University of Málaga

Search and Logic

Luka Kuzyk

2023

Index

1. Search (A-star)

1.1 INTRODUCTION

A-star (A*) algorithm [1] is one of the best known path planning algorithm, that use combination of heuristic searching and searching based on the shortest path. While it is a mathematical and computational concept, the principles behind A* are inspiring efficient decision-making and goal achievement in different field of human life:

- Robotics and Autonomous Vehicles
- Video Games
- Logistics and Route Optimization
- Network Routing
- Map Applications
- Maze Solving
- Medical Imaging

In order to find out the cost of the cheapest solution in the following literature [2] we can find next formula:

$$f(v) = h(v) + g(v)$$

where ***h*(*v*)** is the heuristic distance to the goal, ***g*(*v*)** is the length of the path from the initial state to the current state, and ***f(v)*** is estimated cost of the best solution trough *v*.

To calculate the A-star algorithm [3], we need to follow the following steps :

Step 1: Problem Definition: Defining the problem, including the state space, initial state, goal state, and the cost of moving from one state to another.

Step 2: State Representation: Each state in the state space is represented as a node *v*.

Step 3: Heuristic Function: Defining the heuristic function *h(v),* which evaluate the cost from current state *v* to the goal state.

Step 4: Calculation: Initial state is the first one from which calculation starts. It required to find out all *f(v)* with next nodes that is directly connected to current node. If there are more than 1, the lowest value of *f(v)* is the shortest path. If there is only 1 node – calculate its *f(v)* and follow to next node until current node will not be the goal state or a node with more than 1 connection to next nodes.

1.2 APPLYING OF ALGORITHM

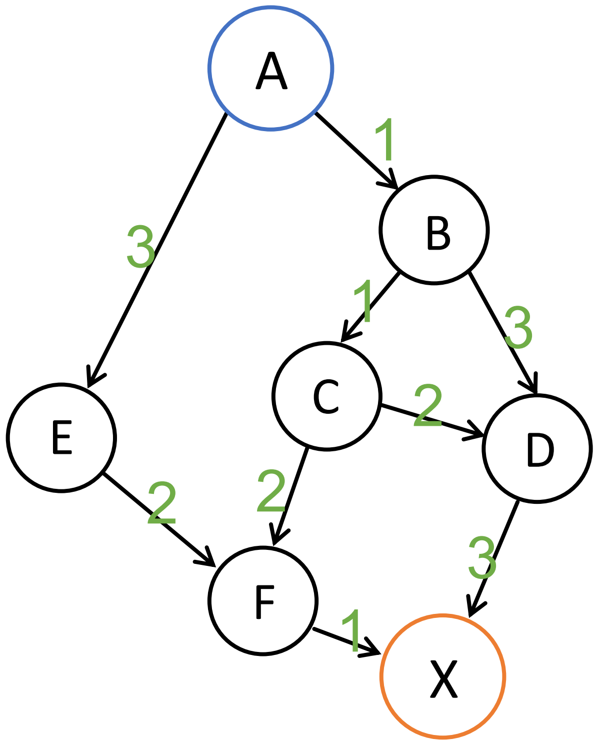

FIGURE 1 TREE OF NODES

NODE	$h(v)$
B	4
C	3
D	2
E	3
F	1
X	0

TABLE 1 NODE'S HEURISTIC COST

From the figure 1, we can see tree structure of nodes, where each node corresponds to a specific state, and the tree grows as the algorithm shows potential paths. Node A represent initial state, and X - goal state. The green numbers above the arrows mean *g(v)* - length between nodes. Table 1 represents heuristic value to each node.

Step 1

Firstly, from the Tree of nodes and table of heuristic value we can see that from initial state A we can go to B and E nodes by following calculation:

A → B: *f(B) = h(B) + g(B) = 4 + 1 =* ***5***

A → E: *f(E) = h(E) + g(E) = 3 + 3 =* ***6***

Since, f(B) < f (E) – the path from node B will be the shortest.

Path: A → B

Step 2

From node B, we can move on to nodes C and D using the following calculations:

A → B → C: *f(C) = h(C) + g(C) = 3 + (1+1) =* ***5***

A → B → D: *f(D) = h(D) + g(D) = 2 + (1+3) =* ***6***

Since, f(C) < f (D) – the path from node C will be the shortest.

Path: A → B → C

Step 3

From node C, we can move on to nodes D and F using the following calculations:

A → B → C → F: *f(F) = h(F) + g(F) = 1 + (1+1+2) =* ***5***

A → B → C → D: *f(D) = h(D) + g(D) = 2 + (1+1+2) =* ***6***

Since, f(F) < f (D) – the path from node B will be the shortest.

Path: A → B → C → F

Step 4

Node F is connected to only one node, so we calculate the path until we reach the goal or the nearest node:

A → B → C → F → **X**: *f(X) = h(X) + g(X) = 0 + (1+1+2+1) =* ***5***

Path: A → B → C → F → X is the most optimal path among all possible ones with a cost of 5.

Results

Also, if we calculate another possible path, we will be able to compare all paths by their cost:

1) **Path: A → B → C → F → X = 5**
2) **Path: A → E → F → X = 6**
3) **Path: A → B → C → D → X = 7**
4) **Path: A → B → D → X = 7**

Therefore, this calculation also confirms that the first path is the most optimal because it is the shortest among the others.

2. Propositional logic

2.1 INTRODUCTION

Propositional logic [4], also known as syntactic logic or statement logic, is a system of mathematical logic used to represent and manipulate propositions truth values Propositions are declarative statements that can be true or false but are not both of these.

The key elements of propositional logic include:

1. **Variables***:* *R*epresent propositions and can be either true or false.

2. **Logical Connectives:** Represent ways to combine or modify propositions. The basic logical connectives include:

   - **Conjunction (∧):** Represents "and." Conjunction of two propositions is true only when both propositions are true.
   - **Disjunction (∨):** Represents "or." Disjunction of two propositions is true when at least one of the propositions is true.
   - **Negation (¬):** Represents "not." Negates the truth value of a proposition.

3. **Laws of Noncontradiction and Excluded Middle:**

   - **Noncontradiction:** A proposition cannot be both true and false simultaneously.
   - **Excluded Middle:** A proposition must be either true or false.

Propositional reasoning allows to formally structure the reasoning processes with variables and logical connectives. It provides a basis for making logical inferences based on relationships between propositions.

2.2 APPLYING OF RESOLUTION ALGORITHM

From literature [5], we can discover how to apply this algorithm:

Knowledge Base:

1. Car 1 is a convertible. => Convertible (Car1)
2. Car 2 is a sedan. => Sedan (Car2)
3. Car 1 is an electric car. => Electric (Car1)
4. Car 2 is a gasoline-powered car. => Gasoline (Car2)

Conclusion (Inference): Is it true that there exists a car that is both a convertible and electric?

Step 1

Convert data form sentences to Conjunctive Normal Form:

1. *Convertible(Car1)*
2. *Sedan(Car2)*
3. *Electric(Car1)*
4. *Gasoline(Car2)*
5. *¬Convertible(Car1) - (Negation of 1)*
6. *¬Sedan(Car2) - (Negation of 2)*
7. *¬Electric(Car1) - (Negation of 3)*
8. *¬Gasoline(Car2) - (Negation of 4)*

Step 2

Now let's apply resolution:

1. By combining 1 and 5 we obtain: *Electric (Car1)*
2. By combining 3 and 7 we obtain: *Convertible (Car1)*

Step 3

The resolution of sentences 1 and 5 leads to the simplified sentence Electric(Car1).

Results

The resolution algorithm simplifies the knowledge base, and the obtained sentence Electric(Car1) implies the existence of an electric car but, it does not explicitly mention whether the car is also convertible. Thus, it is not possible to assume on the basis of a given Knowledge Base that the vehicle is reversible and electric.

3. First-order logic

3.1 Introduction

First-order logic (FOL) [6] is a formal language used for expressing relationships and making inferences about objects and their properties. It is a powerful tool in artificial intelligence, mathematics, and philosophy. In FOL, the basic building blocks include objects, relations, and functions. From

Main components of First-Order Logic [5]:

- **Domain**: The domain is the set of objects or domain elements that the logical statements refer to.
- **Relations**: Relations are sets of tuples of objects that are related to each other, where tuples represent ordered sequences of objects.
- **Functions:** Functions map elements from one set to another. In FOL, functions must be total, meaning they produce an output for every input. An 'invisible' object may be introduced to handle partial functions and convert them into total functions.

Quantifiers and operator:

- ∀ - represents the universal quantifier (for all).
- ∃ - represents the existential quantifier (there exists).
- → - represents the implication (if...then).
- ∧ - represents the logical AND.
- ≠ - represents the not equal operator.

3.2 Applying of algorithm

Sentence:	**Description**	**First-Order Logic:**
All cars have an owner.	For every object *x* that is a car, there exists an object *y* such that *y* is the owner of *x*.	$\forall x$ Car(x) → $\exists y$ OwnerOf(y,x)
Some cars are electric	There exists an object *x* that is both a car and electric.	$\exists x$ [Car(x) ∧ Electric(x)]
No car is both red and green	For every pair of objects *x* and *y* that are both cars, if *x* is red and *y* is green, then *x* is not the same as *y*.	$\forall x \forall y$[Car(x) ∧ Car(y) ∧ Red(x) ∧ Green(y) → $x \neq y$]
If a car has a sunroof, then it is a luxury car.	For every object *x* that is a car and has a sunroof, *x* is a luxury car.	$\forall x$ [Car(x) ∧ HasSunroof(x) → LuxuryCar(x)]

TABLE 2 TRANSLATING OF 4 SENTECES TO FIRSR-ORDER LOGIC

4. Conclusion

In this comprehensive work, we delved into three key areas of logical reasoning: the A* algorithm in search, propositional logic, and first-order logic in problem solving. Each of these algorithm offers valuable tools for problem-solving and decision-making in different contexts.

The A* algorithm, a powerful path-planning tool, was demonstrated in the context of heuristic searching. The step-by-step application of A* was showed through a tree structure and heuristic cost calculations, emphasizing its efficiency in finding optimal paths. Since search A* algorithm heavily depends on heuristics It is important to note it does not produce the shortest path always.

Propositional logic, was introduced as a system for representing and reasoning about truth values of propositions. The resolution algorithm was then applied to a knowledge base about cars, showcasing how logical inference can be employed to derive new information from existing propositions.

First-order logic has demonstrated its flexibility in expressing relations and properties of objects. With the help of quantifiers and operators, we formalized statements about cars and their attributes. The ability of the logic system to work with domains, relations, and functions positions it as a powerful tool in various fields, including artificial intelligence and mathematics.

At the end, I would like to express my gratitude to my teacher Enrique Domínguez Merino for the material he provided and to all the authors of the work mentioned and used in this report.

5. Bibliography

[1]. Duchoň, F., Babinec, A., Kajan, M., Beňo, P., Florek, M., Fico, T., & Jurišica, L. (2014). Path planning with modified a star algorithm for a mobile robot. Procedia engineering, 96, 59-69.

[2]. Domínguez Merino, E., López Rubio, E. Search. Department of Computer Languages and Computer Science. University of Málaga, Spain.

[3]. Yao, J., Lin, C., Xie, X., Wang, A. J., & Hung, C. C. (2010, April). Path planning for virtual human motion using improved A* star algorithm. In 2010 Seventh international conference on information technology: new generations (pp. 1154-1158). IEEE.

[4]. Büning, H. K., & Lettmann, T. (1999). Propositional logic: deduction and algorithms (Vol. 48). Cambridge University Press.

[5]. Domínguez Merino, E., López Rubio, E. First-order logic. Department of Computer Languages and Computer Science. University of Málaga, Spain.

[6]. Barwise, J. (1977). An introduction to first-order logic. In Studies in Logic and the Foundations of Mathematics (Vol. 90, pp. 5-46). Elsevier.