Intro to Artificial Intelligence - Theoretical HW 1

Shay Kricheli

December 2020

1 Agents and Environment Simulators

	Bridge	**Home Repair**	**COVID-19**	**Solitaire**	**Sokoban**
Agent Type	Utility-Based	Utility-Based	Utility-Based	Goal/Utility-Based	Goal-Based
Deterministic	No	No	No	Yes	Yes
Observable	Partially	Partially	Partially	Yes	Yes
Episodic	No	No	No	Yes	Yes
Discrete	Yes	No	Yes	Yes	Yes
Single-Agent	No	No	No	Yes	Yes
Static	No	No	No	No	No

Explanations:

- An agent that plays Bridge:
  - In the game of Bridge, a player is to maximize his own points. The **utility-based** agent is the only one that can do that, so just for that reason - a utility based agent is needed. Moreover, abstraction for a Bridge game will produce too many possible states for a reflex agent with states.
  - Since in Bridge cards are dealt from a mixed deck, the environment is **stochastic**.
  - Since a player can only see his own cards, the environment is **partially observable**.
  - Since we need to remember the past - the environment is **sequential** (and not episodic).
  - The game takes place is a turn for each player and is thus **discrete**.
  - Each Bridge player can be considered an agent and thus the environment is **multi-agent**.
  - The players place cards on the table and take some of them so the environment is **dynamic**.
- A fully autonomous robot for home repair:
  - Home repair is a broad concept that can deal with many variations of maximizations and minimizations and thus a **utility based** agent is still needed.
  - Since this robot will not know beforehand the consequences of his actions and the entire state of the world he's in, the environment is **stochastic** and **partially observable**.
  - Every move in the past can affect the next decision to be made, so the environment is **sequential**.
  - We're talking about a robot that moves in the real world, so the environment is **continuous**.
  - Since there can be other entities around (a cat that disturbs cleaning the floor or a man making dinner) that change the current affair of things the environment is **multi-agent** and **dynamic**.

- An internet agent that optimizes COVID-19 response at the municipal level:
  - This is a task that needs optimization of various sorts and thus a **utility based** agent is needed.
  - Similar to the home repair robot, this agent will not know beforehand the consequences of his actions and the entire state of the world he's in so the environment is **stochastic** and **partially observable**.
  - Every move in the past can affect the next decision to be made, so the environment is **sequential**.
  - We're talking about an internet agent that applies strategies to the real world, so the physical environment is continuous but the agent is internet-based, so he can only do **discrete** moves.
  - Since there are other entities around that help or make trouble and change the current affair of things, the environment is **multi-agent** and **dynamic**.
- An agent that can play peg-and-hole Solitaire:
  - In the game of peg-and-hole Solitaire, the player faces an cross shaped boards with a hole in the middle. The entire board is placed with pegs except for the central hole. The objective is, to empty the entire board except for a single peg in the central hole. This is a purely combinatorial problem that has several known solutions. If the agent is to be able to solve the game without minimizing the number of moves - then it can be a **goal-based** agent. If not, then it has to be a **utility-based** one.
  - Since every moves induces one exact state - the environment is **deterministic**.
  - Since the board is known in advance and in between moves - the environment is **fully observable**.
  - Since past actions do not reflect on the best action to do at a given state - the environment is **episodic**.
  - The game takes place in single moves and is thus **discrete**.
  - The game is played by the player itself and is thus **single-agent**.
  - The players' moves change the pegs on the board and thus the environment is **dynamic**.
- An agent that can solve Sokoban puzzles (with no bonus for shorter plans):
  - Similar to the peg Solitaire, this game is a purely combinatorial problem. Without a bonus for shorter plans, it will require no more than a **goal-based** agent.
  - Since every moves induces one exact state - the environment is **deterministic**.
  - Since the puzzle is known in advance and in between moves - the environment is **fully observable**.
  - Since past actions do not reflect on the best action to do at a given state - the environment is **episodic**.
  - The game takes place in single moves and is thus **discrete**.
  - The game is played by the player itself and is thus **single-agent**.
  - The players' moves change the state of the puzzle and thus the environment is **dynamic**.

2 Search

The heuristic defined in class to the problem posed in assignment 1 is closely related to the Steiner tree problem in graphs:

Definition 2.1. *Given a weighted undirected graph $G = (V, E)$ and a subset of vertices $V' \subseteq V$ in the graph, a **Steiner tree** is a tree $T = (V', E')$ such that the edges in $E' \subseteq E$ span the vertices V' in G.*

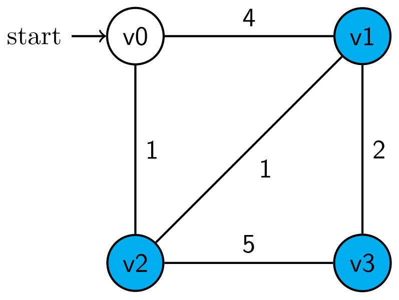
Finding the Steiner tree is a problem which is NP-complete and thus the suggested heuristic is in fact an approximation to the Steiner tree which is solvable in polynomial time.

Definition 2.2. *The **metric closure** of a graph G is the complete graph $G' = (V, E')$ where $\forall e = (u, v) \in E'$, e is weighted by the shortest path distance between u and v in G.*

The heuristic suggests first constructing G' - the metric closure of G by the vertices V' where V' consists of the vertices that have people in them along with the current vertex the agent is in and the edges in G' are weighted with the shortest paths between the vertices of V' in G. Then the minimum spanning tree (MST) value is to be calculated and returned.
Let us denote with $G = (V, E)$ to be our graph, where:

$$\begin{aligned} V = \;& \{v_0,\ v_1,\ v_2,\ v_3\} \\ E = \;& \{e_1 = (v_0, v_1),\ e_2 = (v_0, v_2),\ e_3 = (v_2, v_3),\ e_4 = (v_1, v_3),\ e_5 = (v_1, v_2)\} \\ & w(e_1) = 4,\ w(e_2) = w(e_5) = 1,\ w(e_3) = 5,\ w(e_4) = 2 \end{aligned}$$

According to the people location described in question 6, let us define $V' = \{v_1,\ v_2,\ v_3\}$ to be the set of vertices we must visit. Let us consider the following figure that depicts G (where the cyan colored vertices are ones with people in them - i.e. the requested subset to span):



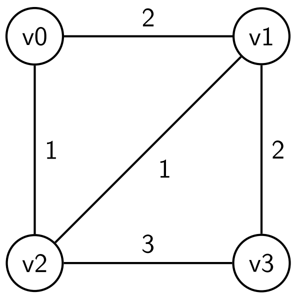
We will show the run of the algorithms on the graph at hand. Let us consider the state space by defining a state as a quadruple

$$S_i = (v_i,\ p1_i,\ p2_i,\ p3_i)$$

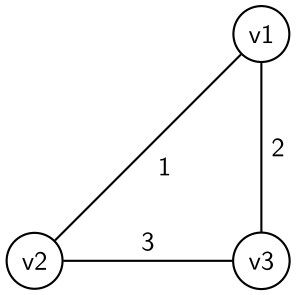
where $v_i \in V$ is the current vertex of the agent at state i and pj_i is true iff there are people at vertex v_j at state i. From this definition one can see that the number of states is finite (to be precise it is $O(|V|2^{|V|})$. According to this definition, let us define the set of all possible goal states:

$$Goals = \{(v_i, F, F, F) \mid v_i \in V\}$$

Let us consider the first step which is the same for all algorithms:
At first, the only state to consider is $S_0 = (v_0, T, T, T)$. To illustrate the heuristic value, let us consider the metric graph G'_0 of the initial state:

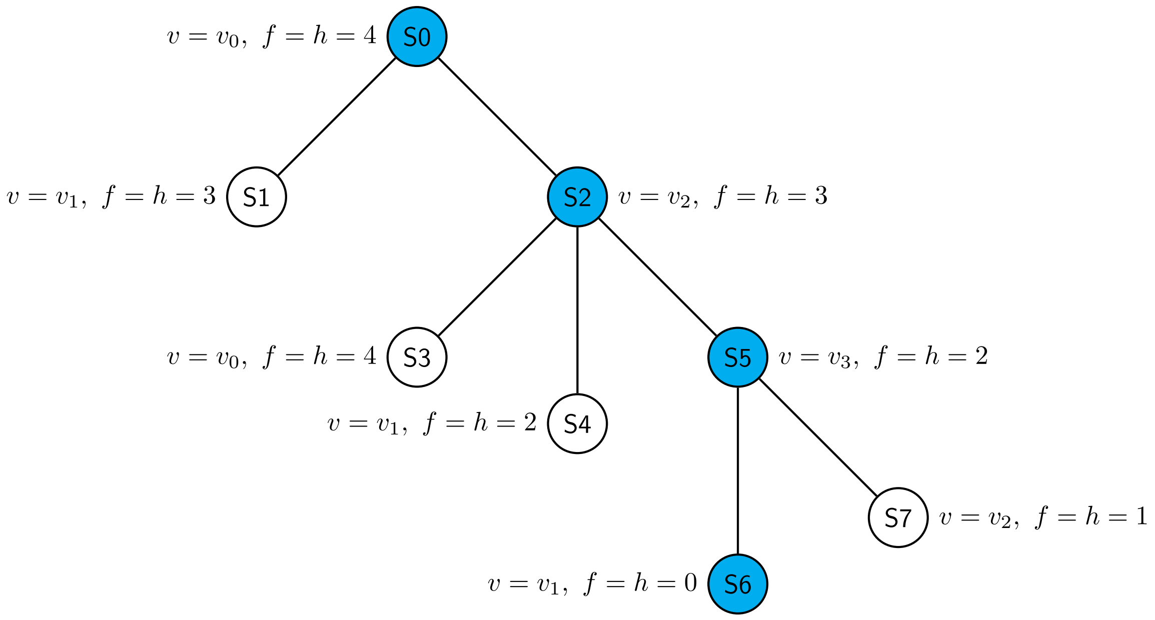


The MST value of G'_0 is 4 and thus the heuristic value is $h(0) = 4,\ h'(0) = 8$ and we have $g(0) = 0$. Since this is the only state, it is chosen. Since $S_0 \notin Goals$, it is not a goal and thus it is expanded to add two possible states to the agenda: $S_1 = (v_1, F, T, T)$ and $S_2 = (v_2, T, F, T)$. Let us consider the metric graph G'_1 of state S_1:



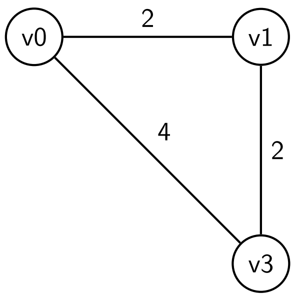
The MST value of G'_1 is $h(1) = 3,\ h'(1) = 6$ and we have $g(1) = w(e_1) = 4$. The metric graph G'_2 of state S_2 will in this case be the same and thus $h(2) = 3,\ h'(2) = 6$ and we have $g(2) = w(e_2) = 1$.
The rest of the run depends on the algorithm.

- **Greedy search** ($f(n) = h(n)$), breaking ties in favor of states where the higher node number: Let us draw the search tree for the greedy agent:

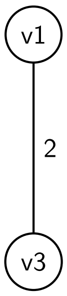


where the cyan colored vertices are ones that the agent visits.

1. By breaking ties in favor of states where the higher node number - we choose S_2. Since $S_2 \notin Goals$, it is not a goal and thus it is expanded to add: $S_3 = (v_0, T, F, T)$, $S_4 = (v_1, F, F, T)$ and $S_5 = (v_3, T, F, F)$. Let us consider the metric graph G'_3 of state S_3:

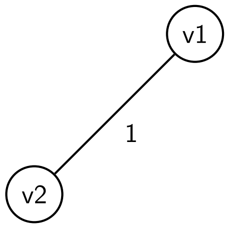
   The MST value of G'_3 is $h(3) = 4$. Let us consider the metric graph G'_4 of state S_4:

   The MST value of G'_4 is $h(4) = 2$. The metric graph G'_5 of state S_5 will in this case be the same and thus $h(5) = 2$.

2. Since S_4 and S_5 have the smallest heuristic value - by breaking ties in favor of states where the higher node number - we choose S_5. Since $S_5 \notin Goals$, it is not a goal and thus it is expanded to add: $S_6 = (v_1, F, F, F)$ and $S_7 = (v_2, T, F, F)$. Let us consider the metric graph G'_6 of state S_6:

   The MST value of G'_6 is $h(6) = 0$. Let us consider the metric graph G'_7 of state S_7:

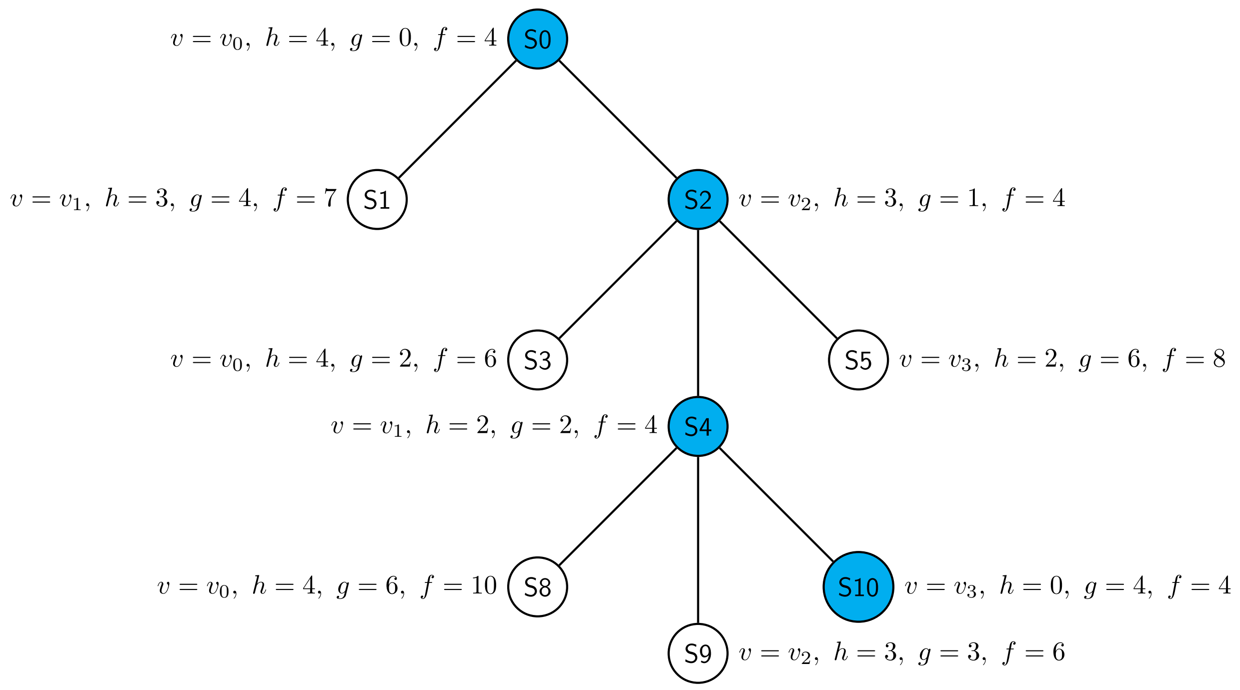
   

   The MST value of G'_7 is $h(7) = 1$.

3. State S_6 has the smallest heuristic value and so it is chosen. We have that $S_6 \in Goals$ so we terminate and return S_6.

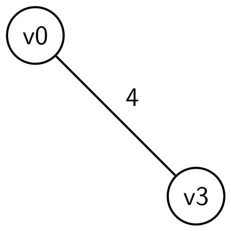
So the resulting path is $P_{geedy} = (v_0, v_2, v_3, v1)$ with a weight of $w(P_{geedy}) = 8$.

- **A* search** ($f(n) = h(n) + g(n)$), breaking ties as above: Let us draw the search tree for the A* agent:

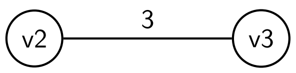


$v = v_0,\ h = 4,\ g = 0,\ f = 4$
S0
$v = v_1,\ h = 3,\ g = 4,\ f = 7$
S1
S2
$v = v_2,\ h = 3,\ g = 1,\ f = 4$
$v = v_0,\ h = 4,\ g = 2,\ f = 6$
S3
S5
$v = v_3,\ h = 2,\ g = 6,\ f = 8$
$v = v_1,\ h = 2,\ g = 2,\ f = 4$
S4
$v = v_0,\ h = 4,\ g = 6,\ f = 10$
S8
S10
$v = v_3,\ h = 0,\ g = 4,\ f = 4$
S9
$v = v_2,\ h = 3,\ g = 3,\ f = 6$

1. As showed before, $f(1) = h(1) + g(1) = 7$ and $f(2) = h(2) + g(2) = 4$ so we choose S_2 - which is as above expanded to yield S_3, S_4 and S_5. We have $g(3) = w(e_2) + w(e_2) = 2$, $g(4) = w(e_2) + w(e_5) = 2$ and $g(5) = w(e_2) + w(e_3) = 6$ so along with the values calculated above; $f(3) = h(3) + g(3) = 6$, $f(4) = h(4) + g(4) = 4$ and $f(5) = h(5) + g(5) = 8$.
2. We choose S_4. Since $S_4 \notin Goals$ we expand it to yield $S_8 = (v_0, F, F, T), S_9 = (v_2, F, F, T)$ and $S_{10} = (v_3, F, F, F)$. Let us consider the metric graph G'_8 of state S_8:

   The MST value of G'_8 is $h(8) = 4$. Let us consider the metric graph G'_9 of state S_9:

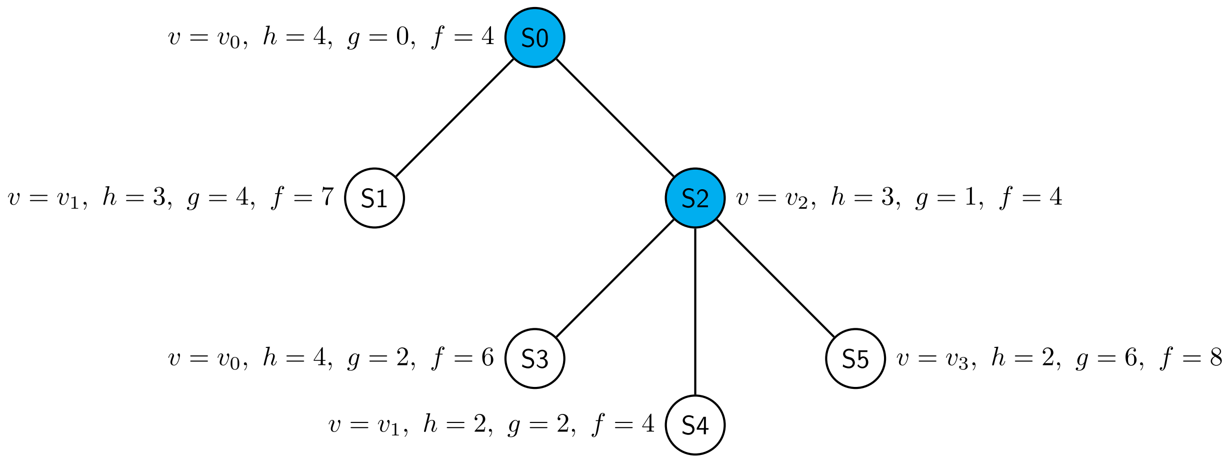
   The MST value of G'_9 is $h(9) = 3$. Let us consider the metric graph G'_{10} of state S_{10}:

   The MST value of G'_{10} is $h(10) = 0$.
3. State S_{10} has the smallest heuristic value and so it is chosen. We have that $S_{10} \in Goals$ so we terminate and return S_{10}.

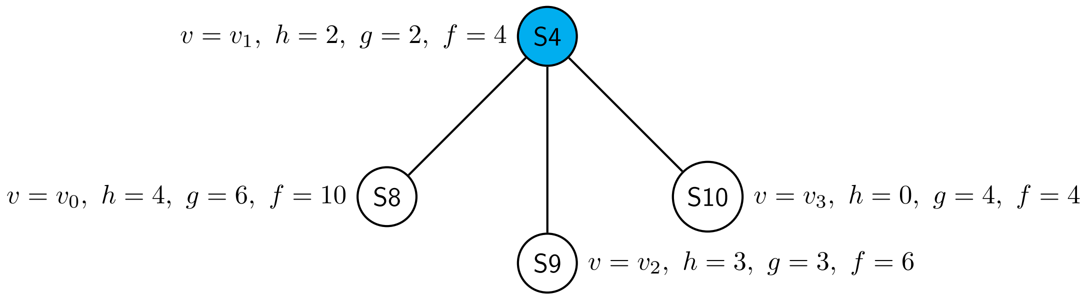
So the resulting path is $P_{A^*} = (v_0, v_2, v_1, v3)$ with a weight of $w(P_{A^*}) = 4$.

- **Simplified RTA* search** with a limit of 2 expansions per real action, breaking ties as above: Let us draw the search trees for the Simplified RTA* agent:

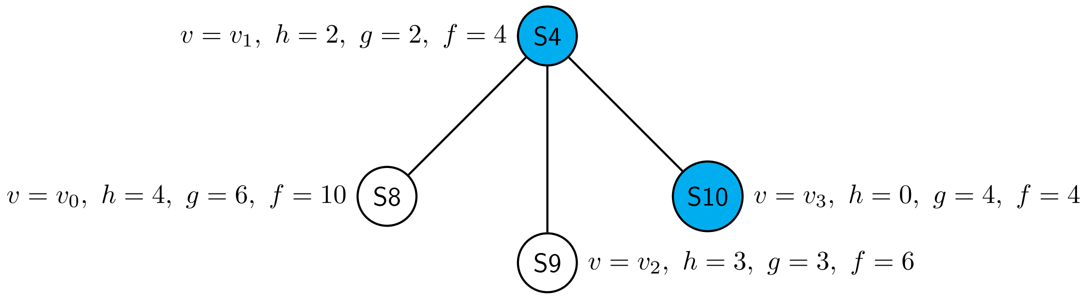
  1. After the first two expansions:


$v = v_0,\ h = 4,\ g = 0,\ f = 4$ S0
$v = v_1,\ h = 3,\ g = 4,\ f = 7$ S1
S2 $v = v_2,\ h = 3,\ g = 1,\ f = 4$
$v = v_0,\ h = 4,\ g = 2,\ f = 6$ S3
S5 $v = v_3,\ h = 2,\ g = 6,\ f = 8$
$v = v_1,\ h = 2,\ g = 2,\ f = 4$ S4

  2. After the first two expansions, the smallest heuristic value if of S_4, so the agent chooses it. Since $S_4 \notin Goals$, the agents reiterates and opens up a new tree where it expands S_4:


$v = v_1,\ h = 2,\ g = 2,\ f = 4$ S4
$v = v_0,\ h = 4,\ g = 6,\ f = 10$ S8
S10 $v = v_3,\ h = 0,\ g = 4,\ f = 4$
S9 $v = v_2,\ h = 3,\ g = 3,\ f = 6$

  3. After the third expansion, the smallest heuristic value if of S_{10}, so the agent chooses it:

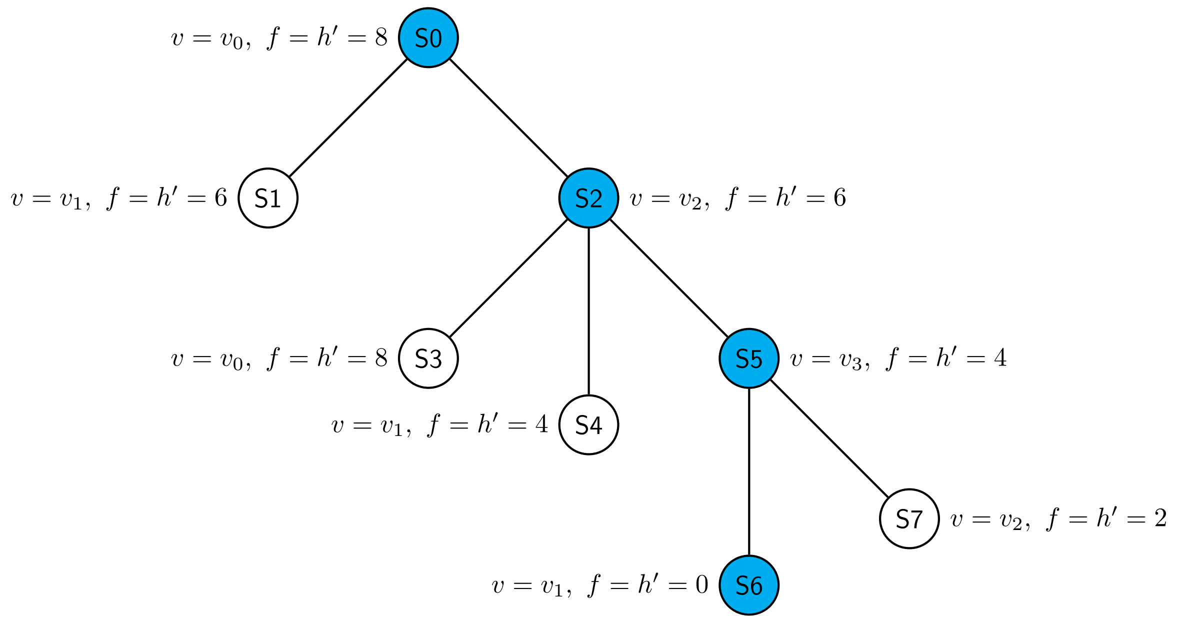

$v = v_1,\ h = 2,\ g = 2,\ f = 4$ S4
$v = v_0,\ h = 4,\ g = 6,\ f = 10$ S8
S10 $v = v_3,\ h = 0,\ g = 4,\ f = 4$
S9 $v = v_2,\ h = 3,\ g = 3,\ f = 6$

     We have that $S_{10} \in Goals$ so we terminate and return S_{10}.

  So the resulting path is $P_{S-RTA^*} = (v_0, v_2, v_1, v3)$ with a weight of $w(P_{S-RTA^*}) = 4$.

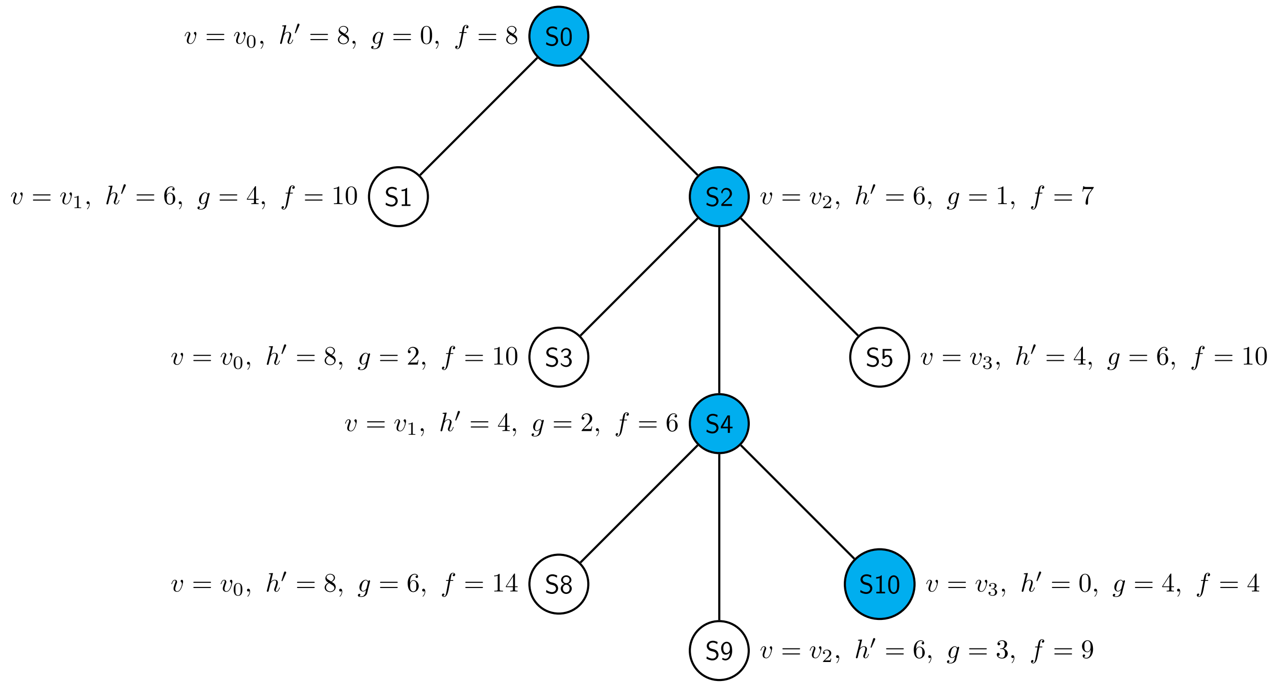
- **Using $h'(n) = 2h(n)$ as the heuristic**:

  1. Greedy search: Let us draw the search tree for the greedy agent in this case:


$v = v_0,\ f = h' = 8$ S0
$v = v_1,\ f = h' = 6$ S1
S2 $v = v_2,\ f = h' = 6$
$v = v_0,\ f = h' = 8$ S3
$v = v_1,\ f = h' = 4$ S4
S5 $v = v_3,\ f = h' = 4$
S7 $v = v_2,\ f = h' = 2$
$v = v_1,\ f = h' = 0$ S6

     The agent will follow the same path of the previous greedy agent since the relative order between the heuristic values of the states remains the same (as will happen always with a greedy agent when multiplying the heuristic with a fixed constant).

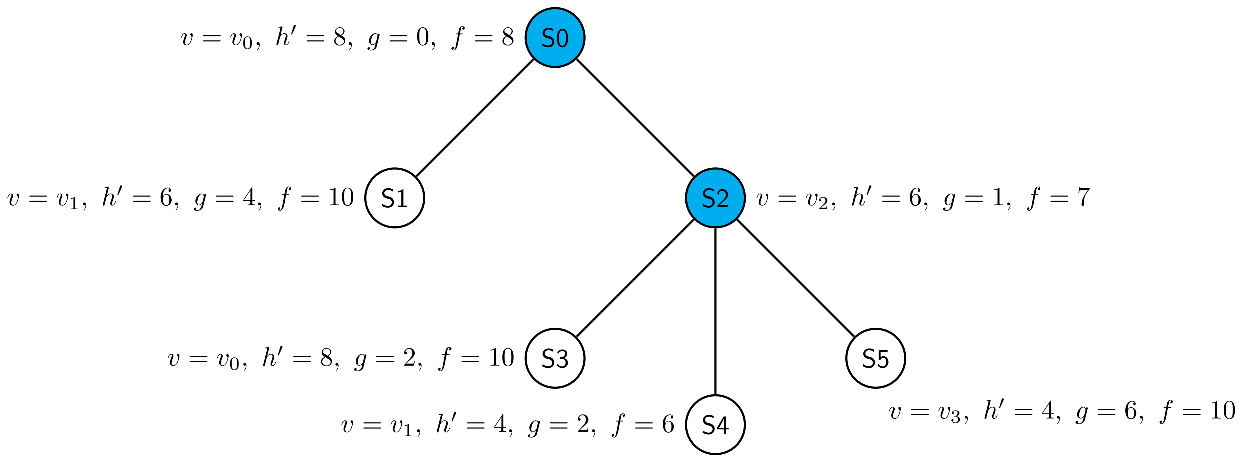
  2. A^* search: Let us draw the search tree for the A^* agent in this case:


$v = v_0,\ h' = 8,\ g = 0,\ f = 8$ S0
$v = v_1,\ h' = 6,\ g = 4,\ f = 10$ S1
S2 $v = v_2,\ h' = 6,\ g = 1,\ f = 7$
$v = v_0,\ h' = 8,\ g = 2,\ f = 10$ S3
S5 $v = v_3,\ h' = 4,\ g = 6,\ f = 10$
$v = v_1,\ h' = 4,\ g = 2,\ f = 6$ S4
$v = v_0,\ h' = 8,\ g = 6,\ f = 14$ S8
S10 $v = v_3,\ h' = 0,\ g = 4,\ f = 4$
S9 $v = v_2,\ h' = 6,\ g = 3,\ f = 9$

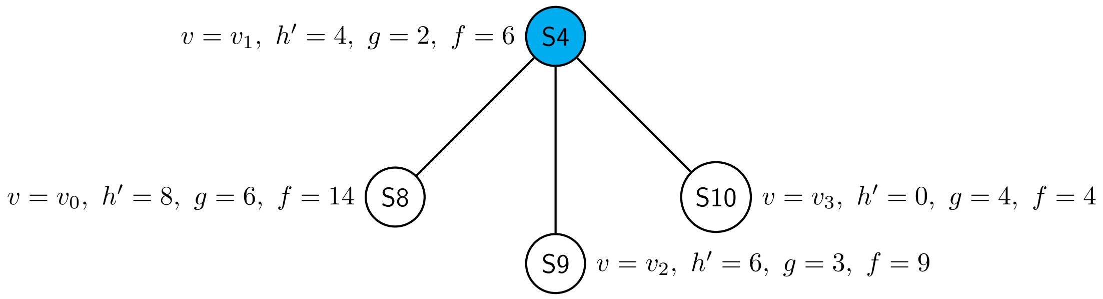
     The agent will follow the same path of the previous A^* agent since in this specific case the relative order between the f values of the states remains the same (and this is not necessarily the case for an A^* agent, since multiplying the heuristic with a fixed constant can change the order of the f values among the states).

3. Simplified RTA* search: Let us draw the search trees for the Simplified RTA* agent in this case:

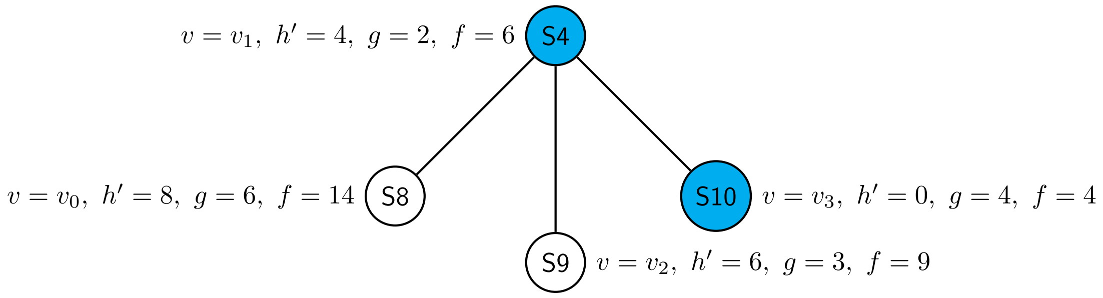
   (a) After the first two expansions:


$v = v_0,\ h' = 8,\ g = 0,\ f = 8$
S0
$v = v_1,\ h' = 6,\ g = 4,\ f = 10$
S1
S2
$v = v_2,\ h' = 6,\ g = 1,\ f = 7$
$v = v_0,\ h' = 8,\ g = 2,\ f = 10$
S3
S5
$v = v_1,\ h' = 4,\ g = 2,\ f = 6$
S4
$v = v_3,\ h' = 4,\ g = 6,\ f = 10$

   (b) After the first two expansions, the smallest heuristic value if of S_4, so the agent chooses it. Since $S_4 \notin Goals$, the agents reiterates and opens up a new tree where it expands S_4:


$v = v_1,\ h' = 4,\ g = 2,\ f = 6$
S4
$v = v_0,\ h' = 8,\ g = 6,\ f = 14$
S8
S10
$v = v_3,\ h' = 0,\ g = 4,\ f = 4$
S9
$v = v_2,\ h' = 6,\ g = 3,\ f = 9$

   (c) After the third expansion, the smallest heuristic value if of S_{10}, so the agent chooses it:


$v = v_1,\ h' = 4,\ g = 2,\ f = 6$
S4
$v = v_0,\ h' = 8,\ g = 6,\ f = 14$
S8
S10
$v = v_3,\ h' = 0,\ g = 4,\ f = 4$
S9
$v = v_2,\ h' = 6,\ g = 3,\ f = 9$

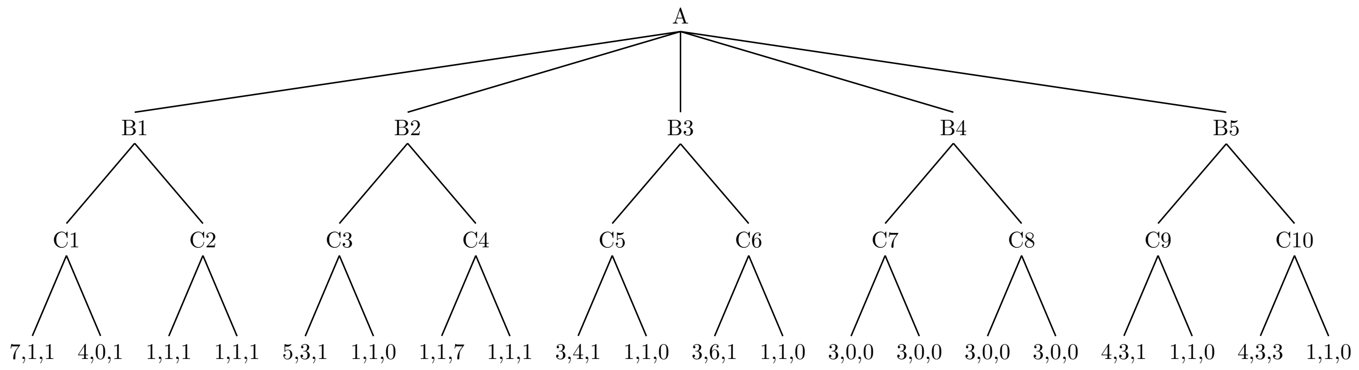
   We have that $S_{10} \in Goals$ so we terminate and return S_{10}.

   The agent will follow the same path of the previous Simplified Real-Time A^* agent from the same reasoning as the regular A^* agent.

h' is not admissible since we can see for example that the value for h' at the initial state S_0 is larger than the actual optimal value we got (since we know A^* is optimal), i.e. $h'(0) = 8 > h^*(0) = 4$ - in contradiction to the definition of admissibility.

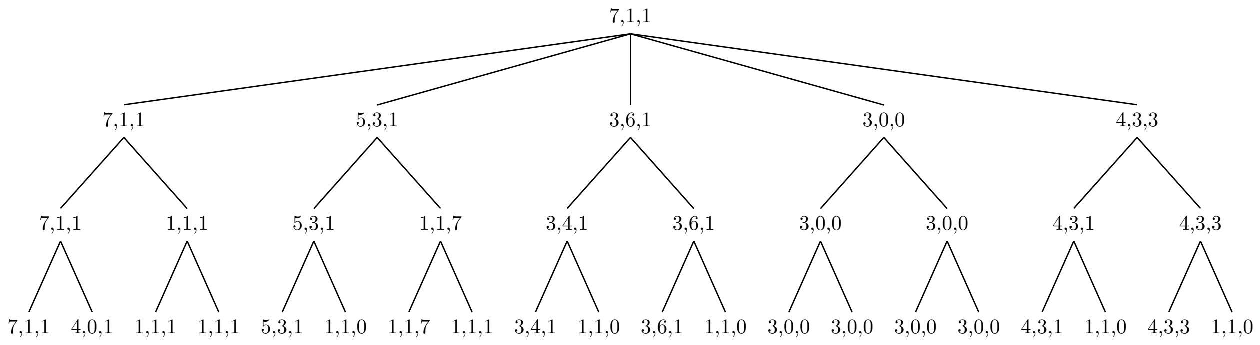
3 Game Trees

Let us observe the following game tree:



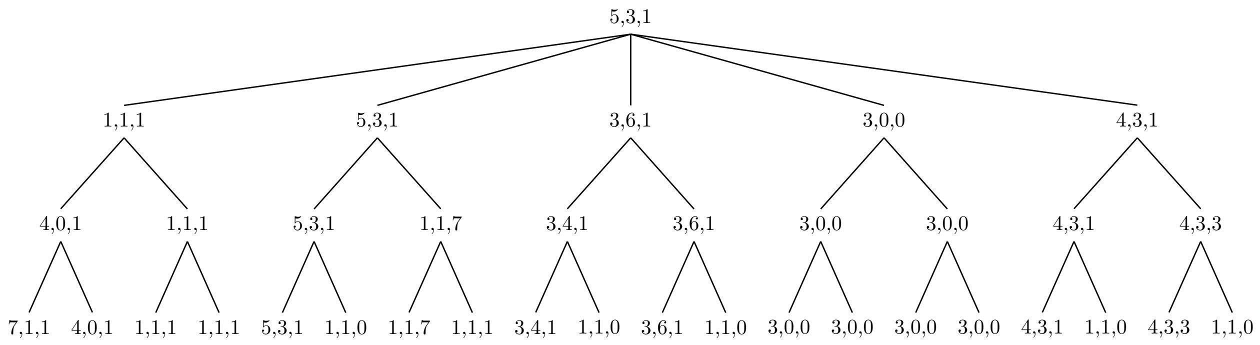
Each triple of the form (a, b, c) corresponds to the appropriate values for players A, B and C accordingly. The edges at the lowest level correspond to choices player C makes, the edges in between correspond to choices player B makes and the edges at the highest level correspond to choices player A makes. Let us consider each of the given tactics:

1. **Each agent out for itself, and they cannot communicate (with ties broken cooperatively)**: In this scenario, the following tree will result:



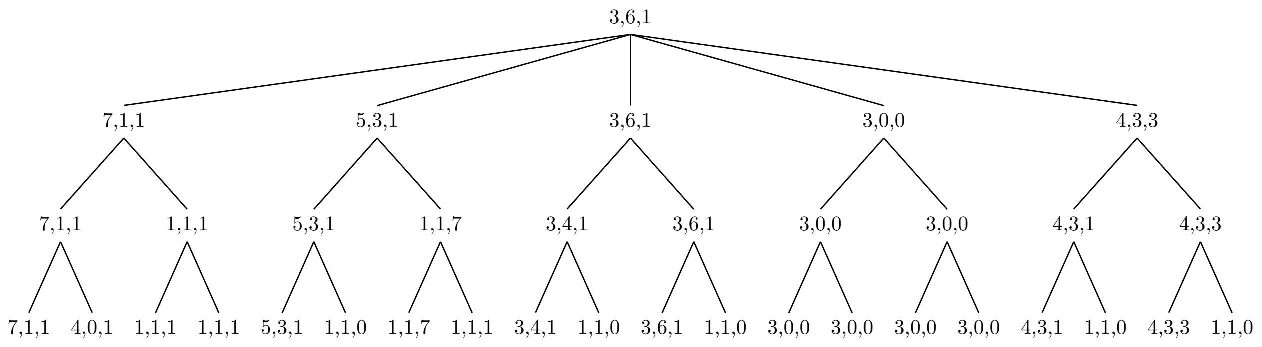
One can see that A takes the leftmost option in this case.

2. **As in 1, except B and C are anti-cooperative (they break ties anti-cooperatively)**: In this scenario, the following tree will result:



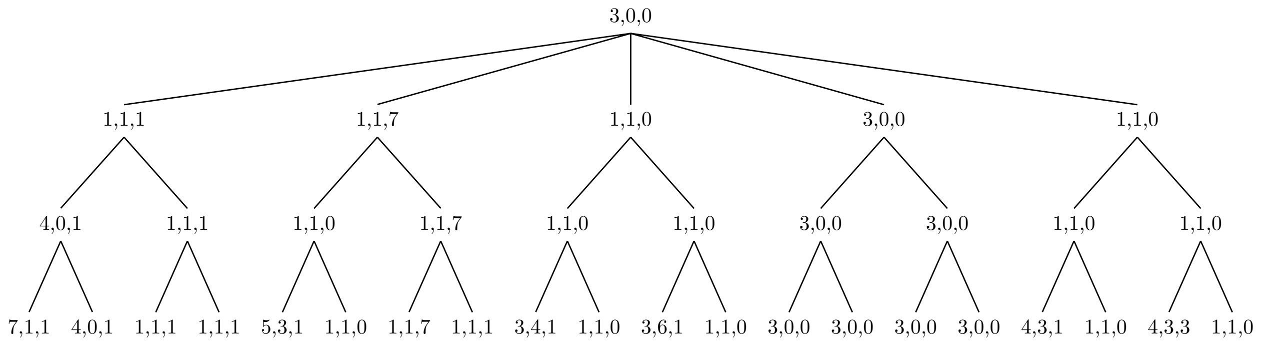
One can see that A takes the second option from the left in this case.

3. **As in 1, except A and B are partners aiming to maximize the sum of their scores**:
   In this scenario, the following tree will result:



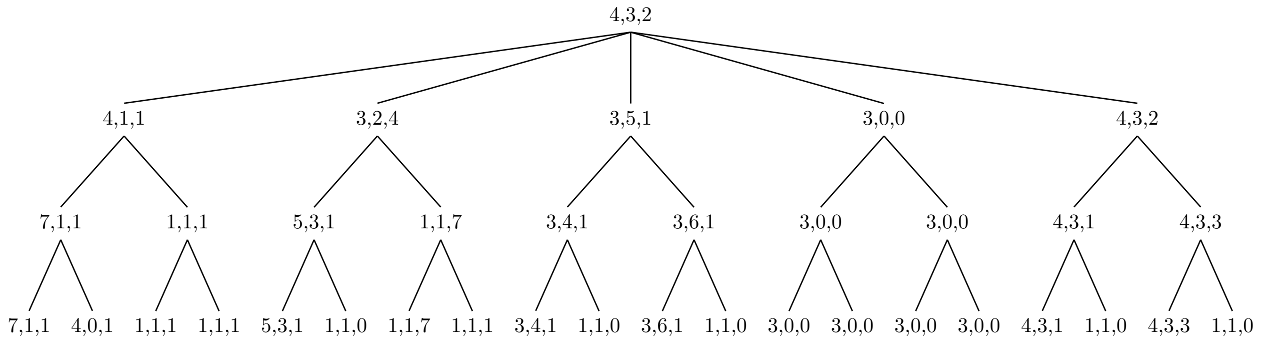
   One can see that A takes the middle option.

4. **Paranoid assumption: B and C are against A, no matter what their score is**:
   In this scenario, the following tree will result:



   One can see that A takes the second option from the right.

5. **As in 1, except B plays randomly with uniform distribution**:
   In this scenario, the following tree will result (with the values for B calculated as the expected value with each edges at the middle level with a probability of 0.5):



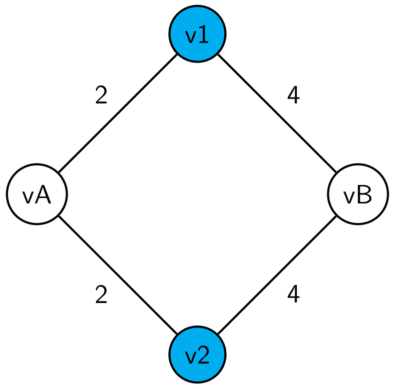
   One can see that A takes the first option from the right.

4 Game-Tree Search - Alpha-Beta Pruning

1. Let us observe the following example to the hurricane evacuation problem: Let there be two agents, A and B, let $D = 7$ be the deadline and let $G = (V, E)$ be a graph where:

$$\begin{aligned} V = &\ \{v_A,\ v_B,\ v_1,\ v_2\} \\ E = &\ \{e_1 = (v_A, v_1),\ e_2 = (v_A, v_2),\ e_3 = (v_B, v_1),\ e_4 = (v_B, v_2)\} \\ &\ w(e_1) = w(e_2) = 2,\ w(e_3) = w(e_4) = 4 \end{aligned}$$

Let us assume there are a million people both at v_1 and v_2. Let us observe the following figure of the graph (where the cyan colored vertices are ones with people in them):



Let us assume no-op moves only consume 1 time unit and that agent A start at V_A, B at V_B and that A plays first. The first move by any player can be either to move in the direction of v_1, move in the direction of v_2 or do no-op.

Claim 4.1. *For the game defined on G, the optimal first move for A would be to do no-op.*

Lemma 4.2. *In any scenario where A starts to move towards v_1 or v_2 at the first clock, there exists a strategy for B that scores even.*

Proof. Let us assume the lemma holds. Thus suffice it to show that there exists a scenario where A first does no-op and wins the game for any strategy of B. Then let us assume A first does no-op.

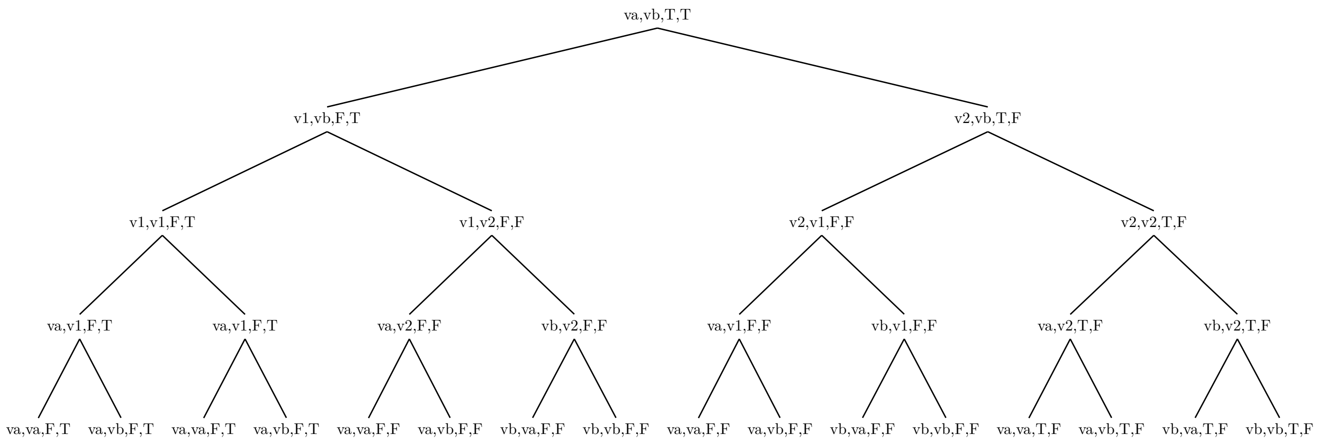
- **Case 1: In the second clock, B also does no-op.**
  In that case let A do no-op again. Then since 3 clocks have passed, if B doesn't start moving, he will never reach any node and in the next clock A can move in any direction and win. Let us assume that B then starts moving towards v_1 w.l.o.g (the following arguments when moving towards v_2 are symmetrical). Then let A start moving towards v_1 as well and make it there before B. B will not have enough time to make it to v_2 and A will win.
- **Case 2: In the second clock, B moves towards v_1 w.l.o.g.**
  In that case let A start moving towards v_1 as well and make it there before B. By the time he makes it to v_1, 3 clocks have passed. Then let him traverse back to v_A and then to v_2 - for a total of 7 clocks up to the deadline. By the time B has made it to v_1, the people there were already picked up by A, and he won't have enough time to head towards v_2 in either way by the time limit. So in this case A also wins.

□

Let us now prove the lemma:

Proof. Let us assume that A starts moving in the first clock. w.l.o.g let us assume the first move by A is towards v_1 (the following arguments when moving towards v_2 are symmetrical). Since A started moving in the direction of v_1, B will never make it to v_1 in time to save the people there. After the first move B is aware that A is moving towards v_1. Thus in the second move B can start moving towards v_2. Let us assume he does so. Since A started moving towards v_1, he will still have another time clock to get there. Once he got there, let us assume he takes the shortest path towards v_2 - which is moving back to v_A and then to v_2. By the time A will get back to v_A, B will already be at v_2. Since A took the shortest path between v_1 and v_2, in any scenario where A starts to move towards v_1 at the first clock, B has a strategy that score even. □

2. Let us consider the example from the last section and plot the search graph of it (each state is represented by a tuple $(a, b, p1, p2)$ corresponding to the current vertices of A and B and whether there are people at v_1 and v_2 respectively):



3. In all branches where there is F,F as the flag for the people - once we hit F,F at the second level - we know searching down is useless as it provides no additional value since all the people were already picked up.

5 Propositional Logic

- a) Satisfiable, not valid and with 5 models.
- b) Satisfiable, not valid and with 30 models.
- c) Not satisfiable, not valid and with 0 models.
- d) Satisfiable, valid and with 16 models.
- e) Not satisfiable, Not valid and with 0 models.
- f) Satisfiable, valid and with 1 models.

6 FOL and situation calculus

1. Let us state the given static facts representing the graph G and the hurricane problem scenario:

$$\begin{aligned}
&Vertex(V0)\\
&Vertex(V1)\\
&Vertex(V2)\\
&Vertex(V3)\\
&\forall\ V_i, V_j\ ;\ i \neq j\ \rightarrow V_i \neq V_j\\
&Edge(E1, V0, V1)\\
&Edge(E2, V0, V2)\\
&Edge(E3, V2, V3)\\
&Edge(E4, V1, V3)\\
&Edge(E5, V1, V2)\\
&\forall\ E_i, E_j\ ;\ i \neq j\ \rightarrow E_i \neq E_j\\
&Weight(E1, 4)\\
&Weight(E2, 1)\\
&Weight(E3, 5)\\
&Weight(E4, 2)\\
&Weight(E5, 1)\\
&Deadline(6)
\end{aligned}$$

Let us declare that the graph is undirected - meaning that each edge is symmetric:

$$\forall\ e, v1, v2\ ;\ Edge(e, v1, v2) \rightarrow Edge(e, v2, v1) \tag{1}$$

Let us state the initial conditions as stated in the problem:

$$\begin{aligned}
&Loc(V0, S0)\\
&PeopleAt(V1, S0)\\
&PeopleAt(V2, S0)\\
&PeopleAt(V3, S0)\\
&Time(0, S0)
\end{aligned}$$

Let us add to the KB a predicate to indicate where were people at the initial state called PO where $PO(v)$ is true iff there were people at vertex v at state S_0. Thus let:

$$\begin{aligned}
&\neg PO(V0)\\
&PO(V1)\\
&PO(V2)\\
&PO(V3)
\end{aligned}$$

Let us state that there will be no people at any time vertices where there weren't any to begin with:

$$\forall\ v, s\ ;\ \neg PO(v) \rightarrow \neg PeopleAt(v, s) \tag{2}$$

Now let us define the order of the natural numbers with a successor function $S(n)$ for all $n \in \mathbb{N}$:

$$\begin{aligned}
&S(0) = 1\\
&S(S(0)) = S(1) = 2\\
&\forall\ n\ ;\ S(n) = n + 1
\end{aligned}$$

Let us define addition on natural numbers:

$$\forall\ n\ ;\ +(x,0,n) \wedge +(0,n,n)$$
$$\forall\ n1,n2,n3\ ;\ +(n1,n2,n3) \rightarrow (+(n1,S(n2),S(n3)) \wedge +(S(n1),n2,S(n3))$$

Let us define the order of the natural numbers:

$$\forall\ n\ ;\ Leq(n,n) \wedge Leq(n,S(n))$$
$$\forall\ n1,n2,n3\ ;\ (Leq(n1,n2) \wedge Leq(n2,n3)) \rightarrow Leq(n1,n3)$$

Let us assume that the KB contains all relevant such facts such as $Leq(100,10000)$ and $+(1,1,2)$. Let us define an additional fluent called OK where $OK(action, source, destination, s)$ is true iff when at state S, the action *action* of moving from *source* to *destination* does not violate the deadline. This condition will hold if there is an edge e from $V1$ to $V2$, the time now is t, the agent is at $V1$, the weight of e is w, and the deadline is $Leq\ p$ (the sum of t and w), then traversing e is OK:

$$\begin{gathered}\forall\ v1,v2,e,s,t,d,p\ ;\ (Edge(e,v1,v2) \wedge Time(t,s) \wedge Loc(v1,s) \\ \wedge Weight(e,w) \wedge Deadline(d) \wedge +(t,w,p) \wedge Leq(p,d)) \rightarrow OK(traverse(e),v1,v2,s)\end{gathered} \tag{3}$$

Using the OK fluent let us define effects of traversal. Let us define an update for the clock:

$$\begin{gathered}\forall\ s,e,v1,v2,t1,t2,w\ ;\ (Time(t1,s) \wedge Weight(e,w)\ \wedge OK(traverse(e),v1,v2,s) \\ \wedge +(t1,w,t2)) \rightarrow Time(t2,Result(traverse(e),s))\end{gathered} \tag{4}$$

Let us define an update for the location **and for the change in number of people (which is equivalent to a FRAME AXIOM)**:

$$\begin{gathered}\forall\ s,e,v1,v2\ ;\ OK(traverse(e),v1,v2,s) \rightarrow Loc(v2,Result(traverse(e),s)) \wedge \\ \neg PeopleAt(v2,Result(traverse(e),s))\end{gathered} \tag{5}$$

2. Let us recall the path the A^* agent found in question 2: $P_{A^*} = (v_0, v_2, v_1, v3)$ with a weight of $w(P_{A^*}) = 4$.

   (a) Let us convert the knowledge base into conjunctive normal form: First let us convert equation (1) into CNF by logical equivalence laws and removal of universal quantifiers:

   $$\neg Edge(e,v1,v2) \vee Edge(e,v2,v1)$$

   Equation (2) converts into:

   $$PO(v) \vee \neg PeopleAt(v,s) \tag{6}$$

   Equation (3) converts into:

   $$\begin{gathered}\neg Edge(e,v1,v2) \vee \neg Time(t,s) \vee \neg Loc(v1,s) \vee \neg Weight(e,w) \vee \neg Deadline(d) \\ \vee \neg + (t,w,p) \vee \neg Leq(p,d) \vee OK(traverse(e),v1,v2,s)\end{gathered} \tag{7}$$

   Equation (4) converts into:

   $$\begin{gathered}\neg Time(t1,s) \vee \neg Weight(e,w) \vee \neg OK(traverse(e),v1,v2,s) \\ \vee \neg + (t1,w,t2)) \vee Time(t2,Result(traverse(e),s))\end{gathered}$$

   Equation (5) converts into:

   $$\begin{gathered}\neg OK(traverse(e),v1,v2,s) \vee \Big(Loc(v2,Result(traverse(e),s)) \wedge \\ \neg PeopleAt(v2,Result(traverse(e),s))\Big)\end{gathered} \tag{8}$$

(b) Let us state what we need to prove:

Claim 6.1. *There exists a sequence of actions that will result in all people being saved before the deadline, starting from S_0.*

We know that traversing $E2$ to $V2$, then $E5$ to $V1$, then $E4$ to $V3$ saves all the people. Thus we need to prove that:

$$\begin{aligned}&\exists t, d\ \forall v\ ;\ Deadline(d) \wedge Leq(1,t) \wedge Leq(t,d)\\ &\wedge Time(t, Result(traverse(E4), Result(traverse(E5), Result(traverse(E2), S0))))\\ &\wedge \neg PeopleAt(V, Result(traverse(E4), Result(traverse(E5), Result(traverse(E2), S0))))\end{aligned}$$

The negation is:

$$\begin{aligned}&\forall t, d\ \exists v\ ;\ \neg Deadline(d) \vee \neg Leq(1,t) \vee \neg Leq(t,d)\\ &\vee \neg Time(t, Result(traverse(E4), Result(traverse(E5), Result(traverse(E2), S0))))\\ &\vee PeopleAt(V, Result(traverse(E4), Result(traverse(E5), Result(traverse(E2), S0))))\end{aligned}$$

and in CNF form:

$$\begin{aligned}&\neg Deadline(d) \vee \neg Leq(1,t) \vee \neg Leq(t,d)\\ &\vee \neg Time(t, Result(traverse(E4), Result(traverse(E5), Result(traverse(E2), S0))))\\ &\vee PeopleAt(V, Result(traverse(E4), Result(traverse(E5), Result(traverse(E2), S0))))\end{aligned}$$

(c) Let us now prove the theorem disregarding the deadline and thus also the time and the weights. For that matter, let us state the reduced theorem we are to prove (in CNF form):

$$PeopleAt(V, Result(traverse(E4), Result(traverse(E5), Result(traverse(E2), S0))))$$

Let us now use Resolution to prove the theorem. For each action, we will first show that it is OK to do that action, then use that to show some or all properties of the situation resulting from the action, and repeat until we can prove the theorem.

Proof.
```
Resolve eq. 7 (neglecting the deadline, time and weights),
  with "Edge(E2, V0, V2)", unifier {e/E2, V1/V0, V2/V2} to get:
 100.   not Loc(V0,s) or OK(traverse(E2), V0, V2, s)

 Resolve 100 with "Loc(V0,S0)", empty unifier, to get:
 101.   OK(traverse(E2), V0, V2, S0)
```
Now we got the execution of traversing $E2$ to $V2$. Since we neglect the time, let us continue on to updating the location and number of people:

```
  Resolve eq. 8 with 101 and the unifier {e/E2, V1/V0, V2/V2, s/S0} to get:
 102.   Loc(V2, Result(traverse(E2), S0)) and not PeopleAt(V2, Result(traverse(E2), S0))

 Now we are done with the first action.
 Resolve eq. 7 again with "Edge(E5, V2, V1)", unifier {e/E5, V1/V2, V2/V1} to get:
 103.   not Loc(V2, s) or OK(traverse(E5), V2, V1, s)

 Resolve 103 with 102 and the unifier {s/Result(traverse(E2), S0)} to get:
 104.   OK(traverse(E5), V2, V1, Result(traverse(E2), S0))
```
Now we got the execution of traversing $E5$ to $V1$. So:

```
  Resolve eq. 8 with 104 and the unifier
  {e/E5, V1/V2, V2/V1, s/Result(traverse(E2), S0)} to get:
 105.   Loc(V1, Result(traverse(E5), Result(traverse(E2), S0)))
```

```
and not PeopleAt(V1, Result(traverse(E5), Result(traverse(E2), S0)))

Now we are done with the second action.
Resolve eq. 7 again with "Edge(E4, V1, V3)",
unifier {e/E4, V1/V1, V2/V3} to get:
106.   not Loc(V1, s) or OK(traverse(E4), V1, V3, s)

Resolve 106 with 105 and the unifier
{s/Result(traverse(E5), Result(traverse(E2), S0))} to get:
107.   OK(traverse(E4), V1, V3, Result(traverse(E4),
Result(traverse(E5), Result(traverse(E2), S0))))
```

Now we got the execution of traversing $E4$ to $V3$. So:

```
 Resolve eq. 8 with 107 and the unifier
 {e/E4, V1/V1, V2/V3, s/Result(traverse(E4), Result(traverse(E5),
 Result(traverse(E2), S0)))} to get:
108.   Loc(V3, Result(traverse(E4), Result(traverse(E5),
Result(traverse(E2), S0))))
and not PeopleAt(V3, Result(traverse(E4), Result(traverse(E5),
Result(traverse(E2), S0))))
```

We end up with clauses that represent statements that are true and thus just by applying themselves as a unifier, we end up with an empty clause and the proof by contradiction is finished. $\square$

3. We couldn't prove the theorem without the frame axioms we detailed beforehand, as they cause the transitions to update the number of people at visited vertices.

4. We used some axioms that were not in Horn form ant thus in this situation the proof cannot be done by backward or forward chaining. If we would to write the formulas in Horn form we would be able to do so.