RL-STaR: Theoretical Analysis of Reinforcement Learning Frameworks for Self-Taught Reasoner

Fu-Chieh Chang[*]
MediaTek Research, Taipei, Taiwan
Graduate Institute of Communication Engineering, National Taiwan University, Taipei, Taiwan
d09942015@ntu.edu.tw

Yu-Ting Lee[*]
Department of Mathematical Sciences, National Chengchi University, Taipei, Taiwan
110308056@g.nccu.edu.tw

Hui-Ying Shih
Department of Mathematics, National Tsing Hua University, Hsinchu, Taiwan
huiyingshih0228@gmail.com

Yi Hsuan Tseng
Department of Psychology, National Taiwan University, Taipei, Taiwan
r12227115@ntu.edu.tw

Pei-Yuan Wu
Graduate Institute of Communication Engineering, National Taiwan University, Taipei, Taiwan
peiyuanwu@ntu.edu.tw

Abstract

The reasoning abilities of large language models (LLMs) have improved with chain-of-thought (CoT) prompting, allowing models to solve complex tasks step-wise. However, training CoT capabilities requires detailed reasoning data, which is often scarce. The self-taught reasoner (STaR) framework addresses this by using reinforcement learning to automatically generate reasoning steps, reducing reliance on human-labeled data. Although STaR and its variants have demonstrated empirical success, a theoretical foundation explaining these improvements is lacking. This work provides a theoretical framework for understanding the effectiveness of reinforcement learning on CoT reasoning and STaR. Our contributions are: (1) criteria for the quality of pre-trained models necessary to initiate effective reasoning improvement; (2) an analysis of policy improvement, showing why LLM reasoning improves iteratively with STaR; (3) conditions for convergence to an optimal reasoning policy; and (4) an examination of STaR's robustness, explaining how it can improve reasoning even when incorporating occasional incorrect steps; This framework aims to bridge empirical findings with theoretical insights, advancing reinforcement learning approaches for reasoning in LLMs.

1 Introduction

With the advancement of large language models (LLMs), their reasoning capabilities have become crucial to their success. This progress is mainly attributed to chain-of-thought (CoT) prompting Wei et al. (2022), which allows LLMs to go beyond pattern matching and handle more complex reasoning

[*]equal contribution

problems by providing step-by-step guidance. GPT4-o1 OpenAI (2024) exemplifies this success, achieving high scores on various mathematical and programming benchmarks.

However, to train models with CoT capabilities, training data must include detailed reasoning steps Malach (2024); Prystawski et al. (2024); Xiao & Liu (2024), which are often absent. To address this challenge, the self-taught reasoner (STaR) approach Zelikman et al. (2022) was proposed, leveraging reinforcement learning to automatically discover reasoning steps. Numerous improvements to STaR have since been introduced Hosseini et al. (2024); Zelikman et al. (2024); Lin et al. (2024); Andukuri et al. (2024); Xiang et al. (2025), demonstrating empirically that LLMs can effectively learn reasoning steps via reinforcement learning without human intervention.

Although some theoretical research exists on CoT techniques (e.g., Prystawski et al. (2024); Malach (2024); Feng et al. (2024); Xiao & Liu (2024); Hu et al. (2024)), these studies are primarily focused on supervised and auto-regressive learning settings that require detailed reasoning steps included in training data. They do not show how reinforcement techniques can enhance reasoning steps. Furthermore, while there are existing reinforcement learning frameworks for theoretical analysis (e.g., Jin et al. (2018); Ayoub et al. (2020); Jin et al. (2021; 2020); Bhandari & Russo (2021); Chen et al. (2022); Yeh et al. (2023); Lai et al. (2024)), none are designed to analyze the self-improvement of LLMs through reinforcement learning. As a result, no theoretical framework explains how LLMs can enhance their reasoning capabilities via reinforcement learning. A detailed literature review is shown in Sec. A.1.

1.1 Our Contributions

In this research, we propose a theoretical framework to analyze the effectiveness of reinforcement learning in Chain-of-Thought (CoT) reasoning and Self-Taught Reasoner (STaR), addressing the following questions:

Q1. Conditions of Pre-trained Models for STaR: ***How competent does the pre-trained LLM need to be to bootstrap the discovery of reasoning steps in the first iteration?*** We show that a pre-trained LLM can initiate effective reasoning improvement if one of the following holds when inferencing on the problems in STaR's training set.

- The pre-trained LLM performs better than a randomly initialized model at each reasoning step (i.e., its probability of producing the correct step exceeds that of random guessing).
- The pre-trained LLM is on par with a randomly initialized model at exactly one reasoning step but exceeds it at all other steps.

Q2. Policy Improvement: ***Can LLMs improve their reasoning capabilities iteratively through STaR?*** We demonstrate that if the pre-trained model satisfies the conditions we mentioned previously, within each iteration of the STaR algorithm, LLMs can consistently improve the correctness of their reasoning trajectories.

Q3. Convergence to the Optimal Policy: ***If an optimal reasoning model exists, can STaR eventually find this optimal reasoner?*** Given sufficient iterations, we prove that if the pre-trained model satisfies the previously mentioned conditions, LLMs can converge to the optimal reasoner, achieving the highest probability of generating correct reasoning trajectories that lead to correct answers.

Q4. Existence of Incorrect Reasoning Steps in STaR: ***Is it possible for a sub-optimal model which would generate incorrect reasoning steps included in the next iteration of training, while still arriving at the correct final answer?*** We show that even when incorrect reasoning steps are included in the training data for a given iteration, the probability of these incorrect reasoning steps being included in the training data will diminish with the increasing iteration of STaR.

To the best of our knowledge, this is the first theoretical analysis to guarantee how LLMs can improve their reasoning capabilities through reinforcement learning.

2 Theoretical Frameworks

2.1 Problem Formulation

In our problem formulation, we consider a chain of thought (CoT) reasoning process composed of N steps where $N > 1$. Let s_0 denote the initial input string, and s_n represent the resulting string after the n-th CoT step, where $1 \leq n \leq N$. We assume that the chain-of-thought steps satisfy the Markov property: each string s_n contains sufficient information to derive the next string s_{n+1}, without relying on information from any the preceding string s_i where $0 \leq i < n$. For instance, in the addition problem $1 + 2 + 3 + 4$, the chain-of-thought steps can be expressed as:

$$s_0 = \texttt{1+2+3+4} \Rightarrow s_1 = \texttt{3+3+4} \Rightarrow s_2 = \texttt{6+4} \Rightarrow s_3 = \texttt{10}.$$

In this example, obtaining $s_2 = \texttt{6+4}$ requires only the information in $s_1 = \texttt{3+3+4}$ and does not depend on $s_0 = \texttt{1+2+3+4}$. Under this Markov assumption, the chain-of-thought (CoT) process can naturally be cast as a Reinforcement Learning (RL) problem, which can be described by a Markov decision process (MDP). Formally, let $\mathcal{M} = (\mathcal{S}, \mathcal{A}, N, r, \mathcal{P})$ represents the MDP, where:

- $\mathcal{S}$ is the space of all possible initial inputs, CoT steps, and final answers, denoted $\{s_i \mid 0 \leq i \leq N\}$. In this formulation, s_0 is the CoT input, $s_1, \ldots, s_{N-1}$ are the subsequent reasoning steps, and s_N is the final answer.
- $\mathcal{A}$ is the action space. Because each action corresponds uniquely to selecting the next state, we identify $\mathcal{A}$ with $\mathcal{S}$. That is, choosing a is equivalent to setting s_{n+1} as the unique next step associated with a.
- $\mathcal{P}$ is the transition function. Given that an action a uniquely determines the next state s_{n+1}, this transition is deterministic:

$$\mathcal{P}\big(S_{n+1} = s_{n+1} \big| A = s_{n+1}, S_n = s_n\big) = 1.$$

- $r(s_0, s_N)$ is the reward function, yielding a nonzero reward solely at the final step if s_N matches the correct answer. Concretely, consider a dataset $\mathcal{D}$ composed of question-answer pairs. For each instance $(s_0, s_N^\star) \in \mathcal{D}$, where s_0 represents the question and $s_N^\star$ the correct final answer, define

$$r(s_0, s_N) = \begin{cases} 1, & \text{if } s_N = s_N^\star \text{ and } (s_0, s_N^\star) \in \mathcal{D}, \\ 0, & \text{otherwise.} \end{cases}$$

  Thus, the agent earns a reward of 1 exclusively when it terminates on $s_N^\star$, and 0 otherwise. Note that we fixed the number of CoT steps to be N; intermediate states s_n (for $1 \leq n < N$) do not produce any positive reward.

Given the RL formulation above, a policy $\pi(A \mid S_n = s_n)$ specifies how the next action $A = s_{n+1}$ is selected based on the current state $S_n = s_n$. Although the transition $\mathcal{P}\big(S_{n+1} \mid A = s_{n+1}, S_n = s_n\big)$ is deterministic (that is, once $A = s_{n+1}$ is chosen, $S_{n+1} = s_{n+1}$ is uniquely determined), the policy $\pi(A \mid S_n = s_n)$ itself can be stochastic if there is uncertainty about which next step to choose given $S_n = s_n$. To streamline notations, we define a stochastic transition $P(S_{n+1} \mid S_n)$ by combining the stochastic policy $\pi(A \mid S_n)$ with the deterministic transition $\mathcal{P}(S_{n+1} \mid A, S_n)$:

$$P\big(S_{n+1} \mid S_n\big) \; = \; \mathcal{P}\big(S_{n+1} \mid \pi(A \mid S_n), S_n\big).$$

In this setup, the LLM serves as the transition function $P(S_{n+1} \mid S_n)$, producing the subsequent CoT step s_{n+1} based on the current CoT step s_n. After the final step s_N is reached, the reward function $r(s_0, s_N)$ measures correctness by comparing s_N to the ground truth answer $s_N^\star$.

To guide LLMs toward selecting a final state s_N that maximizes the reward upon completing CoT, we use a modified version of STaR Zelikman et al. (2022), termed RL-STaR, as outlined in Algorithm 1. This algorithm takes a training dataset $\mathcal{D}$ and a pre-trained LLM as transition P_0 as input, and outputs a trained LLM as transition P_T, where $\mathcal{D}$ consists of K instances. In each iteration t, we repeat the following procedure L times: we uniformly sample a pair $\big(s_0^{(\ell)}, s_N^{\star(\ell)}\big)$ from $\mathcal{D}$ and generate a trajectory $\tau^{(\ell)}$ by sequentially sampling states from the current transition P_{t-1}. Specifically, starting from $s_0^{(\ell)}$, we iteratively draw $s_n^{(\ell)} \sim P_{t-1}\big(S_n \,|\, S_{n-1} = s_{n-1}^{(\ell)}\big)$ for $n = 1, \ldots, N$. If the final state

Algorithm 1 RL-STaR

```
Input: A datasets $\mathcal{D} = \{(s_0^{(k)}, s_N^{\star(k)}) | k \in [K]\}$, a pre-trained LLM as transition $P_0$.
Output: A trained LLM as transition $P_T$.
for $t = 1$ to $T$ do
    # Repeat the following procedure for $L$ times where $L \gg K$.
    for $\ell = 1$ to $L$ do
        $\left(s_0^{(\ell)} s_N^{\star(\ell)}\right) \sim \mathcal{D}$. # Uniformly sample a pair $\left(s_0^{(\ell)}, s_N^{\star(\ell)}\right)$ from $\mathcal{D}$.
        $\tau^{(\ell)} \leftarrow (s_0^{(\ell)}, )$ # Set $s_0^{(\ell)}$ as the initial state of the trajectory $\tau^{(\ell)}$.
        for $n = 1$ to $N$ do
            $s_n^{(\ell)} \sim P_{t-1}(S_n | S_{n-1} = s_{n-1}^{(\ell)})$ # Randomly sample $s_n^{(\ell)}$ with probability $P_{t-1}(S_n = s_n^{(\ell)} | S_{n-1} = s_{n-1}^{(\ell)})$.
            $\tau^{(\ell)} \leftarrow (\tau_{1:n-1}^{(\ell)}, s_n^{(\ell)})$ # Append $s_n^{(\ell)}$ to $\tau^{(\ell)}$.
        end for
    end for
    $\mathcal{D}_t \leftarrow \left\{\tau^{(\ell)} \mid \ell \in [L] \wedge s_N^{(\ell)} = s_N^{\star(\ell)}\right\}$ # Add the trajectories whose final state $s_N^{(\ell)}$ matches $s_N^{\star(\ell)}$ to $\mathcal{D}_t$.
    $P_t \leftarrow \text{Train}(P_{t-1}, \mathcal{D}_t)$ # Use $\mathcal{D}_t$ to update the transition.
end for
```

$s_N^{(\ell)}$ matches the ground truth $s_N^{\star(\ell)}$, we add the entire trajectory $\tau^{(\ell)}$ to $\mathcal{D}_t$. After completing these L samplings, we update P_{t-1} to P_t by retraining on $\mathcal{D}_t$. In particular, $\text{Train}(P_{t-1}, \mathcal{D}_t)$ adjusts the transition probabilities $P_{t-1}(S_n = s_n \mid S_{n-1} = s_{n-1})$ to better align with the transitions observed in the successful trajectories within $\mathcal{D}_t$. After T iterations, this procedure outputs the final transition model P_T. In Sec. 3, we will show that RL-STaR aims to maximize the total expected return $J(P_t)$, defined by

$$J(P_t) = \mathbb{E}_{(s_0, s_N^\star) \sim \mathcal{D}} \mathbb{E}_{(s_1, \cdots, s_N) \sim P_t(\tau | S_0 = s_0)} r(s_0, s_N),$$
$$\text{where } P_t(\tau | S_0 = s_0) = P_t(S_1 = s_1 | S_0 = s_0)(\Pi_{n=2}^N P_t(S_n = s_n | S_{n-1} = s_{n-1})).$$

To demonstrate this, we use the following setup.

2.2 Settings of Our Theoretical Analysis

Markov Assumption of Language Models' Input: For simplicity in analyzing the RL-STaR algorithm, we impose the Markov assumption by allowing the LLM to accept only the current step s_{n-1} as input, rather than the complete history $(s_0, s_1, \ldots, s_{n-2})$. This differs from standard CoT approaches, which incorporate all prior strings into the context. Nonetheless, our simplification mirrors the method in Zekri et al. (2024), where a small context window is used to achieve Markov properties for tractable theoretical analysis.

Simplification of STaR Algorithm: For simplicity in our theoretical analysis, we exclude the *rationalization* from STaR—that is, we do not correct an LLM's incorrect outputs using hints derived from the final answer. As noted in Zelikman et al. (2022), omitting rationalization can substantially reduce STaR's performance. Nonetheless, we accept this trade-off here since our focus lies on preliminary theoretical examination rather than achieving state-of-the-art performance on reasoning benchmarks.

Assumption of the Ground-Truth Reasoner $\bar{\pi}$: We assume that a ground-truth transition $\bar{P}(S_{n+1}|S_n)$ exists, which produces the correct sequence $s_1, \ldots, s_{N-1}$ leading to $s_N = s_N^\star$ given s_0 for every instance $(s_0, s_N^\star)$ in $\mathcal{D}$. This property enables us to apply the analysis of RL-STaR which approximate $\bar{P}(S_{n+1}|S_n)$ with an estimated transition $P_T(s_{n+1}|S_n)$. It is clear that if an estimated transition P_T can perfectly match the ground-truth reasoning step transitions $\bar{P}$, it would be the optimal estimated transition $P^\star$ which maximizes $J(P^\star)$, namely,

$$P^\star(S_{n+1} = s_{n+1} | S_n = s_n) = \bar{P}(S_{n+1} = s_{n+1} | S_n = s_n),$$
$$\text{for all } (s_{n+1}, s_n) \in \text{support}(S_{n+1}) \times \text{support}(S_n), \quad \text{and for all } n \in \{0, 1, \ldots, N-1\}.$$

3 THEORETICAL RESULTS

In this section, we present our theoretical analysis addressing the questions outlined in Sec. 1.1. We outline conditions under which the RL-STaR algorithm can effectively train an LLM to approximate the optimal estimated transition $P^{\star}$. For clarity, the notations are defined in Sec A.3.

3.1 A TOY EXAMPLE

We first illustrate our theoretical results with a toy example. In this scenario, we consider a CoT process with two reasoning steps (i.e., $N = 2$), and each step has two possible states (i.e., $M = 2$). Here, S_0 is a random variable representing the initial state, S_1 the intermediate state, and S_2 the final state. We assume their supports are $\mathrm{support}(S_0) = \{s_{0,1}, s_{0,2}\}$, $\mathrm{support}(S_1) = \{s_{1,1}, s_{1,2}\}$, and $\mathrm{support}(S_2) = \{s_{2,1}, s_{2,2}\}$. The ground-truth reasoning trajectories are defined as $\tau_0^{\star} = (s_{0,1}, s_{1,1}, s_{2,1})$ and $\tau_1^{\star} = (s_{0,2}, s_{1,2}, s_{2,2})$, giving the ground-truth transition $\bar{P}_n(S_n|S_{n-1})$ at step $1 \leq n \leq 2$ as

$$\bar{P}_n(S_n|S_{n-1}) = \begin{cases} 1 & \text{if } (S_n, S_{n-1}) = (s_{n,m}, s_{n-1,m}) \text{ for all } n, m \in [2], \\ 0 & \text{otherwise.} \end{cases}$$

We define $P_{u,n}$ as a uniform distribution such that

$$P_{u,n}(S_n|S_{n-1}) = \left\{ \tfrac{1}{2} \quad \text{if } (S_n, S_{n-1}) = (s_{n,m'}, s_{n-1,m}) \text{ for all } n, m, m' \in [2]. \right.$$

With these assumptions in place, we turn to the questions outlined in Sec. 1.1. Regarding the first question, Theorem 3.1 demonstrates that conditions for the pre-trained models to bootstrap the RL-STaR algorithm are that the pre-trained LLM captures certain critical features of the ground-truth transition, thus enabling it to surpass a randomly initialized model. The following theory subsequently verifies this condition.

Theorem 3.1 (**Sufficient Conditions for Pre-trained Models**). *Given the toy example defined in the previous paragraph, in the RL-STaR algorithm, for every CoT step $n \in [2]$, we assume that $P_{0,n}$ represents the state transition estimated by a pre-trained LLM at this step, which is an interpolation between $\bar{P}_n$ and $P_{u,n}$ with a coefficient $0 \leq \delta_{0,n} < 1$. Specifically, we have*

$$P_{0,n}(S_n|S_{n-1}) = \begin{cases} \frac{1+\delta_{0,n}}{2} & \textit{if } (S_n, S_{n-1}) = (s_{n,m}, s_{n-1,m}) \\ & \quad \textit{for all } n, m \in [2], \\ \frac{1-\delta_{0,n}}{2} & \textit{otherwise.} \end{cases}$$

In the RL-STaR algorithm, we assume that the training dataset is $\mathcal{D} = \{(s_{0,1}, s_{2,1}), (s_{0,2}, s_{2,2})\}$. We assume that before RL-STaR iterations t, for every step $n \in [2]$, there exists $0 \leq \delta_{t-1,n} < 1$ such that $P_{t-1,n}$ satisfies the following transition probabilities

$$P_{t-1,n}(S_n|S_{n-1}) = \begin{cases} \frac{1+\delta_{t-1,n}}{2} & \textit{if } (S_n, S_{n-1}) = (s_{n,m}, s_{n-1,m}) \\ & \quad \textit{for all } n, m \in [2], \\ \frac{1-\delta_{t-1,n}}{2} & \textit{otherwise}, \end{cases}$$

and assume that after $\mathrm{Train}(P_{t-1}, \mathcal{D}_t)$ in RL-STaR, $P_{t,n}$ can perfectly match the conditional transition $P(S_{n,m}|s_{n-1,m})$ based on the probabilities of $(s_{n-1,m}, s_{n,m})$ in $\tau \sim \mathcal{D}_t$, and $(s_{n-1,m} s_{n,m' \neq m})$ in $\tau' \sim \mathcal{D}_t$. Then, for every step $n \in [2]$, $P_{t,n}$ satisfies

$$P_{t,n}(S_n|S_{n-1}) = \begin{cases} \frac{1+\delta_{t,n}}{2} & \textit{if } (S_n, S_{n-1}) = (s_{n,m}, s_{n-1,m}) \\ & \quad \textit{for all } m \in [2], \\ \frac{1-\delta_{t,n}}{2} & \textit{otherwise}, \end{cases}$$

where

$$0 \leq \delta_{t,1} = \delta_{t,2} = \frac{\delta_{t-1,1} + \delta_{t-1,2}}{\delta_{t-1,1}\delta_{t-1,2} + 1} < 1.$$

Besides, conditions on the values of $\delta_{0,1}$ and $\delta_{0,2}$ can be listed as follows:

(a) If $0 < \delta_{0,1}$ and $0 < \delta_{0,2}$, then for all $t \geq 1$,

$$\delta_{t-1,1} < \delta_{t,1} \quad \textit{and} \quad \delta_{t-1,2} < \delta_{t,2}.$$

(b) *If exactly one of $\delta_{0,1}$ or $\delta_{0,2}$ is zero—meaning there exist $n, n' \in \{1, 2\}$ with $\delta_{0,n} = 0$ but $\delta_{0,n'} > 0$—then*

$$\delta_{1,n} = \delta_{1,n'} = \delta_{0,n'} > 0,$$

and for all $t > 1$,

$$\delta_{t-1,1} < \delta_{t,1} \quad \textit{and} \quad \delta_{t-1,2} < \delta_{t,2}.$$

(c) *If both $\delta_{0,1} = 0$ and $\delta_{0,2} = 0$, then for all $t \geq 1$,*

$$\delta_{t,1} = \delta_{t,2} = 0.$$

Proof. The proof can be found in Sec. A.5.1. □

Building upon the preceding theorem, we can establish the convergence rate of $\delta_{t,n}$. This result is presented in Theorem A.1. We also address the remaining questions about the STaR algorithm outlined in Sec. 1.1 for this toy example, as detailed in Sec. A.4.1. We move on to a more general case in the next section.

3.2 Main Theorem

After introducing a toy example, we now present our main theorem, which can be applied to an arbitrary number of reasoning steps N and an arbitrary number of states M for each step. In this scenario, S_0 is a random variable that represents the initial state, $S_1, \ldots, S_{N-1}$ represent the intermediate states of steps 1 to $N-1$ respectively, and S_N represents the final state. Each step $n \in [N]$ contains M possible states, so $\text{support}(S_n) = \{s_{n,1}, s_{n,2}, \ldots, s_{n,M}\}$. Assuming there are M ground-truth reasoning trajectories, we denote these trajectories as $\{\tau_m | m \in [M]\}$ where each trajectory τ_m has the form of $(s_{0,m}, s_{1,m}, \ldots, s_{N,m})$. The transition function for step n, denoted as $\bar{P}_n(S_n|S_{n-1})$, is defined as

$$\bar{P}_n(S_n|S_{n-1}) = \begin{cases} 1 & \text{if } (S_n, S_{n-1}) = (s_{n,m}, s_{n-1,m}) \quad \text{for all } m \in [M], \\ 0 & \text{if } (S_n, S_{n-1}) = (s_{n,m' \neq m}, s_{n-1,m}) \;\; \text{for all } m, m' \in [M], \end{cases}$$

and the uniform transition at step n, denoted as $P_{u,n}(S_n|S_{n-1})$, is defined as

$$P_{u,n}(S_n|S_{n-1}) = \frac{1}{M} \quad \text{for all } m \in [M].$$

We now address the questions listed in Sec. 1.1. In response to the first question, Theorem 3.2 indicates that a condition for the pre-trained models to bootstrap the RL-STaR algorithm is that the pre-trained LLM outperforms a randomly initialized model at every reasoning step.

Theorem 3.2 (Conditions of Pre-trained Models). *Given the scenario defined in the previous paragraph, we assume that for every CoT step $n \in [N]$, there is a transition probability $P_{0,n}$ which is learned by pre-trained LLMs and it is an interpolation between $\bar{P}_{u,n}$ and $P_{u,n}$ with a coefficient $0 \leq \delta_{0,n} < 1$, such that*

$$P_{0,n}(S_n|S_{n-1}) = \begin{cases} \frac{1+(M-1)\delta_{0,n}}{M} & \textit{if } (S_n, S_{n-1}) = (s_{n,m}, s_{n-1,m}), \\ \frac{1-\delta_{0,n}}{M} & \textit{if } (S_n, S_{n-1}) = (s_{n,m' \neq m}, s_{n-1,m}) \ , \end{cases}$$
$$\textit{for all } m, m' \in [M].$$

We also assume $\mathcal{D} = \{(s_{0,1}, s_{N,1}), (s_{0,2}, s_{N,2}), \cdots, (s_{0,M}, s_{N,M})\}$. Before iterations t of RL-STaR, if for every CoT step $n \in [N]$, there exist $0 \leq \delta_{t-1,n} < 1$ such that $P_{t-1,n}$ are the following transition probabilities

$$P_{t-1,n}(S_n|S_{n-1}) = \begin{cases} \frac{1+(M-1)\delta_{t-1,n}}{M} & \textit{if } (S_n, S_{n-1}) = (s_{n,m}, s_{n-1,m}), \\ \frac{1-\delta_{t-1,n}}{M} & \textit{if } (S_n, S_{n-1}) = (s_{n,m' \neq m}, s_{n-1,m}), \end{cases}$$
$$\textit{for all } m, m' \in [M],$$

and assume that after $\text{Train}(P_{t-1}, \mathcal{D}_t)$ in RL-STaR, $P_{t,n}$ can perfectly match the conditional transition $P(S_{n,m}|s_{n-1,m})$ based on the probabilities of $(s_{n-1,m}, s_{n,m})$ in $\tau \sim \mathcal{D}_t$, and $(s_{n-1,m} s_{n,m' \neq m})$

in $\tau' \sim \mathcal{D}_t$. Then, for all $n \in [N]$, $P_{t,n}$ satisfies

$$P_{t,n}(S_n|S_{n-1}) = \begin{cases} \frac{1+(M-1)\delta_{t,n}}{M} & \text{if } (S_n, S_{n-1}) = (s_{n,m}, s_{n-1,m}), \\ \frac{1-\delta_{t,n}}{M} & \text{if } (S_n, S_{n-1}) = (s_{n,m'\neq m}, s_{n-1,m}), \end{cases}$$
$$\text{for all } m, m' \in [M],$$

and

$$\delta_{t,n} = \frac{(M-2)\prod_{k=1}^{N}\delta_{t-1,k} + \prod_{k\neq n}\delta_{t-1,k} + \delta_{t-1,n}}{(M-1)\prod_{k=1}^{N}\delta_{t-1,k} + 1}.$$

Besides, based on the values of $\delta_{0,n}$, we have the following cases:

(a) If $0 < \delta_{0,n} < 1$ for each $n \in [N]$, then for all $t \geq 1$,

$$\delta_{t-1,n} < \delta_{t,n} < 1.$$

(b) If there exists a step $n \in [N]$ satisfying $\delta_{0,n} = 0$ and for any other $n' \neq n,\ n' \in [N]$ we have $\delta_{0,n'} > 0$, then when $t = 1$,

$$\delta_{1,n} = \prod_{n'\neq n}\delta_{0,n'} > 0 \quad \text{and} \quad \delta_{1,n'} = \delta_{0,n'} > 0,$$

and for all $t > 1$,

$$\delta_{t-1,n} < \delta_{t,n} < 1. \quad \text{for all } n \in [N].$$

(c) If there exist two distinct steps $n, n' \in [N]$ such that $\delta_{0,n} = 0 = \delta_{0,n'}$, then for all $t \geq 1$,

$$\delta_{t,n} = \delta_{t-1,n} \quad \text{for all } n \in [N].$$

Proof. The proof can be found in Sec. A.6.2. □

The above theorem establishes three key points:

(a) If $\delta_{0,n} > 0$ for all $n \in [N]$, then $\delta_{t,n}$ is strictly increasing in t.

(b) There can be at most one step n with $\delta_{0,n} = 0$. After the first iteration of RL-STaR, we have $\delta_{1,n} > 0$ for all $n \in [N]$, and hence by (a), $\delta_{t,k} > 0$ will be strictly increasing in t for all $t > 1$ and $k \in [N]$.

(c) Conversely, if more than one step satisfies $\delta_{0,n} = 0$, then $\delta_{t,k}$ remains unchanged for all $t \geq 1$.

In conclusion, the conditions of a pre-trained model to improve itself through RL-STaR are to satisfy (a) or at least (b). The following theorem establishes the convergence speed of $\delta_{t,n}$ toward 1.

Theorem 3.3 (Convergence Speed of $\delta_{t,n}$). *Given the scenario defined in Theorem 3.2 (a) or (b), denote*

$$\gamma = \begin{cases} \frac{1-\prod_{k=1}^{N}\delta_{0,k}}{(M-1)\prod_{k=1}^{N}\delta_{0,k}+1} & \text{if (a) is satisfied}, \\ \frac{1-\prod_{k=1}^{N}\delta_{1,k}}{(M-1)\prod_{k=1}^{N}\delta_{1,k}+1} & \text{if (b) is satisfied}. \end{cases}$$

Then for any $\varepsilon \in \left(0, \frac{M}{N(M+1)}\right)$, it holds that

$$\delta_{t,n} > 1-\varepsilon, \forall t \geq \left\lceil \frac{\log\frac{M+1}{M\gamma} + \log\left(N - \sum_{k=1}^{N}\delta_{0,k}\right)}{\log(1/\gamma)} \right\rceil_+ + \left\lceil \log_2\left(\frac{\log M + \log\frac{1-N\varepsilon}{N\varepsilon}}{\log\frac{(M+1)-M\gamma}{\gamma}}\right) \right\rceil_+$$
$$+ \left\lfloor 1 - \prod_{k=1}^{N}\delta_{0,k} \right\rfloor_+ \quad \text{for every } n \in [N].$$

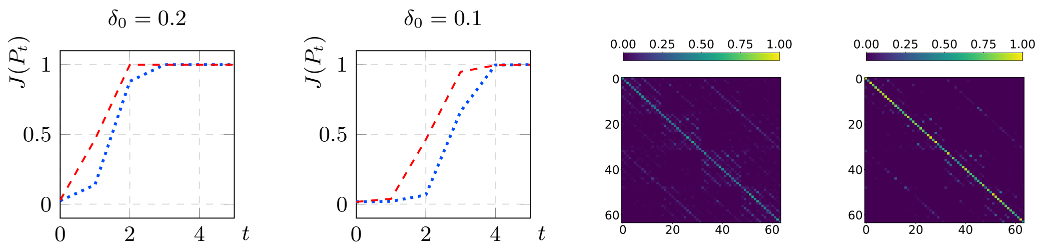

Figure 1: The first two figures on the left illustrate the comparison between theoretical values (blue dotted line) and experimental values (red dashed line) of $J(P_t)$, with the first figure corresponding to $\delta_0 = 0.2$ and the second figure corresponding to $\delta_0 = 0.1$. The remaining two figures on the right depict the comparison of transitions $P(S_1|S_0)$, directly extracted from dataset $\mathcal{D}_1$ (third figure), and the transitions $P_{1,1}(S_1|S_0)$ learned by LLMs during the RL-STaR algorithm (fourth figure).

Proof. The proof can be found in Sec. A.6.3. □

The remaining questions in Sec. 1.1 are answered by the subsequent corollaries.

Corollary 3.4 (Policy Improvement). *Suppose the assumptions of Theorem 3.2 (a) or (b) hold. Let $P_t = \{P_{t,n}\}_{n=1}^N$ represent the estimated transition by the model in the t-th iteration of RL-STaR training. Then the training process improves $J(P_t)$. Specifically, $J(P_{t+1}) \geq J(P_t)$.*

Proof. The proof can be found in Sec. A.6.4. □

Corollary 3.5 (Convergence to the Optimal Policy). *Suppose the assumptions of Theorem 3.2 (a) or (b) hold. If the optimal transition $P^\star$ matches the $M \times M$ identity transition I_M in every CoT step, when the training iteration t of RL-STaR approaches infinity, $P_t = \{P_{t,n}\}_{n=1}^N$ will converge to $P^\star$. That is, $\lim_{t\to\infty} \|P_{t,n} - I_M\|_\infty = 0$ for all $n \in [N]$.*

Proof. The proof can be found in Sec. A.6.5. □

Corollary 3.6 (Diminishing of Incorrect Reasoning Trajectories). *Suppose the assumptions of Theorem 3.2 (a) or (b) hold. In iterations t of RL-STaR, denote $\tau_{t,k}$ to be a trajectory containing k incorrect reasoning steps in $\mathcal{D}_t$. Then, the probability that RL-CoT generates trajectories containing incorrect reasoning steps will diminish as t increases. Specifically, $\lim_{t\to\infty} p\left(\bigcup_k \tau_{t,k}\right) = 0$.*

Proof. The proof can be found in Sec. A.6.6 □

Remark 3.7. In this work, we focus on the scenario that $\delta_{t,n}$ is uniform within each reasoning step n across states $s_{n,m}$ for all $m \in [M]$. Beyond this assumption, there may be additional scenarios under which an pre-trained LLM would still converge via STaR, which will be discussed in Sec. 5.

4 EXPERIMENTS

We experiment to illustrate our theoretical analysis in Sec 3.2 . For the language model, we choose GPT-2 Radford et al. (2019). We restrict the output of LLMs within $M = 64$ valid states within each CoT step. To facilitate reproducibility, we have made our experimental code publicly available[1]. We conducted an experiment to compare the theoretical and experimental values of $J(P_t)$ under the conditions of Theorem 3.2 (a), specifically the case where $\delta_{0,n} > 0$ for every n. The details of the experiment are provided in Sec 4.1. Additionally, we perform further experiments to investigate the convergence behavior of $J(P_t)$ under conditions satisfying Theorem 3.2 (b). Detailed results are provided in Section A.2.

[1] `https://github.com/d09942015ntu/rl_star`

4.1 Theoretical Values of $J(P_t)$ Versus Experimental Values of $J(P_t)$

In our first experiment, we focus on evaluating $J(P_t)$ to compare its theoretically predicted values with those observed in practice. The experimental settings are described below. To facilitate the comparison between theory and practice, we select a straightforward example known as the zip operator[2]. The results of this operator are demonstrated using binary strings of length three. An example of this operation is as follows:

$$s_0 = \texttt{x:101,110} \Rightarrow s_1 = \texttt{x:10,11,y:10} \Rightarrow s_2 = \texttt{x:11,y:01,10} \Rightarrow s_3 = \texttt{y:11,01,10}.$$

Here, the symbols $\texttt{x}$ and $\texttt{y}$ represent the input and output at each step, respectively. At each step, a single token from $\texttt{x}$ is paired with the output $\texttt{y}$. This example has the equal number of states $M = 64$ for each step s_n, as there are 64 possible values of $\texttt{x}$ at s_n. The total number of reasoning steps $N = 3$. For the value of $\delta_{t,n}$, we set $\delta_{t,3} = \delta_{t,2} = \delta_{t,1} = \delta_t$. where $\delta_t \in \{0.1, 0.2\}$.

Theoretical Values: From Theorem 3.2, for the special case $N = 3$ and $\delta_{t,n} = \delta_t$ for all n, the value of δ_t satisfies: $\delta_t = \frac{(M-2)\,\delta_{t-1}^3 + \delta_{t-1}^2 + \delta_{t-1}}{(M-1)\,\delta_{t-1}^3 + 1}$. To estimate $J(P_t)$, it can be shown that when $N = 3$, a direct calculation yields the closed-form solution: $J(P_t) = e_1^T P_{t,3} P_{t,2} P_{t,1} e_1 = \alpha_t^3 + 3(M-1)\alpha_t \beta_t^2 + (M-1)(M-2)\beta_t^3$ where $\alpha_t = \frac{1+(M-1)\delta_t}{M}$ and $\beta_t = \frac{1-\delta_t}{M}$.

Experimental Values: We train large language models (LLMs) using the training dataset $\mathcal{D}$ by the procedures described in Algorithm 1. In this experiment, the pre-trained LLM is obtained from a pre-trained dataset comprising noisy trajectories whose transitions between s_n and s_{n-1} in this dataset match the transition probability $P_0(S_n = s_n | S_{n-1} = s_{n-1})$ referenced in Theorem 3.2.

Results: The results are presented in Fig. 1. These two figures demonstrate that our proposed theory aligns approximately with the actual values obtained from training LLMs. Additionally, LLMs exhibit faster convergence compared to the theoretical predictions. This discrepancy arises because Theorem 3.2 assumes that the transitions learned by LLMs, denoted as P_t, perfectly match the transitions in the dataset $\mathcal{D}_t$. In practice, however, this assumption is not always valid. For example, LLMs may put excessive probability mass on the instances that appear more frequently in the training dataset. An example is provided in Fig. 1, which illustrates the difference between the transitions from the dataset and learned by LLMs. The left-hand heatmap represents the transitions directly derived from the dataset $\mathcal{D}_1$, while the right-hand heatmap depicts the transitions P_1 learned by the LLMs. Notably, the LLM-learned transitions' diagonal elements exhibit higher probabilities than those in the original dataset.

5 Limitations

In this section, we discuss our framework's limits and its divergence from real-world scenarios. We highlight constraints, propose improvements, and extend this theory for more practical applications.

Uniformity of $\delta_{t,n}$ within a Reasoning Step: Our analysis assumes uniform $\delta_{t,n}$ at each reasoning step. However, as noted in Remark 3.7, dropping this requirement can permit other scenarios in which a pre-trained LLM converges through STaR. For instance, consider the following example, where the ground-truth trajectories are $(s_{0,1}, s_{1,1}, s_{2,1})$ and $(s_{0,2}, s_{1,2}, s_{2,2})$. The pre-trained model's transition probabilities (shown on each edge) still enable RL-STaR to discover the optimal policy:

$$\begin{tikzcd}
s_{0,1} \arrow[r, "0.1"] \arrow[dr, "0.9"] & s_{1,1} \arrow[r, "0.6"] \arrow[dr, "0.4"] & s_{2,1} \\
s_{0,2} \arrow[ur, "0.4"] \arrow[r, "0.6"] & s_{1,2} \arrow[ur, "0.01"] \arrow[r, "0.99"] & s_{2,2}.
\end{tikzcd}$$

Analyzing non-uniform $\delta_{t,n}$ in general is significantly more challenging, so establishing the corresponding conditions for pre-trained models in this settings remains future work.

[2]For more details about the zip operator, refer to `https://www.w3schools.com/python/ref_func_zip.asp`

Markov Properties of State Transitions: As noted in Sec. 2.2, in our setup, at CoT step n, the LLM only receives the previous state $S_{n-1} = s_{n-1}$ and does not rely on earlier states $s_0, s_1, \ldots, s_{n-2}$. This assumption grants Markov properties that simplify our RL-based analysis. However, this approach diverges from typical CoT usage, where all prior states are available. Consequently, gaps may exist between our theoretical framework and real-world LLM applications.

Determinism of Ground-Truth Reasoning Trajectories: In our analysis, we assume each question-answer pair $(s_{0,m}, s_{N,m})$ has a single ground-truth reasoning trajectory $\tau = (s_{0,m}, s_{1,m}, \ldots, s_{N,m})$. While this simplifies our theoretical model, in reality, multiple ground-truth reasoning trajectories may lead to the same correct answer. For example, in the arithmetic task $3 \times 2 + 5 \times 4$, both

$$s_0 = \texttt{3 * 2 + 5 * 4} \Rightarrow s_1 = \texttt{6 + 5 * 4} \Rightarrow s_2 = \texttt{6 + 20} \Rightarrow s_3 = \texttt{26}, \quad \text{and}$$
$$s_0 = \texttt{3 * 2 + 5 * 4} \Rightarrow s_1' = \texttt{3 * 2 + 20} \Rightarrow s_2 = \texttt{6 + 20} \Rightarrow s_3 = \texttt{26}.$$

This example illustrates that multiple ground-truth reasoning trajectories yield the same final answer.

Fixed Number of Reasoning Steps N: We adopt a fixed number of CoT reasoning steps N, yet in practice, LLMs can occasionally skip steps while still producing correct answers. For example, in the arithmetic task $3 \times 2 + 5 \times 4$, an LLM may proceed through intermediate steps or jump directly:

$$s_0 = \texttt{3 * 2 + 5 * 4} \Rightarrow s_1 = \texttt{6 + 5 * 4} \Rightarrow s_2 = \texttt{6 + 20} \Rightarrow s_3 = \texttt{26}, \quad \text{and}$$
$$s_0 = \texttt{3 * 2 + 5 * 4} \Rightarrow s_2 = \texttt{6 + 20} \Rightarrow s_3 = \texttt{26}.$$

Thus, although a fixed sequence length simplifies analysis, LLMs have the option to skip steps in real-world settings.

Fixed Number of States M: In our framework, each reasoning step is limited to M possible states. However, this assumption does not fully reflect real-world behavior since LLMs can generate any string rather than being restricted to a predefined set. Consequently, actual LLM outputs may extend beyond these M states, resulting in a broader and potentially less predictable range of responses in real applications.

6 Conclusion

In this work, we introduce a theoretical framework, RL-STaR, to analyze the foundational properties of the Self-Taught Reasoning (STaR) approach. With appropriately bootstrapped pre-trained models, we show that the STaR algorithm can achieve policy improvement and convergence toward the optimal policy, despite the wrong reasoning trajectories being included in the dataset. However, our framework simplifies the complexities inherent in real-world LLM applications. In future work, we plan to extend this framework to encompass more realistic and intricate settings.

Acknowledgment

This work was supported in part by the Asian Office of Aerospace Research & Development (AOARD) under Grant NTU-112HT911020, National Science and Technology Council of Taiwan under Grant NSTC-112-2221-E-002-204- and NSTC-113-2221-E-002-208-, Ministry of Education (MOE) of Taiwan under Grant NTU-113L891406, and Ministry of Environment under Grant NTU-113BT911001

References

Chinmaya Andukuri, Jan-Philipp Fränken, Tobias Gerstenberg, and Noah Goodman. STar-GATE: Teaching language models to ask clarifying questions. In *First Conference on Language Modeling*, 2024. URL `https://openreview.net/forum?id=CrzAj0kZjR`.

Alex Ayoub, Zeyu Jia, Csaba Szepesvari, Mengdi Wang, and Lin Yang. Model-based reinforcement learning with value-targeted regression. In *International Conference on Machine Learning*, pp. 463–474. PMLR, 2020.

Guangsheng Bao, Hongbo Zhang, Cunxiang Wang, Linyi Yang, and Yue Zhang. How likely do llms with cot mimic human reasoning? In *International Conference On Computational Linguistics*, 2025. URL `https://arxiv.org/abs/2402.16048`.

Jalaj Bhandari and Daniel Russo. On the linear convergence of policy gradient methods for finite mdps. In *International Conference on Artificial Intelligence and Statistics*, pp. 2386–2394. PMLR, 2021.

Jalaj Bhandari and Daniel Russo. Global optimality guarantees for policy gradient methods. *Operations Research*, 2024.

Xiaoyu Chen, Han Zhong, Zhuoran Yang, Zhaoran Wang, and Liwei Wang. Human-in-the-loop: Provably efficient preference-based reinforcement learning with general function approximation. In *International Conference on Machine Learning*, pp. 3773–3793. PMLR, 2022.

Guhao Feng, Bohang Zhang, Yuntian Gu, Haotian Ye, Di He, and Liwei Wang. Towards revealing the mystery behind chain of thought: a theoretical perspective. *Advances in Neural Information Processing Systems*, 36, 2024.

Jiafan He, Dongruo Zhou, and Quanquan Gu. Logarithmic regret for reinforcement learning with linear function approximation. In *International Conference on Machine Learning*, pp. 4171–4180. PMLR, 2021.

Arian Hosseini, Xingdi Yuan, Nikolay Malkin, Aaron Courville, Alessandro Sordoni, and Rishabh Agarwal. V-STar: Training verifiers for self-taught reasoners. In *First Conference on Language Modeling*, 2024. URL `https://openreview.net/forum?id=stmqBSW2dV`.

Hao Hu, Yiqin Yang, Qianchuan Zhao, and Chongjie Zhang. The provable benefit of unsupervised data sharing for offline reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=MTTPLcwvqTt`.

Xinyang Hu, Fengzhuo Zhang, Siyu Chen, and Zhuoran Yang. Unveiling the statistical foundations of chain-of-thought prompting methods, 2024. URL `https://arxiv.org/abs/2408.14511`.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *The International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=IkmD3fKBPQ`.

Zhuoxuan Jiang, Haoyuan Peng, Shanshan Feng, Fan Li, and Dongsheng Li. Llms can find mathematical reasoning mistakes by pedagogical chain-of-thought. In *International Joint Conference on Artificial Intelligence*, pp. 3439–3447, 2024. URL `https://doi.org/10.24963/ijcai.2024/381`.

Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I Jordan. Is q-learning provably efficient? In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL `https://proceedings.neurips.cc/paper_files/paper/2018/file/d3b1fb02964aa64e257f9f26a31f72cf-Paper.pdf`.

Chi Jin, Zhuoran Yang, Zhaoran Wang, and Michael I Jordan. Provably efficient reinforcement learning with linear function approximation. In Jacob Abernethy and Shivani Agarwal (eds.), *Proceedings of Thirty Third Conference on Learning Theory*, volume 125 of *Proceedings of Machine Learning Research*, pp. 2137–2143. PMLR, 09–12 Jul 2020. URL `https://proceedings.mlr.press/v125/jin20a.html`.

Ying Jin, Zhuoran Yang, and Zhaoran Wang. Is pessimism provably efficient for offline rl? In *International Conference on Machine Learning*, pp. 5084–5096. PMLR, 2021.

Juno Kim and Taiji Suzuki. Transformers provably solve parity efficiently with chain of thought, 2024. URL `https://arxiv.org/abs/2410.08633`.

Dingwen Kong and Lin Yang. Provably feedback-efficient reinforcement learning via active reward learning. *Advances in Neural Information Processing Systems*, 35:11063–11078, 2022.

Yen-Ru Lai, Fu-Chieh Chang, and Pei-Yuan Wu. Leveraging unlabeled data sharing through kernel function approximation in offline reinforcement learning. *arXiv preprint arXiv:2408.12307*, 2024.

Zhiyuan Li, Hong Liu, Denny Zhou, and Tengyu Ma. Chain of thought empowers transformers to solve inherently serial problems. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=3EWTEy9MTM.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

Haohan Lin, Zhiqing Sun, Yiming Yang, and Sean Welleck. Lean-star: Learning to interleave thinking and proving. *arXiv preprint arXiv:2407.10040*, 2024.

Eran Malach. Auto-regressive next-token predictors are universal learners. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 34417–34431. PMLR, 21–27 Jul 2024.

OpenAI. Chatgpt. https://chatgpt.com/, 2024. Accessed: 2024-10-21.

Ian Osband and Benjamin Van Roy. Model-based reinforcement learning and the eluder dimension. *Advances in Neural Information Processing Systems*, 27, 2014.

Ben Prystawski, Michael Li, and Noah Goodman. Why think step by step? reasoning emerges from the locality of experience. *Advances in Neural Information Processing Systems*, 36, 2024.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In *The Association for Computational Linguistics*, pp. 2609–2634, 2023. URL https://aclanthology.org/2023.acl-long.147/.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023. URL https://arxiv.org/abs/2201.11903.

Violet Xiang, Charlie Snell, Kanishk Gandhi, Alon Albalak, Anikait Singh, Chase Blagden, Duy Phung, Rafael Rafailov, Nathan Lile, Dakota Mahan, et al. Towards system 2 reasoning in llms: Learning how to think with meta chain-of-though. *arXiv preprint arXiv:2501.04682*, 2025.

Changnan Xiao and Bing Liu. A theory for length generalization in learning to reason. *arXiv preprint arXiv:2404.00560*, 2024.

Lin Yang and Mengdi Wang. Sample-optimal parametric q-learning using linearly additive features. In *International conference on machine learning*, pp. 6995–7004. PMLR, 2019.

Zhuoran Yang, Chi Jin, Zhaoran Wang, Mengdi Wang, and Michael I Jordan. On function approximation in reinforcement learning: Optimism in the face of large state spaces. *Advances in Neural Information Processing Systems*, 2020, 2020.

Sing-Yuan Yeh, Fu-Chieh Chang, Chang-Wei Yueh, Pei-Yuan Wu, Alberto Bernacchia, and Sattar Vakili. Sample complexity of kernel-based q-learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 453–469. PMLR, 2023.

Oussama Zekri, Ambroise Odonnat, Abdelhakim Benechehab, Linus Bleistein, Nicolas Boullé, and Ievgen Redko. Large language models as markov chains, 2024. URL `https://arxiv.org/abs/2410.02724`.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488, 2022.

Eric Zelikman, Georges Raif Harik, Yijia Shao, Varuna Jayasiri, Nick Haber, and Noah Goodman. Quiet-STar: Language models can teach themselves to think before speaking. In *First Conference on Language Modeling*, 2024. URL `https://openreview.net/forum?id=oRXPiSOGH9`.

Zhuosheng Zhang, Aston Zhang, Mu Li, and Alex Smola. Automatic chain of thought prompting in large language models, 2022. URL `https://arxiv.org/abs/2210.03493`.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. Large language models are human-level prompt engineers. In *The International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=92gvk82DE-`.

A Appendix

A.1 Related Works

A.1.1 Approaches to Improve Chain-of-thought

By encouraging models to generate intermediate reasoning steps, Chain-of-Thought (CoT) prompting has been shown to significantly improve reasoning capabilities Wei et al. (2023). However, since simply increasing the model size does not enhance its understanding of causal structures Bao et al. (2025), selecting appropriate CoT prompts or identifying and rectifying flawed CoTs Jiang et al. (2024) has become a key focus in recent research. Over the past two years, scholars have explored various perspectives to address this issue, developing diverse improvement methods. For instance, Zhang et al. (2022) uses Retrieval-Q-CoT, which classifies questions based on cosine similarity, to select suitable CoT prompts for the model. Lightman et al. (2024) leveraged Process Supervision to train more reliable reward models, thereby enhancing the stability of the reasoning process. Zhou et al. (2023) explored methods for finding the most suitable instructions, further enhancing the model's adaptability to specific tasks. Additionally, to eliminate the manual effort, Wang et al. (2023) introduced strategies specifically to improve Zero-Shot CoT. Notably, Huang et al. (2024) demonstrated that models cannot perform self-correction in the absence of external feedback, emphasizing the importance of feedback mechanisms. Consequently, leveraging feedback mechanisms to strengthen LLM reasoning remains a crucial goal.

A.1.2 Reinforcement Learning for Boosting Chain-of-thought

To harness external feedback for autonomously improving LLM reasoning, the Self-Taught Reasoner (STaR) framework Zelikman et al. (2022) applies a reinforcement learning strategy. STaR initially generates reasoning steps through in-context learning to elicit chain-of-thought processes. Only the reasoning steps that lead to correct answers are added to the training data, strengthening the model iteratively as the LLM generates new reasoning paths and which are added to the training data in each round. Several STaR extensions have been introduced to further enhance the framework. For instance, Zelikman et al. (2024) proposed Quiet-STaR, a variant where language models produce token-level rationales to justify upcoming text, refining their predictions. V-STaR, introduced in Hosseini et al. (2024), trains a verifier using DPO that evaluates both correct and incorrect self-generated solutions to improve answer verification. Lean-STaR Lin et al. (2024) guides models to generate informal thought steps preceding each proof, boosting theorem-proving abilities. STaR-GATE Andukuri et al. (2024) rewards models for generating insightful questions as part of a self-improvement process. Finally, Meta-STaR Xiang et al. (2025) integrates meta-cognition (System 2 reasoning) into LLM by leveraging self-generated meta-CoT. While these adaptations have demonstrated significant empirical success, none has provided a theoretical explanation for why reinforcement learning enables LLMs to enhance their reasoning capabilities independently.

A.1.3 Theories of Reinforcement Learning

The theory behind reinforcement learning seeks to explain how reinforcement learning algorithms improve a policy and ultimately achieve optimal performance. In its simplest form, Tabular Q-learning, the work of Jin et al. (2018) offers an analysis of the convergence of reinforcement learning algorithms, demonstrating polynomial time and space convergence to the optimal policy. This algorithm can be extended to more complex reinforcement learning scenarios, such as Q-learning with linear reward and transition functions Jin et al. (2020); Yang & Wang (2019); He et al. (2021), and Q-learning with kernel-based approximations of reward and transition functions Yang et al. (2020); Yeh et al. (2023). Additionally, convergence to the optimal policy has been theoretically analyzed for other reinforcement learning algorithms, including policy gradient methods Bhandari & Russo (2021; 2024), human-in-the-loop reinforcement learning Chen et al. (2022); Kong & Yang (2022), model-based reinforcement learning Osband & Van Roy (2014); Ayoub et al. (2020), and offline reinforcement learning Hu et al. (2023); Jin et al. (2021); Lai et al. (2024). These theoretical analyses provide valuable insights into various types of reinforcement learning algorithms. However, they do not address the unique challenges that arise in the reasoning processes of LLMs. Consequently, there is a need for a new theoretical framework to analyze reinforcement learning applications in LLM reasoning steps.

A.1.4 Theories of Chain-of-thought

The Chain-of-Thought (CoT) techniques Wei et al. (2022) enable large language models (LLMs) to tackle complex reasoning tasks by breaking down solutions into a series of sequential steps. Beyond empirical success, some theoretical insights into CoT reasoning have emerged. For instance, Prystawski et al. (2024) models the CoT process using Bayesian networks, where questions, answers, and reasoning steps are nodes within the network. Providing a structured path of reasoning steps has been shown to boost LLM performance. Additionally, Xiao & Liu (2024) introduces the concept of length generalization, where LLMs can solve complex problems by generalizing patterns from simpler training examples. In Malach (2024), the authors extend the PAC supervised learning framework to a PAC auto-regressive framework, demonstrating that an auto-regressive learner can learn linear threshold circuits when CoT steps are provided. Furthermore, Feng et al. (2024) shows that with CoT, transformers can address problem classes solvable by dynamic programming, even when problem sizes grow polynomially. Hu et al. (2024) examine CoT prompting through the lens of statistical estimation, offering a detailed analysis of its sample complexity. Li et al. (2024) theoretically analyze the effectiveness of CoT for decoder-only transformers, showing that it enhances expressiveness by enabling inherently sequential computations, which are otherwise absent in shallow transformers. Kim & Suzuki (2024) presents the first theoretical analysis of training transformers to tackle complex problems by recursively generating intermediate states, akin to fine-tuning for CoT reasoning. Although these works lay a theoretical foundation for CoT, they fall short of explaining why reinforcement learning could enhance CoT capabilities in LLMs. Moreover, these studies underscore the necessity of ample reasoning step examples in training data to develop strong CoT abilities during inference. Uncovering the reasons behind CoT improvement through reinforcement learning could suggest strategies to reduce labeling demands for CoT data in LLM pre-training.

A.2 Additional Experiment

A.2.1 Experiment of the Convergence of $J(P_t)$ with One $\delta_{0,n} = 0$

In this experiment, we examine the convergence behavior of $J(P_t)$ when one $\delta_{0,n} = 0$. We apply the setting of $(\delta_{0,1}, \delta_{0,2}, \delta_{0,3}) = (0, 0.2, 0.2)$. Except for the value of $\delta_{0,n}$, the other settings are the same as described in the experimental part of Sec. 4.1.

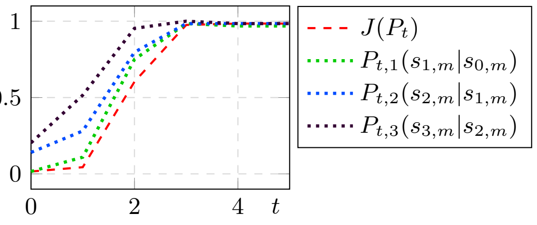

Figure 2: Values of $J(P_t)$ when $(\delta_{0,1}, \delta_{0,2}, \delta_{0,3}) = (0, 0.2, 0.2)$.

Results: Fig. 2 presents the results. In this figure, only one $\delta_{0,n}$ is zero, and $J(P_t)$ converges to nearly optimal value, roughly consistent with Theorem 3.2. Beside $J(P_t)$, we show the probability of pre-trained model P_t generating the correct transition $(s_{n,m}, s_{n-1,m})$ for each CoT step n, denoted by $P_{t,n}(s_{n,m}|s_{n-1,m})$, across iterations t of RL-STaR. Notably, the probability of transition generated by the pre-trained model $P_{0,n}(s_{n,m}|s_{n-1,m})$ does not perfectly align with the distributions in the pre-training dataset. For instance, $P_{0,2}(s_{2,m}|s_{1,m})$ is lower than $P_{0,3}(s_{3,m}|s_{2,m})$ even though $\delta_{0,2}$ and $\delta_{0,3}$ share the same value in the pre-training dataset. Moreover, under the condition of Theorem 3.2(b), the values of $P_{t,2}(s_{2,m}|s_{1,m})$ and $P_{t,3}(s_{3,m}|s_{2,m})$ should remain unchanged between $t = 0$ and $t = 1$. In practice, however, they increase on the first iteration. These differences reflect the discrepancy between the theoretical values and experiment values observed in the previous experiment.

A.3 Notations

Before presenting the additional theorems and proofs, we define the following notation for clarity:

- T: The number of RL-STaR iterations.
- N: The number of CoT steps.
- M: The number of states at each CoT step.
- $s_{n,m'\neq m}$: A state $s_{n,m'}$ where $m' \neq m$ for some $m' \in [M]$.
- $s_{n,m}$: The m-th state at the n-th CoT step.
- $\mathrm{support}(S)$: The support of a random variable S.
- $[n]$: The set $\{1, 2, \dots, n\}$.
- $\tau = (a, b, c)$: An ordered set containing elements a, b, c sequentially.
- $(s_i, s_k) \Subset \tau$: Indicates that elements s_i and s_k are both in the ordered set $\tau = (s_i, s_j, \dots, s_k, s_l)$, with s_i preceding s_k in τ.
- $(\mathbf{s})_i$: The i-th element of the vector $\mathbf{s}$.
- $\|P\|_\infty$: The maximum element in the matrix P.
- $[P]_{i,j}$: The element with index (i, j) in matrix P.
- $\{x_i\}_{i=0}^{\infty}$: An infinite sequence $x_0, x_1, \dots, x_i, \dots$.
- $a \wedge b : \min(a, b)$.
- $a \vee b : \max(a, b)$.
- $\lceil x \rceil_+ : \min(\{n | n \in \mathbb{N} \text{ and } n \geq x\})$.
- $\lfloor x \rfloor_+ : \max(\{n | n \in \mathbb{N} \text{ and } n \leq x\})$.
- $\mathbb{I}\{a = b\} : \mathbb{I}\{a = b\} = 1$ if $a = b$; otherwise $\mathbb{I}\{a = b\} = 0$.

A.4 Additional Theorems and Proofs for Toy example

A.4.1 Additional Theorems for Toy Examples

Theorem A.1 (Convergence Speed of δ_t). *Given the toy example defined in Sec. 3.1, if $\delta_{t,1} = \delta_{t,2} > 0$ for all $t \geq 1$, we define $\delta_t = \delta_{t,1} = \delta_{t,2}$. Then, $\delta_t = \left(\left(\frac{\delta_0^{-1}+1}{\delta_0^{-1}-1}\right)^{2^t} - 1\right) \Big/ \left(\left(\frac{\delta_0^{-1}+1}{\delta_0^{-1}-1}\right)^{2^t} + 1\right)$.*

Proof. Let $f(x) = \frac{2x}{1+x^2}$. Using Taylor's theorem to expand about $x = 1$, one has

$$f(\delta_t) - 1 = f'(1)(\delta_t - 1) + \frac{f''(1)}{2!}(\delta_t - 1)^2 + \frac{f^{(3)}(\xi_t)}{3!}(\delta_t - 1)^3,$$

for each $t \geq 0$ and some $\xi_t \in (\delta_t, 1)$. It's straightforward to see that $f'(1) = 0$, $f''(1) = -1$ and $f^{(3)}(x) = -\frac{12\left(x^4-6x^2+1\right)}{(x^2+1)^4}$. Hence,

$$\lim_{t\to\infty} \frac{|\delta_{t+1} - 1|}{|\delta_t - 1|^2} = 0.5,$$

which shows quadratic convergence.
To find its closed-form expression, put $x_t = \delta_t^{-1}$ and rewrite our recurrence as $x_{t+1} = (1 + x_t^2)/2x_t$. Observe that

$$x_{t+1} + 1 = \frac{(x_t + 1)^2}{2x_t}, \quad \text{and}$$
$$x_{t+1} - 1 = \frac{(x_t - 1)^2}{2x_t}.$$

One now sees that

$$\frac{x_t+1}{x_t-1} = \left(\frac{x_0+1}{x_0-1}\right)^{2^t}.$$

Directly solving for x_t and δ_t, we obtain

$$x_t = \frac{\left(\frac{x_0+1}{x_0-1}\right)^{2^t}+1}{\left(\frac{x_0+1}{x_0-1}\right)^{2^t}-1} \quad \text{and} \quad \delta_t = \frac{\left(\frac{\delta_0^{-1}+1}{\delta_0^{-1}-1}\right)^{2^t}-1}{\left(\frac{\delta_0^{-1}+1}{\delta_0^{-1}-1}\right)^{2^t}+1}.$$

□

Corollary A.2 (Policy Improvement in the Toy Example). *Given the toy example defined in Sec. 3.1, let P_t be the transition model at iteration t of RL-STaR training. If $\delta_{t,1} = \delta_{t,2} > 0$ for all $t \geq 1$, we define $\delta_t = \delta_{t,1} = \delta_{t,2}$. Then,*

$$J(P_t) > J(P_{t-1}).$$

Proof. From Eq. equation 2, the reward $J(P_0)$ is the probability that $p(\tau)$ satisfies $(s_{0,m}, s_{2,m}) \Subset \tau$. Hence,

$$J(P_0) \;=\; \left(\frac{1+\delta_{0,1}}{2}\right)\left(\frac{1+\delta_{0,2}}{2}\right) \;+\; \left(\frac{1-\delta_{0,1}}{2}\right)\left(\frac{1-\delta_{0,2}}{2}\right) \;=\; \frac{1+\delta_{0,1}\,\delta_{0,2}}{2}.$$

This expression remains valid for all $t \geq 0$. Moreover, since $\delta_t = \delta_{t,1} = \delta_{t,2}$ when $t \geq 1$, and it is obvious that $J(P_t)$ is an increasing function. Therefore,

$$J(P_t) \;=\; \frac{1+\delta_t^2}{2} \;>\; \frac{1+\delta_{t-1}^2}{2} \;=\; J(P_{t-1}), \quad \text{for all } t \geq 1,$$

which completes the proof. □

Corollary A.3 (Convergence to Optimal Policy in the Toy Example). *Given the toy example defined in Sec. 3.1, If $\delta_{t,1} = \delta_{t,2} > 0$ for all $t \geq 1$, we define $\delta_t = \delta_{t,1} = \delta_{t,2}$. Define $P^\star$ as the optimal estimated transition, which maximizes the reward $J(P^\star)$. This maximum is achieved when*

$$J(P^\star) = \lim_{\delta \to 1}\left(\frac{1+\delta^2}{2}\right) = 1.$$

Proof. We need to show that, for any $0 < \delta_t < 1$, the limit of δ_t approaches 1 as t tends to infinity. Since $0 < \delta_t < 1$, we have $(\delta_0^{-1}+1)\big/(\delta_0^{-1}-1) > 1$. It is straightforward that by applying Corollary A.1, we have

$$\lim_{t\to\infty} \delta_t = \frac{\left(\frac{\delta_0^{-1}+1}{\delta_0^{-1}-1}\right)^{2^t}-1}{\left(\frac{\delta_0^{-1}+1}{\delta_0^{-1}-1}\right)^{2^t}+1} = 1.$$

□

Corollary A.4 (Diminishing of Incorrect Reasoning Trajectories). *Given the toy example defined in Sec. 3.1, If $\delta_{t,1} = \delta_{t,2} > 0$ for all $t \geq 1$, we define $\delta_t = \delta_{t,1} = \delta_{t,2}$. Denote τ' as the incorrect reasoning trajectories included in the dataset $\mathcal{D}_t$. There are two types of τ' in this toy example:*

$$\tau' = (s_{0,1}, s_{1,2}, s_{2,1}), \quad \textit{and} \quad \tau' = (s_{0,2}, s_{1,1}, s_{2,2}).$$

When t increases, the probability of $\tau' \in D$ diminishes. Specifically,

$$\lim_{t\to\infty} p(\tau' \in \mathcal{D}_t) = 0.$$

Proof. Note that $\delta_t = \delta_{t,1} = \delta_{t,2}$ when $t \geq 1$. Apply Eq.equation 3 and we have

$$p(\tau' \in \mathcal{D}_t) = \frac{(1-\delta_t)^2}{4(1+\delta_t^2)} \quad \text{if } \tau = (s_{0,1}, s_{1,2}, s_{2,1}) \text{ or } \tau = (s_{0,2}, s_{1,1}, s_{2,2}). \tag{1}$$

We complete the proof by applying Corollary A.1. □

A.5 Proof of Theorems for Toy example

A.5.1 Proof of Theorem 3.1

Proof. Without loss of generality, we prove the case when $t = 1$. This is the case of the first iteration of RL-STaR Algorithm. First, we illustrate the transition probability of the pre-trained LLM $P_{0,n}$ as follows

$$\begin{tikzcd}
s_{0,1} \arrow[r, "P_{0,1}(s_{1,1}|s_{0,1})=\frac{1+\delta_{0,1}}{2}"] \arrow[dr, "P_{0,1}(s_{1,2}|s_{0,1})=\frac{1-\delta_{0,1}}{2}"'] & s_{1,1} \arrow[r, "P_{0,2}(s_{2,1}|s_{1,1})=\frac{1+\delta_{0,2}}{2}"] \arrow[dr, "P_{0,2}(s_{2,2}|s_{1,1})=\frac{1-\delta_{0,2}}{2}"'] & s_{2,1} \\
s_{0,2} \arrow[r, "P_{0,1}(s_{1,2}|s_{0,2})=\frac{1+\delta_{0,1}}{2}"'] \arrow[ur, "P_{0,1}(s_{1,1}|s_{0,2})=\frac{1-\delta_{0,1}}{2}"] & s_{1,2} \arrow[r, "P_{0,2}(s_{2,2}|s_{1,2})=\frac{1+\delta_{0,2}}{2}"'] \arrow[ur, "P_{0,2}(s_{2,1}|s_{1,2})=\frac{1-\delta_{0,2}}{2}"] & s_{2,2}.
\end{tikzcd}$$

To begin with, we have an equal probability of selecting either sample $(s_{0,1}, s_{2,1})$ or $(s_{0,2}, s_{2,2})$ from $\mathcal{D}$. Consequently, we obtain the trajectories τ from RL-CoT$(s_{0,m}, P_0)$ for $m \in \{1, 2\}$, with the following probabilities

$$p(\tau) = \begin{cases} \frac{1}{2}\left(\frac{1+\delta_{0,1}}{2}\right)\left(\frac{1+\delta_{0,2}}{2}\right) & \text{if } \tau = (s_{0,1}, s_{1,1}, s_{2,1}), \\ \frac{1}{2}\left(\frac{1+\delta_{0,1}}{2}\right)\left(\frac{1-\delta_{0,2}}{2}\right) & \text{if } \tau = (s_{0,1}, s_{1,1}, s_{2,2}), \\ \frac{1}{2}\left(\frac{1-\delta_{0,1}}{2}\right)\left(\frac{1-\delta_{0,2}}{2}\right) & \text{if } \tau = (s_{0,1}, s_{1,2}, s_{2,1}), \\ \frac{1}{2}\left(\frac{1-\delta_{0,1}}{2}\right)\left(\frac{1+\delta_{0,2}}{2}\right) & \text{if } \tau = (s_{0,1}, s_{1,2}, s_{2,2}), \\ \frac{1}{2}\left(\frac{1-\delta_{0,1}}{2}\right)\left(\frac{1+\delta_{0,2}}{2}\right) & \text{if } \tau = (s_{0,2}, s_{1,1}, s_{2,1}), \\ \frac{1}{2}\left(\frac{1-\delta_{0,1}}{2}\right)\left(\frac{1-\delta_{0,2}}{2}\right) & \text{if } \tau = (s_{0,2}, s_{1,1}, s_{2,2}), \\ \frac{1}{2}\left(\frac{1+\delta_{0,1}}{2}\right)\left(\frac{1-\delta_{0,2}}{2}\right) & \text{if } \tau = (s_{0,2}, s_{1,2}, s_{2,1}), \\ \frac{1}{2}\left(\frac{1+\delta_{0,1}}{2}\right)\left(\frac{1+\delta_{0,2}}{2}\right) & \text{if } \tau = (s_{0,2}, s_{1,2}, s_{2,2}), \end{cases} \tag{2}$$

where $m, n \in [2]$ and $m \neq m'$. In the first iteration of the RL-STaR algorithm, the trajectories τ that satisfy $(s_{0,m}, s_{2,m}) \Subset \tau$ can be exclusively collected in the dataset $\mathcal{D}_1$. Therefore, the conditional probability for τ such that $\tau \in \mathcal{D}_1$ is

$$p(\tau|\tau \in \mathcal{D}_1) = \begin{cases} \frac{(1+\delta_{0,1})(1+\delta_{0,2})}{4(1+\delta_{0,1}\delta_{0,2})} & \text{if } \tau = (s_{0,1}, s_{1,1}, s_{2,1}), \\ \frac{(1-\delta_{0,1})(1-\delta_{0,2})}{4(1+\delta_{0,1}\delta_{0,2})} & \text{if } \tau = (s_{0,1}, s_{1,2}, s_{2,1}), \\ \frac{(1-\delta_{0,1})(1-\delta_{0,2})}{4(1+\delta_{0,1}\delta_{0,2})} & \text{if } \tau = (s_{0,2}, s_{1,1}, s_{2,2}), \\ \frac{(1+\delta_{0,1})(1+\delta_{0,2})}{4(1+\delta_{0,1}\delta_{0,2})} & \text{if } \tau = (s_{0,2}, s_{1,2}, s_{2,2}). \end{cases} \tag{3}$$

Based on this dataset, we assume that the LLMs can perfectly learn the conditional transition $P(S_{n+1,m}|s_{n,m})$ based on the probabilities of $(s_{n,m}, s_{n+1,m}) \Subset \tau$ and $(s_{n,m}, s_{n+1,m'\neq m}) \Subset \tau$ from the $\tau \sim p(\tau|\tau \in \mathcal{D}_1)$. For example, $P(S_{1,1}|s_{0,1})$ can be obtained from

$$\begin{aligned} P(s_{1,1}|s_{0,1}) &= \frac{p((s_{0,1}, s_{1,1}) \Subset \tau|\tau \in \mathcal{D}_1)}{p((s_{0,1}, s_{1,1}) \Subset \tau|\tau \in \mathcal{D}_1) + p((s_{0,1}, s_{1,2}) \Subset \tau|\tau \in \mathcal{D}_1)} \\ &= \frac{\frac{(1+\delta_{0,1})(1+\delta_{0,2})}{4(1+\delta_{0,1}\delta_{0,2})}}{\frac{(1+\delta_{0,1})(1+\delta_{0,2})}{4(1+\delta_{0,1}\delta_{0,2})} + \frac{(1-\delta_{0,1})(1-\delta_{0,2})}{4(1+\delta_{0,1}\delta_{0,2})}} \\ &= \frac{(1+\delta_{0,1})(1+\delta_{0,2})}{2(1+\delta_{0,1}\delta_{0,2})}. \end{aligned}$$

Hence, the transition P_1 is

$$P_1(S_n|S_{n-1}) = \begin{cases} \frac{(1+\delta_{0,1})(1+\delta_{0,2})}{2(1+\delta_{0,1}\delta_{0,2})} & \text{if } S_{n-1} = s_{n-1,m} \text{ and } S_n = s_{n,m} \text{ for all } n, m \in [2], \\ \frac{(1-\delta_{0,1})(1-\delta_{0,2})}{2(1+\delta_{0,1}\delta_{0,2})} & \text{if } S_{n-1} = s_{n-1,m} \text{ and } S_n = s_{n,m'\neq m} \text{ for all } n, m, m' \in [2], \end{cases}$$

which can be shown as

$$\begin{tikzcd}[column sep=huge, row sep=huge]
s_{0,1} \arrow[r, "P_1(s_{1,1}|s_{0,1})=\frac{(1+\delta_{0,1})(1+\delta_{0,2})}{2(1+\delta_{0,1}\delta_{0,2})}"] \arrow[dr, "P_1(s_{1,2}|s_{0,1})=\frac{(1-\delta_{0,1})(1-\delta_{0,2})}{2(1+\delta_{0,1}\delta_{0,2})}" description] & s_{1,1} \arrow[r, "P_1(s_{2,1}|s_{1,1})=\frac{(1+\delta_{0,1})(1+\delta_{0,2})}{2(1+\delta_{0,1}\delta_{0,2})}"] \arrow[dr, "P_1(s_{2,2}|s_{1,1})=\frac{(1-\delta_{0,1})(1-\delta_{0,2})}{2(1+\delta_{0,1}\delta_{0,2})}" description] & s_{2,1} \\
s_{0,2} \arrow[r, "P_1(s_{1,1}|s_{0,2})=\frac{(1+\delta_{0,1})(1+\delta_{0,2})}{2(1+\delta_{0,1}\delta_{0,2})}"'] \arrow[ur, "P_1(s_{1,2}|s_{0,2})=\frac{(1-\delta_{0,1})(1-\delta_{0,2})}{2(1+\delta_{0,1}\delta_{0,2})}" description] & s_{1,2} \arrow[r, "P_1(s_{2,1}|s_{1,2})=\frac{(1+\delta_{0,1})(1+\delta_{0,2})}{2(1+\delta_{0,1}\delta_{0,2})}"'] \arrow[ur, "P_1(s_{2,2}|s_{1,2})=\frac{(1-\delta_{0,1})(1-\delta_{0,2})}{2(1+\delta_{0,1}\delta_{0,2})}" description] & s_{2,2}.
\end{tikzcd}$$

If RL-STaR improves the transition probabilities in each iteration, then the probabilities of transitions matching the ground-truth trajectories will increase. Specifically, we have $P_1(S_n = s_{n,m}|S_{n-1} = s_{n-1,m}) > P_1(S_n = s_{n,m'\neq m}|S_{n-1} = s_{n-1,m})$. Now we need to prove that

$$\delta_{t-1,m} < \delta_{t,m} < 1.$$

We can get $\delta_{1,1} = \frac{\delta_{0,1}+\delta_{0,2}}{1+\delta_{0,1}\delta_{0,2}}$ by

$$\begin{aligned}
\frac{(1+\delta_{0,1})(1+\delta_{0,2})}{2(1+\delta_{0,1}\delta_{0,2})} &= \frac{(1+\delta_{0,1}\delta_{0,2}) + ((1+\delta_{0,1})(1+\delta_{0,2}) - (1+\delta_{0,1}\delta_{0,2}))}{2(1+\delta_{0,1}\delta_{0,2})} \\
&= \frac{1}{2} + \frac{(1+\delta_{0,1})(1+\delta_{0,2}) - (1+\delta_{0,1}\delta_{0,2})}{2(1+\delta_{0,1}\delta_{0,2})} \\
&= \frac{1}{2} + \frac{\delta_{0,1}+\delta_{0,2}}{2(1+\delta_{0,1}\delta_{0,2})} = \frac{1}{2} + \frac{\delta_{1,1}}{2}.
\end{aligned}$$

We can also show that $\delta_{1,1} < 1$, since

$$\delta_{1,1} - 1 = \frac{\delta_{0,1}+\delta_{0,2}}{1+\delta_{0,1}\delta_{0,2}} - 1 = \frac{(\delta_{0,1}+\delta_{0,2}) - (1+\delta_{0,1}\delta_{0,2})}{1+\delta_{0,1}\delta_{0,2}} = \frac{-(\delta_{0,1}-1)(\delta_{0,2}-1)}{1+\delta_{0,1}\delta_{0,2}} < 0.$$

Besides, it is straightforward that the above analysis is true if we replace $\delta_{0,1}$ by $\delta_{t-1,1}$, or replace $\delta_{t-1,1}$ by $\delta_{t-1,2}$.

Now we consider the three conditions on the values of $\delta_{0,1}$ and $\delta_{0,2}$ as follows.

(a) If both $\delta_{0,1} > 0$ and $\delta_{0,2} > 0$, we claim $\delta_{1,1} - \delta_{0,1} > 0$. Indeed,

$$\begin{aligned}
\delta_{1,1} - \delta_{0,1} &= \frac{\delta_{0,1}+\delta_{0,2}}{1+\delta_{0,1}\,\delta_{0,2}} \;-\; \delta_{0,1} \\
&= \frac{\delta_{0,1}+\delta_{0,2}}{1+\delta_{0,1}\,\delta_{0,2}} \;-\; \frac{\delta_{0,1}\big(1+\delta_{0,1}\,\delta_{0,2}\big)}{1+\delta_{0,1}\,\delta_{0,2}} \\
&= \frac{\delta_{0,2}\,\big(1-\delta_{0,1}^2\big)}{1+\delta_{0,1}\,\delta_{0,2}} \;>\; 0,
\end{aligned}$$

because $\delta_{0,1}, \delta_{0,2} > 0$ and $1 - \delta_{0,1}^2 > 0$ (note that $\delta_{0,1} < 1$). A similar argument holds for $\delta_{1,2} - \delta_{0,2}$, ensuring $\delta_{1,2} > \delta_{0,2}$ whenever $\delta_{0,1} > 0$ and $\delta_{0,2} > 0$.

(b) If exactly one of $\delta_{0,1}$ or $\delta_{0,2}$ is zero, we write $\delta_{0,n'} > 0$ and $\delta_{0,n} = 0$. Then,

$$\delta_{1,1} \;=\; \delta_{1,2} \;=\; \frac{0+\delta_{0,n'}}{0+1} \;=\; \delta_{0,n'} \;>\; 0.$$

Thus, starting from $t = 1$, both $\delta_{1,1}$ and $\delta_{1,2}$ are strictly positive, reducing this scenario to the first case above for all subsequent iterations that $\delta_{t,1} > 0$ and $\delta_{t,2} > 0$.

(c) If both $\delta_{0,1} = 0$ and $\delta_{0,2} = 0$, then

$$\delta_{t,1} \;=\; \delta_{t,2} \;=\; \frac{0+0}{0+1} \;=\; 0,$$

for all $t > 0$. In other words, once both $\delta_{0,1}$ and $\delta_{0,2}$ are zero, they remain zero for every iteration t.

□

A.6 Proof of Theorems

A.6.1 Propositions for Proving the Main Theorem

Proposition A.5. *Consider a non-homogeneous Markov sequence $S = (S_n)_{n=0}^N$ with state space $\mathcal{S} = [M]$, initial distribution μ, and transition matrices $P_n \in \mathbb{R}^{M\times M}$. More specifically, for each $n \in [N]$*

$$S_0 \sim \mu, \quad [P_n]_{i,j} = P\left[S_n = j \mid S_{n-1} = i\right].$$

Denote $\mathcal{T} = \left\{(s_n)_{n=0}^N \in \mathcal{S}^{N+1} : s_0 = s_N\right\}$, $\mu_i = \mu(\{i\})$, and

$$\left[\tilde{P}_n\right]_{i,j} = P\left[S_n = j \mid S \in \mathcal{T}, S_{n-1} = i\right] = \frac{\sum_{m=1}^M P\left[S_0 = S_N = m, S_{n-1} = i, S_n = j\right]}{\sum_{m=1}^M P\left[S_0 = S_N = m, S_{n-1} = i\right]}$$
$$= \frac{\sum_{m=1}^M \mu_m \left[\prod_{k=1}^{n-1} P_k\right]_{m,i} [P_n]_{i,j} \left[\prod_{k=n+1}^N P_k\right]_{j,m}}{\sum_{m=1}^M \mu_m \left[\prod_{k=1}^{n-1} P_k\right]_{m,i} \left[\prod_{k=n}^N P_k\right]_{i,m}}.$$

Suppose μ is uniform and that $\left[P_n\right]_{i,j} = \alpha\left(\delta_n\right)\mathbb{I}\{i=j\} + \beta\left(\delta_n\right)\mathbb{I}\{i\neq j\}$ with $\delta_n \in [0,1]$ for each $n \in [N]$, where

$$\alpha(\delta) = \frac{1+(M-1)\delta}{M}, \beta(\delta) = \frac{1-\delta}{M}.$$

Then $\left[\tilde{P}_n\right]_{i,j} = \alpha(\tilde{\delta}_n)\mathbb{I}\{i=j\} + \beta(\tilde{\delta}_n)\mathbb{I}\{i\neq j\}$ satisfies

$$\tilde{\epsilon}_n = \frac{1-\prod_{k\neq n}\delta_k}{1+(M-1)\prod_{k=1}^N \delta_k}\epsilon_n, \tag{4}$$

where $\tilde{\epsilon}_n = 1 - \tilde{\delta}_n$ and $\epsilon_n = 1-\delta_n$.

Proof. Let $\{u_m\}_{m=1}^M$ be an orthonormal basis of $\mathbb{R}^M$ where $u_1 = \frac{1}{\sqrt{M}}\mathbf{1}$. Then

$$P_n = \delta_n I_M + \frac{1-\delta_n}{M}\mathbf{1}\mathbf{1}^\top = \delta_n \sum_{m=1}^M u_m u_m^\top + (1-\delta_n)\, u_1 u_1^\top = u_1 u_1^\top + \delta_n \sum_{m=2}^M u_m u_m^\top = U\Lambda_n U^\top,$$

where $U = [u_1 \cdots u_M]$ and $\Lambda_n = \operatorname{diag}(1, \delta_n, \cdots, \delta_n)$. Denote $\delta_{k:l} = \prod_{t=k}^l \delta_t$ and $\Lambda_{k:l} = \prod_{t=k}^l \Lambda_t = \delta_{k:l} I_M + (1-\delta_{k:l})\, e_1 e_1^\top$. One has

$$\left[\tilde{P}_n\right]_{i,j} = [P_n]_{i,j} \frac{\sum_{m=1}^M \mu_m e_m^\top U \Lambda_{1:n-1} U^\top e_i e_j^\top U \Lambda_{n+1:N} U^\top e_m}{\sum_{m=1}^M \mu_m e_m^\top U \Lambda_{1:n-1} U^\top e_i e_i^\top U \Lambda_{n:N} U^\top e_m}$$
$$= [P_n]_{i,j} \frac{\sum_{m=1}^M e_m^\top \left(\delta_{1:n-1} I_M + (1-\delta_{1:n-1})\, u_1 u_1^\top\right) e_i e_j^\top \left(\delta_{n+1:N} I_M + (1-\delta_{n+1:N})\, u_1 u_1^\top\right) e_m}{\sum_{m=1}^M e_m^\top \left(\delta_{1:n-1} I_M + (1-\delta_{1:n-1})\, u_1 u_1^\top\right) e_i e_i^\top \left(\delta_{n:N} I_M + (1-\delta_{n:N})\, u_1 u_1^\top\right) e_m}$$
$$= [P_n]_{i,j} \frac{\sum_{m=1}^M \left(\frac{1-\delta_{1:n-1}}{M} + \delta_{1:n-1} e_m^\top e_i\right)\left(\frac{1-\delta_{n+1:N}}{M} + \delta_{n+1:N} e_j^\top e_m\right)}{\sum_{m=1}^M \left(\frac{1-\delta_{1:n-1}}{M} + \delta_{1:n-1} e_m^\top e_i\right)\left(\frac{1-\delta_{n:N}}{M} + \delta_{n:N} e_i^\top e_m\right)}. \tag{5}$$

Note that

$$\left[\tilde{P}_n\right]_{i,j} = \beta\left(\delta_n\right) \frac{(M-2)\beta\left(\delta_{1:n-1}\right)\beta\left(\delta_{n+1:N}\right) + \beta\left(\delta_{1:n-1}\right)\alpha\left(\delta_{n+1:N}\right) + \alpha\left(\delta_{1:n-1}\right)\beta\left(\delta_{n+1:N}\right)}{(M-1)\beta\left(\delta_{1:n-1}\right)\beta\left(\delta_{n:N}\right) + \alpha\left(\delta_{1:n-1}\right)\alpha\left(\delta_{n,N}\right)},$$

are identical for all $i \neq j$. Hence there uniquely exists a $\tilde{\delta}_n \in [0,1]$ such that $\left[\tilde{P}_n\right]_{i,j} = \beta\left(\tilde{\delta}_n\right)$ and $\left[\tilde{P}_n\right]_{i,i} = \alpha\left(\tilde{\delta}_n\right)$ for all $i \neq j$. Moreover, (cf. equation 5)

$$\begin{aligned}
\alpha\left(\tilde{\delta}_n\right) &= \alpha\left(\delta_n\right) \frac{(M-1)\beta\left(\delta_{1:n-1}\right)\beta\left(\delta_{n+1:N}\right) + \alpha\left(\delta_{1:n-1}\right)\alpha\left(\delta_{n+1,N}\right)}{(M-1)\beta\left(\delta_{1:n-1}\right)\beta\left(\delta_{n:N}\right) + \alpha\left(\delta_{1:n-1}\right)\alpha\left(\delta_{n,N}\right)} \\
&= \alpha\left(\delta_n\right) \frac{\left(1-\alpha\left(\delta_{1:n-1}\right)\right)\beta\left(\delta_{n+1:N}\right) + \alpha\left(\delta_{1:n-1}\right)\alpha\left(\delta_{n+1,N}\right)}{\left(1-\alpha\left(\delta_{1:n-1}\right)\right)\beta\left(\delta_{n:N}\right) + \alpha\left(\delta_{1:n-1}\right)\alpha\left(\delta_{n,N}\right)} \\
&= \alpha\left(\delta_n\right) \frac{\beta\left(\delta_{n+1:N}\right) + \alpha\left(\delta_{1:n-1}\right)\delta_{n+1:N}}{\beta\left(\delta_{n:N}\right) + \alpha\left(\delta_{1:n-1}\right)\delta_{n:N}} \\
&= \alpha\left(\delta_n\right) \frac{1 + (M-1)\delta_{1:n-1}\delta_{n+1:N}}{1 + (M-1)\delta_{1:N}}.
\end{aligned}$$

Hence,

$$\begin{aligned}
&(M-1)\beta\left(\tilde{\delta}_n\right) = 1 - \left(1 - (M-1)\beta\left(\delta_n\right)\right) \frac{1 + (M-1)\delta_{1:n-1}\delta_{n+1:N}}{1 + (M-1)\delta_{1:N}} \\
&= \frac{(M-1)\left(\delta_{1:N} - \delta_{1:n-1}\delta_{n+1:N}\right) + (M-1)\beta\left(\delta_n\right)\left(1 + (M-1)\delta_{1:n-1}\delta_{n+1:N}\right)}{1 + (M-1)\delta_{1:N}} \\
&= \frac{(M-1)\left(-M\beta\left(\delta_n\right)\delta_{1:n-1}\delta_{n+1:N}\right) + (M-1)\beta\left(\delta_n\right)\left(1 + (M-1)\delta_{1:n-1}\delta_{n+1:N}\right)}{1 + (M-1)\delta_{1:N}} \\
&= (M-1)\beta\left(\delta_n\right) \frac{1 - \delta_{1:n-1}\delta_{n+1:N}}{1 + (M-1)\delta_{1:N}}.
\end{aligned}$$

Note that $\epsilon_n = M\beta\left(\delta_n\right)$ and $\tilde{\epsilon}_n = M\beta\left(\tilde{\delta}_n\right)$ yield Eq. equation 4. □

Proposition A.6. *Consider the scenario defined in Proposition A.5. Let $(\theta_t)_{t=0}^{\infty}, (\vartheta_t)_{t=0}^{\infty}$ be two non-negative non-increasing sequences satisfying $\vartheta_t \leq \theta_t$, $\vartheta_0 \in (0,1)$, and*

$$\theta_{t+1} \leq \frac{\vartheta_t}{M - (M-1)\vartheta_t}\theta_t.$$

We have the following:

(a) Denote $\theta = \sum_{k=1}^{N} 1 - \delta_k$ and $\vartheta = 1 - \prod_{k=1}^{N} \delta_k$. Then, $\vartheta \leq \theta$.

(b) Following (a), if we further assume $\delta_n > 0$ for all n, we have

$$\sum_{k=1}^{N} \tilde{\epsilon_k} \leq \frac{\vartheta\theta}{M - (M-1)\vartheta}.$$

(c) Denote $\gamma = \frac{\vartheta_0}{M-(M-1)\vartheta_0} \in (0,1)$. Then $\theta_t \leq \gamma^t\theta_0$. That is, for $\varepsilon > 0$, it holds that $\theta_t \leq \varepsilon$ whenever $t \geq \frac{\log(\theta_0/\varepsilon)}{\log(1/\gamma)} \vee 0$.

(d) If $\theta_0 < \frac{M}{M+1}$, then

$$\theta_t \leq \frac{M\left(\theta_0\right)^{2^t}}{M\left(\theta_0\right)^{2^t} + \left(M\left(1-\theta_0\right)\right)^{2^t}}.$$

Furthermore, for $\varepsilon \in \left(0, \frac{M}{M+1}\right)$, it holds that

$$\theta_t \leq \varepsilon \quad \forall t \geq \log_2\left(\frac{\log M + \log\frac{1-\varepsilon}{\varepsilon}}{\log M + \log\frac{1-\theta_0}{\theta_0}}\right) \vee 0.$$

(e) For $\varepsilon \in \left(0, \frac{M}{M+1}\right)$, we have

$$\theta_t \le \varepsilon \quad \forall t \ge \left\lceil \frac{\log \frac{M+1}{M\gamma} + \log(\theta_0)}{\log(1/\gamma)} \right\rceil_+ + \left\lceil \log_2 \left(\frac{\log M + \log \frac{1-\varepsilon}{\varepsilon}}{\log \frac{(M+1)-M\gamma}{\gamma}} \right) \right\rceil_+ .$$

Proof.

(a) Because $0 \le \delta_k \le 1$ for all $k \in [N]$, we can use a probability space $(\Omega, \mathcal{F}, \mu)$ to show $\vartheta \le \theta$. Let $\{E_1, E_2, \dots, E_N\} \subset \mathcal{F}$ be events with $\mu(E_k) = \delta_k$. Then the complements E_k^c satisfy $\mu(E_k^c) = 1 - \delta_k$. We can define

$$\vartheta = \mu\Big(\bigcup_{k=1}^N E_k^c\Big) \quad \text{and} \quad \theta = \sum_{k=1}^N \mu(E_k^c).$$

If $E_1^c, E_2^c, \dots, E_N^c$ are disjoint, $\vartheta = \theta$ by countable additivity. In general, E_k^c may overlap, so

$$\mu\Big(\bigcup_{k=1}^N E_k^c\Big) < \sum_{k=1}^N \mu(E_k^c),$$

implying $\vartheta < \theta$.

(b) From the proof of Proposition A.5, we can derive

$$\sum_{k=1}^N \tilde{\epsilon}_k = \frac{\theta - \delta_{1:N} \sum_{k=1}^N \frac{\epsilon_k}{1-\epsilon_k}}{1 + (M-1)\delta_{1:N}} \le \frac{\theta - (1-\vartheta)\frac{N\theta}{N-\theta}}{M - (M-1)\vartheta} = \theta \frac{(N\vartheta - \theta)/(N-\theta)}{M - (M-1)\vartheta} \le \frac{\vartheta\theta}{M - (M-1)\vartheta},$$

where the inequalities follow by noting that $\vartheta \le \theta$ and

$$\theta^2 = \left(\sum_{k=1}^N \frac{\epsilon_k}{\sqrt{1-\epsilon_k}} \cdot \sqrt{1-\epsilon_k} \right)^2 \le \left(\sum_{k=1}^N \frac{\epsilon_k^2}{1-\epsilon_k} \right) \left(\sum_{k=1}^N (1-\epsilon_k) \right) = (N-\theta) \sum_{k=1}^N \frac{\epsilon_k^2}{1-\epsilon_k}$$

$$\Rightarrow \sum_{k=1}^N \frac{\epsilon_k}{1-\epsilon_k} = \sum_{k=1}^N \left(\epsilon_k + \frac{\epsilon_k^2}{1-\epsilon_k} \right) \ge \theta + \frac{\theta^2}{N-\theta} = \frac{N\theta}{N-\theta}.$$

(c) Since $\vartheta_t \le \vartheta_0$, it is clear that $\theta_{t+1} \le \gamma\theta_t$ and the rest is trivial.

(d) Denote $f : x \in [0,1] \mapsto \frac{x^2}{M-(M-1)x}$ and $g : x \in [0,1) \mapsto \left(\varphi \circ h \circ \varphi^{-1}\right)(x)$ where

$$\varphi : x \in [0,\infty) \mapsto \frac{Mx}{Mx+1}, \quad h : x \in [0,\infty) \mapsto x^2, \quad \varphi^{-1} : y \in [0,1) \mapsto \frac{y}{M(1-y)}.$$

Note that

$$g(x) = \frac{M\left(\frac{x}{M(1-x)}\right)^2}{M\left(\frac{x}{M(1-x)}\right)^2 + 1} = \frac{x^2}{M - 2Mx + (M+1)x^2} \ge \frac{x^2}{M - 2Mx + (M+1)x} = f(x),$$

for all $x \in [0,1)$. Since $\vartheta_t \le \theta_t < 1$, one has $\theta_{t+1} \le f(\theta_t) \le g(\theta_t)$, which further implies $\theta_t \le g^{(t)}(\theta_0)$ due to g being monotonically increasing. Thus

$$\theta_t \le g^{(t)}(\theta_0) = \left(\varphi \circ h^{(t)} \circ \varphi^{-1}\right)(\theta_0) = \frac{M\left(\frac{\theta_0}{M(1-\theta_0)}\right)^{2^t}}{M\left(\frac{\theta_0}{M(1-\theta_0)}\right)^{2^t} + 1}.$$

We complete the proof by noting that (provided $t \ge 0$)

$$g^{(t)}(\theta_0) \le \varepsilon \Leftrightarrow \left(\varphi \circ h^{(t)} \circ \varphi^{-1}\right)(\theta_0) \le \varepsilon \Leftrightarrow \left(\varphi^{-1}(\theta_0)\right)^{2^t} = \left(h^{(t)} \circ \varphi^{-1}\right)(\theta_0) \le \varphi^{-1}(\varepsilon)$$

$$\Leftrightarrow 2^t \log\left(1/\varphi^{-1}(\theta_0)\right) \ge \log\left(1/\varphi^{-1}(\varepsilon)\right) \Leftrightarrow t \ge \log_2 \left(\frac{\log\left(1/\varphi^{-1}(\varepsilon)\right)}{\log\left(1/\varphi^{-1}(\theta_0)\right)} \right).$$

(e) Take $\tau_1 = \left\lceil \frac{\log \frac{M+1}{M\gamma} + \log(\theta_0)}{\log(1/\gamma)} \right\rceil_+$ and $\tau_2 = \left\lceil \log_2 \left(\frac{\log M + \log \frac{1-\varepsilon}{\varepsilon}}{\log \frac{M+1-M\gamma}{\gamma}} \right) \right\rceil_+$. Then (a) implies $\theta_{\tau_1} \le \frac{M\gamma}{M+1}$, and (b) further implies $\theta_{\tau_1+\tau_2} \le \varepsilon$.

□

A.6.2 Proof of Theorem 3.2

Proof. The proof of this theorem is based primarily on Proposition A.5. Without loss of generality, we assume that $s_{0,m} = s_{1,m} = \cdots = s_{N,m} = m$ for every $m \in [M]$ and only consider the first iteration of RL-STaR (i.e., $t = 1$ and $t-1 = 0$). We omit subscripts $(0, n)$ and simply write $P_{0,n} = P_n$, $\delta_{0,n} = \delta_n$, and $S_{0,n} = S_n$, etc. Similarly, let $P_{1,n} = \tilde{P}_n$, $\delta_{1,n} = \tilde{\delta}_n$, $\tilde{\epsilon}_n = 1 - \delta_{1,n}$, and $\epsilon_n = 1 - \delta_n$, etc. Under this formulation, the scenario given in Theorem 3.2 can be seen as the Markov process introduced in Proposition A.5. Applying Eq. equation 4, we have

$$\delta_{t,n} = \frac{(M-2)\prod_{k=1}^N \delta_{t-1,k} + \prod_{k\neq n} \delta_{t-1,k} + \delta_{t-1,n}}{1 + (M-1)\prod_{k=1}^N \delta_{t-1,k}}. \tag{6}$$

We now examine the three cases with respect to the values of $\delta_{t=1,n}$.

(a) If $0 < \delta_{0,n} < 1$ for all $n \in [N]$, it is obvious that $0 < \tilde{\epsilon}_n < \epsilon_n$ and hence
$$\delta_{1,n} < \delta_{0,n} < 1.$$
It is straightforward that this inequality will hold for every subsequent iteration $t > 1$.

(b) Suppose there exists a $n \in [n]$ such that $\delta_{0,n} = 0$ and for any other $n' \neq n,\ n' \in [N]$ we have $\delta_{0,n} > 0$. Then, $\prod_{k=1}^N \delta_{0,k} = 0$, $\prod_{n'\neq n} \delta_{0,n'} > 0$ and $\prod_{k\neq n'} \delta_{0,k} = 0$ for any $k \neq n', n' \neq n,\ k, n' \in [N]$. One now sees that
$$\delta_{1,n} = \frac{0 + \prod_{n'\neq n} \delta_{0,n'} + 0}{1+0} = \prod_{n'\neq n} \delta_{0,n'} > 0, \quad \text{and} \quad \delta_{1,n'} = \frac{0+0+\delta_{0,n'}}{1+0} = \delta_{0,n'} > 0.$$
After the first iteration, apply the analysis in (a).

(c) Suppose there exist two (or more than two) steps $n, n' \in [N]$ satisfying $n' \neq n$, $\delta_{0,n} = 0$ and $\delta_{0,n'} = 0$. We have $\prod_{k=1}^N \delta_{0,k} = 0$, and $\prod_{k\neq n} \delta_{0,k} = 0$ for any $k \in [N]$. Then
$$\delta_{1,n} = \frac{0+0+\delta_{0,n}}{1+0} = \delta_{0,n} \quad \text{for all } n \in [N].$$
It is obvious that above equation holds for every subsequent iteration $t > 1$.

□

A.6.3 Proof of Theorem 3.3

Proof. For the case of $\delta_{0,n} > 0$ for all $n \in [N]$, we can apply Proposition A.6, where we set
$$\gamma = \frac{\vartheta_0}{M - (M-1)\vartheta_0} = \frac{1 - \prod_{k=1}^N \delta_{0,k}}{(M-1)\prod_{k=1}^N \delta_{0,k} + 1},$$
and let $\varepsilon = \frac{\varepsilon'}{N}$ for $\varepsilon' \in \left(0, \frac{M}{M+1}\right)$. Therefore, we have
$$\max_k \delta_{t,k} \ge \frac{\sum_{n=1}^K \delta_{t,k}}{N} \ge 1 - \frac{\varepsilon'}{N} \quad \forall t \ge \left\lceil \frac{\log \frac{M+1}{M\gamma} + \log(\theta_0)}{\log(1/\gamma)} \right\rceil_+ + \left\lceil \log_2 \left(\frac{\log M + \log \frac{1-\varepsilon'}{\varepsilon'}}{\log \frac{(M+1)-M\gamma}{\gamma}} \right) \right\rceil_+ .$$
On the other hand, if there exists only one $n \in [N]$ such that $\delta_{0,n} = 0$, an additional iteration $t = 1$ is needed to ensure $\delta_{1,n} > 0$. Once $\delta_{1,n}$ becomes strictly positive for all $n \in [N]$, we set
$$\gamma = \frac{1 - \prod_{k=1}^N \delta_{1,k}}{(M-1)\prod_{k=1}^N \delta_{1,k} + 1},$$
and then the argument proceeds as before, completing the proof. □

A.6.4 Proof of Corollary 3.4

Proof. By applying Proposition A.5 to the t-th iteration of RL-STaR, we abuse the notation of $P_{t,n}$ to denote the transition matrix in the t-th iteration of RL-STaR, at the n-th CoT step. Recall that

$$\prod_{k=1}^{N} P_{t,k} = U\left(\prod_{k=1}^{N} \delta_{t,k} I_M + (1-\prod_{k=1}^{N}\delta_{t,k})e_1 e_1^\top\right) U^T,$$

where $U = [u_1, u_2, \ldots, u_M]$, and $u_1 = \frac{1}{\sqrt{M}}\mathbf{1}$. One sees that

$$\Big[\prod_{k=1}^{N} P_{t,k}\Big]_{i,i} = \Big[\prod_{k=1}^{N} P_{(t,k)}\Big]_{1,1} = \prod_{k=1}^{N}\delta_{t,k} + \frac{1-\prod_{k=1}^{N}\delta_{t,k}}{M}.$$

Since the initial state is chosen uniformly, $J(P_t) = \sum_{m=1}^{M}\frac{1}{M}\big[\prod_{k=1}^{N} P_{(t,k)}\big]_{m,m} = \frac{1}{M} + (\frac{M-1}{M})\prod_{k=1}^{N}\delta_{t,k}$. Assuming Theorem 3.2 (a) or (b) hold, we conclude as $\delta_{1,n} \geq \delta_{0,n}$ and $\delta_{t,n} > \delta_{t-1,n}$ for all $t > 1$ and $n \in [N]$. □

A.6.5 Proof of Corollary 3.5

Proof. Theorem 3.3 implies that

$$\lim_{t\to\infty}\delta_{t,n} = 1 \quad \text{for all } n \in [N].$$

By Proposition A.5, we know that $P_{t,n}$ has diagonal elements $\alpha(\delta_{t,n})$ and off-diagonal elements $\beta(\delta_{t,n})$ such that

$$\lim_{t\to\infty}\alpha(\delta_{t,n}) = \lim_{t\to\infty}\frac{1+(M-1)\delta_{t,n}}{M} = 1 \quad \text{and} \quad \lim_{t\to\infty}\beta(\delta_{t,n}) = \lim_{t\to\infty}\frac{1-\delta_{t,n}}{M} = 0 \quad \text{for all } n \in [N].$$

Hence,

$$\lim_{t\to\infty}\|P_{t,n} - I_M\|_\infty = 0 \quad \text{for all} \quad n \in [N].$$

□

A.6.6 Proof of Corollary 3.6

Proof. This is a simple consequence of convergence to the optimal policy. Since the incorrect reasoning steps may appear in any CoT steps, we see that

$$p(\tau_{t,k}) = \frac{1}{M}\left(\beta(\delta_{t,n_1})\cdots(\beta(\delta_{t,n_k})\right)\prod_{j\neq n_i \forall i}\alpha(\delta_{t,j})$$

where $2 \leq k \leq N$, for some subsequence $n_1 < n_2 < \cdots < n_k$. Using $\lim_{t\to\infty}\alpha(\delta_{t,n}) = 1$ and $\lim_{t\to\infty}\beta(\delta_{t,n}) = 0$ for each n, we have $0 \leq p(\tau_{t,k}) \xrightarrow{t\to\infty} 0$. Because $|\bigcup_k \tau_{t,k}| < \infty$, we obtain the desired result. □