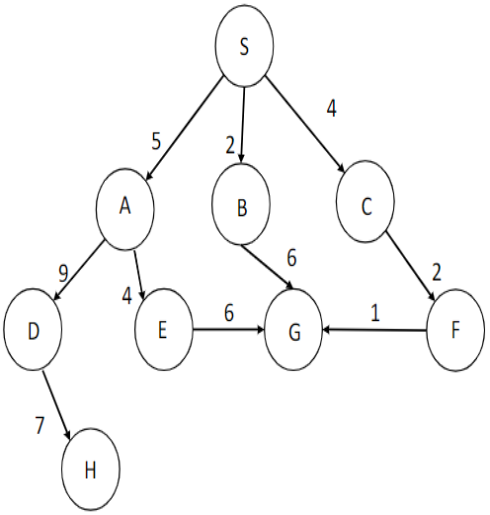

Figure 4:This figure is an example graph to show the IDS[7]

Expanding	Frontier nodes
	S
S	A, B, C
A	B, C
B	C
C	
S	A, B, C
A	D, E, B, C
D	E, B, C
E	B, C
B	G, C
G	C

Figure5: This figure shows traversal order of iterative deepening search

Methodology for Iterative deepening search:

- Start at the goal node G
- Iterate through depth limit d from 0 to max
  Visit all children nodes at depth d
- After visiting all nodes in the graph iterating through depth 0 and to depth max, stop the search.

Heuristic Search:

Heuristic search is a form of informed search that uses domain specific information in some way. All domain knowledge used in the search is encoded in the heuristic function h and h(n) is >=0 for all nodes. In our case, to select an appropriate motion we are using the motion success rate heuristic.

Our search starts from the goal node and ends when all the nodes are being visited.

Greedy Breadth First search :

In greedy breadth first search algorithm ,the path is selected based on the evaluation function using different heuristic functions.If the selected heuristic is optimal,the search will be performed efficiently

- Start at the goal node G
- Check input nodes of the goal node and search for them.
- If the nodes are already in the kitchen, do not explore them
- If the nodes are not in the kitchen, explore them and that selection of motion depends on heuristic

Heuristic 1:

- Heuristic 1 lets the system select the motion which has the highest motion success rate.
- Here h(n) is defined as success rate of the motion
- If the success rate is >=90 then the robot can do the task.
- If the success rate is <=10 then the robot is not able to perform the action.

Heuristic 2:

- When it comes to the same object, heuristic 2 lets the system select a functional unit which has fewer ingredients and visit it.
- Here h(n) is defined as no of input objects in the functional unit including the ingredients in the utensils.
- In this heuristic instead of selecting a candidate, we need to select heuristic.

Stop the search when all nodes of the graph are being visited.

Heuristic search is an example of a weak method for AI because of the limited way that domain specific information is used to solve a problem.

4.EXPERIMENTAL DISCUSSION

The task trees of the three algorithms either have the same number of functional units or if not closer.Here we can see that all task trees of sweet potato,ice and macaroni have the same no of nodes.Greek salad and whipped cream gave different no of motion nodes.

Goal nodes Methods	Whipped cream	Sweet potato	Macaroni	Ice	Greek salad
Iterative deepening	10	3	7	1	31
Heuristic1	15	3	7	1	30
Heuristic 2	15	3	7	1	30

Figure6 : This table represents the no of motions in each of the retrieved task tree of three algorithms.

5.REFERENCES

[1] D. Paulius, Y. Huang, R. Milton, W. D. Buchanan, J. Sam, and Y. Sun. Functional Object-oriented network for manipulation learning. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pages 2655–2662. IEEE, 2016.

[2] Shaogang Ren and Yu Sun. Human-object-object-interaction affor- dance. In *Workshop on Robot Vision*, 2013.

[3] Y. Sun, S. Ren, and Y. Lin. Object-object interaction affordance learning. *Robotics and Autonomous Systems*, 2013

[4] David Paulius and Yu Sun. A survey of knowledge representation in service robotics. *Robotics and Autonomous Systems*, 118:13–30, 2019.

[5] J.J. Gibson. The theory of affordances. In R. Shaw and J. Bransford, editors, *Perceiving, Acting and Knowing*. Hillsdale, NJ: Erlbaum, 1977..

[6] David Paulius, Ahmad B Jelodar, and Yu Sun. Functional Object- Oriented Network: Construction & Expansion. In *2018 IEEE Interna- tional Conference on Robotics and Automation (ICRA)*, pages 5935– 5941. IEEE, 2018.

[7] Y. Lin and Y. Sun. Robot grasp planning based on demonstrated grasp strategies, *Intl. Journal of Robotics* D. Paulius, K. S. P. Dong, and Y. Sun. Task Planning with a Weighted Functional Object-Oriented Network. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021.

[8] FOON Website: Graph Viewer and Videos. http://www.foonets.com. Accessed: 2021-05-31.