Proof for Proposition 2

Proposition 2. *Given a planning instance P and an admissible set T, the following claims hold.*

1. *There exists an atomic conjunction $t \in T$ such that any optimal plan to t is also an optimal plan for P.*
2. *There exists an admissible sequence $\tau = (t_0, t_1, \ldots, t_m)$ with $m \geq 0$ and $t_i \in T$ for all $0 \leq i \leq m$.*

Proof. Let $T = \{t_0, t_1, \ldots, t_n\}$. In order to prove the first claim, we show that (i) there exists an atomic conjunction $t \in T$ such that one of its optimal plan is an optimal plan for P and then we prove that (ii) for this t, all of its optimal plans are optimal plans for P. For the second claim, we construct a sequence of atomic conjunctions and show it is an admissible sequence.

First Claim (i) Assume, towards a contradiction, that for all i, any optimal plans to t_i is not an optimal plan for P.

Since T is admissible, there exists an atomic conjunction which is true in the initial state. W.O.L.G., let $t_0 \subseteq s_0$. The empty plan ( ) is an optimal plan for t_0 and as we assume that it is not an optimal plan for P, so there must exist an atomic conjunction in T (W.O.L.G., call it t_1) such that the empty plan ( ) can be extended by one action a_1 to reach t_1 optimally. Therefore $\pi_1 = (a_1)$ is an optimal plan for t_1, but not an optimal plan for P. As T is admissible, (a_1) can be extended by one action a_2 to reach t_2 optimally, i.e., $\pi_2 = (a_1, a_2)$ is an optimal plan for t_2. However π_2 is not an optimal plan for P. We continue this process and obtain a sequence of actions $(a_1, a_2, \ldots)$ and a sequence of atomic conjunctions $(t_0, t_1, t_2, \ldots)$. It can be seen that for each i, the action sequence π_i is an optimal plan for t_i, but not an optimal plan for P per assumption.

Now we show that all atomic conjunctions in the sequence $(t_0, t_1, t_2, \ldots,)$ are distinct. If not, assume $t_j = t_k$ with $j < k$, then both π_j and π_k can reach t_j optimally, but the length of π_j is strictly shorter than the length of π_k, contradicts.

Since T contains finitely many atomic conjunctions, this sequence must end and let t_m be the last atomic conjunction in this sequence and denote it as $\tau \triangleq (t_0, t_1, \ldots, t_m)$. We show that $\pi_m = (a_1, a_2, \ldots, a_m)$ is an optimal plan for P. If not, π_m can be extended by one action to reach another atomic conjunction in T optimally, so t_m is not the last element in this atomic conjunction sequence, contradicts. Therefore, π_m is an optimal plan for P and we know the length of the optimal plans for P is m.

First Claim (ii) Now we show that any optimal plan to t_m must be an optimal plan for P.

Assume that there exists an optimal plan π to t_m which is not optimal for P. We know the length of π is m from above. Since T is admissible, there must exist an atomic conjunction t'_1 in T such that π can be extended by one action a'_1 to reach it optimally and we can continue the process and obtain a sequence of actions $(a'_1, a'_2, \ldots)$ and a sequence of atomic conjunctions $(t'_1, t'_2, \ldots,)$. Similarly as above, this sequence of atomic conjunctions will end and we assume the last one in this sequence is t'_k. We know that an optimal plan to t'_k is an optimal plan for P and the length of this optimal plan is $k + |\pi| = k + m$, which is strictly greater than m. Therefore, it cannot be an optimal plan for P, contradicts. So we have shown that any optimal plan to reach t_m must be an optimal plan for the planning instance P.

Second Claim Assume $t_0 \subseteq s_0$. The empty plan ( ) is an optimal plan to reach t_0. If it is also an optimal plan for P, let $\tau = (t_0)$ and it is an admissible sequence. Otherwise, since T is an admissible set, any optimal plan to t_0 which is not optimal for P can be extended by one action to reach another atomic conjunction in T optimally (W.O.L.G., let it be t_1). If an optimal plan to t_1 is optimal for P, then let $\tau = (t_0, t_1)$ and it is easy to verify that it is an admissible sequence. Otherwise, we know that every optimal plan to t_1 can be extended to t_2 optimally and so on.

From above construction, we can obtain a sequence of atomic conjunctions $(t_0, t_1, \ldots,)$ and every optimal plan to t_i can be extended to reach t_{i+1} optimally. From proof for the first claim, we know this sequence will end and we assume the last one is t_m. Let $\tau = (t_1, \ldots, t_m)$ and it can be verified that it is an admissible sequence since 1) t_0 is true in the initial state s_0, 2) every optimal plan to t_i can be extended to t_{i+1} optimally for all $i = 0, \ldots, m-1$, and 3) every optimal plan to t_m is an optimal plan for P. □

Proof for Lemma 5

Lemma 5. *The problem* `FCP` *is in the complexity class of* $\mathsf{OptP}[\mathcal{O}(\log L)]$.

Proof. Given an input $\langle P \in \mathcal{P}_{\text{poly}}, t, t' \rangle$ where t, t' are two atomic conjunctions of P. Let L be the size of P, t, t' as input, n and m be the numbers of atoms and actions respectively, and $p(n)$ be the polynomial which bounds the length of optimal plans for P. Trivially, $n + m < L$. We consider the set $\mathcal{R} \subseteq \mathbb{N}_0 \times \mathbb{N}_0 \times \mathbb{N}_0$ equipped with the lexicographical order, i.e., $\langle a_1, a_2, a_3 \rangle \prec \langle b_1, b_2, b_3 \rangle$ if and only if $(a_1 < b_1)$ or $(a_1 = b_1$ and $a_2 < b_2)$ or $(a_1 = b_1$ and $a_2 = b_2$ and $a_3 < b_3)$. We execute the following algorithm.

guess an action sequence $a_1, \ldots, a_K, b_1, b_2, \ldots, b_J$ with $K, J \leq p(n)$;
let $l_1 = p(n) + 1$, $l_2 = p(n) + 3$;
for $0 \leq i \leq K - 1$ **do**
  if $t \subseteq s_i$ **then**
    $l_1 = i$, break;
  end if
  if a_{i+1} applicable in s_i **then**
    let $s_{i+1} = f(s_i, a_{i+1})$;
  else
    break;
  end if
end for
Let $s'_0 = s_0$;
for $0 \leq i \leq K - 1$ **do**
  if $t' \subseteq s'_i$ **then**
    $l_2 = i$, break;
  end if
  if b_{i+1} applicable in s'_i **then**
    let $s'_{i+1} = f(s'_i, b_{i+1})$;
  else

21: break;
22: **end if**
23: **end for**

24: **if** $l_1 = p(n) + 1$ or $l_2 = p(n) + 3$ **then**
25: write $\langle l_1, l_2, 0\rangle$ and end computation;
26: **end if**
27: **if** $l_1 = 0$ and $l_2 = 0$ **then**
28: write $\langle 0, 0, 0\rangle$ and end computation;
29: **end if**
30: **if** $l_1 = 0$ and $l_2 = 1$ **then**
31: write $\langle 0, 1, 1\rangle$ and end computation;
32: **end if**
33: **for** action $a \in \mathcal{A}$ **do**
34: let $s' = f(s_{l_1}, a)$ if a is applicable in s_{l_1};
35: **if** $t' \subseteq s'$ **then**
36: write $\langle l_1, l_2, 1\rangle$ and end computation;
37: **end if**
38: **end for**
39: write $\langle l_1, l_2, 0\rangle$ and end computation;

Call the OTM $\mathcal{M}$ to find the minimum among all above values based on lexicographical order and we denote it as $\langle l, l', d\rangle$. If $l' = l + 1$, it returns $\langle l, d\rangle$, otherwise it returns $\langle l, 0\rangle$.

We now explain the above computation.

First of all, l_1 and l_2 represent the lengths of plans to reach t and t' respectively as in Lines 3-23 and since the action sequences $a_1, \ldots, a_K$ and $b_1, \ldots, b_J$ are independent, the minimum values l, l' correspond to $pos(t)$, $pos(t')$ respectively.

Then we consider the indicator d. If $l+1 \neq l'$, that means none optimal plans can be extended by one action to reach t' optimally, therefore $d = 0$. If $l + 1 = l'$, we know that the length of the optimal plans to t is one less than the length of the optimal plans to t', but still we need to check if every optimal plan to t can be extended to reach t' optimally and this check is done via Lines 33-38.

Under the condition $l + 1 = l'$, we consider the following cases in each guess.

- If l_1 is greater than the minimum value l, the result of the third element computed in Lines 36-42 will not have any impact on the final result d since the tuple $\langle l_1, -, -\rangle$ will not be selected by the OTM $\mathcal{M}$.
- If the minimum value $l = l_1$, the guessed action sequence reaches t optimally (in l_1 steps). Since $l' = l + 1$, we know that if an action sequence can reach t' by one more step from t, it must be optimal for t'. Now we need to check if this action sequence be extended by one action to reach t' or not. If there is one action to extend, a value 1 is written as the third element of the tuple. Since the OTM $\mathcal{M}$ computes the minimum value of the third element, if it is 0, that means there exists an optimal plan to t but cannot be extended to t'.

Finally, the above computation runs no more than polynomial time of $p(\mathit{size})$ with $\mathit{size} \geq m + n$ as K is upper bounded by $p(n)$, each state has no more than n atoms and the check in Lines 36-42 runs at most m steps.

Moreover, $\langle pos(t), d\rangle$ in binary can be represented by no more than $\log p(n) + 1$ bits, therefore, according to Definition 7, it is in the complexity class of $\mathsf{OptP}[\mathcal{O}(\log L)]$. □

Proof for Lemma 6

Lemma 6. *The problem of* `FCP` *is* $\mathsf{OptP}[\mathcal{O}(\log L)]$*-hard.*

Proof. Similar to the proof of Theorem 4, we metrically reduce any Optimal Turing machine and an input to an instance of `Find-ConjunctionPos`. The main difference between this proof and the proof in Theorem 4 is we construct an "enumeration" of decimal numbers instead of binary numbers.

Let the OTM $\mathcal{M} = \langle Q, \Gamma, \Sigma, \delta, q_0, Q_F, \square\rangle$, where Q is a set of states, Γ is a tape alphabet which contains all symbols that can be written on the tape, $\Sigma \subseteq \Gamma$ is the input alphabet, $\square \notin \Sigma$ is the blank symbol and $\square \in \Gamma$, q_0 is the initial state, $Q_F \subseteq Q$ is the set of final accepting states, $\delta : Q \times \Gamma \to 2^{Q\times\Gamma\times\{\text{Left,Right}\}}$ is the transition function. Finally, if n is the size of the input, let $g_1(n)$ be the polynomial bounding the execution time of $\mathcal{M}$, and $g_3(n) \in \mathcal{O}(\log n)$ be the strict bound (non-equal) of the number of binary digits of the optimal value computed by $\mathcal{M}$. W.L.O.G., we assume no transition is enabled in final accepting states.

Now given an input x on the tape of $\mathcal{M}$, we shall build below a planning instance $P_{\langle\mathcal{M},x\rangle}$ and an atomic conjunction $t_{\langle\mathcal{M},x\rangle}$ and $t'_{\langle\mathcal{M},x\rangle}$ such that the output from `Find-ConjunctionPos` on $\langle P_{\langle\mathcal{M},x\rangle}, t_{\langle\mathcal{M},x\rangle}, t'_{\langle\mathcal{M},x\rangle}\rangle$ is closely related to the minimum value computed from $\mathcal{M}$ when run x. More specifically, output from `Find-ConjunctionPos` is $\langle pos(t_{\langle\mathcal{M},x\rangle}), d\rangle$, and from $pos(t_{\langle\mathcal{M},x\rangle})$, we can derive the minimum value computed from $\mathcal{M}$ "easily". A trace of $P_{\langle\mathcal{M},x\rangle}$ will first encode an execution branch of $\mathcal{M}$ on x, and then perform a kind of "counting" of decimal numbers that reach $t_{\langle\mathcal{M},x\rangle}$ in a desirable number of actions.

Let $G \triangleq g_1(|x|)$ be the maximum time $\mathcal{M}$ may run on input x, and of course, the maximum number of tape positions used. We let $K = (2^{g_3(|x|)} - 1)$ denote the decimal value of the binary number $\underbrace{111\ldots11}_{g_3(|x|)\text{ copies of 1's}}$. If a computation branch writes a number greater than K, then our planning instance will just write K instead. Therefore, the numbers in decimal that can be written on each computational branch must be in the set $\{0, 1, \ldots, K\}$ where K is upper bounded by $|x|^{\mathcal{O}(1)}$ since $g_3(|x|) \in \mathcal{O}(\log n)$.

We define the planning instance $P_{\langle\mathcal{M},x\rangle} = \langle\mathcal{F}, \mathcal{A}, \mathcal{I}, \mathcal{G}\rangle \in \mathcal{P}_{\text{poly}}$ and the atomic conjunction $t_{\langle\mathcal{M},x\rangle}$ as follows.

Atoms. The set $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2 \cup \bigcup_{k=1}^{K}\{count_k\} \cup \{goal_1, goal_2\}$ where:

- $t_{\langle\mathcal{M},x\rangle} = \{goal_1\}$ is the atomic conjunction whose pos we wish to compute.

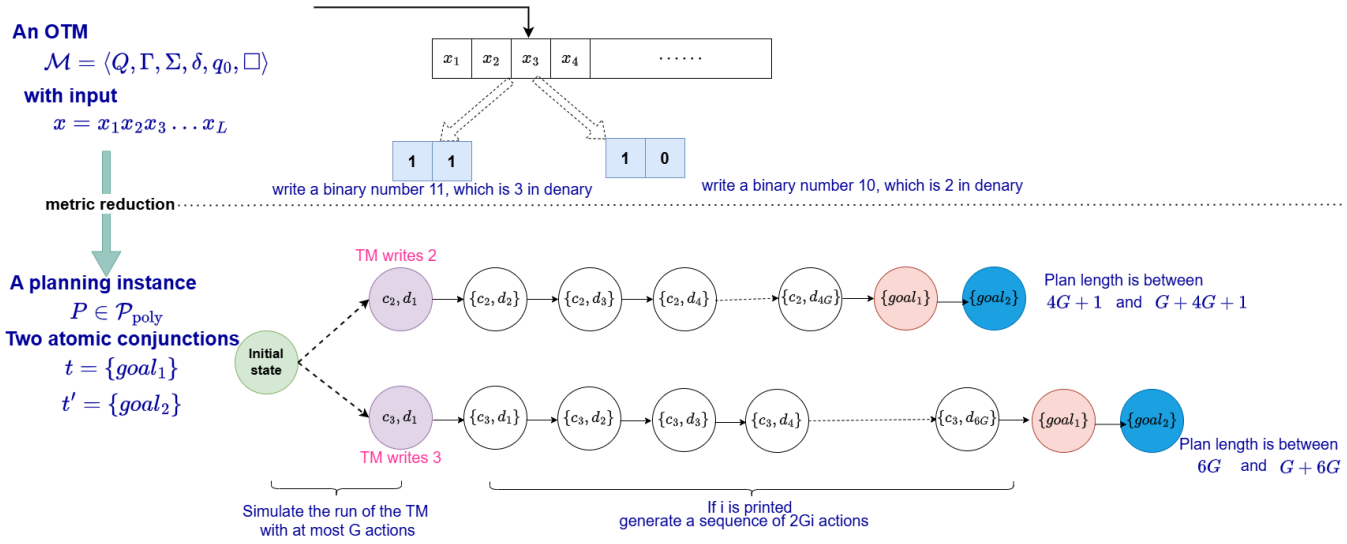

Figure 2: Metric reduction for hardness proof in Lemma 6.

- $t'_{\langle \mathcal{M}, x \rangle} = \{goal_2\}$.
- $\mathcal{F}_1 = \{in(j, x), at(j, q)\}_{q \in Q, x \in \Gamma, j \in [G]}$, as in (Bylander 1994) to capture all $\mathcal{M}$'s workings: $in(i, x)$ means that the symbol x is in position i. $at(i, q)$ means that at the current position i and state q, $\mathcal{M}$ is ready to perform the transition according to δ.
- $\mathcal{F}_2 = \{d_i\}_{i \in [2GK]}$, atoms used to represent and implement listing of a sequence of atomic conjunctions that reach the goal in a desired number of steps.
- Atom $count_k$ signals that the Turing machine branch left k (as decimal) in the tape and hence the counting process will start.
- Atom $goal_2$ will be used as the dummy goal.

There are at most $(|Q|G + |\Gamma|G + 2GK + K + 2)$ atoms. Since K is polynomially upper bounded, the number of atoms is upper bounded polynomially in terms of the size of the description of $\mathcal{M}$ and its input x.

Initial and goal states. The initial state describes the input $x = x_1, \ldots, x_{|x|}$ in the tape with the rest of the cells on the tape blank, and the OTM $\mathcal{M}$'s initial configuration.

$$\mathcal{I} = \{in(1, x_1), in(2, x_2), \ldots, in(|x|, x_{|x|})\} \cup \{in(i, \square) \mid i \in \{0, |x| + 1, \ldots, G\}\} \cup \{at(1, q_0)\}$$

The goal of the planning task is to reach a state where distinguished atom $goal_2$ holds true, that is, $\mathcal{G} = \{goal_2\}$.

Actions. The set of actions is built from three different "components".

First, we have actions dedicated to model the dynamics of the OTM $\mathcal{M}$. This is basically as in (Bylander 1994) except that we use one single action to model the Turing machine transition rather than three. Concretely, for every $i \in [G]$ (representing the position in the tape), and every transition $(q, x, q', y, d) \in \delta$, the planning instance includes the action $\alpha(i, q, x, q', y, d)$ with precondition $\{at(i, q), in(i, x)\}$ and post-conditions:

- $\{at(i?1, q'), \neg at(i, q), in(i, y), \neg in(i, x)\}$, if $x \neq y$; or
- $\{at(i?1, q'), \neg at(i, q)\}$, if $x = y$.

where $? = +$ if $d =$ Right and $? = -$ if $d =$ Left

Once the Turing Machine reaches an accepting state, special actions will convert the binary number left in the tape to an atom representing its decimal value. Concretely, for every final accepting state $q \in Q_F$, possible tape position $i \in [G]$, and every possible decimal number left on the tape $k \leq K$ (via atoms $in(\cdot, \cdot)$), the planning instance has an action $decimal^i_k$ with precondition $at(i, q)$ plus the corresponding set of atoms $in(\cdot, \cdot)$ encoding the binary representation of number k; its post-condition is $\{count_k, d_1\}$.

When the atom d_1 becomes true, it signals the start of the next phase, which implements a sequential "counting" of decimal numbers from 1 up to $2Gk$. For each $i \in [2GK - 1]$, action add_i has precondition $\{d_i\}$ and its post-condition is $\{d_{i+1}, \neg d_i\}$.

Next, we define a set of *semi-final actions* that complete the enumeration process when the last number is reached. Concretely, for every $k \in [K]$, there is an action $finish_k$ whose precondition is $\{d_{2Gk}, count_k\}$ and post-condition is $\{goal_1\}$.

Finally, we define an action $goal - transfer$ with precondition $\{goal_1\}$ and post-condition $\{goal_2\}$ to mark the end of the entire computation.

There are at most $2G|Q|^2|\Gamma|^2$ actions $\alpha(i, q, x, q', y, d)$, $KG|Q|$ actions $decimal^i_k$, $2GK - 1$ actions add_i, K actions $finish_k$ and 1 action $goal - transfer$, so the total number of actions is polynomial in terms of the size of the description of $\mathcal{M}$ and its input x.

Since the longest plan length of P will be upper bounded by $2GK + G + 1$ and given K is polynomially bounded, which means that $P \in \mathcal{P}_{\text{poly}}$.

Given both the number of atoms and the number of actions are polynomially bounded with respect to the size of the description $\mathcal{M}$ and its input x, we know that it takes polynomial time to reduce any OTM $\mathcal{M}$ and its input x to an input for `Find-ConjunctionPos`, i.e., a planning instance $P \in \mathcal{P}_{\text{poly}}$ and an atomic conjunction t.

Consider two decimal numbers $i < j$ written by $\mathcal{M}$, we know that the plan which encodes the computation branch which writes i has length at most $2Gi + G + 1$ and the plan which encodes the one with j as length at least $2Gj + 1$. Since $2Gi + G + 1 < 2Gi + 2G + 1 = 2G(i + 1) + 1 \leq 2Gj + 1$, we know the plan encoding i is shorter than plan encoding j. Thus the length of optimal plan to reach $t_{\langle \mathcal{M},x \rangle} = \{goal_1\}$ should be between $2Gr$ and $2Gr + G$, where r is the minimum number computed by $\mathcal{M}$. And we know the output of this `Find-ConjunctionPos` instance will be $\langle pos(t_{\langle \mathcal{M},x \rangle}), 1 \rangle$ since the only optimal plan to $t_{\langle \mathcal{M},x \rangle}$ can be extended by the action $goal - transfer$ to reach $t'_{\langle \mathcal{M},x \rangle}$ optimally.

Therefore, from $pos(t_{\langle \mathcal{M},x \rangle})$, the minimum value r from the OTM $\mathcal{M}$'s computation should be obtained easily, i.e. $\lfloor \frac{pos(t_{\langle \mathcal{M},x \rangle})}{2G} \rfloor$.

Finally $pos(t_{\langle \mathcal{M},x \rangle}) \leq G + 2GK$, therefore its binary representation is in $\mathcal{O}(\log L)$ as both G and K are polynomially upper bounded.

From Definition 9, we have proved that `Find-ConjunctionPos` is $\mathsf{OptP}[\mathcal{O}(\log L)]$-hard. $\square$