A Contrastive Plan Explanation Framework for Hybrid System Models

*Mir Md Sajid Sarwar, *Rajarshi Ray, †Ansuman Banerjee

*Indian Association for the Cultivation of Science, Kolkata, India
†Indian Statistical Institute, Kolkata, India

Formal Methods Update Meeting 2022,
July 4-5, 2022, IIT Delhi, INDIA

Outline

- Motivation
- Preliminaries
- An example domain and a planning problem instance
- Our contrastive explanation framework
- A set of contrastive questions
- Proving the absence of plan
- Evaluation
- Conclusion

Motivation

- **AI Planning in Real Applications**

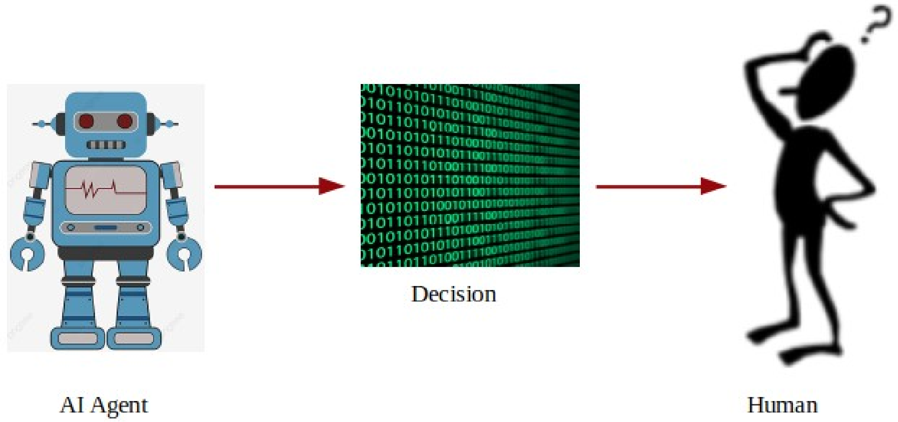
  In the last few years, planners are becoming more powerful, and planning is used in new (critical) domains.

  - Mining
  - Underwater Robotics
  - Smart Energy
  - Air Traffic Control
  - Urban Traffic Control
  - Search and Rescue Missions
  - Human-Autonomy Teaming
- **Plans are more complex than before**

  (continuous nonlinear methods, differential equations, fluid dynamics, etc...)
- **Explaining planning is more needed now than before**

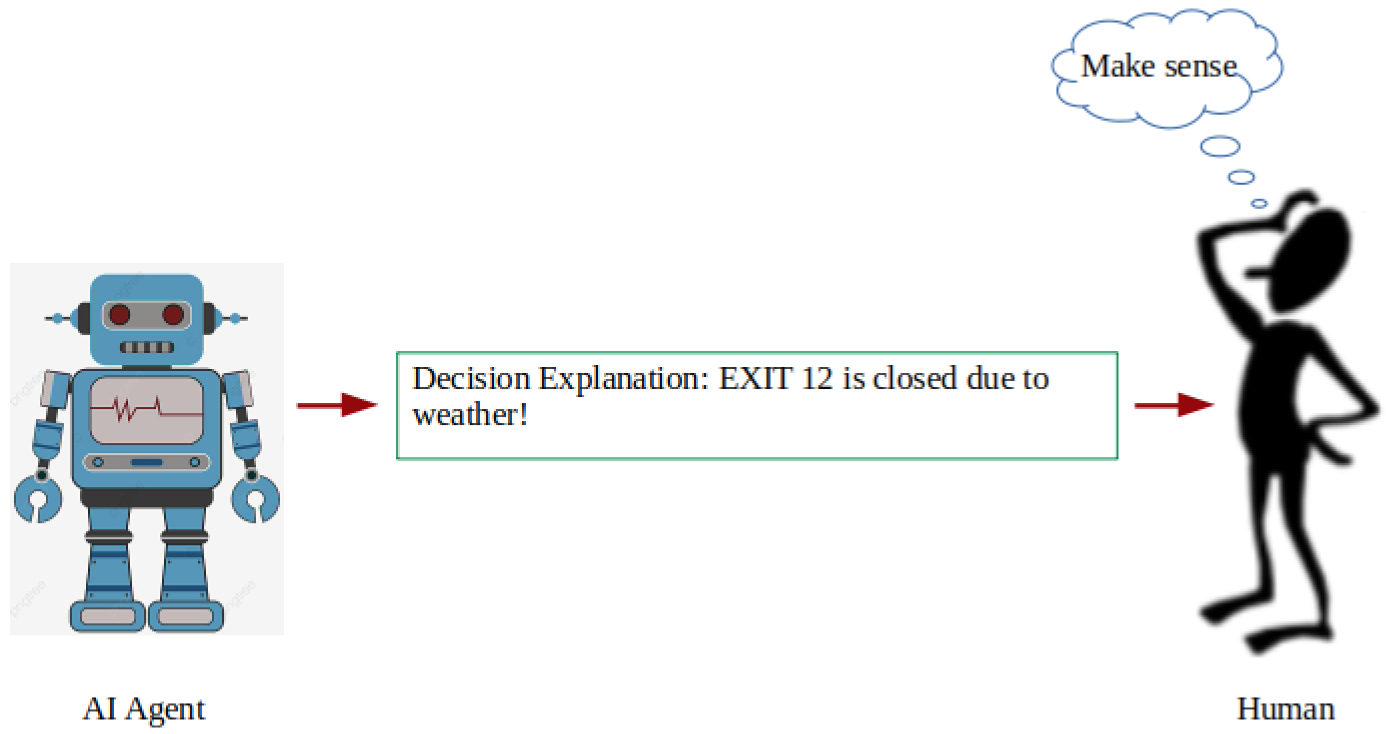
Motivation

- **Explainable AI Planning**



Motivation

- **Explainable AI Planning**



Motivation

- A detailed overview of the explainable artificial intelligence planning landscape and different terms used in this domain are introduced in [T. Chakraborti et al., 2018].
- [David E. Smith, 2012] put forward the challenge of planning as an iterative process for better modeling preferences and providing explanations. While improving the user's level of understanding and building trust in the system are the main purposes of these explanations, they can be local (regarding a specific plan) or global (concerning how the planning system works in general).
- [T. Chakraborti et al., 2017] considered explanation as a model reconciliation problem assuming that the agent and the human may have possibly different models of the environment. In such a scenario, the agent explains those actions to the human which are not expected to be executed taking the human model as reference. Explanation here can be seen as a reconciliation between the agent and the human.
- [B. Krarup et al., 2019] focuses on local explanations of temporal and numeric planning problems, introducing a formal description of the compilation from user questions to constraints in a PDDL2.1 planning setting and explaining why a planner has made a certain decision.

Motivation

- Contrastive explanations is a popular approach in recent literature for explanations of plans in AI planning.

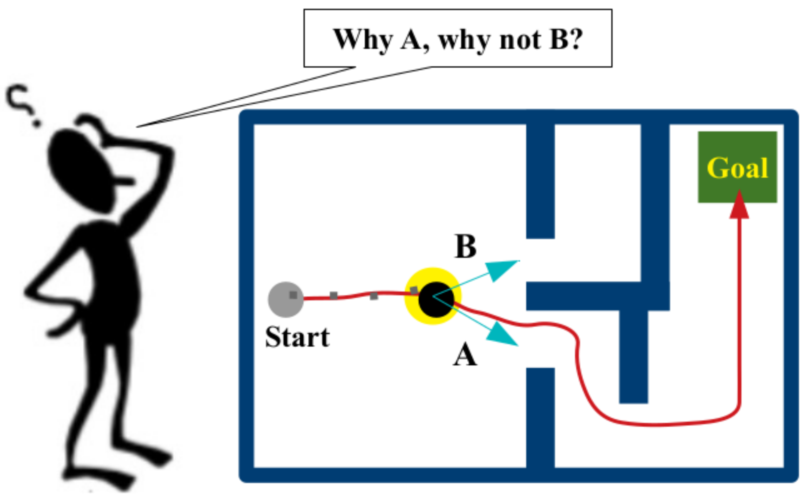


Figure 1: Original plan

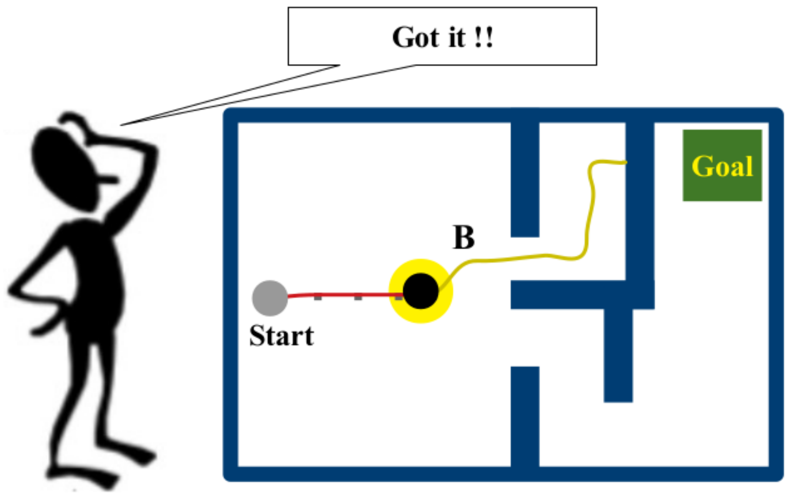


Figure 2: Contrastive plan

Preliminaries

- A **hybrid system** exhibits an interplay of discrete and continuous dynamics.

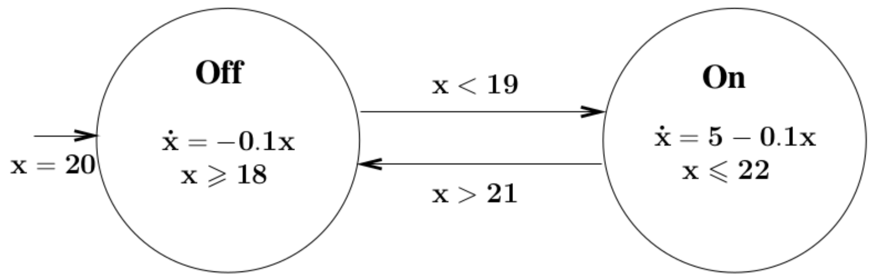


Figure 1: Hybrid thermostat

- PDDL+ extends PDDL2.1 for representing mixed discrete-continuous domains and planning problems.
- The key features supported in PDDL+ are the ability to model exogenous events and continuous evolution of the system.

- A **planning instance** Π in PDDL+ is a pair (Dom, Prob), where Dom is a 6-tuple (Fs, Rs, As, Es, Ps, arity) called the domain. Prob is a triplet (Os, I, G).
- A **plan** φ is a tuple ⟨λ, makespan⟩. For a planning instance with a set of ground actions A, λ is a finite set of triplets ⟨t, act, dur ⟩ together with the plan makespan ∈ R. In the triplet λ, t ∈ R^+ is the time instant of executing the action act ∈ A and dur ∈ R^+ is the duration for which the action act remains active in the plan. The makespan is the overall duration of the plan.
- Given a planning instance Π and a plan φ, an **action sequence** is an ordered set of ground actions in φ ordered by their time of appearance in φ.

Preliminaries

- An **explanation problem** is a tuple $E = \langle \Pi, \varphi, Q \rangle$, where Π represents the planning instance, φ is the plan generated by a planner, and Q represents the contrastive question posed by the user.
- **HModel** is a new planning instance Π' constructed from the planning instance Π which encapsulates the constraints of a user question Q. Π' can be defined as:

$$\Pi' = (\langle Fs', Rs', As', Es', Ps', arity' \rangle, \langle Os', I', G' \rangle)$$

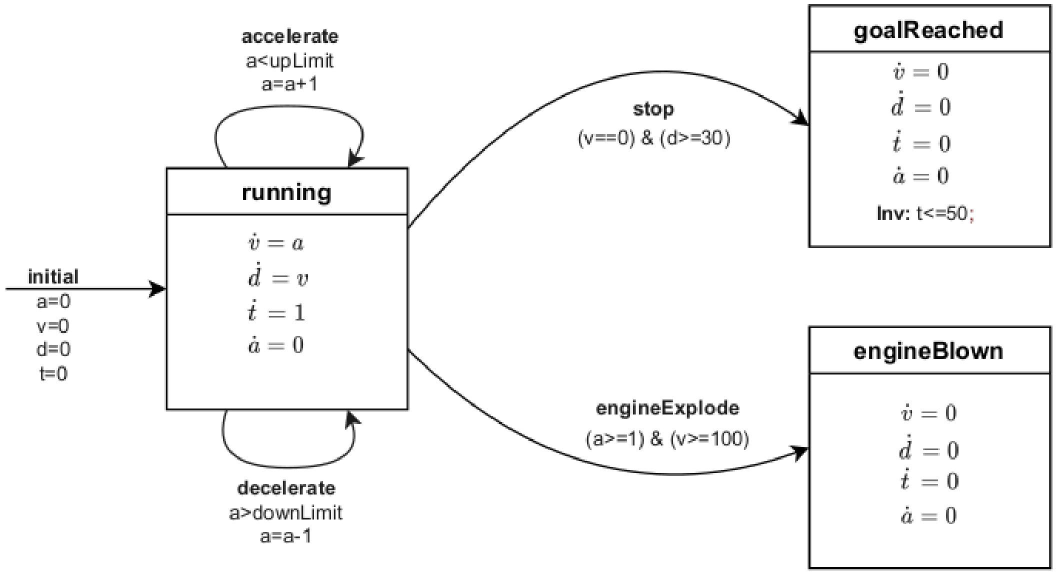
- **HPlan** is a hypothetical plan φ' produced by the planner over an HModel Π' which satisfies the constraint(s) posed in question Q by a user.

An example domain and a planning problem instance

- **Figure 1: Car domain in PDDL+:**

```
(define (domain car)
(:predicates (running) (engineBlown) (goalReached))
(:functions (d) (v) (a) (upLimit) (downLimit)
        (runningTime))
(:process moving
  :parameters ()
  :precondition (and (running))
  :effect (and (increase (v) (* #t (a)))
       (increase (d) (* #t (v)))
           (increase (runningTime) (* #t 1))))
(:action accelerate
    ...
(:action decelerate
    ...
(:event engineExplode
    ...
(:action stop
    ...
```

- **Figure 3: Hybrid automaton model of the domain:**



- **Figure 2: A problem instance in PDDL+:**

```
(define (problem car_prob)
    (:domain car)
    (:init (running) (= (runningTime) 0)
           (= (upLimit) 1) (= (downLimit) −1)
           (= d 0) (= a 0) (= v 0))
    (: goal ( and ( goalReached ) ( not ( engineBlown ) )
           ( ≤ ( runningTime ) 50 ) ) )
    (:metric minimize(total−time)))
```

- **Figure 4: The original plan by SMTPlan+:**

Time	**Action**	**Duration**
0.0:	(accelerate)	[0.0]
1.0:	(decelerate)	[0.0]
31.0:	(decelerate)	[0.0]
32.0:	(stop)	[0.0]

Our contrastive explanation framework

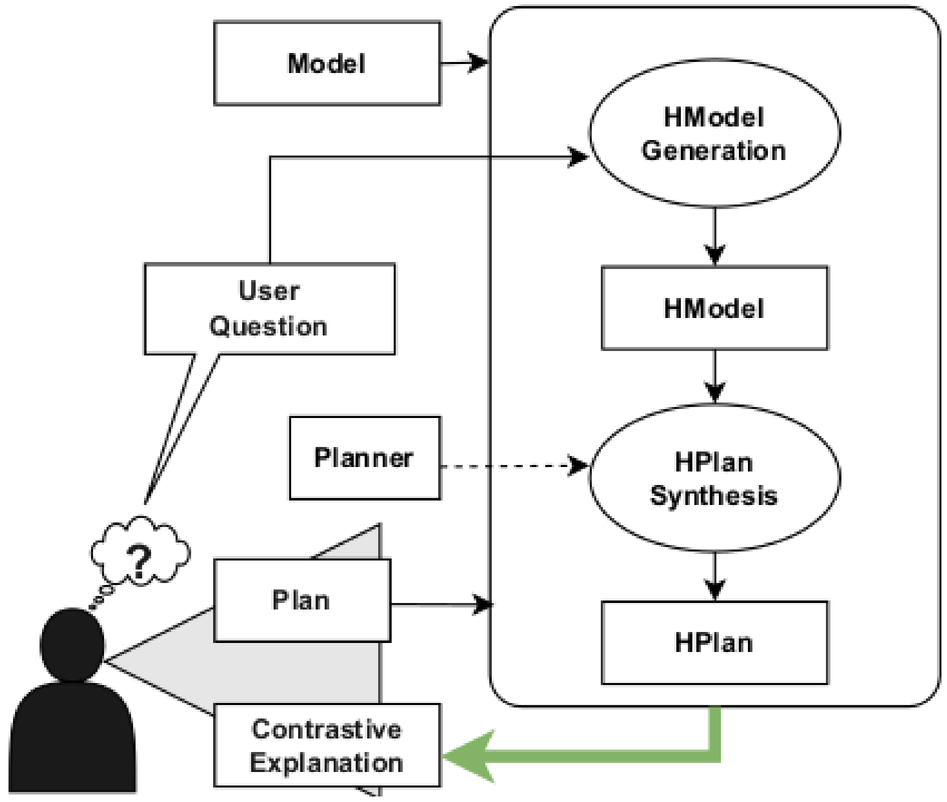


Figure 1: The Contrastive Plan Explanation Framework.

A set of contrastive questions

1. Why did the planner choose to do action A and not B instead?
2. Why did the planner not choose to do an action later in the plan?
3. Why did the planner not choose to do an action earlier in the plan?
4. Why did the planner choose to do an action in the plan, instead of not doing it?
5. Why not have fewer occurrences of an action in the plan?
6. Why is the accumulative duration of the plan not less?
7. Why did an action sequence appear in the plan?
8. Why is the length of the plan not less?

A. Why did the planner choose to do action A and not B instead?

Construction of the HModel for A:

- Let a plan φ consisting of an action sequence ⟨a_1 , a_2 , . . ., a_{i-1}, a_i , . . . , a_n⟩ where we want to replace the action a_i with an action **b**.
- We construct the HModel in such a way that any valid plan must consists of ⟨a_1, . . ., a_{i-1}, **b** , c_1, . . . , c_m⟩
  - Preserves ⟨a_1 , a_2 , . . ., a_{i-1}⟩.
  - Followed by **b** which replaces a_i in the original plan.
  - Followed by any other sequence of actions ⟨c_1, c_2, . . . , c_m⟩ which is leading to the goal.
- For example, the plan φ in the car domain is ⟨accelerate-decelerate-decelerate-stop⟩
- Suppose the user wants to replace the decelerate action at time 1.0 with an another accelerate action

Figure1 : The original plan.

Time	Action	Duration
0.0:	(accelerate)	[0.0]
1.0:	(decelerate)	[0.0]
31.0:	(decelerate)	[0.0]
32.0:	(stop)	[0.0]

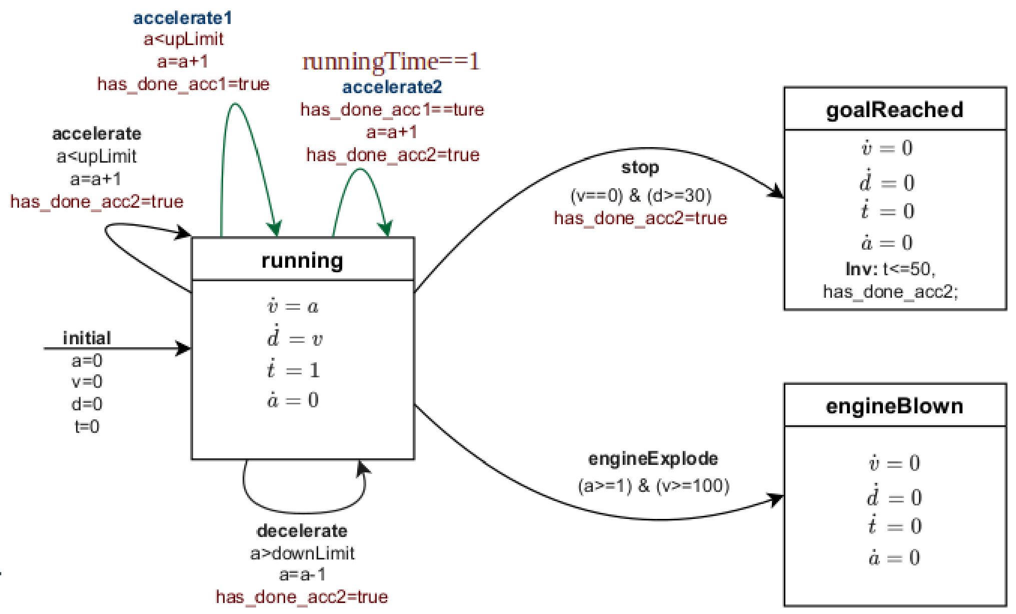
A. Why is the action (decelerate) used at time 1.0, and not action (accelerate)?

- **Construction of the HModel for A:**
- **The new planning instance Π′ (HModel) is:**
- **Π′ = ((Fs, Rs′ , As′ , Es, Ps, arity), (Os, I, G′))**

where

- **Rs′ : Rs ∪ {has_done_acc1, has_done_acc2}**
- **As′ : As ∪ {accelerate1, acccelerate2}**
- **G′ : G ∧ (has_done_acc1) ∧ (has_done_acc2)**

- **Figure 1: Hybrid automaton for the Hmodel:**



- **Figure 2: Hypothetical plan in SMTPlan+**

Time	Action	Duration
0.0:	(accelerate1)	[0.0]
1.0:	(accelerate2)	[0.0]
3.0:	(decelerate)	[0.0]
5.0:	(decelerate)	[0.0]
7.0:	(decelerate)	[0.0]
14.0:	(stop)	[0.0]

makespan: = 14.0 units.

Contrastive Explanation of A

- **Contrastive Explanation for the above user question is as follows:**
  - Remove: accelerate action is removed from the original plan;
  - Add: accelerate1 and accelerate2 are added to the alternate plan;
  - Common: decelerate and stop;
  - dwell-diff: {moving, 18};
  - diff-cost$_\tau$: -18;
  - diff-cost$_{len}$: 2.

- **User conclusion:**

  Replacing decelerate with an accelerate action at time instant 1 provides a shorter plan in terms of the plan duration but longer plan in terms of the number of applied actions.

B. Why did the planner not choose to do an action later in the plan?

Construction of the HModel for B:

- Let a plan φ consisting of an action sequence ⟨a_1 , a_2 , . . ., a_i , . . . , a_n⟩ where the action a_i appears in the plan at time **t**. We want to restrict the action a_i so that it may appear in a plan after the time t.
- HModel is constructed such that any valid plan if contains the action a_i it must appear after the time t
  - Introduce a time variable T_v, which captures the elapsed time in the system.
  - Restrict a_i to appear after $T_v > t$.
  -
- For example, the plan φ in the car domain is ⟨accelerate-**decelerate**-decelerate-stop⟩
- Suppose the user wants to restrict the **decelerate** action to appear after the **time 1.0**.

Figure1 : The original plan

Time	Action	Duration
0.0:	(accelerate)	[0.0]
1.0:	(decelerate)	[0.0]
31.0:	(decelerate)	[0.0]
32.0:	(stop)	[0.0]

B. Why is the action (decelerate) used at time 1.0, why not later?

- Construction of the HModel for B:
- The new planning instance Π′ is:

  Π′ = ((Fs, Rs, As′, Es, Ps, arity), (Os, I, G))

  where
- As′ : As ∪ {decelerate1} \ {decelerate}

- Figure 1: Hybrid Automaton for HModel of B.

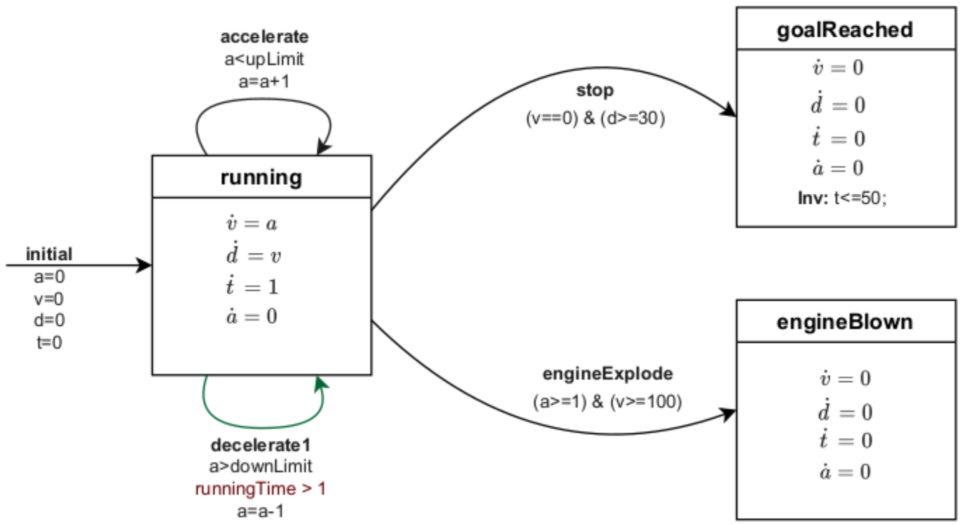
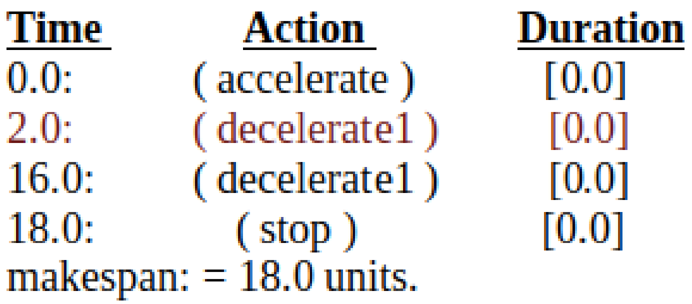


Figure 2: Resulting Hplan.



Time	Action	Duration
0.0:	(accelerate)	[0.0]
2.0:	(decelerate1)	[0.0]
16.0:	(decelerate1)	[0.0]
18.0:	(stop)	[0.0]

makespan: = 18.0 units.

Contrastive Explanation of B

- **Contrastive Explanation for the above user question is as follows:**
  - Remove: decelerate action is removed from the original plan;
  - Add: decelerate1 action is added to the alternate plan;
  - Common: accelerate, and stop;
  - dwell-diff: {moving, 14};
  - diff-cost$_\tau$: -14;
  - diff-cost$_{len}$: 0.

- **User conclusion:**
  Using the decelerate action later than 1 time unit results in plan of lesser duration in comparison to the original plan.

C. Why did the planner not choose to do an action earlier in the plan?

Construction of the HModel for C:

- **Let a plan φ consisting of an action sequence ⟨a_1 , a_2 , . . ., a_i , . . . , a_n⟩ where the action a_i appears in the plan at time t. We want to restrict the action a_i to appear in a plan before the time t.**
- **HModel is constructed such that the action a_i appears before the time t:**
  - **Introduce a time variable T_v, which captures the elapsed time in the system.**
  - **Restrict a_i to appear before $T_v < t$.**
  -
- **For example, the plan φ in the car domain is ⟨accelerate-decelerate-decelerate-stop⟩**
- **Suppose the user wants to restrict the decelerate action to appear before the time 1.0.**

- **Figure1 : The original plan:**

Time	**Action**	**Duration**
0.0:	(accelerate)	[0.0]
1.0:	(decelerate)	[0.0]
31.0:	(decelerate)	[0.0]
32.0:	(stop)	[0.0]

C. Why is that at time instant 1.0 the action (decelerate) is used, why not earlier?

- Construction of the HModel for C:
- The new planning instance Π′ is:

  Π′ = ((Fs, Rs′, As′, Es, Ps, arity), (Os, I, G′))

  where
- Rs′ = Rs ∪ {do_before_1}
- As′ = {decelerate1} ∪ As
- G′ = G ∧ (do_before_1)

- Figure 1: Hybrid automaton for HModel of C:

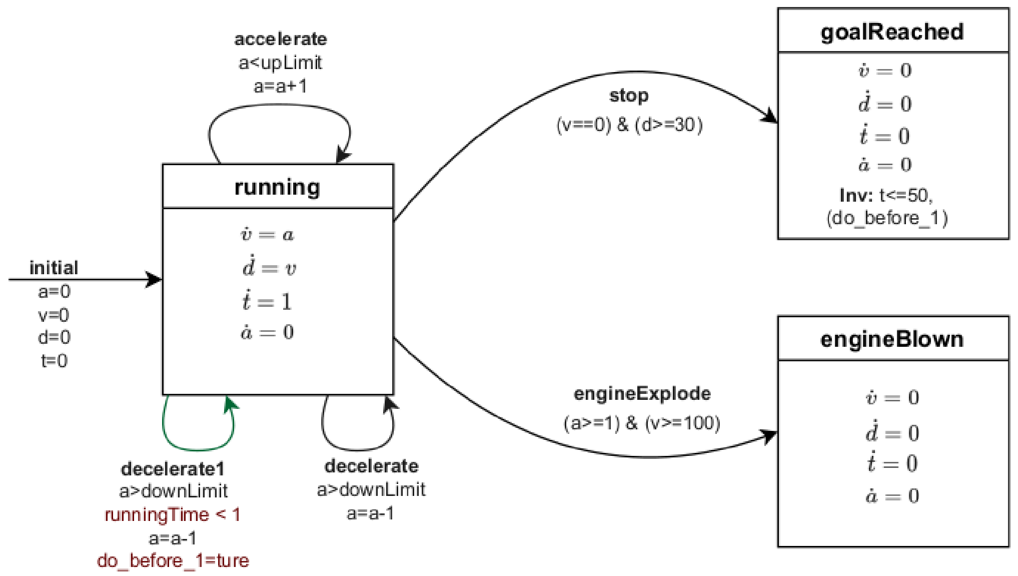


Figure 2: Resulting HPlan.

Time	Action	Duration
0.0:	(accelerate)	[0.0]
0.75:	(decelerate1)	[0.0]
40.75:	(decelerate)	[0.0]
41.5:	(stop)	[0.0]

makespan: = 41.5 units.

Contrastive Explanation of C

- **Contrastive Explanation for the above user question is as follows:**
  - Remove: No action is removed;
  - Add: decelerate1 action is added;
  - Common: accelerate, decelerate and stop;
  - dwell-diff: {moving, −9.5};
  - Diff-cost$_{\tau}$: 9.5;
  - diff-cost$_{len}$: 0.

- **User conclusion:**

  The application of (decelerate) action before than time instance 1.0 results in a sub-optimal plan in terms of the plan duration for this problem.

D. Why did the planner choose to do an action in the plan, instead of not doing it?

Construction of the HModel for D:

- Let the plan φ consisting of actions ⟨a_1 , a_2 , . . ., a_i , . . . , a_n⟩, the user might ask "why is the action a_i used in the plan, rather than not being used?"..
- A compilation is formed such that the action a_i is barred from appearing in a generated HPlan:
  - $As' = As \setminus \{a_i\}$, (\ represents the set difference operation)
- For example, the plan φ in the car domain is ⟨accelerate-decelerate-decelerate-stop⟩
- Suppose the user wants to bar the decelerate action to appear in a plan.

Figure1 : The original plan.

Time	Action	Duration
0.0:	(accelerate)	[0.0]
1.0:	(decelerate)	[0.0]
31.0:	(decelerate)	[0.0]
32.0:	(stop)	[0.0]

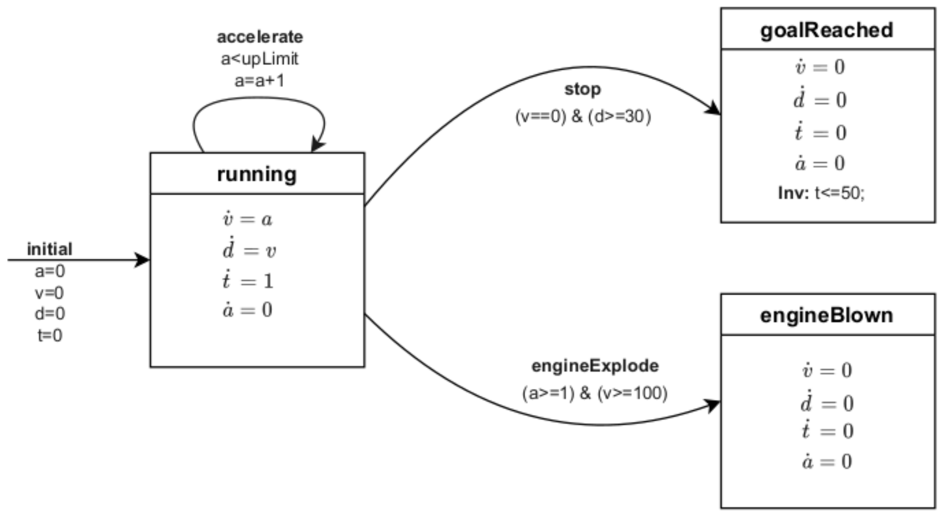
D. Why is action (decelerate) used in the plan, rather than not not being used?

- Construction of the HModel for D:
- The new planning instance Π′ is:

  Π′ = ((Fs, Rs, As′, Es, Ps, arity), (Os, I, G))

  where
- As′ = As \ {decelerate}

- Figure 1: Hybrid automaton for HModel of D.



- Hplan: No plan generated.

Contrastive Explanation of D

- **Contrastive Explanation for the above user question is as follows:**
  - Remove: Undefined, as there is no alternate plan;
  - Add: Undefined, as there is no alternate plan;
  - Common: Undefined, as there is no alternate plan;
  - dwell-diff: Can not be defined;
  - Diff-cost$_\tau$: ∞;
  - diff-cost$_{len}$: ∞.

- **User conclusion:**

  Barring the decelerate action to appear in the plan leads to no valid plan in this domain for this problem.

Note: Since the planning problem is undecidable for hybrid systems in general, when the planner does not generate a valid plan for a planning problem, it cannot be asserted whether it is due to the underlying undecidability or the problem is actually unsolvable. This leads to a limitation in our explanation framework since incorrect explanations may result in such cases.

E. Why not have fewer occurrences of an action in the plan?

Construction of the HModel for E:

- **Let the plan φ, where an action b appears n number of times. A user might ask "Why did the planner choose to do action b n number of times, why not less?".**
- **A compilation is formed such that the action b is restricted to appear less than n number of times in a generated HPlan:**
  - **Introduce a variable s that keeps track of each occurrance of b in a plan by increasing its value by 1.**
  - **($s < n$) is added as a goal contraint.**
- **For example, the plan φ in the car domain is ⟨accelerate-decelerate-decelerate-stop⟩**
- **Suppose the user wants to restrict the decelerate action to appear less than twice in a plan.**

Figure1 : The original plan.

Time	Action	Duration
0.0:	(accelerate)	[0.0]
1.0:	(decelerate)	[0.0]
31.0:	(decelerate)	[0.0]
32.0:	(stop)	[0.0]

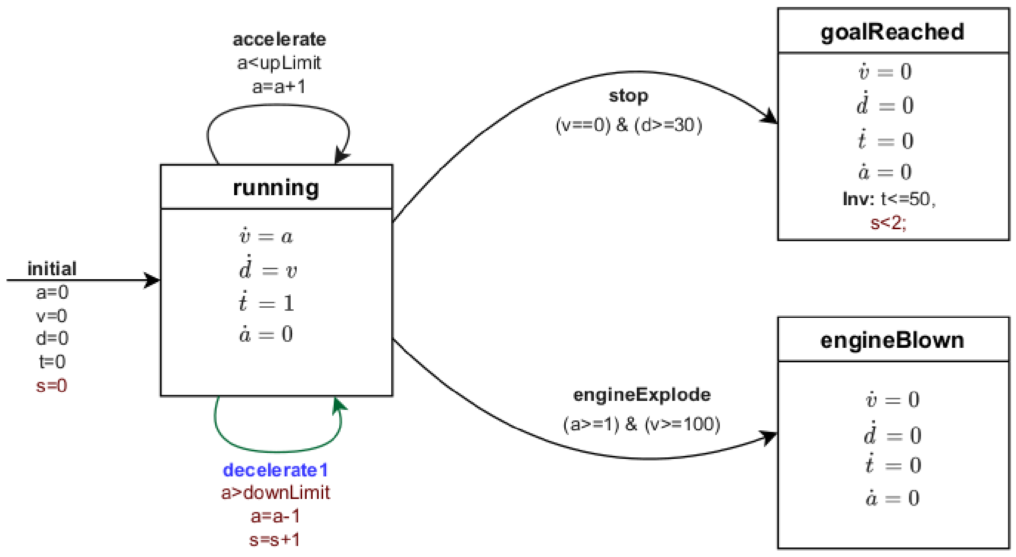
E. Why is the action (decelerate) taken twice in the plan, why not once?

- Construction of the HModel for E:
- The new planning instance Π′ is:

  Π′ = ((Fs′, Rs, As′, Es, Ps, arity), (Os, I′, G′))

  where
- Fs′ = Fs ∪ {s}
- As′ = As ∪ {decelerate1} \ {decelerate}
- I′ = I ∪ {PNE(s = 0), s ∈ Fs}
- G′ = G ∧ constraint(PNE(s) < 2)

- Figure 1: Hybrid automaton of HModel for E.



- Resulting Hplan: No plan generated.

Contrastive Explanation of E

- **Contrastive Explanation for the above user question is as follows:**
  - Remove: Undefined, as there is no alternate plan;
  - Add: Undefined, as there is no alternate plan;
  - Common: Undefined, as there is no alternate plan;
  - dwell-diff: Can not be defined;
  - Diff-cost$_\tau$: ∞;
  - diff-cost$_{len}$: ∞.

- **User conclusion:**

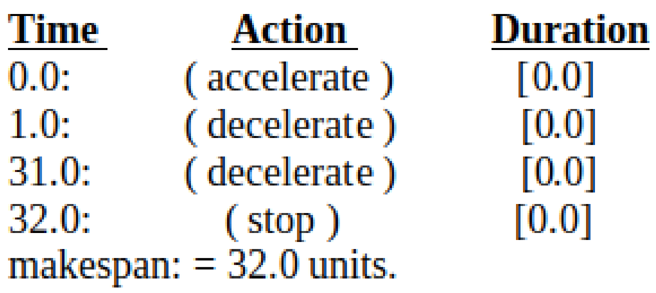
  No valid plan can be generated that has less than two executions of the action decelerate for this problem.

F. Why is the accumulative duration of the plan not less?

Construction of the HModel for F:

- **A user might question the optimality of the observed plan in terms of duration and ask "Why is the makespan of the plan not less?".**
- **Iterative HModel Formation: For a planning instance Π and a plan φ, let ψ_i be the set of user imposed constraints derived from φ_{i-1} which is initially empty, i.e. $\psi_0 = \emptyset$. Each stage i (initially 0) of this process starts with the planner producing a HPlan φ_i' for the Hmodel $\Pi_i' = \Pi \times \psi_i$, where × is the constraint operator which encapsulates ψ_i in Π_i'.**
- **For the plan φ in Figure 1, the question might be "Why is the makespan of the plan not less than 32?".**

Figure1 : The original plan.



```
Time        Action           Duration
0.0:      ( accelerate )      [0.0]
1.0:      ( decelerate )      [0.0]
31.0:     ( decelerate )      [0.0]
32.0:        ( stop )         [0.0]
makespan: = 32.0 units.
```

F. Questioning optimality of the Plan Duration: Why is the accumulative duration of the plan not less than 32?

- **Construction of the HModel for F:**
- **The new planning instance Π′ is given below:**
  - $\Pi' = \Pi'_{prev\_iter} \cup \{constraints(\varphi'_{prev\_iter})\}$

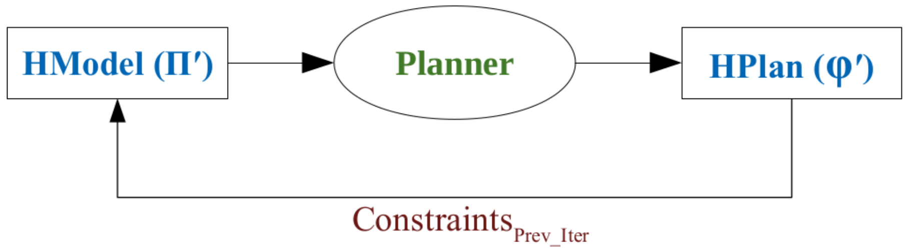


Figure 1: Iterative Hmodel.

- **For example, in our original problem where the constraint *runningTime ≤ 50* leads to a solution of duration 32.0. So in the next iteration in Π′ , the constraint is modified to *runningTime < 32* to find a plan with duration less than 32.0 time units if any such plan exists.**

- **Table 1: Duration of the Hypothetical plan in each re-plan iteration:**

Iteration	Constraint	Duration
1	≤ 50	32
2	< 32	31.5
3	< 31	18
4	< 18	17.5
5	< 17	14
6	< 14	13.5
7	< 13	12
8	< 12	11.7.5
9	< 11	10.96
10	< 10.96	10.95
11	< 10.95	∞

Contrastive Explanation of F

- **Contrastive Explanation for the above user question is as follows:**
  - Remove: No action is removed from the original plan;
  - Add: No new action is added in each iteration;
  - Common: accelerate, decelerate and stop;
  - The dwell-diff and diff-cost$_\tau$ of the HModel Π' for each iteration is given in table I
  - The diff-cost$_{len}$ for each iteration is 0 as all alternate plans are of same length except for the no-plan where length is ∞.

- **Table 1: dwell-diff and diff-cost$_\tau$ of the Hypothetical plan in each re-plan iteration:**

Iteration	dwell-diff	diff-cost$_\tau$
1	–	–
2	{moving, 0.5}	−0.5
3	{moving, 13.5}	−13.5
4	{moving, 0.5}	−0.5
5	{moving, 3.5}	−3.5
6	{moving, 0.5}	−0.5
7	{moving, 1.5}	−1.5
8	{moving, 0.25}	−0.25
9	{moving, 0.79}	−0.79
10	{moving, 0.01}	−0.01
11	NA	∞

- **User conclusion:**

  The accumulative duration of the original plan is not optimal and there can be alternate plans with lesser duration for this given problem.

G. Why did an action sequence appear in the plan?

Construction of the HModel for G:

- Let a plan φ consisting of an action sequence ⟨a_1, a_2, … , a_{i-1}, $\mathbf{a_i}$, $\mathbf{a_{i+1}}$, … , $\mathbf{a_{i+k}}$, a_{i+k+1}, … , a_m⟩. A user might ask why does the action sequence ⟨$\mathbf{a_i}$, $\mathbf{a_{i+1}}$, … , $\mathbf{a_{i+k}}$⟩ appear in the plan, why not any other sequence?
- We construct the HModel in such a way that any valid plan
  - Preserves the pre-sequence ⟨a_1 , a_2 , . . ., a_{i-1}⟩ in the plan.
  - Preserves the post-sequence ⟨a_{i+k+1}, … , a_m⟩ in the plan.
  - Any sequence other than the forbidden sequence may appear in between the pre-sequence and the post-sequence.
- For example, the plan φ in the car domain is ⟨accelerate-**decelerate-decelerate**-stop⟩
- Suppose the user wants to know is there any other sequence than **decelerate-decelerate** that can appear a plan.

Figure1 : The original plan.

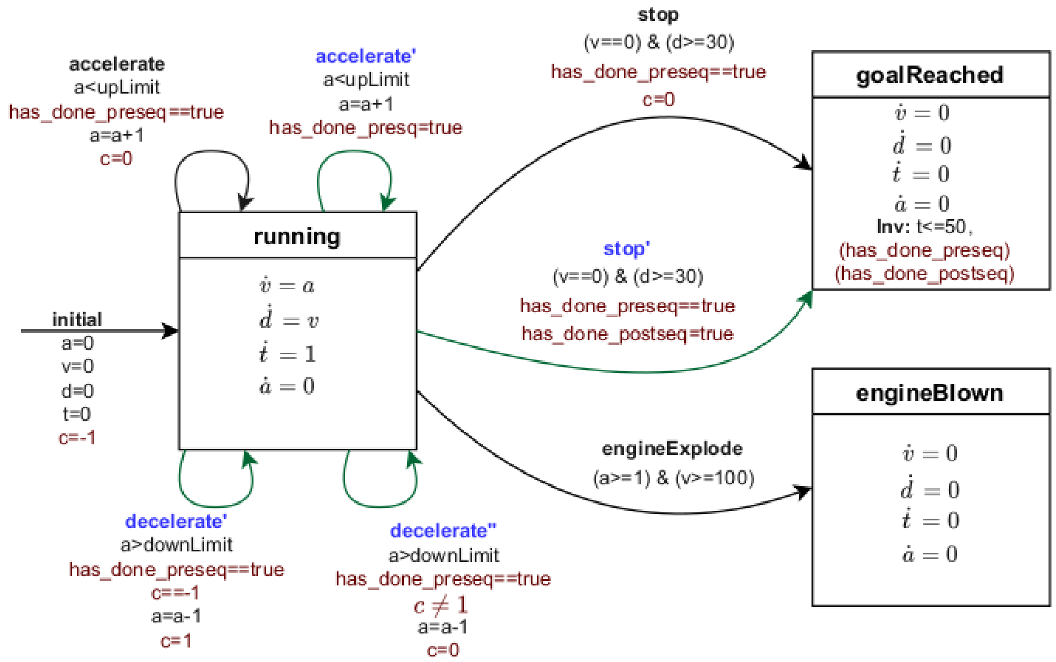
Time	Action	Duration
0.0:	(accelerate)	[0.0]
1.0:	(decelerate)	[0.0]
31.0:	(decelerate)	[0.0]
32.0:	(stop)	[0.0]

G.Why does the action sequence decelerate-decelerate appear in the plan ⟨accelerate-decelerate-decelerate-stop⟩, why not some other sequence?

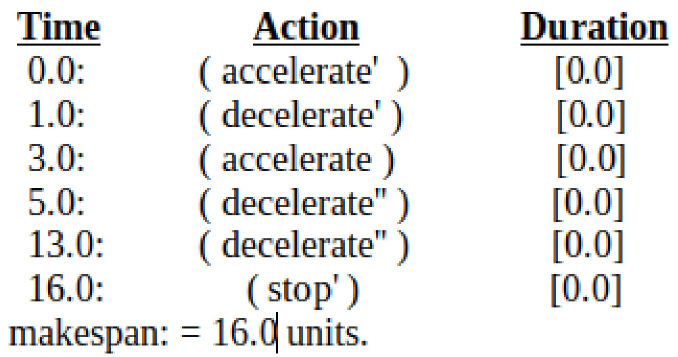
- **Construction of the HModel for G:**
- **The new planning instance Π′ is:**
- **Π′ = ((Fs′, Rs′, As′, Es, Ps, arity), (Os, I′, G′))**

  where
- **Fs′ = Fs ∪ {c}**
- **Rs′= Rs ∪ {has_done_preseq, has_done_postseq}**
- **As′ ={accelerate′, decelerate′, decelerate′′, stop′}∪As \ {decelerate}**
- **I′ = I ∪ {PNE(c = -1), c ∈ Fs}**
- **G′ = G ∧ (has_done_preseq) ∧ (has_done_postseq)**

- **Figure 1: Hybrid automaton of HModel for G.**



- **Figure 2: Resulting HPlan:**



Time	Action	Duration
0.0:	(accelerate')	[0.0]
1.0:	(decelerate')	[0.0]
3.0:	(accelerate)	[0.0]
5.0:	(decelerate'')	[0.0]
13.0:	(decelerate'')	[0.0]
16.0:	(stop')	[0.0]

makespan: = 16.0 units.

Contrastive Explanation of G

- **Contrastive Explanation for the above user question is as follows:**
  - Remove: decelerate and stop action are removed in the alternate plan;
  - Add: accelerate', decelerate', decelerate'', and stop' are added;
  - Common: accelerate;
  - dwell-diff: {moving, 16};
  - Diff-cost$_\tau$: -16;
  - diff-cost$_{len}$: 2.

- **User conclusion:**

  For the given problem instance, there exist plans that consist of action sequences other than the sequence decelerate-decelerate, however, the plan length is greater than that of the original one.

H. Why is the length of the plan not less?

Construction of the HModel for H:

- **Let φ is a plan of length k. A user might question the optimality of the observed plan in terms of plan-length and ask "Why is the length of the plan k and not less?".**
- **We construct the HModel in such a way that any valid plan should be of plan-length less than k.**
  - **Introduces a variable s which keeps track the number of actions that appear in a plan.**
  - **(s < k) is added as a goal contraint.**
- **For example, the plan φ in the car domain is ⟨accelerate-decelerate-decelerate-stop⟩ which is of plan-length 4. The question might be "Why is the length of the plan not less than 4?".**

- **Figure1 : The original plan:**

Time	Action	Duration
0.0:	(accelerate)	[0.0]
1.0:	(decelerate)	[0.0]
31.0:	(decelerate)	[0.0]
32.0:	(stop)	[0.0]

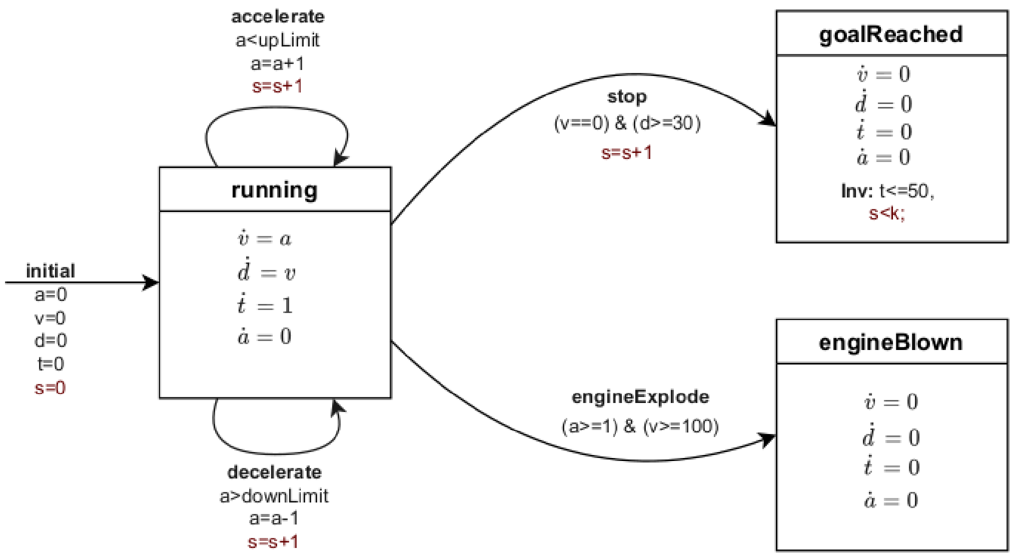
H. Why is the length of the plan is not less than 4?

- Construction of the HModel for E:
- The new planning instance Π′ is:

  Π′ = ((Fs′, Rs, As, Es, Ps, arity), (Os, I′, G′))

  where
- Fs′ = Fs ∪ {s}
- I′ = I ∪ { (s = 0), s ∈ Fs}
- G′ = G ∧ constraint (s < 4)

- Figure 1: Hybrid automaton of HModel for E.



- Resulting Hplan: No plan generated.

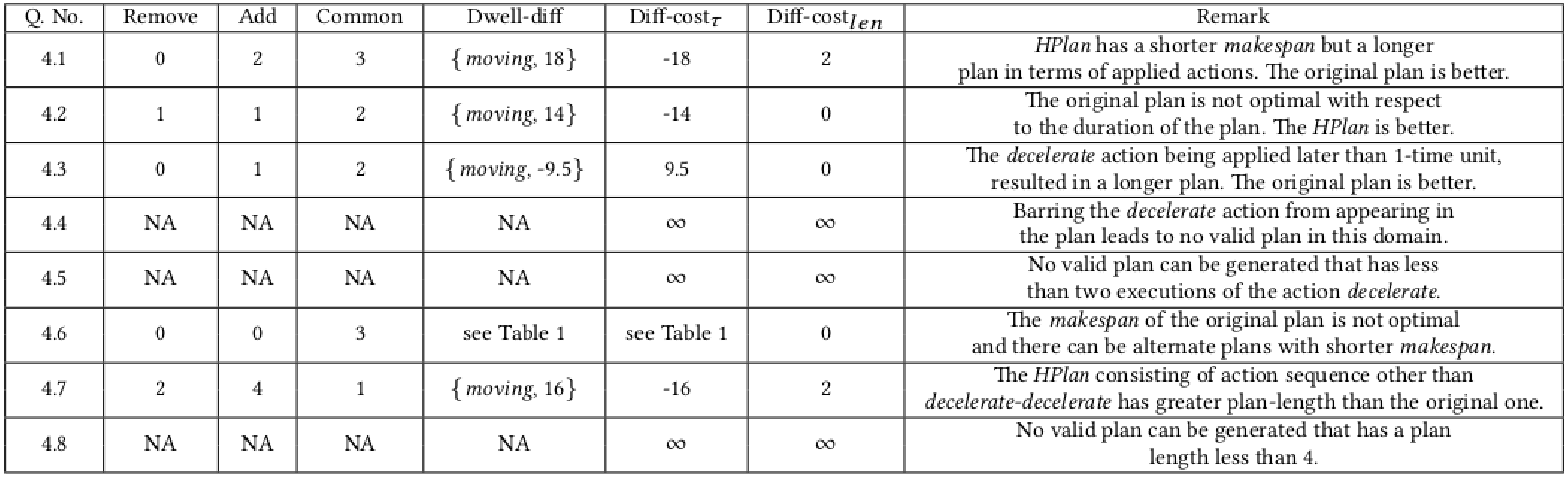
Contrastive Explanation of H

- **Contrastive Explanation for the above user question is as follows:**
  - Remove: Undefined, as there is no alternate plan;
  - Add: Undefined, as there is no alternate plan;
  - Common: Undefined, as there is no alternate plan;
  - dwell-diff: Can not be defined;
  - Diff-cost_{τ}: ∞;
  - diff-cost_{len}: ∞.

- **User conclusion:**

  For the problem instance, no valid plan can be generated that has a plan length less than 4.

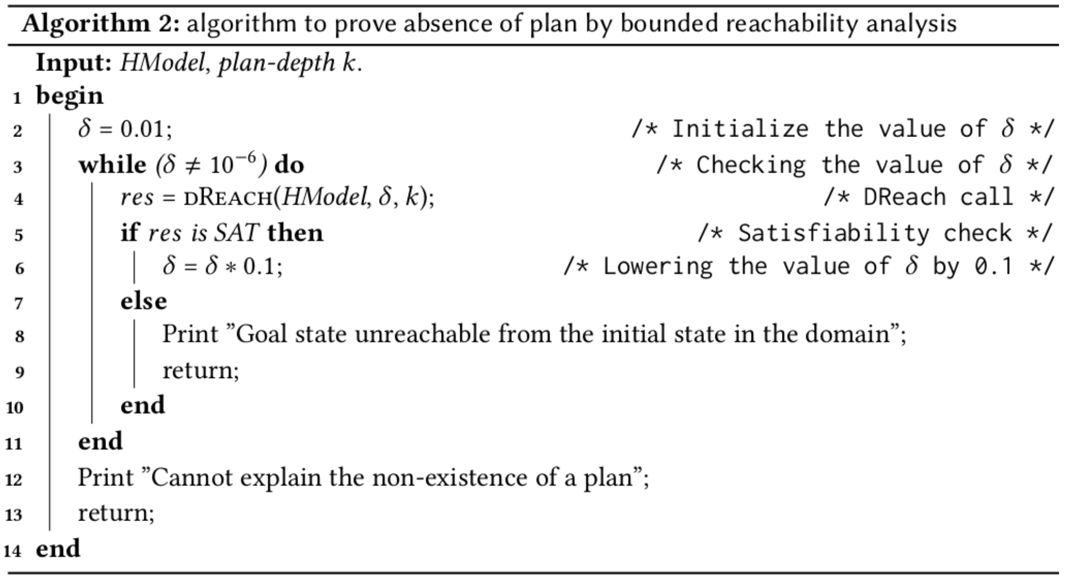
Summary of Contrastive Explanations



Q. No.	Remove	Add	Common	Dwell-diff	Diff-cost$_{\tau}$	Diff-cost$_{len}$	Remark
4.1	0	2	3	$\{$*moving*, 18$\}$	-18	2	*HPlan* has a shorter *makespan* but a longer plan in terms of applied actions. The original plan is better.
4.2	1	1	2	$\{$*moving*, 14$\}$	-14	0	The original plan is not optimal with respect to the duration of the plan. The *HPlan* is better.
4.3	0	1	2	$\{$*moving*, -9.5$\}$	9.5	0	The *decelerate* action being applied later than 1-time unit, resulted in a longer plan. The original plan is better.
4.4	NA	NA	NA	NA	∞	∞	Barring the *decelerate* action from appearing in the plan leads to no valid plan in this domain.
4.5	NA	NA	NA	NA	∞	∞	No valid plan can be generated that has less than two executions of the action *decelerate*.
4.6	0	0	3	see Table 1	see Table 1	0	The *makespan* of the original plan is not optimal and there can be alternate plans with shorter *makespan*.
4.7	2	4	1	$\{$*moving*, 16$\}$	-16	2	The *HPlan* consisting of action sequence other than *decelerate-decelerate* has greater plan-length than the original one.
4.8	NA	NA	NA	NA	∞	∞	No valid plan can be generated that has a plan length less than 4.

Proving The Absence of a Plan

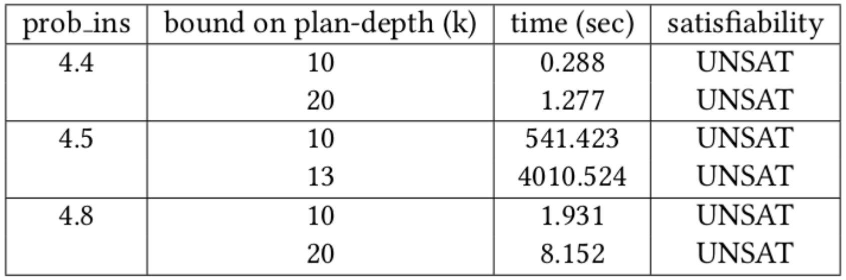
- **δ-Approximation: The δ-approximation of a system can be best explained in terms the δ-weakening of the corresponding hybrid automata. Let $\delta \in Q^{+} \cup \{0\}$ be an arbitrary rational number and H = (Loc, Var, Flow, Init, Lab, Edge, Inv) represents a hybrid automaton.**
- **A δ-weakened hybrid system as H^{δ} : H^{δ} = (Loc, Var, $Flow^{\delta}$, $Init^{\delta}$, Lab, $Edge^{\delta}$, Inv^{δ})**
- **For a given problem instance Π, absence of any plan in H^{δ} implies the absence of any plan in H.**



Algorithm 2: algorithm to prove absence of plan by bounded reachability analysis

```
Input: HModel, plan-depth k.
begin
    $\delta = 0.01$;                                  /* Initialize the value of $\delta$ */
    while ($\delta \neq 10^{-6}$) do                   /* Checking the value of $\delta$ */
        res = DREACH(HModel, $\delta$, k);            /* DReach call */
        if res is SAT then                        /* Satisfiability check */
            $\delta = \delta * 0.1$;                  /* Lowering the value of $\delta$ by 0.1 */
        else
            Print "Goal state unreachable from the initial state in the domain";
            return;
        end
    end
    Print "Cannot explain the non-existence of a plan";
    return;
end
```

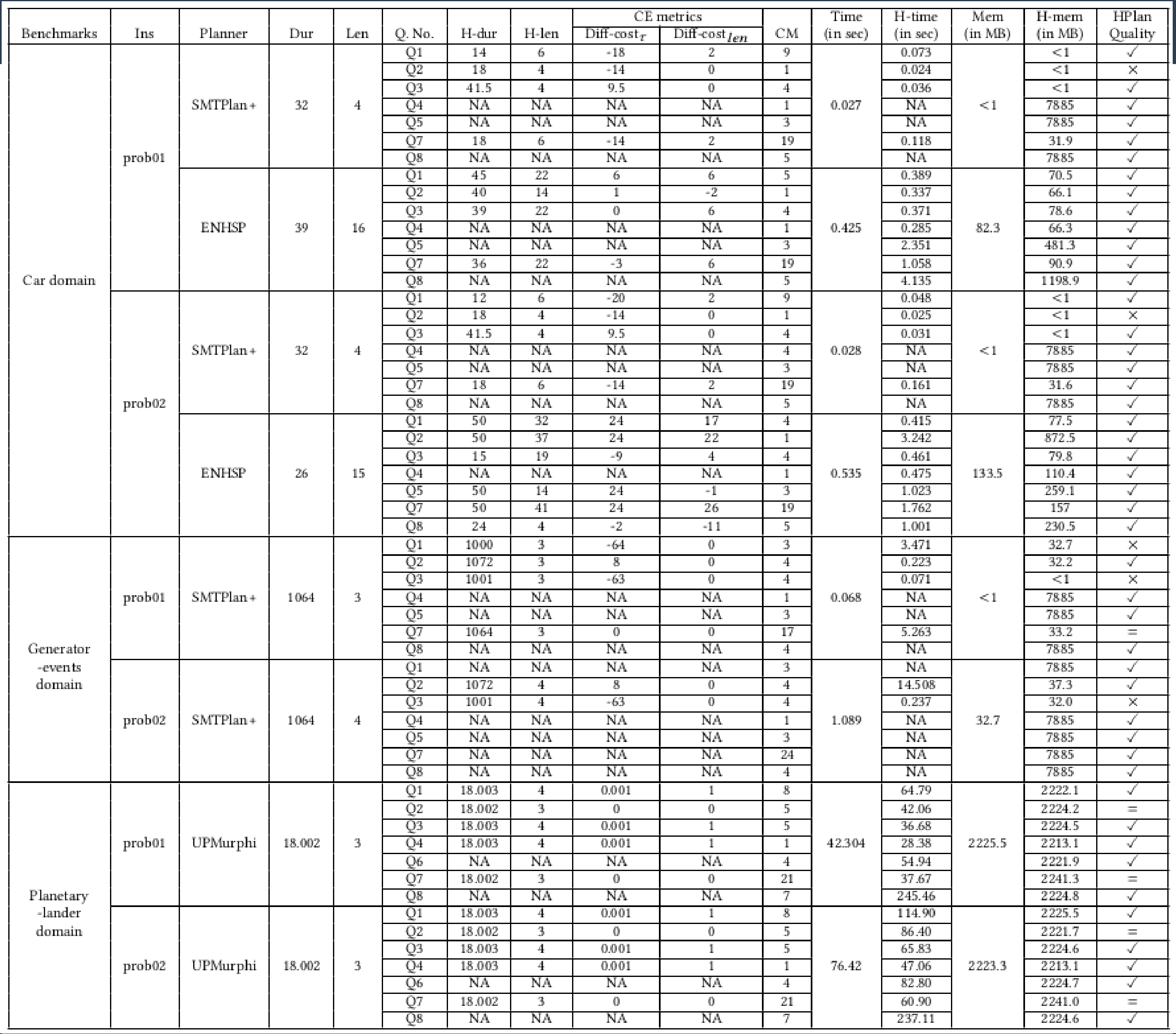
Figure 1: Algorithm for bounded reachability analysis



prob_ins	bound on plan-depth (k)	time (sec)	satisfiability
4.4	10	0.288	UNSAT
	20	1.277	UNSAT
4.5	10	541.423	UNSAT
	13	4010.524	UNSAT
4.8	10	1.931	UNSAT
	20	8.152	UNSAT

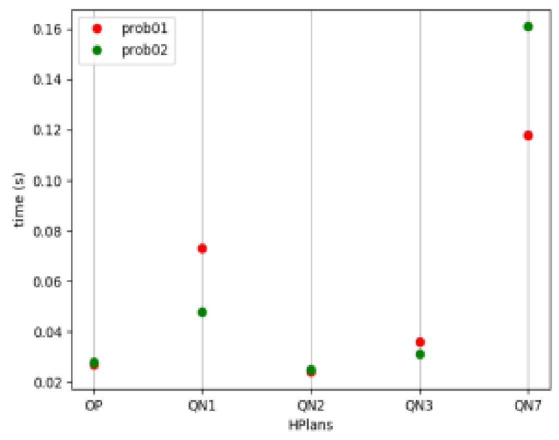
Figure 2: Bounded reachability analysis of the no-plan-models for a given δ perturbation of 0.01.

Evaluation : **We evaluated the performance of our framework with respected to the contrastive questions presented, working with three benchmark PDDL+ planning domains and three state-of-the-art planners.**

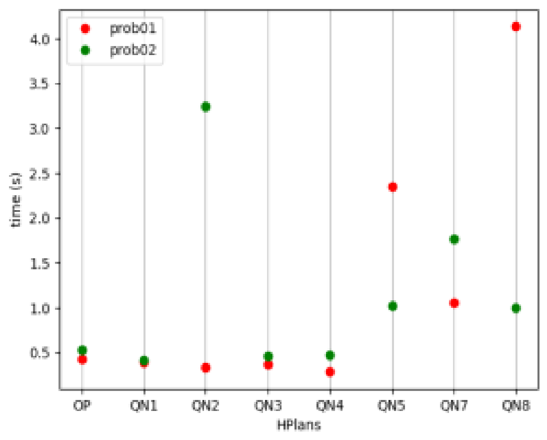


Benchmarks	Ins	Planner	Dur	Len	Q. No.	H-dur	H-len	CE metrics		CM	Time	H-time	Mem	H-mem	HPlan
								Diff-cost_T	Diff-cost_{len}		(in sec)	(in sec)	(in MB)	(in MB)	Quality
Car domain	prob01	SMTPlan+	32	4	Q1	14	6	-18	2	9	0.027	0.073	<1	<1	✓
					Q2	18	4	-14	0	1		0.024		<1	×
					Q3	41.5	4	9.5	0	4		0.036		<1	✓
					Q4	NA	NA	NA	NA	1		NA		7885	✓
					Q5	NA	NA	NA	NA	3		NA		7885	✓
					Q7	18	6	-14	2	19		0.118		31.9	✓
					Q8	NA	NA	NA	NA	5		NA		7885	✓
		ENHSP	39	16	Q1	45	22	6	6	5	0.425	0.389	82.3	70.5	✓
					Q2	40	14	1	-2	1		0.337		66.1	✓
					Q3	39	22	0	6	4		0.371		78.6	✓
					Q4	NA	NA	NA	NA	1		0.285		66.3	✓
					Q5	NA	NA	NA	NA	3		2.351		481.3	✓
					Q7	36	22	-3	6	19		1.058		90.9	✓
					Q8	NA	NA	NA	NA	5		4.135		1198.9	✓
	prob02	SMTPlan+	32	4	Q1	12	6	-20	2	9	0.028	0.048	<1	<1	✓
					Q2	18	4	-14	0	1		0.025		<1	×
					Q3	41.5	4	9.5	0	4		0.031		<1	✓
					Q4	NA	NA	NA	NA	4		NA		7885	✓
					Q5	NA	NA	NA	NA	3		NA		7885	✓
					Q7	18	6	-14	2	19		0.161		31.6	✓
					Q8	NA	NA	NA	NA	5		NA		7885	✓
		ENHSP	26	15	Q1	50	32	24	17	4	0.535	0.415	133.5	77.5	✓
					Q2	50	37	24	22	1		3.242		872.5	✓
					Q3	15	19	-9	4	4		0.461		79.8	✓
					Q4	NA	NA	NA	NA	1		0.475		110.4	✓
					Q5	50	14	24	-1	3		1.023		259.1	✓
					Q7	50	41	24	26	19		1.762		157	✓
					Q8	24	4	-2	-11	5		1.001		230.5	✓
Generator -events domain	prob01	SMTPlan+	1064	3	Q1	1000	3	-64	0	3	0.068	3.471	<1	32.7	×
					Q2	1072	3	8	0	4		0.223		32.2	✓
					Q3	1001	3	-63	0	4		0.071		<1	×
					Q4	NA	NA	NA	NA	1		NA		7885	✓
					Q5	NA	NA	NA	NA	3		NA		7885	✓
					Q7	1064	3	0	0	17		5.263		33.2	=
					Q8	NA	NA	NA	NA	4		NA		7885	✓
	prob02	SMTPlan+	1064	4	Q1	NA	NA	NA	NA	3	1.089	NA	32.7	7885	✓
					Q2	1072	4	8	0	4		14.508		37.3	✓
					Q3	1001	4	-63	0	4		0.237		32.0	×
					Q4	NA	NA	NA	NA	1		NA		7885	✓
					Q5	NA	NA	NA	NA	3		NA		7885	✓
					Q7	NA	NA	NA	NA	24		NA		7885	✓
					Q8	NA	NA	NA	NA	4		NA		7885	✓
Planetary -lander domain	prob01	UPMurphi	18.002	3	Q1	18.003	4	0.001	1	8	42.304	64.79	2225.5	2222.1	✓
					Q2	18.002	3	0	0	5		42.06		2224.2	=
					Q3	18.003	4	0.001	1	5		36.68		2224.5	✓
					Q4	18.003	4	0.001	1	1		28.38		2213.1	✓
					Q6	NA	NA	NA	NA	4		54.94		2221.9	✓
					Q7	18.002	3	0	0	21		37.67		2241.3	=
					Q8	NA	NA	NA	NA	7		245.46		2224.8	✓
	prob02	UPMurphi	18.002	3	Q1	18.003	4	0.001	1	8	76.42	114.90	2223.3	2225.5	✓
					Q2	18.002	3	0	0	5		86.40		2221.7	=
					Q3	18.003	4	0.001	1	5		65.83		2224.6	✓
					Q4	18.003	4	0.001	1	1		47.06		2213.1	✓
					Q6	NA	NA	NA	NA	4		82.80		2224.7	✓
					Q7	18.002	3	0	0	21		60.90		2241.0	=
					Q8	NA	NA	NA	NA	7		237.11		2224.6	✓

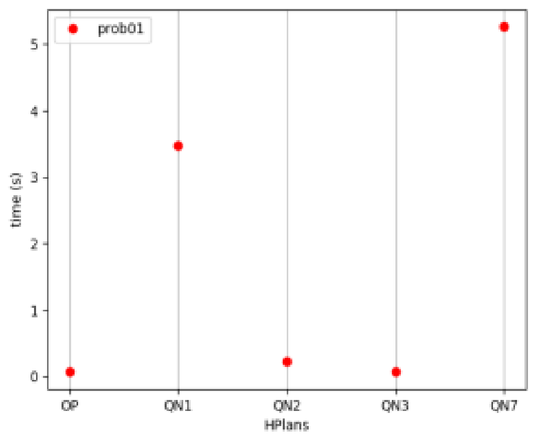
Evaluation :



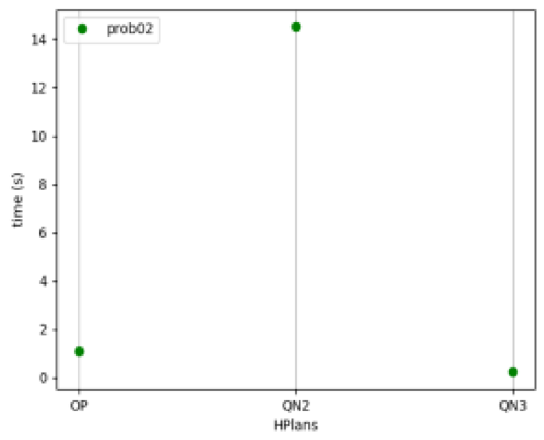
(a) The *HPlan* generation times using SMTPlan+ for two problem instances in Car domain.



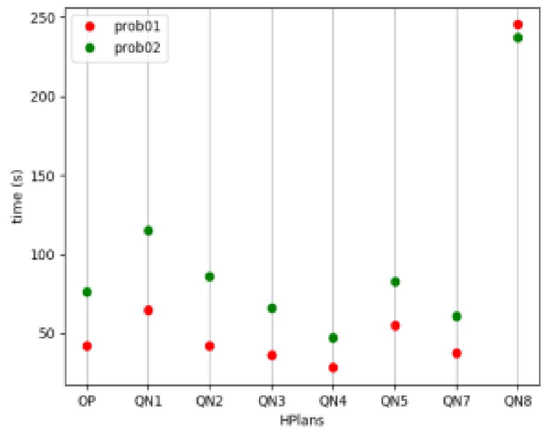
(b) The *HPlan* generation times using ENHSP for two problem instances in Car domain.



(c) The *HPlan* generation times using SMTPlan+ for prob-ins 1 in Generator-events domain.

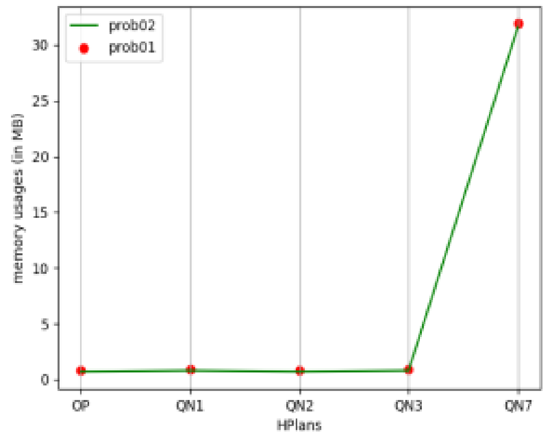


(d) The *HPlan* generation times using SMTPlan+ for prob-ins 2 in Generator-events domain.

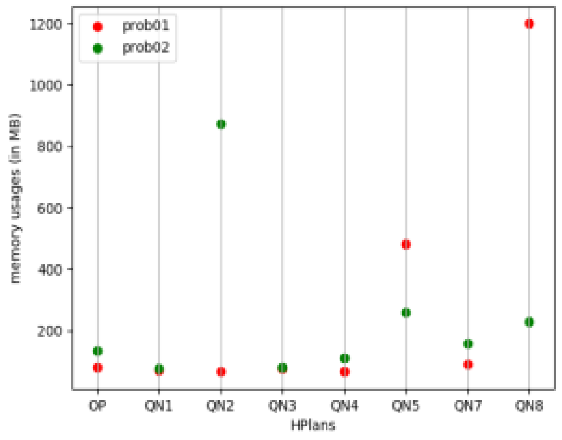


(e) The *HPlan* generation times using UPMurphi for instances in Planetary-lander domain.

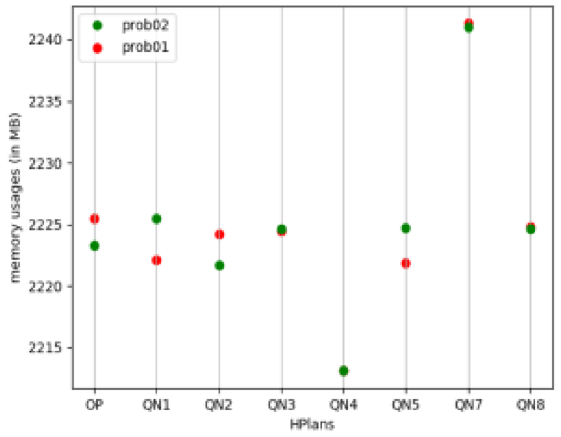
Evaluation :



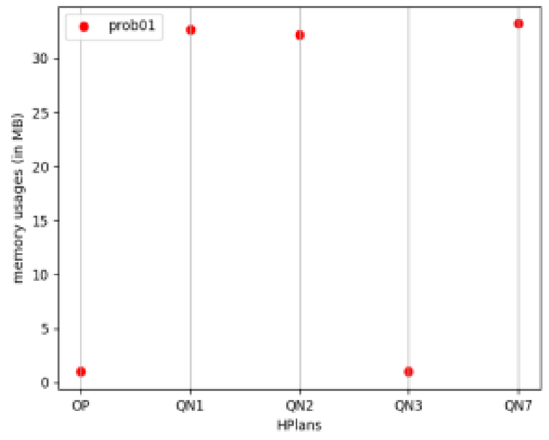
(a) The memory usage in *HPlan* generation using SMTPlan+ for problem instances in Car domain.



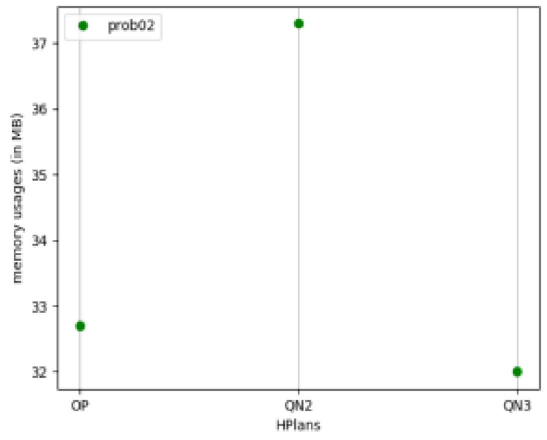
(b) The memory usage in *HPlan* generation using ENHSP for problem instances in Car domain.



(e) The memory usage of the *HPlan* generation by UPMurphi for instances in Planetary-lander.



(c) The memory usage in *HPlan* generation using SMTPlan+for prob-ins 1 in Generator-events.



(d) The memory usage in *HPlan* generation using SMTPlan+ for prob-ins 2 in Generator-events.

Conclusion

- We propose a contrastive explanation framework to provide explanations to the plans in Hybrid System models.
- We show that our contrastive explanations can draw conclusions about the planning domain and the planning tool as well, such as identifying that the plan is not always necessarily cost optimal.
- No-plans are also helpful to figure out the critical actions for a certain planning problem.
- Further, we provide a no-plan explanation algorithm for our no-plan-models through bounded reachability analysis to verify the reachability of the problem instances.
- We believe our framework can be of immense importance to the hybrid systems planning community for synthesizing better explainable plans.

References

1. Tathagata Chakraborti, Anagha Kulkarni, Sarath Sreedharan, David E. Smith, and Subbarao Kambhampati. 2018. Explicability? Legibility? Predictability? Transparency? Privacy? Security? The Emerging Landscape of Interpretable Agent Behavior. arXiv e-prints, Article arXiv:1811.09722 (Nov. 2018), arXiv:1811.09722 pages. arXiv:1811.09722 [cs.AI]

2. Tathagata Chakraborti, Sarath Sreedharan, Yu Zhang, and Subbarao Kambhampati. 2017. Plan explanations as model reconciliation: Moving beyond explanation as soliloquy. In 26th International Joint Conference on Artificial Intelligence, IJCAI 2017. International Joint Conferences on Artificial Intelligence, 156–163. 26th International Joint Conference on Artificial Intelligence, IJCAI 2017 ; Conference date: 19-08-2017 Through 25-08-2017.

3. Benjamin Krarup, Michael Cashmore, Daniele Magazzeni, and Tim Miller. 2019. Model-based contrastive explanations for explainable planning. In ICAPS 2019 Workshop on Explainable AI Planning (XAIP). AAAI Press, USA. https://strathprints.strath.ac.uk/69957/

4. David E. Smith. 2012. Planning as an Iterative Process. In Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence (Toronto, Ontario, Canada) (AAAI'12). AAAI Press, 2180–2185.

5. Sicun Gao, Soonho Kong, Wei Chen, and Edmund M. Clarke. 2014. Delta-Complete Analysis for Bounded Reachability of Hybrid Systems. CoRR abs/1404.7171 (2014). arXiv:1404.7171 http://arxiv.org/abs/1404.7171

THANK YOU