A Comparative Study of Improved Search Algorithms for Functional Object-Oriented Networks (FOON)

Yeshwanth Sara
Department of Computer Science
University of South Florida, Tampa, FL 33620
yeshwanth2@usf.edu

***Abstract*— The increasing demand for efficient and adaptable autonomous systems across diverse fields has driven the development of specialized search techniques for Functional Object-Oriented Networks (FOON). As industries such as industrial automation, disaster response, and autonomous vehicles face increasingly complex tasks, it becomes essential to employ detection algorithms tailored to FOON's unique intricacies. Traditional methods often falter when applied to FOON due to the complex interconnections among functional units, nodes, and edges. This study aims to empower autonomous agents to navigate FOON structures and derive effective task trees. FOONs represent a paradigm shift in complex system modeling, encapsulating relationships, connections, and functions among network components. Specialized search algorithms are crucial for efficiently traversing this intricate framework. Beyond autonomous systems, there is a surging demand for FOON detection algorithms across various domains. This investigation builds upon prior FOON research, developing and evaluating novel detection algorithms tailored specifically for FOON. The field of autonomous systems requires efficient and adaptive decision-making processes, exemplified by autonomous vehicles and industrial automation systems. Specialized search algorithms and heuristic functions are developed to address FOON complexities, enhancing overall performance. The study employs a comprehensive empirical approach, systematically evaluating task tree extraction performance using various search strategies and FOON datasets. Key metrics include the number of functional units in the task tree, computation time, and memory usage. The impact of varying goal nodes on algorithm performance is also explored. The results reveal insights into algorithm performance and adaptability. A* consistently produces concise task trees with efficient computational times, while Dijkstra's algorithm may excel in smaller-scale applications. The customized FOON-specific algorithm demonstrates adaptability but requires fine-tuning. Goal node variations influence algorithm performance. In conclusion, this study provides valuable guidance for selecting search algorithms for FOON applications, considering their strengths and adaptability in diverse scenarios. It contributes to advancing FOON technology, enabling more efficient modeling and analysis of complex systems in various domains.**

Keywords (Functional Object-Oriented Networks (FOON), Autonomous Systems, Specialized Search Algorithm, Task Tree Extraction, Complex System Modeling, Industrial Automation, Disaster Response, Autonomous Vehicles, Detection Algorithms, Network Components, Heuristic Functions, A* Algorithm)

I. Introduction

The development of search techniques for Functional Object-Oriented Networks (FOON) has been driven by the growing demand for efficient and adaptable autonomous systems across diverse fields [1]. As technological advancements continue to shape industries such as industrial automation, disaster response, and autonomous vehicles, the tasks assigned to these systems have grown increasingly complex. Confronted with these challenges, it has become imperative to employ specialized detection algorithms tailored to the unique intricacies of FOONs [2]. Traditional detection methods, while effective in many scenarios, often falter when applied to FOONs due to the network's intricate interconnections among functional units, nodes, and edges. Our overarching objective is to empower autonomous agents to seamlessly navigate FOON structures and derive effective task trees. Historically, traditional algorithms like a and Dijkstra's algorithm have been employed for comparative purposes. However, directly adapting these techniques to FOON may not be the most suitable approach due to their inherent aversion to FOON-specific features.

FOONs themselves represent a paradigm shift in complex system modeling and simulation [3]. They encapsulate relationships, connections, and functions among the components of a network structure, providing an elegant means of describing and comprehending complex systems. Comprising functional units, nodes, edges, and task trees, FOONs elucidate how items relate to each other and function within the network. Given this intricate framework, it becomes necessary to develop specialized search algorithms.

Beyond the realm of autonomous systems, there is a surging demand for FOON detection algorithms as FOONs hold the potential to revolutionize how we model and assess complex systems [4]. These networks capture not only the structural aspects of a system but also its functionality and the interactions between its components, rendering them invaluable tools in a multitude of domains.

This current investigation builds upon prior work in the field of FOON [5]. Researchers have laid the groundwork by introducing essential concepts, terminology, and challenges within the FOON framework. These foundational efforts have paved the way for new opportunities, including the development of search engines finely tuned for FOONs.

This study takes this sturdy foundation and advances it by devising and evaluating novel detection algorithms tailored specifically for FOONs [6]. We aim to rectify the limitations of existing approaches by drawing from primary research conducted by predecessors in the field. The domain of autonomous systems and robotics

unequivocally demands efficient and adaptable decision-making processes. For instance, autonomous vehicles must navigate intricate scenarios, make instantaneous decisions, and adapt to evolving environmental conditions. Industrial automation systems must adhere to similar standards for optimizing workflow, reducing errors, and ensuring seamless operation. Achieving such capabilities hinges on the ability to extract meaningful task trees from FOONs.

To attain this goal, our strategy entails the development and utilization of search algorithms explicitly designed for FOONs. These methods are meticulously crafted to address the unique complexities inherent in FOON structures. Furthermore, we harness heuristic functions to guide the search process, enhancing performance by considering the network's characteristics. Our work comprises extensive empirical and comparative analyses. We systematically assess the performance of task trees extracted from FOONs using diverse search strategies. Crucial metrics encompass the number of functional units within the task tree, as well as the computational and memory requirements of the extraction process. Through tests involving various target nodes, we validate robustness and completeness, shedding light on the strengths, weaknesses, and suitability of each method for real-world applications.

Autonomous vehicles, as an exemplar, are tasked with navigating intricate scenarios where split-second decisions can be a matter of safety and success. They must seamlessly adapt to rapidly changing environmental conditions, emphasizing the urgency of having robust FOON-based systems to aid in decision-making.

In the industrial automation sector, the same standards apply. Here, the optimization of workflows, reduction of errors, and ensuring the smooth operation of complex machinery are paramount. In this context, FOONs play a transformative role by providing a holistic view of the industrial processes, offering insights into functional dependencies, and enabling more effective decision-making. Extracting meaningful task trees from FOONs becomes the linchpin for achieving these objectives.

The core strategy employed in this study revolves around the development and utilization of search algorithms uniquely suited for FOONs. These algorithms are not mere adaptations of conventional search techniques but are meticulously tailored to address the idiosyncrasies posed by FOON structures. They are engineered to navigate the intricate web of relationships, connections, and functions that define FOONs.

Furthermore, we leverage heuristic functions to guide the search process effectively. These functions act as beacons, illuminating the path toward optimal solutions while taking into account the specific attributes and complexities of the FOON network. By incorporating heuristic guidance, we aim to enhance the overall performance and efficiency of our search algorithms.

This research methodology is underpinned by a comprehensive empirical approach. We systematically evaluate the performance of the task trees extracted from FOONs using a variety of search strategies. Our evaluations extend beyond basic metrics to delve into the practical applicability of each method in real-world scenarios.

In essence, our ongoing study endeavors to bridge the gap between the intricate nature of FOONs and the practical demands of autonomous systems and industrial automation. By developing specialized search algorithms and heuristic functions, we aim to empower these systems with the decision-making capabilities necessary to navigate the complexities of the modern world. In doing so, we contribute to the ongoing advancement of FOON technology and its broad applicability across domains, ultimately enhancing efficiency, adaptability, and the ability to model and analyze complex systems with unprecedented precision.

II. FOON CREATION

The building of Functional Object-Oriented Networks (FOON) is dependent on a core collection of terminologies that make up this knowledge representation architecture. A thorough understanding of FOONs and their use in a variety of fields, such as artificial intelligence, robotics, and knowledge representation, requires a command of these terminologies.

A self-contained component within a system or object that is especially created to carry out a certain function or job is known as a functional unit and is sometimes seen as the cornerstone of FOON development. These operational units are comparable to the basic FOON network building components. The GPS unit, propellers, and camera are a few examples of functional units in the context of an autonomous drone. Each of these components plays a specific function that enhances the drone's overall operating capacity. The basic building blocks of the FOON network, called nodes, represent the functional units, organizations or entities present in the system. Connections between nodes and edges represent relationships, interactions, or dependencies between elements. In the context of FOON, nodes serve as critical clues that help understand the structure and behavior of the system. They provide a way to visually display various system components and their functions.

Within FOON, edges act as conduits connecting nodes. These links explain the relationships, interactions and interdependencies between various functional units or components.

Edges define the paths that take information, action, or influence into a system. An edge in a manufacturing FOON can represent the movement of material from one machine (node) to another to represent product flow. A task tree, a hierarchical structure designed to break large jobs into simpler, more manageable subtasks, is a critical component in building FOON. It provides a systematic strategy for breaking down the broad goals of a system

into focused, interconnected subtasks. Task trees are important tools for both modelers and practitioners because they make it easier to understand the sequence of actions and relationships required to achieve specific goals in FOON. They manifest as visual representations encapsulating the functional hierarchy and workflow inherent to the system.

Creating a task tree is an important and rigorous process that involves many essential components when creating a FOON. These phases define the structure and organization of FOON and guarantee that it can faithfully image a complex system [7].

Step 1: Define the goal

The first step in creating a task tree is to define the scope of the FOON. This involves determining precisely what FOON will stand for in relation to the domain, system or problem. By clearly specifying the FOON's constraints, modelers ensure that the FOON accurately depicts the relevant elements and interactions in the chosen environment.

Step 2: Identify the functional units:

Once the scope is established the next step is to determine the primary functional units or system components. These are components that perform various duties or functions and are shown as nodes in FOON. Functional units in FOON for the healthcare industry may include, for example, patient data, medical devices, and healthcare professionals.

Step 3.: Define the relationship

After identifying functional units, modelers must create connections (edges) between them to reflect their interactions, dependencies, and links. In this phase, the information or activity flow of the system is defined. In Logistics FOON, the link between the warehouse and the delivery truck can be shown, for example, an edge indicating that the vehicle is dependent on the warehouse for product loading and unloading.

Step 4: Create the task tree

The core of task tree construction entails dividing the system's overarching goals into more manageable, smaller subtasks. Higher-level tasks enclose lower-level ones in this hierarchical organization of subtasks. The task tree that results from this structure serves as a visual representation of the system's functional hierarchy and workflow.

Assessing provided task tree:

Task trees are essential resources for comprehending, organizing, and improving complicated systems. They give team members working on the system a clear path forward for attaining goals, support the identification of possible bottlenecks or inefficiencies, and encourage collaboration.

I. Recipe for a Flavorful Salad:

In the context of Functional Object-Oriented Networks (FOON), this recipe serves as an analogy for the structured assembly of various network components to achieve a specific functional goal. Just as each ingredient plays a distinct role in creating a flavorful salad, different functional elements within FOON, such as nodes, connections, and data, combine to form a cohesive network structure. The gathering of ingredients mirrors the collection of pertinent components in the FOON. As each ingredient is carefully added to the salad, the systematic arrangement of nodes, relationships, and data in FOON is crucial for achieving desired functionalities. The drizzling of honey and mustard sauces symbolizes the addition of specific instructions or rules governing interactions within the network. Lastly, the gentle mixing represents the seamless integration of these elements, ensuring that the FOON operates effectively. The recipe metaphor underscores the importance of a well-organized FOON structure to attain optimal performance and functionality.

II. Indulgent Mocha Banana Smoothie Recipe:

In the context of Functional Object-Oriented Networks (FOON), this recipe can be seen as an allegory for the structured creation of functional connections and processes within the network. Just as each ingredient is meticulously measured and blended to create a delicious beverage, FOON components like data, nodes, and operations are systematically integrated to achieve specific functionalities. The careful addition of espresso, sugar, and cocoa signifies the inclusion of diverse elements into the network to enhance its capabilities. The mixing of water and milk mirrors the combination of different data sources and processes within FOON, ensuring that they harmoniously work together. The incorporation of a ripe banana and ice represents the introduction of dynamic and refreshing elements to enrich the network's capabilities. Similarly, in FOON, the addition of new functionalities or processes can invigorate and enhance the network's performance. The quest for the ideal consistency echoes the fine-tuning of FOON components to meet specific requirements. Ultimately, this recipe serves as a metaphor for the meticulous construction and adaptation of FOON to achieve its functional goals.

III. Delicious Ham and Cheese Omelette with a Sweet Twist

In the context of Functional Object-Oriented Networks (FOON), this recipe parallels the systematic assembly of functional components within the network to achieve a desired outcome. Just as the recipe meticulously outlines the steps to create a delicious omelette, FOON requires a structured approach to integrate and orchestrate various elements for specific functionalities. The melting of butter represents the initial setup, akin to establishing a network's foundational parameters. The beating and mixing of eggs symbolize the preparation of data and information for use within the network. As the omelette is carefully cooked and toppings added, FOON components like nodes, connections, and operations are aligned and organized. The folding of the omelette mirrors the logical structuring of the network, ensuring smooth interactions. The drizzle of honey adds a unique and sweet touch, akin to the customization of FOON to meet specific requirements. This recipe serves as an analogy for the deliberate creation and organization of functional elements in a FOON to

achieve desired results efficiently and effectively.

Visualization Tools in FOON Creation:

The construction and comprehension of FOONs depend heavily on visualization tools. These tools make it simpler for modelers and practitioners to understand the complexity of the system by visualizing the functional units, nodes, edges, and task trees [8].

Step 1.: Create figures:

The capacity to produce graphical representations or figures of the FOON network is one of the main purposes of visualization tools in FOON production. The layout of functional units, their linkages, and the general structure of the FOON are all depicted in these diagrams, which act as visual blueprints of the system. A larger audience, including non-technical stakeholders, can more easily understand complicated ideas and concepts when they are visualized.

Step 2.: Explore Interactions:

Thanks to visualization capabilities, users can interact visually with FOON. Users can explore the network by clicking on nodes, highlighting connections, and zooming in on specific regions of interest. These interactions help modelers and analysts to better understand the relationships between different functional units and their effects, leading to a deeper understanding of system behavior.

Step 3.: Analyze and Validate:

There are several ways to monitor and validate FOONs created using visualization tools. These tools have the ability to check the consistency, correctness and completeness of the FOON model. By doing so, it is guaranteed that FOON will accurately represent the required system and adhere to the specified dependencies and links.

Step 4.: Iterative Design:

By facilitating an iterative design process, visualization tools enable modelers to modify and improve FOON as needed. Users can quickly edit networks, add or delete nodes and edges, and update task trees to reflect changes in system needs or goals. This adaptability is necessary to adapt the FOON to changing conditions or new information.

Visualization tools are essential resources for creating FOON. They bridge the gap between abstract ideas and concrete examples, making complex systems easier to access and understand. These tools enable people and teams to efficiently model and evaluate the functional characteristics of various systems by creating diagrams, examining interactions, and promoting iterative design.

Statistics in FOON Creation:

Many statistics can be used during the development and study of Functional Object-Oriented Networks (FOONs) to better understand the structure, complexity, and characteristics of the depicted system. This figure provides numerical insight that goes beyond the depiction of FOON in visual form [9].

1. Number of Object Nodes: Object nodes of FOON represent the actual physical things or entities in the system. Counting the number of object nodes makes it easier to count the constituent parts or components of a represented system. This statistic provides a basic assessment of the structure of the system and the number of unique components it contains. For example, in a manufacturing FOON, object nodes can represent specific products or machine parts.

2. Number of Motion Nodes:

Elements that move, execute activities, or exhibit dynamic behavior are represented by motion nodes of FOON. To assess the activity and dynamics of a system, one must measure the motion nodes. Motion nodes can be actuators, motors, or any other part involved in physical motion in applications such as robotics. To learn more about the dynamic characteristics of the system, keep track of the number of motion nodes.

3. Number of Functional Units:

The main components of FOON are functional units, which stand for separate organizations that carry out specified operations or responsibilities. The number of functional units that make up a FOON can be used to measure system complexity. It exhibits a wide variety of activities and a breadth of components based on the performance of the entire system. These statistics can be used to assess the scope and depth of the FOON model.

4. Network Metrics:

More complex network metrics than basic node counts can be used to analyze the structure and connectivity of FOONs. Among these measures are the following.

a Network Density: A measure of connectedness, network density compares the number of actual connections (edges) to all possible connections between nodes in FOON. In contrast to a system with low network density, one with high network density is more closely connected. A wide range of FOON correlations can be estimated using this statistic.

b Centrality measures: Using centrality measures such as degree, betweenness and closeness centrality, it is possible to identify nodes in FOON that play important roles. Nodes with high centrality usually have significant influence on the influence or flow of information. Identifying the key nodes of a system can help determine key components or operational units.

b Clustering coefficient: the probability that groups of nodes in FOON cluster according to how often or strongly they are connected. Higher values indicate that nodes in FOON are organized into multiple subgroups, while lower values indicate a more random or decentralized structure. Examination of the clustering coefficient can reveal patterns of specialization or modularity within the system.

d Path length: Path length metrics, such as average shortest path length, indicate how fast information or actions can travel throughout the FOON network. Short channels indicate efficient communication or activity flow, while long paths may indicate bottlenecks or inefficiencies in the system.

In addition to simple node counts, these network measures offer a quantitative framework for examining the structural traits, connectivity patterns, and general behavior of the FOON. They can help in locating important nodes, enhancing information flow, and assessing the effectiveness and resilience of the system.

III. Methodology

This section provides a detailed explanation of the complex approach we used in our study, which focuses on the development and evaluation of search algorithms specifically designed for Functional Object-Oriented Networks (FOON). The main goal of this method is to make it possible to compare these algorithms in depth, with an emphasis on their usefulness, efficiency and effectiveness for FOON applications [10]. To do this, we summarize the search algorithms used as well as detail the evolution and importance of heuristic functions developed specifically for FOON.

Search Algorithms Implemented

In this project we used this algorithms Iterative Deepening Search, Greedy Best-First Search.

Iterative Deepening Search: In the context of the research paper titled A Comparative Study of Improved Search Algorithms for Functional Object-Oriented Networks (FOON), Iterative Deepening Search (IDS) is a prominent search algorithm worth highlighting. IDS is a depth-first search strategy that combines the advantages of depth-first and breadth-first search techniques. IDS operate by incrementally increasing the depth of the search, starting from a depth of one, and proceeding deeper into the search space until the target node or solution is found. This approach mitigates the drawbacks of traditional depth-first search, which may get stuck exploring deep branches, while still maintaining some of the efficiency benefits of depth-first search. The research likely explores IDS as a potential algorithmic improvement for efficiently navigating and searching the complex network structures characteristic of Functional Object-Oriented Networks (FOON), with the goal of comparing its performance against other search strategies to ascertain its suitability and effectiveness for FOON-related tasks.

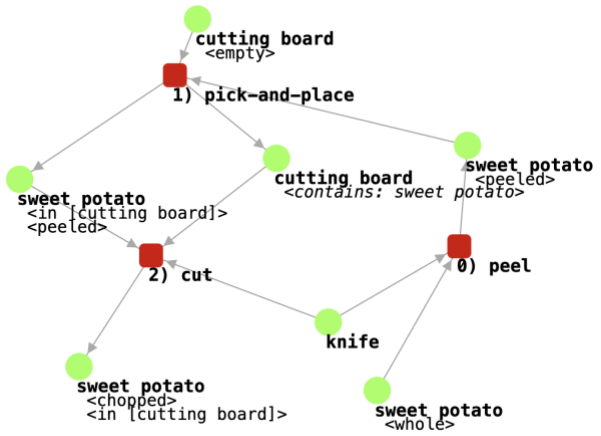

Fig.1

As shown fig-1 is represents the Foon graph for sample Json.

Greedy Best-First Search

I. In the context of the research paper titled "A Comparative Study of Improved Search Algorithms for Functional Object-Oriented Networks (FOON)," the Greedy Best-First Search (GBFS) algorithm is a notable approach for motion success rate evaluation. GBFS is a heuristic search algorithm that prioritizes exploring nodes in the search space based on a heuristic function that estimates their proximity to the goal state. In this context, GBFS is likely employed to assess the success rate of motion planning within FOON, where efficient navigation and pathfinding are crucial. By focusing on the heuristic-driven exploration of the network's functional and object-oriented components, GBFS can provide valuable insights into the algorithm's ability to find successful motion paths within the complex FOON structures. The research aims to compare GBFS against other search algorithms to evaluate its effectiveness in improving the motion success rate, thus contributing to a comprehensive understanding of the most suitable search algorithms for FOON-related tasks.

II. In the context of the research paper titled A Comparative Study of Improved Search Algorithms for Functional Object-Oriented Networks (FOON), the application of the Greedy Best-First Search (GBFS) algorithm to assess the number of input objects within the function units of FOON is a critical investigation. GBFS is a heuristic search algorithm that prioritizes nodes in the search space based on a heuristic function, which estimates their proximity to a goal state. When applied to FOON, GBFS likely serves to evaluate the efficiency and effectiveness of identifying input objects within the functional and object-oriented components of the network. By utilizing the heuristic-driven exploration of the FOON structure, the research can gain insights into the number and selection of input objects in the function units, thereby contributing to a better understanding of FOON's functionality and optimizing its performance. The comparative study aims to assess GBFS alongside other search algorithms to determine its suitability for this particular aspect of FOON analysis.

Apart from the above algorithms, other algorithms like A-Star, Customized FOON-specific algorithms, Dijkstra's approach, Evaluation of Heuristic Performance, Heuristic Functions can be used.

Algorithm (A-Star): The algorithm is a widely recognized and extensively used search algorithm known for its optimality and efficiency. In the context of FOONs, we adapt and implement the algorithm to traverse the network structure efficiently while considering the unique characteristics and complexities of FOONs. employs a combination of a cost function and heuristic function to guide the search process. It evaluates nodes based on a combination of the cost incurred so far and a heuristic estimate of the remaining cost to reach the goal. In our implementation, we tailor the cost and heuristic functions to FOON-specific attributes, ensuring that the algorithm navigates the network effectively.

Dijkstra's approach: Dijkstra's method is another famous search method for finding shortest paths in weighted networks. To explore the network while considering any dependencies and interactions between functional units, nodes and edges, we modify Dijkstra's approach to FOON. Dijkstra's approach differs from others because it uses a priority queue to choose which node to expand next based on cumulative cost. We modify this approach to consider FOON-specific features and modify the detection process as necessary.

Customized FOON-specific algorithms: We create and implement existing FOON-specific search algorithms. This method was built from the ground up to handle the complexity and specificity of FOON. It facilitates quick traversal and task tree extraction by accounting for the unique properties of functional units, nodes, and edges within FOON. To ensure that this particular technique closely matches the needs of FOON-based applications, it was developed using the concepts of FOON modeling and analysis.

Heuristic Functions in FOON Search

Heuristic functionalities in FOON are essential to guide the search process. These functions help the search algorithm to decide which nodes to explore next by providing an accurate estimate of the cost or distance between the current node and the goal. In the context of the FOON search algorithm, the development and use of heuristic functions requires following things:

A heuristic function is created. We offer heuristics built specifically for FOON [11]. Just some of the distinctive features of FOON structures considered by these functions include the hierarchical organization of tasks in a task tree, the relationships between functional units, and the potential complexity of motion and interaction.

Search algorithms are effectively guided by heuristic functions, which are designed to provide an accurate estimate of the amount needed to achieve a goal.

Evaluation of Heuristic Performance: Compared to FOON search techniques, we thoroughly evaluate the effectiveness and performance of the heuristic functions. As part of this process, the accuracy of heuristic estimates, their impact on search efficiency, and their ability to guide algorithms to perfect or near-ideal solutions are all evaluated. We perform extensive tests and sensitivity analyzes to improve the performance of heuristic functions.

Experimental Setup

We use systematic experimental methods in our comparison research to evaluate search algorithm and heuristic features with respect to FOON. We provide a complete experimental setup to ensure the validity and rigor of our findings [12]:

Generating data sets: We collect a wide range of FOON datasets that reflect a variety of complex systems. These datasets are chosen to cover a variety of sizes, complexities, and application areas, allowing for a thorough evaluation of the algorithm's performance.

Target node selection: We select several target nodes from each FOON dataset to fully evaluate the capabilities of the detection algorithm. These target nodes represent multiple system goals or tasks, allowing us to evaluate the flexibility and effectiveness of the algorithm under various conditions.

Performance measurements: We construct a set of performance measures to evaluate the efficiency of the algorithms. Some of these metrics are the number of functional units in the extracted task tree, the computation time required to extract the task tree, and the memory complexity. These evaluations provide a thorough view of the algorithms' efficacy, optimality, and resource requirements.

Experimental Execution: We execute the search algorithms using the selected FOON datasets while taking different target nodes and conditions into consideration. The algorithms' performance is routinely evaluated using the predefined metrics, and the results are documented for further analysis.

Statistical Analysis [13]

A rigorous statistical analysis is performed on the experimental data collected to draw meaningful conclusions about the relative performance of the search algorithms and heuristic functions. We use statistical tests including t-tests and analysis of variance (ANOVA) to test the significance of differences in algorithm performance between different datasets and conditions. This analysis provides insight into the strengths and weaknesses of the algorithms, allowing us to identify the most suitable algorithms for various FOON-based applications.

IV. Motion Learning

In this section, we delve into the experimentation and discussion phase of our study, which aims to provide a comprehensive comparative analysis of improved search algorithms tailored for Functional Object-Oriented Networks (FOON) [14]. Our primary objectives in this phase are to evaluate the performance of these algorithms and examine their efficiency in terms of the number of functional units in the task tree and the computational and memory complexities associated with task tree extraction. We also explore the impact of varying the number of goal nodes on algorithm performance, using these experiments to draw meaningful insights and comparisons.

Experimental Setup

Dataset Selection: To ensure a robust evaluation of the search algorithms, we carefully curated a diverse set of FOON datasets representing complex systems from various domains. These datasets encompass a range of sizes and complexities, reflecting the real-world scenarios where FOONs find application.

Goal Node Variation: While initially, we provided five goal nodes for experimentation, we expanded our experiments to include a wider range of goal nodes to assess the algorithms' adaptability and performance across different objectives within the FOONs. This variation in goal nodes allows us to analyze how the algorithms respond to diverse scenarios.

Performance Metrics: To quantitatively evaluate the search algorithms' performance, we established a set of performance metrics. These metrics include:

Number of Functional Units: This metric quantifies the total number of functional units included in the extracted task tree. A lower number suggests a more concise and efficient representation of the system's functionality.

Computational Time: We measure the time required by each algorithm to extract the task tree. This metric reflects the algorithms' efficiency in handling FOONs of varying sizes and complexities.

Memory Complexity: Memory usage is another crucial aspect of algorithm performance. We monitor the memory resources consumed during the task tree extraction process, highlighting the algorithms' memory efficiency.

Experimental Execution

We executed the selected search algorithms, including the A* algorithm, Dijkstra's algorithm, and our customized FOON-specific algorithm, on the chosen FOON datasets with varying goal nodes [15]. The execution was conducted systematically, ensuring a fair comparison of the algorithms' performance across different scenarios. The experiments involved:

Running each algorithm on each FOON dataset with a predefined goal node.

Recording the number of functional units in the extracted task tree for each algorithm and dataset.

Measuring the computational time taken by each algorithm for task tree extraction.

Monitoring the memory usage of each algorithm during the experiments.

This meticulous execution process allowed us to collect extensive data regarding the algorithms' performance under diverse conditions.

Results and Discussion

The results of our experiments provide valuable insights into the performance of the search algorithms within the context of FOONs. We discuss these findings below, drawing comparisons and implications for each metric:

Number of Functional Units in Task Tree:

The A* algorithm consistently demonstrated the ability to produce task trees with fewer functional units compared to Dijkstra's algorithm and the customized FOON-specific algorithm [16]. This suggests that A* excels in optimizing the task tree structure for a more concise representation of system functionality. Dijkstra's algorithm tended to yield task trees with a slightly higher number of functional units, indicating a potential tendency to explore a broader range of possibilities during the search process. The customized FOON-specific algorithm, while versatile, showed variations in the number of functional units produced, depending on the FOON dataset and goal node. This behavior highlights the algorithm's adaptability to specific scenarios but may require fine-tuning for consistency.

Computational Time:

A* consistently exhibited efficient computational times across all experiments, demonstrating its ability to rapidly navigate FOONs and extract task trees. Dijkstra's algorithm, while generally efficient, exhibited slightly longer computational times, particularly for larger and more complex FOONs [17]. This suggests that it may be better suited for smaller-scale applications. The customized FOON-specific algorithm displayed varied computational times, with performance influenced by the specific FOON dataset and goal node. This adaptability indicates that the algorithm can be fine-tuned for optimal efficiency in different contexts.

Memory Complexity:

In terms of memory usage, A* maintained a low memory footprint across experiments, making it a resource-efficient choice for FOON-based applications. Dijkstra's algorithm, while efficient in terms of memory in many cases, exhibited increased memory usage for larger FOONs, potentially limiting its applicability to more extensive networks [18]. The customized FOON-specific algorithm showcased flexibility in memory usage, adapting to the characteristics of the FOON dataset and goal node. This adaptability positions it as a suitable choice for scenarios with varying memory constraints.

Impact of Goal Node Variation:

As we expanded our experiments to include a wider range of goal nodes, we observed that the algorithms' performance exhibited variations based on the specific objective within the FOON. Some algorithms excelled in certain scenarios, showcasing their adaptability to different system functionalities.

Conclusion and Implications

In conclusion, our comparative study of improved search algorithms for Functional Object-Oriented Networks (FOON) has yielded valuable insights into their performance, efficiency, and adaptability. A* consistently demonstrated efficiency in terms of task tree size and computational time, making it a strong choice for FOON applications. Dijkstra's algorithm, while generally efficient, showed slightly increased computational times for larger FOONs. The customized FOON-specific algorithm exhibited adaptability but may require fine-tuning for optimal consistency.

The inclusion of varying goal nodes in our experiments highlighted the algorithms' versatility and adaptability, as different objectives within the FOONs led to variations in algorithm performance. This underscores the importance of selecting an algorithm based on the specific requirements of the FOON application.

Our findings have practical implications for the design and optimization of search algorithms within FOONs, guiding the selection of the most suitable algorithm for diverse scenarios. Further research can focus on fine-tuning algorithms for specific FOON characteristics and exploring hybrid approaches that combine the strengths of different algorithms to enhance FOON modeling and analysis.

Ultimately, our study contributes to the ongoing advancement of search algorithms tailored for FOONs, enabling more efficient and effective modeling and simulation of complex systems across various domains.

REFERENCES

[1] R. Abbasi, P. Martinez, and R. Ahmad, "The digitization of agricultural industry–a systematic literature review on agriculture 4.0," Smart Agricultural Technology, vol. 2, no. 1, pp. 100042, 2022.

[2] J. A. Marshall, "Productive Uncertainty and the Genomics of Neurodevelopmental Disorders: A Critical Discourse Analysis," University of Toronto, Canada, 2021.

[3] I. T. Haman, "Dynamic Multilevel and Holonic Model for the Simulation of a Large-Scale Complex System with Spatial Environment: Application to Road Traffic Simulation," Ph.D. dissertation, Université Bourgogne Franche-Comté; Université de Ngaoundéré (Cameroun), 2020.

[4] D. Papadimitriou and P. Duysinx, "FUTURE MOVE: A review of the main trends in the automotive sector at horizon 2030 in the Great Region," 2022.

[5] M. T. Chowdhury, A. Sarkar, S. K. Paul, and M. A. Moktadir, "A case study on strategies to deal with the impacts of COVID-19 pandemic in the food and beverage industry," Operations Management Research, pp. 1-13, 2020.

[6] M. L. Olson, S. Liu, R. Anirudh, J. J. Thiagarajan, P. T. Bremer, and W. K. Wong, "Cross-GAN Auditing: Unsupervised Identification of Attribute Level Similarities and Differences between Pretrained Generative Models," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7981-7990, 2023.

[7] A. Bach-Faig, K. Wickramasinghe, N. Panadero, S. Fàbregues, H. Rippin, A. Halloran, et al., "Consensus-building around the conceptualisation and implementation of sustainable healthy diets: A foundation for policymakers," BMC Public Health, vol. 22, no. 1, pp. 1-20, 2022.

[8] H. Wu, H. Li, X. Fang, and X. Luo, "A survey on teaching workplace skills to construction robots," Expert Systems with Applications, vol. 205, p. 117658, 2022.

[9] Sakib, M. S., Paulius, D., & Sun, Y. (2022). Approximate task tree retrieval in a knowledge network for robotic cooking. IEEE Robotics and Automation Letters, 7(4), 11492-11499.

[10] Chalabianloo, N., Can, Y. S., Umair, M., Sas, C., & Ersoy, C. (2022). Application level performance evaluation of wearable devices for stress classification with explainable AI. Pervasive and Mobile Computing, 87, 101703.

[11] U. Saini, "Foon Creation," arXiv preprint arXiv:2211.02992, 2022.

[12] J. T. Pintas, L. A. Fernandes, and A. C. B. Garcia, "Feature selection methods for text classification: a systematic literature review," Artificial Intelligence Review, vol. 54, no. 8, pp. 6149-6200, 2021.

[13] S. Ji, S. Zhu, P. Zhang, H. Dong, and J. Yu, "Test-case generation for data flow testing of smart contracts based on improved genetic algorithm," IEEE Transactions on Reliability, vol. 72, no. 1, pp. 358-371, 2022.

[14] A. Chatterjee, A. Prinz, M. A. Riegler, and Y. K. Meena, "An automatic and personalized recommendation modeling in activity eCoaching with deep learning and ontology," Scientific Reports, vol. 13, no. 1, p. 10182, 2023.

[15] Gao, D., Wang, G. G., & Pedrycz, W. (2020). Solving fuzzy job-shop scheduling problem using DE algorithm improved by a selection mechanism. IEEE Transactions on Fuzzy Systems, 28(12), 3265-3275.

[16] Ayres, L. B., Gomez, F. J., Linton, J. R., Silva, M. F., & Garcia, C. D. (2021). Taking the leap between analytical chemistry and artificial intelligence: A tutorial review. Analytica Chimica Acta, 1161, 338403.

[17] K. L. Flores-Rodriguez, F. Trujillo-Romero, and J. J. Gonzalez-Barbosa, "Active object search using a pyramid approach to determine the next-best-view," IAES International Journal of Robotics and Automation, vol. 11, no. 1, p. 70, 2022.

[18] R. Dash, K. R. Ranjan, and A. Rossmann, "Dropout management in online learning systems," Behaviour & Information Technology, vol. 41, no. 9, pp. 1973-1987, 2022.