Proof of main result

Recall that to prove Theorem 1, it must first be shown that the regret of the algorithm is related to the smallest $\alpha$ among expanded nodes (which will be done in Lemma 6), and then that the algorithm always works to decrease this smallest $\alpha$ (done in Lemma 7). A preliminary result is also needed.

**Lemma 5.** The $\nu$-values of the near-optimal policy classes increase over iterations: $\nu(H^*_t + 1) \geq \nu(H^*_t)$, where $H^*_t \in \arg \max_{H \in T_t} \nu(H)$.

*Proof.* Consider first one policy class $H$, split by expanding some leaf node $s \in L(T_H)$. One child class $H'$ is obtained for each action $u$, and we have $L(T_{H'}) = (L(T_H) \setminus \{s\}) \cup c(s, u)$. By easy calculations, since the rewards are positive, the terms that nodes $c(s, u)$ contribute to $\nu(H')$ add up to more than the term of $s$ in $\nu(H)$, and the other terms remain constant. Thus $\nu(H') \geq \nu(H)$. Then, among the policy classes $H_t \in T_t$, some are split in $T_{t+1}$ and some remain unchanged. For the children of split classes $\nu$-values are larger than their parents; while $\nu$-values of unchanged classes remain constant. Thus, the maximal $\nu$-value increases across iterations. Note it can similarly be shown that $b(H^*_t + 1) \leq b(H^*_t)$.

**Lemma 6.** Define $\alpha_t = \alpha(s_t)$, the $\alpha$ value of the node expanded at iteration $t$; and $\alpha^* = \min_{t=0,\ldots,n-1} \alpha_t$. The regret after $n$ expansions satisfies $R_n \leq \frac{N}{\gamma} \alpha^*$.

*Proof.* We will first bound, individually at each iteration $t$, the suboptimality of $\nu(H^*_t)$, by showing:

$$v^* - \nu(H^*_t) \leq \text{diam}(H^*_t) \leq \frac{N}{\gamma} \alpha_t \tag{7}$$

To this end, observe that:

$$\nu(H^*_t) \leq \nu(H^*_t) \leq v^* \leq b(H^*_t) \tag{8}$$

The inequality $\nu(H^*_t) \leq v^*$ is true by definition ($\nu(H^*_t)$ is a lower bound on the value of some policy, itself smaller than $v^*$). For the leftmost inequality, $H^*_t$ maximizes the lower bound across all policy classes compatible with the current tree, so its lower bound is at least as large as that of the optimistic policy class $H^*_t$. Similarly, for the rightmost inequality, since $H^*_t$ maximizes the upper bound, its upper bound is immediately larger than the true optimal value. Using this string of inequalities, we get:

$$v^* - \nu(H^*_t) \leq b(H^*_t) - \nu(H^*_t) = \text{diam}(H^*_t) = \sum_{s \in L(T_{H^*_t})} c(s) \tag{9}$$

We now investigate the relationship between this diameter and $\alpha_t$. Consider the subtree $T_{H^*_t}$ of policy class $H^*_t$, represented schematically in Figure 4 using a black continuous outline (this subtree has a branching factor of $N$). We are thus interested in finding an upper bound for $\sum_{s \in L(T_{H^*_t})} c(s)$ as a function of $\alpha_t$. Consider the tree $T_h$, as introduced earlier in the definition of $n(s)$, which is included in $T_{H^*_t}$ and is the same for any $h \in H^*_t$. To see this, recall that $s_t$ maximizes $c$ among the leaves of $T_{H^*_t}$. Since additionally $c$ strictly decreases along paths, any node with a contribution larger than $c(s_t)$ must be above these leaves, and this holds for any $h \in H^*_t$.

Denote in this context $T_{h,s}$ more simply by $T'$, shown in gray in the figure, and its leaves by $L'$, shown as a gray outline. Denote the children of $L'$ by $L''$, shown as a dashed line.

![Figure 4: Tree of the optimistic policy class and various subtrees.](image)

Recall that for any $h$ and $s \in T_h$, $\sum_{s' \in C(h, s)} c(s') = \gamma c(s)$. This also means the sum of contributions for the leaves of any subtree of $T_h$ having some $s$ as its root is smaller than $c(s)$. Using these properties, we have:

$$\sum_{s \in L(T_{H^*_t})} c(s) \leq \sum_{s' \in L'} c(s') = \frac{1}{\gamma} \sum_{s'' \in L''} c(s'') \leq \frac{1}{\gamma} \sum_{s \in L''} c(s) = \frac{1}{\gamma} N c(s_t) \leq \frac{1}{\gamma} N n(s_t) c(s_t) = \frac{N}{\gamma} \alpha_t$$
Proof. We show first that condition (i) in the definition (5) of \( S \) size of \( W \) study the whose tree \( n \). By Lemma 6, the first inequality is true because \( \max_u Q^*(s_0, u) = v^* \) and \( Q^*(s_0, H_n^*(s_0)) \geq v^*(H_n^*) \) (the return \( Q^*(s_0, H_n^*(s_0)) \) is obtained by choosing optimal actions below level 0, whereas \( H_n^* \) may make other suboptimal choices). The proof is complete.

**Lemma 7.** All nodes expanded by the algorithm belong to \( S_{\alpha^*} \), so that \( n \leq |S_{\alpha^*}| \).

*Proof.* We show first that \( s_t \in S_{\alpha_t} \) at any iteration \( t \). Condition (i) in the definition (5) of \( S_{\alpha_t} \) is immediately true. For condition (ii), an \( \frac{N}{\gamma} \alpha_t \)-optimal policy \( h \) whose tree \( T_h \) contains \( s_t \) is needed. Choose any \( h \in H_1^t \), then \( s_t \in T_h \) and:

\[
v^* - \nu(h) \leq b(H_1^t) - \nu(H_1^t) = \text{diam}(H_1^t) \leq \frac{N}{\gamma} \alpha_t
\]

where we used some of the inequalities derived in the proof of Lemma 6. Thus \( s_t \in S_{\alpha_t} \). Furthermore, \( \alpha^* \leq \alpha_t \) implies \( S_{\alpha_t} \subseteq S_{\alpha^*} \), and we are done.

*Proof of Theorem 1.* Exploiting Lemma 7 in combination with (6):

- if \( \beta > 0 \), then \( \tilde{O}(\alpha^{* - \beta}) \), thus for large \( n \), \( \alpha^* = O\left(n^{-\frac{1}{\beta}}\right) \);
- if \( \beta = 0 \), then \( a \left( \frac{1}{\gamma} \right)^b \), thus \( \alpha^* \leq \exp\left[-(\frac{n}{\gamma})^b\right] \).

By Lemma 6, \( R_n \leq \frac{N}{\gamma} \alpha^* \) which immediately leads to the desired results.

**Proofs for values of \( \beta \) in special cases**

*Proof of Proposition 2 (uniform case).* We study the size of \( S_{\varepsilon} \). Due to the equal rewards all the policies are optimal, and condition (ii) in (5) does not eliminate any nodes. The contribution of a node is \( c(s) = R(s) \left( \frac{\alpha^*}{1 - \gamma} \right)^d(s) \left( \frac{1}{1 - \gamma} \right) \), since the probability of reaching a node at depth \( d(s) \) is \( \left( \frac{\alpha^*}{1 - \gamma} \right)^d(s) \). This also means that, for any policy \( h \), the tree \( T_{h_\varepsilon} \) consists of all the nodes \( s' \) up to the depth of \( s \). The number of leaves of this tree is \( N^{d(s)} \) (recall that a policy tree has only branching factor \( N \), and since this number does not depend on the policy, \( n(s) \) is also \( N^{d(s)} \). Therefore, \( \alpha(s) = n(s)c(s) = \frac{\alpha^*}{1 - \gamma} \) and condition (i) eliminates nodes with depths larger than \( D = \log \frac{\varepsilon(1 - \gamma)}{\log \gamma} \). The remaining nodes in the whole tree, with branching factor \( NK \), form \( S_{\varepsilon} \), which is of size:

\[
|S_{\varepsilon}| = O(\left(\frac{NK}{\log \frac{\varepsilon(1 - \gamma)}{\log \gamma}}\right)^D) = O(\left(\frac{NK}{\log \frac{\varepsilon(1 - \gamma)}{\log \gamma}}\right) D) = O(\frac{\varepsilon}{\log \frac{\varepsilon(1 - \gamma)}{\log \gamma}})
\]

yielding for \( \beta \) the value: \( \beta_{\text{rew}} = \frac{\log N}{\log \frac{\varepsilon(1 - \gamma)}{\log \gamma}} \). So, for large \( n \), the regret \( R_n = \tilde{O}(n^{-\frac{1}{\gamma}}) \). In fact, as can be easily checked by examining the proof of Theorem 1, the logarithmic component disappears in this case and \( R_n = O(n^{-\frac{1}{\gamma}}) \).

*Proof of Proposition 3 (structured rewards).* Since \( \alpha(s) \) depends only on the probabilities, condition (i) leads to the same \( D = \log \frac{\varepsilon(1 - \gamma)}{\log \gamma} \) as in the uniform case. However, now condition (ii) becomes important, so to obtain the size of \( S_{\varepsilon} \), we must only count near-optimal nodes up to depth \( D \).

Consider the set of nodes in \( T_\infty \) which do not belong to the optimal policy, but lie below nodes that are at depth \( d' \) on this policy. An example is enclosed by a dashed line in Figure 3, where \( d' = 1 \). All these nodes are sub-optimal to the extent of the loss incurred by not choosing the optimal action at their parent, namely: \( \left( \frac{\gamma}{N} \right)^{d'} \left( \frac{1}{1 - \gamma} \right) \). Note these nodes do belong to a policy that is near-optimal to this extent, one which makes the optimal choices everywhere except at their parent. Looking now from the perspective of a given depth \( d \), for any \( m \leq d \) there are \( N^d K^m \) nodes at this depth that are \( \left( \frac{\gamma}{N} \right)^{d-m} \left( \frac{1}{1 - \gamma} \right) \)-optimal. Condition (ii), written \( \left( \frac{\gamma}{N} \right)^{d-m} \left( \frac{1}{1 - \gamma} \right) \leq \frac{N}{\gamma} \left( \frac{n}{\gamma} \right)^b \), leads to \( m \leq \frac{\log N}{\log \frac{\varepsilon(1 - \gamma)}{\log \gamma}} + 1 \).

Then:

\[
|S_{\varepsilon}| \leq \sum_{d=0}^{D} N^d K^{d - \frac{\log N}{\log \frac{\varepsilon(1 - \gamma)}{\log \gamma}}} + 1 \leq K \sum_{d=0}^{D} \left( \frac{NK}{\log \frac{\varepsilon(1 - \gamma)}{\log \gamma}} \right)^d
\]

If \( N > 1 \):

\[
|S_{\varepsilon}| = O\left(\left(\frac{NK}{\log \frac{\varepsilon(1 - \gamma)}{\log \gamma}}\right)^D\right) = O\left(\left(\frac{NK}{\log \frac{\varepsilon(1 - \gamma)}{\log \gamma}}\right) \left(\frac{\varepsilon(1 - \gamma)}{\log \gamma}\right)\right)\]

yielding the desired value of \( \beta_{\text{rew}} = \frac{\log N}{\log \frac{\varepsilon(1 - \gamma)}{\log \gamma}} \).

If \( N = 1 \) (deterministic case), \( \beta_{\text{rew}} = 0 \) and:

\[
|S_{\varepsilon}| = \sum_{d=0}^{D} 1 \cdot K = (D + 1)K = \left(\frac{\log \varepsilon(1 - \gamma)}{\log \gamma} + 1\right) K \leq a \log 1/\varepsilon
\]
for small ε and some constant a, which is of the form (6) for b = 1. From Theorem 1, the regret is $O(\exp(-\frac{b}{a}))$.

Proof of Proposition 4 (structured probabilities). We will show that the quantities of nodes with sizable contributions on the subtree of one policy, and respectively on the whole tree, satisfy:

$$n(\lambda) = \{s \in T_\infty | c(s) \geq \lambda\} = \bar{O}(\lambda^{-\delta})$$
$$n_h(\lambda) = \{s \in T_h | c(s) \geq \lambda\} = \bar{O}(\lambda^{-\delta_h})$$

for constants $\delta_h$ and $\delta$; and we will find values for these constants. (Note $n_h(\lambda)$ is not a function of $h$, since all policies have the same probability structure.) Then, since condition (ii) always holds and nodes in $pC$ policies have the same probability structure.) Then, above a certain maximum depth $d$, we count all the nodes up to $m$ and we will find values for these constants. (Note $n_h(\lambda)$ is not a function of $h$, since all policies have the same probability structure.) Then, since condition (ii) always holds and nodes in $S_c$ only have to satisfy condition (i):

$$|S_c| = \{s \in T_\infty | n(s)c(s) \geq \varepsilon\}\geq \{s \in T_\infty | n_h(c(s))c(s) \geq \varepsilon\} \leq \{s \in T_\infty | \beta c(s)\log 1/c(s)\geq \varepsilon\} = \bar{O}(\varepsilon^{-\frac{1}{\log \lambda}})$$

where we used $n(s) \leq n_h(c(s))$ and $n_h(c(s)) = \bar{O}(c(s)^{-\delta_h})$. Thus $\beta = \frac{\delta}{1-\delta_h}$.

Consider now $n_h(\lambda)$. The nodes at each depth $d$ correspond to a binomial distribution with $d$ trials, so there are $C^m_d$ nodes with contribution $c(s) = p_d^m(1-p)^{d-m}$, for $m = 0, 1, \ldots, d$. Since these contributions decrease monotonically with $d$, and as with $m$ at a certain depth, condition $c(x) \geq \lambda$ eliminates all nodes above a certain maximum depth $D$, as well as at every depth $d$ all nodes above a certain $m(d)$, where:

$$\frac{(p\gamma)^d}{1-\gamma} \geq \lambda \Rightarrow d \leq \frac{\log 1/(\lambda(1-\gamma))}{\log 1/(p\gamma)} = D$$
$$m \leq \frac{\log 1/(\lambda(1-\gamma))}{\log p/(1-p)} - d \frac{\log 1/(p\gamma)}{\log p/(1-p)} = m(d)$$

Note in the condition for $D$ we set $m = 0$ to obtain the largest probability. So, $m(d)$ decreases linearly with $d$, so that up to some depth $m^*$, $m(d) \geq d$ and we count all the nodes up to $m = d$; while above $m^*$, $m(d) < d$ and we count fewer nodes. The depth $m^*$ is obtained by solving $m(d) = d$, leading to $m^* = \frac{\log 1/(\lambda(1-\gamma))}{\log 1/(p\gamma)} = \frac{\log 1/(p\gamma)}{\log 1/(1-p)}D = \eta D$ with the notation $\eta = \frac{\log 1/(p\gamma)}{\log 1/(1-p)}$. The structure of the subtree satisfying $c(s) \geq \lambda$ is represented in Figure 5.

Figure 5: Schematic representation of the subtree satisfying $c(s) \geq \lambda$, shown in gray. Nodes with larger probabilities are put to the left. The thick line represents the fringe $m(d)$ where nodes stop being counted.

Now:

$$n_h(\lambda) = \sum_{d=0}^{D} \sum_{m=0}^{\min\{m(d),d\}} C^m_d \leq \sum_{d=0}^{D} \sum_{m=0}^{\min\{m(d),d\}} \left(\frac{de}{m}\right)^m$$
$$\leq \sum_{d=0}^{D} \sum_{m=0}^{\eta D} \left(\frac{De}{m}\right)^{m^*} = Dm^* \left(\frac{De}{m^*}\right)^{m^*}$$
$$= \eta D^2 \left(\frac{e}{\eta}\right)^{\eta D} = \bar{O}\left(\left(\frac{e}{\eta}\right)^{\eta D}\right)$$

where we used $C^m_d \leq (\frac{de}{m})^m$ as well as $\left(\frac{de}{m}\right)^m \leq (\frac{De}{m})^{m^*}$. The latter inequality can be shown by noticing that $\left(\frac{De}{m}\right)^m$, as a function of $m$, increases up to $m = D$, and $m^* \leq D$ is on the increasing part. Denoting now $\eta' = \frac{e}{\eta}$ and continuing:

$$n_h(\lambda) = \bar{O}(\eta D) = \bar{O}(\eta^{\frac{\log 1/(\lambda(1-\gamma))}{\log 1/(p\gamma)}}) = \bar{O}(\lambda^{-\frac{\log \eta'}{\log 1/(p\gamma)}})$$

leading to the value for $\delta_h = \frac{\log \eta'}{\log 1/(p\gamma)}$. 3

Similarly, it is shown that $n(\lambda) = \bar{O}(\lambda^{-\frac{\log K}{\log 1/(p\gamma)}})$ and thus $\delta = \frac{\log K}{\log 1/(p\gamma)}$, where the extra $K$ comes from the fact we count the nodes corresponding to all $K^d$ policies rather than just one.

The desired result is immediate: $\beta_{\text{prob}} = \frac{\delta}{1-\delta_h} = \frac{\log K}{\log 1/(p\gamma \eta')}$. Note throughout, we silently used the fact that $p$ is close to 1; indeed, this is required for some of the steps to be meaningful, such as having $\log 1/(p\gamma \eta') > 0$. 3

3 The definition of $n(s)$ in fact only requires counting the leaves of the subtree corresponding to $n_h(\lambda)$ (thick line in Figure 5), while we counted all the nodes (gray area). Exploiting this property is unlikely to be helpful, however, since in the upper bound derived for $n_h(\lambda)$ the inner term in the sum (corresponding to $C^m_d$, the number of nodes having a certain probability) is dominant. The fact that the whole tree is taken into account only enters the logarithmic component of the bound.