Middle East Technical University Department Of Computer Engineering

# CENG424

LOGIC FOR CS

Fall 2024-2025

Term Project

Name Surname: Alkım Doğan
Student ID: 2521482

## 1 Pseudocode & Formalization

**IMPORTANT:** My algorithm first gives how to represent relations and some axioms above the pseudocode. In pseudocode, my algorithm is focused on how to make resolution. My algorithm is the answer of the question of *Which order of resolution should we choose to solve this problem?* . I make use of depth first search to answer this question since traverce of the automata (graph) is needed for the maximum weigth given the output string

Below is my representation for the edges

```
t( state_from, state_to, output, weight )
```

Below is my axiom for transitivity property of the edges

$$\forall q_i \forall q_j \forall q_k \forall s_i \forall s_j \forall w_i \forall w_j t(q_i, q_j, s_i, w_i) \wedge t(q_j, q_k, s_j, w_j) \rightarrow t(q_i, q_k, s_i s_j, w_i + w_j)$$

Below is for representing the highest so far

```
highest_weight(x)
```

Below is for finding the highest. I made use of mathematics axiom to compare the weights of the paths. I deliberately did not find a way to compare each path but instead I update the maximum weight as I traverse the graph(automata).

$$\forall qi \forall qj \forall s \forall w \forall x t(q_i, q_j, s, w) \wedge highest\_weight(x) \wedge (w > x) \rightarrow highest\_weight(w)$$

**IMPORTANT:** I could have also used some rules for checking each edge. This would be as the following.

$\forall q_i \forall q_j \forall q_k \forall s_1 \forall s_2 \forall w_1 \forall w_2 t(q_i, q_j, s_1, w_1) \wedge edge(q_j, q_k, s_2, w_2) \wedge prefix(s_1 s_2, input\_string) \rightarrow t(q_i, q_k, s_1 s_2, w_1 + w_2)$

The semantics of the above rule is the fact that you should choose that edge if and only if the concatenation of the string you gathered so far and the string of the current edge is the substring of the input string, then you should continue depth first search from this edge. However, doing this operation for checking each edge as well as comparing the weights, would make resolution 5 or 6 times longer ( 2 resolution for each edge and assume graph as average of 3 edges per vertex, making $2 \times 3 = 6$ ). This means it would take something about 500-600 steps, which is meaningless and impossible to deal with in a real life scenerio without an automated tools as mentioned in the 3rd part. Additionally, we somehow need to check prefix of a string. We can also do this in resolution by defining some set of rules that require resolution. Therefore, I made an optimistic assumption about the resolution algorithm where we do not have to deal with each if check inside for loop.

Below is my pseudocode. The **Algorthm 1** refers to the main function. There are 2 different functions.

**Algorithm 1** Highest Weighted Path

```
Require: Input Automata N, Input String S

    // Initializing the set of premises $\Delta$
    $\Delta \leftarrow \{\}$

    INSERT($\Delta, AutomataEdges$)

    INSERT($\Delta, MathematicAxioms$)

    INSERT($\Delta, TransivityPremise$)

    $max\_weight \leftarrow 0$

    // Start Resolution in a Depth First Search Manner with Respect to the Edges
    DEPTH FIRST SEARCH($N, \Delta, N.start\_state, 0, max\_weight, "", S$)

    Return $max\_weight$
```

**Algorithm 2** Depth First Search

```
Require: Automata N, Premises Δ, Current State q, Current Weight w, Max Weight
  max_weight, Current String curS, Input String S

  // Update the max_weight to check for the maximum value
  if curS == S and weight > max_weight then
    $max\_weight \leftarrow weight$ // Do this step by resolving
  end if

  for each edge E of the current state e do
    // If check not to Traverse unnecessary paths
    if curS + E.output is a prefix of S then

      // Update the path by resolving with Transitive Axiom.
      // Resolve the current path with the current edge
      $resolvent \leftarrow$ ResolveTransitive($(N.startState, q, curS, w), E$)
      insert($\Delta, resolvent$)

      // Recursive Call to go one step Further in DFS
      // I concatenate the character coming from the edge
      // I also added the weight of the edge to the current weight
      Depth First Search($N, \Delta, E.q, w + E.weigth, max_weight, curS + E.output, S$)
    end if
  end for
```

## 2 Solution Using My Approach

Let me first Convert representation into **CNF**. I will make use of **INSEADO** for this purpose.

For transitivity premise,

- **I**: $\forall q_i \forall q_j \forall q_k \forall s_i \forall s_j \forall w_i \forall w_j \neg t(q_i, q_j, s_i, w_i) \vee \neg t(q_j, q_k, s_j, w_j) \vee t(q_i, q_k, s_i s_j, w_i + w_j)$
- **N**: $\forall q_i \forall q_j \forall q_k \forall s_i \forall s_j \forall w_i \forall w_j \neg t(q_i, q_j, s_i, w_i) \vee \neg t(q_j, q_k, s_j, w_j) \vee t(q_i, q_k, s_i s_j, w_i + w_j)$
- **S**: $\forall q_i \forall q_j \forall q_k \forall s_i \forall s_j \forall w_i \forall w_j \neg t(q_i, q_j, s_i, w_i) \vee \neg t(q_j, q_k, s_j, w_j) \vee t(q_i, q_k, s_i s_j, w_i + w_j)$
- **E**: $\forall q_i \forall q_j \forall q_k \forall s_i \forall s_j \forall w_i \forall w_j \neg t(q_i, q_j, s_i, w_i) \vee \neg t(q_j, q_k, s_j, w_j) \vee t(q_i, q_k, s_i s_j, w_i + w_j)$
- **A**: $\neg t(q_i, q_j, s_i, w_i) \vee \neg t(q_j, q_k, s_j, w_j) \vee t(q_i, q_k, s_i s_j, w_i + w_j)$
- **D**: $\neg t(q_i, q_j, s_i, w_i) \vee \neg t(q_j, q_k, s_j, w_j) \vee t(q_i, q_k, s_i s_j, w_i + w_j)$
- **O**: $\{\neg t(q_i, q_j, s_i, w_i), \neg t(q_j, q_k, s_j, w_j), t(q_i, q_k, s_i s_j, w_i + w_j)\}$

For finding the current highest,

- **I**: $\forall qi \forall qj \forall s \forall w \forall x \neg t(q_i, q_j, s, w) \vee \neg highest\_weight(x) \vee \neg(w > x) \vee highest\_weight(w)$
- **N**: $\forall qi \forall qj \forall s \forall w \forall x \neg t(q_i, q_j, s, w) \vee \neg highest\_weight(x) \vee \neg(w > x) \vee highest\_weight(w)$
- **S**: $\forall qi \forall qj \forall s \forall w \forall x \neg t(q_i, q_j, s, w) \vee \neg highest\_weight(x) \vee \neg(w > x) \vee highest\_weight(w)$
- **E**: $\forall qi \forall qj \forall s \forall w \forall x \neg t(q_i, q_j, s, w) \vee \neg highest\_weight(x) \vee \neg(w > x) \vee highest\_weight(w)$
- **A**: $\neg t(q_i, q_j, s, w) \vee \neg highest\_weight(x) \vee \neg(w > x) \vee highest\_weight(w)$
- **D**: $\neg t(q_i, q_j, s, w) \vee \neg highest\_weight(x) \vee \neg(w > x) \vee highest\_weight(w)$
- **D**: $\{\neg t(q_i, q_j, s, w), \neg highest\_weight(x), \neg(w > x), highest\_weight(w)\}$

**IMPORTANT:** $h(w) = highest\_weight(w)$. I will move on with this notation since it fits into the table below much better.

| Step | Clause | Reason |
|---|---|---|
| 1 | $\{t(q_0, q_1, a, 4)\}$ | Premise |
| 2 | $\{t(q_0, q_2, b, 2)\}$ | Premise |
| 3 | $\{t(q_0, q_3, b, 4)\}$ | Premise |
| 4 | $\{t(q_0, q_3, a, 3)\}$ | Premise |
| 5 | $\{t(q_1, q_1, b, 4)\}$ | Premise |
| 6 | $\{t(q_1, q_3, e, 2)\}$ | Premise |
| 7 | $\{t(q_2, q_0, a, 2)\}$ | Premise |
| 8 | $\{t(q_2, q_3, a, 4)\}$ | Premise |
| 9 | $\{t(q_3, q_1, b, 1)\}$ | Premise |
| 10 | $\{t(q_3, q_2, e, 7)\}$ | Premise |
| 11 | Mathematics Axiom (+,<,>) | Premise |
| 12 | $\{\neg t(q_i, q_j, s_i, w_i), \neg t(q_j, q_k, s_j, w_j), t(q_i, q_k, s_i s_j, w_i + w_j)\}$ | Premise |
| 13 | $\{h(0)\}$ | Premise ( No edge gives 0 weight ) |
| 14 | $\{\neg t(q_i, q_j, s, w), \neg h(x), \neg(w > x), h(w)\}$ | Premise |
| 15 | $\{\neg t(q_1, q_k, s_j, w_j), t(q_0, q_k, a s_j, 4 + w_j)\}$ | 1 - 12 $\{q_i \leftarrow q_0, q_j \leftarrow q_1, s_i \leftarrow a, w_i \leftarrow 4\}$ |
| 16 | $\{t(q_0, q_1, ab, 4 + 4)\}$ | 5 - 15 $\{q_k \leftarrow q_1, s_j \leftarrow b, w_j \leftarrow 4\}$ |

| | | |
|---|---|---|
| 17 | $\{t(q_0, q_1, ab, 8)\}$ | 11 - 17 |
| 18 | $\{\neg t(q_1, q_k, s_j, w_j), t(q_0, q_k, abs_j, 8 + w_j)\}$ | 12 - 17 $\{q_i \leftarrow q_0, q_j \leftarrow q_1, s_i \leftarrow ab, w_i \leftarrow 8\}$ |
| 19 | $\{t(q_0, q_3, ab, 8 + 2)\}$ | 6 - 17 $\{q_k \leftarrow q_3, s_j \leftarrow e, w_j \leftarrow 2\}$ |
| 20 | $\{t(q_0, q_3, ab, 10)\}$ | 11- 19 |
| 21 | $\{\neg t(q_3, q_k, s_j, w_j), t(q_0, q_k, abs_j, 10 + w_j)\}$ | 12 - 19 $\{q_i \leftarrow q_0, q_j \leftarrow q_3, s_i \leftarrow ab, w_i \leftarrow 10\}$ |
| 22 | $\{t(q_0, q_1, abb, 10 + 1)\}$ | 9 - 21 $\{q_k \leftarrow 1, s_j \leftarrow b, w_j \leftarrow 1\}$ |
| 23 | $\{t(q_0, q_1, abb, 11)\}$ | 11 - 22 |
| 24 | $\{\neg t(q_i, q_j, s, w), \neg(w > 0), h(w)\}$ | 13-14 $\{x \leftarrow 0\}$ |
| 25 | $\{\neg(11 > 0), h(11)\}$ | 23-34 $\{q_i \leftarrow q_0, q_j \leftarrow q_1, s \leftarrow abb, w \leftarrow 11\}$ |
| 26 | $\{h(11)\}$ | 11 - 25 |
| 27 | $\{\neg t(q_1, q_k, s_j, w_j), t(q_0, q_k, abbs_j, 11 + w_j)\}$ | 12-23 $\{q_i \leftarrow q_0, q_j \leftarrow q_1, s_i \leftarrow abb, w_i \leftarrow 11\}$ |
| 28 | $\{t(q_0, q_3, abb, 11 + 2)\}$ | 6 - 26 $\{q_k \leftarrow 3, s_j \leftarrow e, w_j \leftarrow 2\}$ |
| 29 | $\{t(q_0, q_3, abb, 13)\}$ | 11 - 28 |
| 30 | $\{\neg t(q_i, q_j, s, w), \neg(w > 11), h(w)\}$ | 13-26 $\{x \leftarrow 11\}$ |
| 31 | $\{\neg(13 > 11), h(13)\}$ | 20-30 $\{q_i \leftarrow q_0, q_j \leftarrow q_3, s \leftarrow abb, w \leftarrow 13\}$ |
| 32 | $\{h(13)\}$ | 11 - 31 |
| 33 | $\{\neg t(q_3, q_k, s_j, w_j), t(q_0, q_k, abbs_j, 13 + w_j)\}$ | 12-29 $\{q_i \leftarrow q_0, q_j \leftarrow q_3, s_i \leftarrow abb, w_i \leftarrow 13\}$ |
| 34 | $\{t(q_0, q_2, abb, 13 + 7)\}$ | 10-33 $\{q_k \leftarrow q_2, s_j \leftarrow e, w_j \leftarrow 8\}$ |
| 35 | $\{t(q_0, q_2, abb, 20)\}$ | 11-34 |
| 36 | $\{\neg t(q_i, q_j, s, w), \neg(w > 13), h(w)\}$ | 13-32 $\{x \leftarrow 11\}$ |
| 37 | $\{\neg(20 > 13), h(20)\}$ | 35-36 $\{q_i \leftarrow q_0, q_j \leftarrow q_2, s \leftarrow abb, w \leftarrow 20\}$ |
| 38 | $\{h(20)\}$ | 11 - 38 |
| 39 | $\{\neg t(q_1, q_k, s_j, w_j), t(q_0, q_k, abs_j, 8 + w_j)\}$ | 12- 17 $\{q_i \leftarrow q_0, q_j \leftarrow q_1, s_i \leftarrow ab, w_i \leftarrow 8\}$ |
| 40 | $\{t(q_0, q_1, abb, 8 + 4)\}$ | 5 - 39 $\{q_k \leftarrow q_0, s_j \leftarrow b, w_j \leftarrow 4\}$ |
| 41 | $\{t(q_0, q_1, abb, 12)\}$ | 11-40 |
| 42 | $\{\neg t(q_1, q_k, s_j, w_j), t(q_0, q_k, abbs_j, 12 + w_j)\}$ | 12 - 41 $\{q_i \leftarrow q_0, q_j \leftarrow q_1, s_i \leftarrow abb, w_i \leftarrow 12\}$ |
| 43 | $\{t(q_0, q_3, abb, 12 + 2)\}$ | 6 - 42 $\{q_k \leftarrow q_0, s_j \leftarrow e, w_i \leftarrow 2\}$ |
| 44 | $\{t(q_0, q_3, abb, 14)\}$ | 11 - 43 |
| 45 | $\{\neg t(q_3, q_k, s_j, w_j), t(q_0, q_k, abbs_j, 14 + w_j)\}$ | 12 - 44 $\{q_i \leftarrow q_0, q_j \leftarrow q_3, s_i \leftarrow abb, w_i \leftarrow 14\}$ |
| 46 | $\{t(q_0, q_2, abb, 14 + 7)\}$ | 10 - 45 $\{q_k \leftarrow q_2, s_i \leftarrow e, w_j \leftarrow 7\}$ |
| 47 | $\{t(q_0, q_2, abb, 21)\}$ | 11 - 46 |
| 48 | $\{\neg t(q_i, q_j, s, w), \neg(w > 20), h(w)\}$ | 13-38 $\{x \leftarrow 11\}$ |
| 49 | $\{\neg(21 > 20), h(21)\}$ | 47-48 $\{q_i \leftarrow q_0, q_j \leftarrow q_2, s \leftarrow abb, w \leftarrow 21\}$ |
| 50 | $\{h(21)\}$ | 11 - 49 |
| 51 | $\{t(q_0, q_3, a, 4 + 2)\}$ | 6 - 15 $\{q_j \leftarrow q_3, s_j \leftarrow e, w_j \leftarrow 2\}$ |
| 52 | $\{t(q_0, q_3, a, 6)\}$ | 11 - 51 |
| 53 | $\{\neg t(q_3, q_k, s_j, w_j), t(q_0, q_k, as_j, 6 + w_j)\}$ | 12 - 52 $\{q_i \leftarrow q_0, q_j \leftarrow q_3, s_i \leftarrow a, w_i \leftarrow 6\}$ |
| 54 | $\{t(q_0, q_1, ab, 6 + 1)\}$ | 9 - 53 $\{q_k \leftarrow q_1, s_j \leftarrow b, w_j \leftarrow 1\}$ |
| 55 | $\{t(q_0, q_1, ab, 7)\}$ | 11 - 54 |
| 56 | $\{\neg t(q_1, q_k, s_j, w_j), t(q_0, q_k, abs_j, 7 + w_j)\}$ | 12 - 56 $\{q_i \leftarrow q_0, q_j \leftarrow q_1, s_i \leftarrow ab, w_i \leftarrow 7\}$ |
| 57 | $\{t(q_0, q_3, ab, 7 + 2)\}$ | 6 - 56 $\{q_k \leftarrow q_3, s_j \leftarrow e, w_j \leftarrow 2\}$ |
| 58 | $\{t(q_0, q_3, ab, 9)\}$ | 11 - 57 |
| 59 | $\{\neg t(q_3, q_k, s_j, w_j), t(q_0, q_k, abs_j, 9 + w_j)\}$ | 12 - 58 $\{q_i \leftarrow q_0, q_j \leftarrow q_3, s_i \leftarrow ab, w_i \leftarrow 9\}$ |
| 60 | $\{t(q_0, q_1, abb, 9 + 1)\}$ | 9 - 59 $\{q_k \leftarrow q_1, s_j \leftarrow b, w_j \leftarrow 1\}$ |
| 61 | $\{t(q_0, q_1, abb, 10)\}$ | 11 - 60 $\{q_k \leftarrow q_1, s_j \leftarrow b, w_j \leftarrow 1\}$ |
| 62 | $\{\neg t(q_1, q_k, s_j, w_j), t(q_0, q_k, abs_j, 10 + w_j)\}$ | 12 - 61 $\{q_i \leftarrow q_0, q_j \leftarrow q_1, s_i \leftarrow abb, w_i \leftarrow 10\}$ |
| 63 | $\{t(q_0, q_3, abb, 10 + 2)\}$ | 6 - 62 $\{q_k \leftarrow q_3, s_j \leftarrow e, w_i \leftarrow 2\}$ |
| 64 | $\{t(q_0, q_3, abb, 12)\}$ | 11 - 63 |
| 65 | $\{\neg t(q_1, q_k, s_j, w_j), t(q_0, q_k, abbs_j, 12 + w_j)\}$ | 12 - 64 $\{q_i \leftarrow q_0, q_j \leftarrow q_3, s_i \leftarrow abb, w_i \leftarrow 12\}$ |
| 66 | $\{t(q_0, q_3, abb, 12 + 7)\}$ | 12 - 64 $\{q_k \leftarrow q_3, s_j \leftarrow e, w_j \leftarrow 7\}$ |
| 67 | $\{t(q_0, q_3, abb, 19)\}$ | 11 - 66 |

| | | |
|---|---|---|
| 68 | $\{\neg t(q_1, q_k, s_j, w_j), t(q_0, q_k, abs_j, 7 + w_j)\}$ | 12 - 55 $\{q_i \leftarrow q_0, q_j \leftarrow q_1, s_i \leftarrow ab, w_i \leftarrow 7\}$ |
| 69 | $\{t(q_0, q_1, abb, 7 + 4)\}$ | 5 - 68 $\{q_k \leftarrow q_1, s_j \leftarrow b, w_j \leftarrow 4\}$ |
| 70 | $\{t(q_0, q_1, abb, 11)\}$ | 11 - 69 |
| 71 | $\{\neg t(q_1, q_k, s_j, w_j), t(q_0, q_k, abbs_j, 11 + w_j)\}$ | 12 - 70 $\{q_i \leftarrow q_0, q_j \leftarrow q_1, s_i \leftarrow abb, w_i \leftarrow 11\}$ |
| 72 | $\{t(q_0, q_3, abb, 11 + 2)\}$ | 6 - 71 $\{q_k \leftarrow q_3, s_j \leftarrow e, w_j \leftarrow 2\}$ |
| 73 | $\{t(q_0, q_3, abb, 13)\}$ | 11 - 72 |
| 74 | $\{\neg t(q_3, q_k, s_j, w_j), t(q_0, q_k, abbs_j, 13 + w_j)\}$ | 12 - 73 $\{q_i \leftarrow q_0, q_j \leftarrow q_3, s_i \leftarrow abb, w_i \leftarrow 13\}$ |
| 75 | $\{t(q_0, q_2, abb, 13 + 7)\}$ | 10 - 74 $\{q_k \leftarrow q_2, s_j \leftarrow eb, w_i \leftarrow 7\}$ |
| 76 | $\{t(q_0, q_2, abb, 20)\}$ | 11 - 75 |
| 77 | $\{\neg t(q_3, q_k, s_j, w_j), t(q_0, q_k, as_j, 3 + w_j)\}$ | 4 - 12 $\{q_i \leftarrow q_0, q_j \leftarrow q_3, s_i \leftarrow a, w_i \leftarrow 3\}$ |
| 78 | $\{t(q_0, q_1, ab, 3 + 1)\}$ | 9 - 77 $\{q_k \leftarrow q_1, s_j \leftarrow b, w_j \leftarrow 1\}$ |
| 79 | $\{t(q_0, q_1, ab, 4)\}$ | 11 - 78 |
| 80 | $\{\neg t(q_1, q_k, s_j, w_j), t(q_0, q_k, abs_j, 4 + w_j)\}$ | 12 - 79 $\{q_i \leftarrow q_0, q_j \leftarrow q_1, s_i \leftarrow ab, w_i \leftarrow 4\}$ |
| 81 | $\{t(q_0, q_3, ab, 4 + 2)\}$ | 6 - 80 $\{q_k \leftarrow q_3, s_j \leftarrow e, w_j \leftarrow 2\}$ |
| 82 | $\{t(q_0, q_3, ab, 6)\}$ | 11 - 81 |
| 83 | $\{\neg t(q_3, q_k, s_j, w_j), t(q_0, q_k, abs_j, 6 + w_j)\}$ | 12 - 82 $\{q_i \leftarrow q_0, q_j \leftarrow q_3, s_i \leftarrow ab, w_i \leftarrow 6\}$ |
| 84 | $\{t(q_0, q_1, abb, 6 + 1)\}$ | 9 - 83 $\{q_k \leftarrow q_1, s_j \leftarrow b, w_i \leftarrow 1\}$ |
| 85 | $\{t(q_0, q_1, abb, 7)\}$ | 11 - 84 |
| 86 | $\{\neg t(q_1, q_k, s_j, w_j), t(q_0, q_k, abb, 7 + w_j)\}$ | 12 - 84 $\{q_i \leftarrow q_0, q_j \leftarrow q_1, s_i \leftarrow abb, w_i \leftarrow 7\}$ |
| 87 | $\{t(q_0, q_3, abb, 7 + 2)\}$ | 6 - 86 $\{q_k \leftarrow q_3, s_i \leftarrow e, w_i \leftarrow 2\}$ |
| 88 | $\{t(q_0, q_3, abb, 9)\}$ | 11 - 87 |
| 89 | $\{\neg t(q_3, q_k, s_j, w_j), t(q_0, q_k, abb, 9 + w_j)\}$ | 12 - 88 $\{q_i \leftarrow q_0, q_j \leftarrow q_3, s_i \leftarrow abb, w_i \leftarrow 9\}$ |
| 90 | $\{t(q_0, q_2, abb, 9 + 7)\}$ | 10 - 89 $\{q_k \leftarrow q_2, s_j \leftarrow e, w_j \leftarrow 7\}$ |
| 91 | $\{t(q_0, q_2, abb, 16)\}$ | 11 - 90 |
| 92 | $\{t(q_0, q_1, abb, 4 + 4)\}$ | 5 - 80 $\{q_k \leftarrow q_1, s_j \leftarrow b, w_j \leftarrow 4\}$ |
| 93 | $\{t(q_0, q_1, abb, 8)\}$ | 11 - 92 |
| 94 | $\{\neg t(q_1, q_k, s_j, w_j), t(q_0, q_k, abb, 8 + w_j)\}$ | 12 - 93 $\{q_i \leftarrow q_0, q_j \leftarrow q_1, s_i \leftarrow abb, w_i \leftarrow 8\}$ |
| 95 | $\{t(q_0, q_3, abb, 8 + 2)\}$ | 6 - 94 $\{q_k \leftarrow q_3, s_j \leftarrow e, w_j \leftarrow 2\}$ |
| 96 | $\{t(q_0, q_3, abb, 10)\}$ | 11 - 95 |
| 97 | $\{\neg t(q_3, q_k, s_j, w_j), t(q_0, q_k, abb, 10 + w_j)\}$ | 12 - 95 $\{q_i \leftarrow q_0, q_j \leftarrow q_3, s_i \leftarrow abb, w_i \leftarrow 10\}$ |
| 98 | $\{t(q_0, q_2, abb, 10 + 7)\}$ | 10 - 98 $\{q_k \leftarrow q_2, s_j \leftarrow e, w_j \leftarrow 7\}$ |
| 99 | $\{t(q_0, q_2, abb, 17)\}$ | 11 - 98 |

Line 50 gives the most recent highest value, which is **21**. Above, my algorithm has traversed the entire tree looking for the string **aab** and updated the highest weight as it traversed. I did the resolution above in a depth first manner since I applied the pseudocode line by line. We can also say that my algorithm partially applies **input restriction** strategy to solve. When I traverse, I only look for the edges and the edges are in the initial set of premises.

## 3 Existing Tools

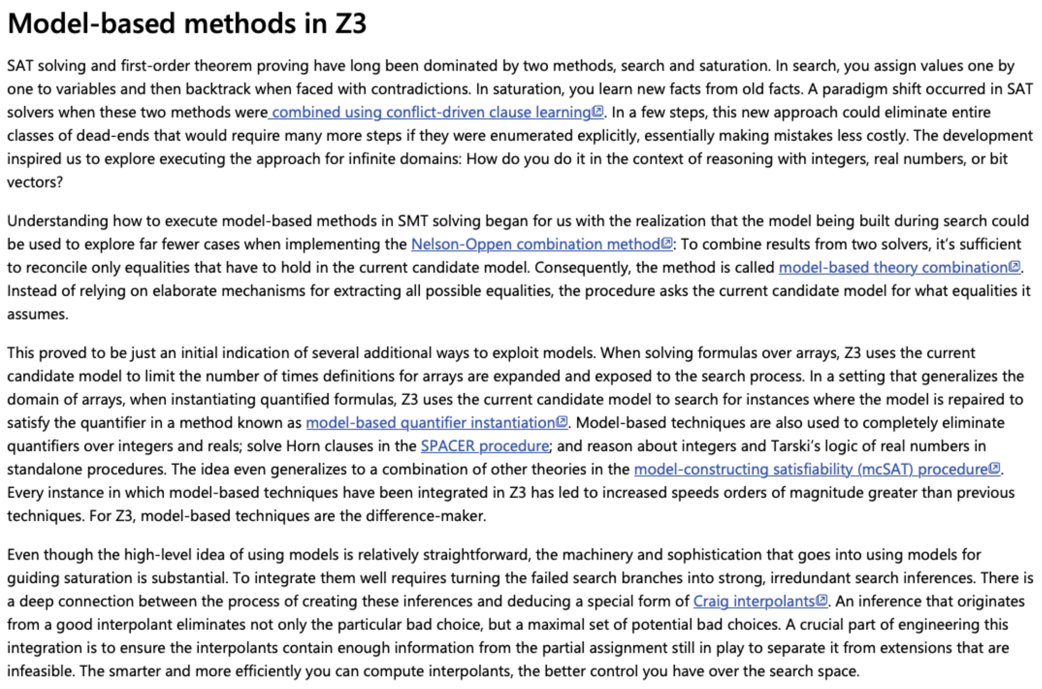
I decided to choose **SMT solver (Satisfiability modulo theories)**. One of the example tool for this solver is called **Z3** by Microsoft. I chose it since this tool already deals with **relational** logic. I was torn between this and prolog but I thought that prolog was a programming language rather than a tool. I **mostly** look at the Wikipedia for my research. Links are Click here for **SMT** and Click here for **Z3**

## 3.1

For SMT solvers, there is no semantic change for the format since both our case and the tools are in relational format. If I chose propositional solvers such as **SAT solvers**, then I would have to convert my project into propositional logic. But I should just adjust my rules in terms of syntax.

## 3.2

**Z3** first handles the inputs such as parsing and so on. Then, it **iteratively** looks for a solution while doing optimizations. I was not able to fully understand the logic behind it since it was very advanced but I found out the fact that **Z3** does not use resolution. This tool leverages SAT-solving techniques like CDCL, and the best part is this tool directly supports maximization and minimization.



**Model-based methods in Z3**

SAT solving and first-order theorem proving have long been dominated by two methods, search and saturation. In search, you assign values one by one to variables and then backtrack when faced with contradictions. In saturation, you learn new facts from old facts. A paradigm shift occurred in SAT solvers when these two methods were combined using conflict-driven clause learning. In a few steps, this new approach could eliminate entire classes of dead-ends that would require many more steps if they were enumerated explicitly, essentially making mistakes less costly. The development inspired us to explore executing the approach for infinite domains: How do you do it in the context of reasoning with integers, real numbers, or bit vectors?

Understanding how to execute model-based methods in SMT solving began for us with the realization that the model being built during search could be used to explore far fewer cases when implementing the Nelson-Oppen combination method: To combine results from two solvers, it's sufficient to reconcile only equalities that have to hold in the current candidate model. Consequently, the method is called model-based theory combination. Instead of relying on elaborate mechanisms for extracting all possible equalities, the procedure asks the current candidate model for what equalities it assumes.

This proved to be just an initial indication of several additional ways to exploit models. When solving formulas over arrays, Z3 uses the current candidate model to limit the number of times definitions for arrays are expanded and exposed to the search process. In a setting that generalizes the domain of arrays, when instantiating quantified formulas, Z3 uses the current candidate model to search for instances where the model is repaired to satisfy the quantifier in a method known as model-based quantifier instantiation. Model-based techniques are also used to completely eliminate quantifiers over integers and reals; solve Horn clauses in the SPACER procedure; and reason about integers and Tarski's logic of real numbers in standalone procedures. The idea even generalizes to a combination of other theories in the model-constructing satisfiability (mcSAT) procedure. Every instance in which model-based techniques have been integrated in Z3 has led to increased speeds orders of magnitude greater than previous techniques. For Z3, model-based techniques are the difference-maker.

Even though the high-level idea of using models is relatively straightforward, the machinery and sophistication that goes into using models for guiding saturation is substantial. To integrate them well requires turning the failed search branches into strong, irredundant search inferences. There is a deep connection between the process of creating these inferences and deducing a special form of Craig interpolants. An inference that originates from a good interpolant eliminates not only the particular bad choice, but a maximal set of potential bad choices. A crucial part of engineering this integration is to ensure the interpolants contain enough information from the partial assignment still in play to separate it from extensions that are infeasible. The smarter and more efficiently you can compute interpolants, the better control you have over the search space.

Figure 1: Further details of **Z3**