A Intersection of WFAs

The intersection of two WFAs $\mathcal{A}_1$ and $\mathcal{A}_2$ is a WFA denoted by $\mathcal{A}_1 \cap \mathcal{A}_2$ that accepts the set of sequences accepted by both $\mathcal{A}_1$ and $\mathcal{A}_2$ and is defined for all $\mathbf{x}$ by

$$(\mathcal{A}_1 \cap \mathcal{A}_2)(\mathbf{x}) = \mathcal{A}_1(\mathbf{x})\, \mathcal{A}_2(\mathbf{x}).$$

There exists a standard efficient algorithm for computing the intersection WFA [Mohri, 2009]. States of $\mathcal{A}_1 \cap \mathcal{A}_2$ are identified with pairs of states Q_1 of $\mathcal{A}_1$ and Q_2 of $\mathcal{A}_2$: $Q \subseteq Q_1 \times Q_2$, as are the set of initial and final states. Transitions are obtained by matching pairs of transitions from each weighted automaton and multiplying their weights following the rule

$$\left(q_1 \xrightarrow{a/w_1} q_1',\ q_2 \xrightarrow{a/w_2} q_2'\right) \quad \Rightarrow \quad (q_1, q_2) \xrightarrow{a/(w_1 w_2)} (q_1', q_2').$$

The worst-case space and time complexity of the intersection of two deterministic weighted finite automata (WFA) is linear in the size of the automaton the algorithm returns. In the worst case, this can be as large as the product of the sizes of the WFA intersected (i.e. $O(|\mathcal{A}_1||\mathcal{A}_2|)$, where $|\mathcal{A}_1|$ is the sum of the number of states and transitions of $\mathcal{A}_1$ and similarly with $|\mathcal{A}_2|$. This corresponds to the case where every transition of $\mathcal{A}_1$ can be paired up with every transition of $\mathcal{A}_2$. In practice far fewer transitions can be matched.

Notice that when both $\mathcal{A}_1$ and $\mathcal{A}_2$ are deterministic, then $\mathcal{A}_1 \cap \mathcal{A}_2$ is also deterministic since there is a unique initial state (pair of initial states of each WFA) and since there is at most one transition leaving $q_1 \in Q_1$ or $q_2 \in Q_2$ labeled with a given symbol $a \in \Sigma$.

In the case of $\mathcal{C} \cap S_T$, the WFA returned is $\mathcal{B}$, which has the same size as $\mathcal{A}$. $\mathcal{A}$ has more transitions than states since each state admits at least on outgoing transition, so its size is dominated by its number of transitions. Therefore, the complexity of intersection here is in $O(|E_\mathcal{A}|)$, where $|E_\mathcal{A}|$ is at most $|\mathcal{C}|NT$.

B WEIGHT-PUSHING algorithm

Here, we briefly describe the WEIGHT-PUSHING algorithm for a WFA $\mathcal{A}$ in the context of this paper [Mohri, 1997, 2009]. We denote by $Q_\mathcal{A}$ the set of states of $\mathcal{A}$, by $E_\mathcal{A}$ the set of transitions of $\mathcal{A}$, by $I_\mathcal{A}$ its initial state, by $F_\mathcal{A}$ the set of its final states, and by $\rho_\mathcal{A}(q)$ the final weight at a final state q – for the WFAs considered in this paper the final weights are all equal to one.

For any state $q \in Q_\mathcal{A}$, let $d[q]$ denote the sum of the weights of all paths from q to final states:

$$d[q] = \sum_{\pi \in P(q, F_\mathcal{A})} \text{weight}[\pi]\, \rho(\text{dest}[\pi]),$$

where $P(q, F_\mathcal{A})$ denotes the set of paths from q to a state in $F_\mathcal{A}$. For an acyclic WFA $\mathcal{A}$, the weights $d[q]$ can be computed in linear time in the size of $\mathcal{A}$, that is in $O(|Q_\mathcal{A}| + |E_\mathcal{A}|)$, or $O(|E_\mathcal{A}|)$ when every state of $\mathcal{A}$ admits at least one outgoing or incoming transition. This can be done using a general shortest-distance algorithm [Mohri, 1997, 2009].

The weight-pushing algorithm then consists of the following steps. For any transition $e \in E_\mathcal{A}$ such that $d[\text{src}[e]] \neq 0$, we update its weight as follows:

$$\text{weight}[e] \leftarrow d[\text{src}[e]]^{-1}\, \text{weight}[e]\, d[\text{dest}[e]].$$

For any state $q \in F_\mathcal{A}$ with $d[q] \neq 0$, we update its final weight as follows:

$$\rho_\mathcal{A}[q] \leftarrow d[q]^{-1}\, \rho_\mathcal{A}[q].$$

The resulting WFA is guaranteed to be stochastic (at any state q, the sum of the weights of all outgoing transitions, and the final weight if q is final, is equal to one) [Mohri, 2009]. Furthermore, if $d[I_\mathcal{A}] = 1$, that is if the sum of the weights of all paths is one, then path weights are preserved by this weight-pushing operation. Otherwise, the weights of all paths starting at the initial state is divided by $d[I_\mathcal{A}]$.

C Proof of Theorem 1

Theorem 1. *Let* q *denote the probability distribution defined by* $\mathcal{C}_T = \mathcal{C} \cap \mathcal{S}_T$ *and let* K *denote the number of accepting paths of* $\mathcal{C}_T$*. Then, the following upper bound holds for the weighted regret of* AWM*:*

$$\mathrm{Reg}_T(\mathrm{AWM}, \mathcal{C}) \leq \frac{\eta T}{8} + \frac{1}{\eta} \log \left[K^\eta \sum_{\mathbf{x} \in \Sigma^T} \mathsf{q}[\mathbf{x}]^\eta \right] \leq \frac{\eta T}{8} + \frac{1}{\eta} \log K.$$

Furthermore, when $K \geq 2$*, for any* $T > 0$*, there exists* $\eta^* > 0$*, decreasing function of* T*, such that:*

$$\mathrm{Reg}_T(\mathrm{AWM}, \mathcal{C}) \leq \sqrt{\frac{T H_{\eta^*}(\mathsf{q})}{2}} - H_{\eta^*}(\mathsf{q}) + \log K,$$

where $H_\eta(\mathsf{q}) = \frac{1}{1-\eta} \log \left(\sum_{\mathbf{x} \in \Sigma^T} \mathsf{q}[\mathbf{x}] \right)$ *is the* η*-Rényi entropy of* q*. The unweighted regret of* AWM *can be upper-bounded as follows:*

$$\mathrm{Reg}^0_T(\mathrm{AWM}, \mathcal{C}) \leq \frac{\eta T}{8} + \frac{1}{\eta} \log K.$$

Proof. We will use a standard potential-based argument. For any $t \geq 1$ and sequence $\mathbf{x} \in \Sigma^T$, let $w_t[\mathbf{x}]$ denote the sequence weight defining q_t via normalization, $\mathsf{q}_t[\mathbf{x}] = \frac{w_t[\mathbf{x}]}{\sum_{\mathbf{x}} w_t[\mathbf{x}]}$, that is $w_1[\mathbf{x}] = \mathsf{q}[\mathbf{x}]^\eta$ and, for $t \geq 2$, $w_t[\mathbf{x}] = w_1[\mathbf{x}] e^{-\eta \sum_{s=1}^{t-1} l_s[\mathbf{x}[s]]}$. Let Φ_t be the potential defined by $\Phi_t = \log \left(\sum_{\mathbf{x}} w_t[\mathbf{x}] \right)$ for $t \geq 1$. Then, by Hoeffding's inequality, we can write

$$\begin{aligned}
\Phi_{t+1} - \Phi_t &= \log \left[\frac{\sum_{\mathbf{x}} w_t[\mathbf{x}] \, e^{-\eta l_t[\mathbf{x}[t]]}}{\sum_{\mathbf{x}} w_t[\mathbf{x}]} \right] \\
&= \log \left[\mathop{\mathbb{E}}_{\mathbf{x} \sim \mathsf{q}_t} \left[e^{-\eta l_t[\mathbf{x}[t]]} \right] \right] \\
&\leq -\eta \mathop{\mathbb{E}}_{\mathbf{x} \sim \mathsf{q}_t} \left[l_t[\mathbf{x}[t]] \right] + \frac{\eta^2}{8} = -\eta \mathop{\mathbb{E}}_{a \sim \mathsf{p}_t} \left[l_t[a] \right] + \frac{\eta^2}{8}.
\end{aligned}$$

Summing up these inequalities over $t \in [1, T]$ results in the following upper bound:

$$\Phi_{T+1} - \Phi_1 \leq -\eta \sum_{t=1}^T \mathop{\mathbb{E}}_{a \sim \mathsf{p}_t} [l_t[a]] + \frac{\eta^2 T}{8}.$$

We can straightforwardly derive a lower bound for the same quantity for any sequence $\mathbf{x}_0 \in \Sigma^T$:

$$\begin{aligned}
\Phi_{T+1} - \Phi_1 &= \log \Big[\sum_{\mathbf{x}} w_{T+1}[\mathbf{x}] \Big] - \log \Big[\sum_{\mathbf{x}} w_1[\mathbf{x}] \Big] \\
&\geq \log[w_{T+1}[\mathbf{x}_0]] - \log \Big[\sum_{\mathbf{x}} w_1[\mathbf{x}] \Big] \\
&= -\eta \sum_{t=1}^T l_t[\mathbf{x}_0[t]] + \log[\mathsf{q}[\mathbf{x}_0]^\eta] - \log \Big[\sum_{\mathbf{x}} \mathsf{q}[\mathbf{x}]^\eta \Big].
\end{aligned}$$

Comparing the upper and lower bounds gives

$$-\eta \sum_{t=1}^T l_t[\mathbf{x}_0[t]] + \log[\mathsf{q}[\mathbf{x}_0]^\eta] - \log \Big[\sum_{\mathbf{x}} \mathsf{q}[\mathbf{x}]^\eta \Big] \leq -\eta \sum_{t=1}^T \mathop{\mathbb{E}}_{a \sim \mathsf{p}_t} [l_t[a]] + \frac{\eta^2 T}{8},$$

which can be rearranged as

$$\begin{aligned}
&\sum_{t=1}^T \mathop{\mathbb{E}}_{a \sim \mathsf{p}_t} [l_t[a]] - \sum_{t=1}^T l_t[\mathbf{x}_0[t]] \leq \frac{\eta T}{8} - \log[\mathsf{q}[\mathbf{x}_0]] + \frac{1}{\eta} \log \Big[\sum_{\mathbf{x}} \mathsf{q}[\mathbf{x}]^\eta \Big] \\
\Leftrightarrow &\sum_{t=1}^T \mathop{\mathbb{E}}_{a \sim \mathsf{p}_t} [l_t[a]] - \sum_{t=1}^T l_t[\mathbf{x}_0[t]] + \log[K \mathsf{q}[\mathbf{x}_0]] \leq \frac{\eta T}{8} + \frac{1}{\eta} \log \Big[K^\eta \sum_{\mathbf{x}} \mathsf{q}[\mathbf{x}]^\eta \Big].
\end{aligned}$$

Since the inequality holds for any sequence $\mathbf{x}_0 \in \Sigma^T$, it implies the following upper bound on the weighted regret:

$$\mathrm{Reg}_T \leq \frac{\eta T}{8} + \frac{1}{\eta} \log \Big[K^\eta \sum_{\mathbf{x}} \mathsf{q}[\mathbf{x}]^\eta\Big].$$

By Jensen's inequality, the inequality $\frac{1}{K}\sum_{\mathbf{x}} \mathsf{q}[\mathbf{x}]^\eta \leq \Big(\frac{1}{K}\sum_{\mathbf{x}} \mathsf{q}[\mathbf{x}]\Big)^\eta = \frac{1}{K^\eta}$ holds for $\eta \in (0,1)$. This implies the following general upper bounds on the weighted regret:

$$\mathrm{Reg}_T \leq \frac{\eta T}{8} + \frac{1}{\eta} \log K.$$

The weighted regret can also be upper bounded in terms of the Rényi entropy. Observe that

$$\frac{\eta T}{8} + \frac{1}{\eta} \log \Big[K^\eta \sum_{\mathbf{x}} \mathsf{q}[\mathbf{x}]^\eta\Big] = \frac{\eta T}{8} + \frac{1-\eta}{\eta} H_\eta(\mathsf{q}) + \log K.$$

$\eta \mapsto H_\eta(\mathsf{q})$ is known to be a non-increasing function (see e.g. [Van Erven and Harremos, 2014]). It follows that $\eta \mapsto \frac{\eta}{\sqrt{H_\eta(\mathsf{q})}}$ is an increasing function that increases at least linearly. If we assume that q is supported on more than a single sequence, then, we have $H_0(\mathsf{q}) > 0$. Thus, for any T, there exists a unique η^* such that $\frac{\eta^*}{\sqrt{H_{\eta^*}(\mathsf{q})}} = \sqrt{\frac{8}{T}}$. Furthermore, for $\eta \leq \eta^*$, the following inequality holds: $\frac{\eta}{\sqrt{H_\eta(\mathsf{q})}} \leq \sqrt{\frac{8}{T}}$. Thus, we can write

$$\begin{aligned}\frac{\eta T}{8} + \frac{1}{\eta} \log \Big[K^\eta \sum_{\mathbf{x}} \mathsf{q}[\mathbf{x}]^\eta\Big] &\leq \inf_{\eta \leq \eta^*} \frac{\eta T}{8} + \frac{1}{\eta} H_\eta(\mathsf{q}) - H_\eta(\mathsf{q}) + \log K \\ &\leq \sqrt{\frac{T H_{\eta^*}(\mathsf{q})}{2}} - H_{\eta^*}(\mathsf{q}) + \log K.\end{aligned}$$

The upper bound on the unweighted regret is obtained straightforwardly from the previous derivations using $\mathsf{q}[\mathbf{x}] = \frac{1}{K}$. □

Note that when the losses are *mixing*, we can also derive better constant-in-time regret guarantees by avoiding the use of Hoeffding's inequality.

D Proof of Theorem 2

Theorem 2. *The weighted regret of the* AWM *algorithm with respect to the WFA $\mathcal{C}$ when run with $\widehat{\mathcal{C}}_T$ instead of $\mathcal{C}_T$ can be upper bounded as follows:*

$$\mathrm{Reg}_T(\mathcal{A}, \mathcal{C}) \leq \frac{\eta T}{8} + \frac{1}{\eta} \log \Big[K^\eta \sum_{\mathbf{x}} \widehat{\mathsf{q}}[\mathbf{x}]^\eta\Big] + D_\infty(\mathsf{q} \| \widehat{\mathsf{q}}) \leq \frac{\eta T}{8} + \frac{1}{\eta} \log K + D_\infty(\mathsf{q} \| \widehat{\mathsf{q}}).$$

Its unweighted regret can be upper bounded as follows:

$$\mathrm{Reg}^0_T(\mathcal{A}, \mathcal{C}) \leq \max_{\mathcal{C}(\mathbf{x})>0} \frac{\eta T}{8} + \frac{1}{\eta} \log \left[\frac{1}{\mathsf{q}[\mathbf{x}]}\right] + \frac{1}{\eta} D_\infty(\mathsf{q} \| \widehat{\mathsf{q}}).$$

Proof. By Theorem 1 (and its proof), for any sequence $\mathbf{x}_0 \in \Sigma^T$, the following upper bound holds for the cumulative loss of AWM run with $\widehat{\mathcal{C}}_T$:

$$\sum_{t=1}^T \mathsf{p}_t \cdot \mathbf{l}_t - \sum_{t=1}^T l_t[\mathbf{x}_0[t]] + \log[\widehat{\mathsf{q}}[\mathbf{x}_0]] \leq \frac{\eta T}{8} + \frac{1}{\eta} \log \Big[\sum_{\mathbf{x}} \widehat{\mathsf{q}}[\mathbf{x}]^\eta\Big].$$

Thus, for any sequence $\mathbf{x}_0 \in \Sigma^T$ accepted by $\mathcal{C}_T$, we can write

$$\sum_{t=1}^T \mathsf{p}_t \cdot \mathbf{l}_t - \sum_{t=1}^T l_t[\mathbf{x}_0[t]] + \log[\mathsf{q}[\mathbf{x}_0] K] \leq \frac{\eta T}{8} + \frac{1}{\eta} \log \Big[K^\eta \sum_{\mathbf{x}} \widehat{\mathsf{q}}[\mathbf{x}]^\eta\Big] + \log \left[\frac{\mathsf{q}[\mathbf{x}_0]}{\widehat{\mathsf{q}}[\mathbf{x}_0]}\right],$$

which implies the following upper bound on the weighted regret:

$$\begin{aligned}\mathrm{Reg}_T(\mathcal{A}, \mathcal{C}) &\leq \frac{\eta T}{8} + \frac{1}{\eta} \log \left[K^\eta \sum_{\mathbf{x}} \widehat{\mathsf{q}}[\mathbf{x}]^\eta \right] + \sup_{\mathcal{C}_T(\mathbf{x}_0) > 0} \log \left[\frac{\mathsf{q}[\mathbf{x}_0]}{\widehat{\mathsf{q}}[\mathbf{x}_0]} \right] \\ &\leq \frac{\eta T}{8} + \frac{1}{\eta} \log \left[K^\eta \sum_{\mathbf{x}} \widehat{\mathsf{q}}[\mathbf{x}]^\eta \right] + D_\infty(\mathsf{q} \| \widehat{\mathsf{q}})\end{aligned}$$

As in the proof of Theorem 1, by Jensen's inequality, $\log \left[K^\eta \sum_{\mathbf{x}} \widehat{\mathsf{q}}[\mathbf{x}]^\eta \right] \leq \log K$, which implies the second inequality.

Similarly, by the proof of Theorem 1, the unweighted regret of AWM run with $\widehat{\mathcal{C}}_T$ can be upper bounded as follows:

$$\sum_{t=1}^{T} \mathsf{p}_t \cdot \mathbf{l}_t - \sum_{t=1}^{T} l_t[z_t] \leq \max_{\mathcal{C}(\mathbf{x}) > 0} \frac{\eta T}{8} + \frac{1}{\eta} \log \left[\frac{1}{\widehat{\mathsf{q}}[\mathbf{x}]} \right] = \max_{\mathcal{C}(\mathbf{x}) > 0} \frac{\eta T}{8} + \frac{1}{\eta} \log \left[\frac{1}{\mathsf{q}[\mathbf{x}]} \right] + \frac{1}{\eta} \log \left[\frac{\mathsf{q}[\mathbf{x}]}{\widehat{\mathsf{q}}[\mathbf{x}]} \right],$$

which completes the proof. $\square$

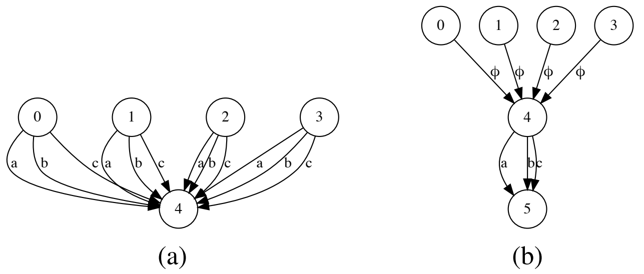

Figure 5: Example of the compression achieved by introducing a failure transition. (a) Standard automaton. (b) φ-automaton.

Algorithm 3: φ-CONVERT.

```
Algorithm: $\varphi$-CONVERT($\mathcal{C}$)
for each non-initial state $q \in \mathcal{C}$ do
    $S^*, Q^* \leftarrow \varphi$-SOURCESUBSET($\mathcal{C}$, $q$)
    if $|S^*| + |Q^*| < |S^*||Q^*|$ then
        $\tilde{q} \leftarrow$ NEWSTATE($\mathcal{C}$)
        $E_{\mathcal{C}} \leftarrow E_{\mathcal{C}} \cup \{(q, \varphi, 1, \tilde{q})\}$
        for each $q' \in Q^*$ do
            for each $e' \in E_{\mathcal{C}}[q']$ do
                if $(\mathrm{lab}[e'], \mathrm{weight}[e']) \in S^*$ then
                    $E_{\mathcal{C}} \leftarrow E_{\mathcal{C}} \cup \{(\tilde{q}, \mathrm{lab}[e'], \mathrm{weight}[e'], q)\}$
                    DELETETRANSITION[$E_{\mathcal{C}}, e'$]
```

E Failure transition algorithms

The computational complexity of the AWM algorithm presented in Section 3 is based on the size of the composed automaton $\mathcal{C} \cap \mathcal{S}_T$, which itself is related to the original size of $\mathcal{C}$. Similarly, if we were to apply AWM to an n-gram approximation, the computational complexity of the algorithm depends on the size of the approximating automaton. In this section, we introduce a technique to improve the computational cost of AWM by reducing the size of the automaton, using the notion of *failure transition (or φ-transition).*

φ-transitions are special transitions characterized by the semantic of "other". If, at a state q, there is no outgoing transition labeled with $a \in \Sigma$ and there is a φ-transition leaving q and reaching q', then the failure transition is taken instead without consuming the label, and the next state is determined using the transitions leaving q'. A φ-automaton is an automaton with φ-transitions. We assume that there is no φ-cycle in any of our φ-automata, and that there is at most one failure transition leaving any state. This implies that the number of consecutive failure transitions taken is bounded.

A failure transition can often replicate the role of multiple standard transitions when there is "symmetry" within an automaton, that is when there are many transitions leading to the same state from different states that consume the same set of labels. Figure 5 illustrates such a case.

E.1 Conversion

Notice that in Figure 5, the introduction of a failure transition removed $|S|$ transitions from $|Q|$ parent states while introducing $|Q|$ φ-transitions from each of the parent states to a new state q', and $|S|$ transitions from q' to q. Thus, the change in the number of transitions is $|S^*| + |Q^*| - |S^*||Q^*|$. This fact can be exploited to design an algorithm that iterates through the states of an automaton, and for each state, determines whether it is beneficial to introduce a failure transition between that state and (a subset of) its parents. We call this algorithm, φ-Convert, which uses another algorithm, φ-SOURCESUBSET as a subroutine to greedily select a candidate set of parent states from which to introduce a φ-transition for each state. The pseudocode for φ-CONVERT and φ-SOURCESUBSET are presented in Algorithm 3 and Algorithm 4 respectively.

Recall that the two main automata operations required for AWM are intersection and shortest-distance. While these two operations are standard for weighted automata, it is not as clear how one can perform them over weighted φ-automata. We now extend both to φ-automata.

Algorithm 4: φ-SourceSubset.

```
Algorithm: $\varphi$-SourceSubset($\mathcal{C}, q$)
$(S_0, Q_0) \leftarrow (\emptyset, \emptyset)$
$k^* \leftarrow 1$
for $k \leftarrow 1$ to $|$ Parents $[q]|$ do
    $q_k \leftarrow \text{argmax}_{q' \in \text{Parents}[q] \setminus Q_{k-1}} |(a, w) \in \Sigma \times \mathbb{R}_+ : \forall \tilde{q} \in \text{Parents}[q] \cup \{q'\}, (\tilde{q}, a, w, q) \in E_{\mathcal{C}}|$
    $S_k \leftarrow |(a, w) \in \Sigma \times \mathbb{R}_+ : \forall \tilde{q} \in \text{Parents}[q] \cup \{q_k\}, (\tilde{q}, a, w, q) \in E_{\mathcal{C}}|$
    $Q_k \leftarrow Q_{k-1} \cup \{q_k\}$
    $k^* \leftarrow \text{argmax}_{j \in \{k, k^*\}} \{|S_j||Q_j| - (|S_j| + |Q_j|)\}$
return $(S_{k^*}, Q_{k^*})$
```

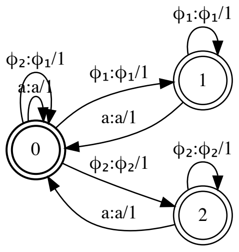

Figure 6: Illustration of the φ-filter $\mathcal{F}$.

E.2 Intersection using a φ-filter

One of the main automata operations required for AWM is intersection. The standard algorithm for intersection of automata (Appendix A), which is based on matching transitions, can return an incorrect result in the presence of φ-transitions. Specifically, the algorithm may produce multiple φ-paths between two states, which leads to redundancy and incorrect weights.

Redundant φ-paths are generated by standard intersection algorithms because when the algorithm is in state q_1 in WFA $\mathcal{C}_1$ and state q_2 in $\mathcal{C}_2$, both of which contain outgoing φ-transitions, the algorithm may take any of the following steps: (1) move forward on a φ-transition in $\mathcal{C}_1$ while staying at q_2; (2) move forward on a φ-transition in $\mathcal{C}_2$ while staying in $\mathcal{C}_1$; or (3) move forward in both $\mathcal{C}_1$ and $\mathcal{C}_2$.

To avoid this situation, we introduce the concept of a φ-*filter*, which is a *finite state transducer (FST)* that can filter out all but one φ-path between any two states.

Our φ-filter is designed to modify the two input automata in a way that will distinguish between the above cases. In $\mathcal{C}_1$, for every φ-transition, we rename the label φ as φ_2. Moreover, at the source and destination states of every φ-transition, we introduce new self-loop transitions labeled with φ_1 and with weight 1. Thus, a transition labeled with φ_2 will indicate a "move forward," while a transition labeled with φ_1 will indicate a "stay." Similarly, in $\mathcal{C}_2$, we rename the φ labels as φ_1, and we introduce self-loops labeled with φ_2 and weight 1 at the source and destination states of every φ-transition. With these modifications, any φ-path resulting from the composition algorithm will include transitions of the form: (1) $(\varphi_2 : \varphi_2)$; (2) $(\varphi_1 : \varphi_1)$; or (3) $(\varphi_2 : \varphi_1)$.

Now consider the finite-state transducer $\mathcal{F}$ illustrated in Figure 6, which will serve as our φ-filter. The composition of any two φ-automata and the φ-filter $\mathcal{F}$, $\mathcal{C}_1 \circ \mathcal{F} \circ \mathcal{C}_2$, will result in a finite-state transducer whose transitions have labels in $\{(a : a)\}_{a \in \Sigma} \cup \{(\varphi_2 : \varphi_2), (\varphi_1 : \varphi_1), (\varphi_2 : \varphi_1)\}$.[8] Moreover, we identify all label pairs in $\{(\varphi_2 : \varphi_2), (\varphi_1 : \varphi_1), (\varphi_2 : \varphi_1)\}$ using the same semantic of "other" as we did with φ. Thus, we can identify all label pairs in $\{(\varphi_2 : \varphi_2), (\varphi_1 : \varphi_1), (\varphi_2 : \varphi_1)\}$ with the single pair $(\varphi : \varphi)$ and treat the result of composition as simply a weighted finite automaton.

[8]Composition is a standard algorithm for weighted finite-state transducers which coincides with the intersection operation in the special case of WFA (see Mohri [2009]).

Algorithm 5: φ-AUTOMATAWEIGHTEDMAJORITY(φ-AWM).

```
Algorithm: $\varphi$-AWM($\mathcal{C}$, $\eta$)
$\mathcal{C} \leftarrow \varphi$-CONVERT($\mathcal{C}$)
$\mathcal{B} \leftarrow \mathcal{C} \cap \mathcal{F} \cap \mathcal{S}_T$
$\mathcal{A} \leftarrow$ WEIGHT-PUSHING($\mathcal{B}^\eta$)
$\boldsymbol{\beta} \leftarrow$ BWDDIST($\mathcal{A}$)
$\boldsymbol{\alpha} \leftarrow 0$; $\boldsymbol{\alpha}[I_\mathcal{A}] \leftarrow 1$
for each $e \in E_\mathcal{A}^{0\to 1}$ do
    $\mathsf{p}_1[\mathrm{lab}[e]] \leftarrow \mathrm{weight}[e]$.
for $t \leftarrow 1$ to $T$ do
    $i_t \leftarrow$SAMPLE($\mathsf{p}_t$); PLAY($i_t$); RECEIVE($\mathbf{l}_t$)
    $Z \leftarrow 0$; $\mathbf{w} \leftarrow 0$
    for each $e \in E_\mathcal{A}^{t\to t+1}$ do
        $\mathrm{weight}[e] \leftarrow \mathrm{weight}[e]\, e^{-\eta l_t[\mathrm{lab}[e]]}$
        $\mathbf{w}[\mathrm{lab}[e]] \leftarrow \mathbf{w}[\mathrm{lab}[e]] + \boldsymbol{\alpha}[\mathrm{src}[e]]\,\mathrm{weight}[e]\,\boldsymbol{\beta}[\mathrm{dest}[e]]$
        $Z \leftarrow Z + \mathbf{w}[\mathrm{lab}[e]]$
        $\boldsymbol{\alpha}[\mathrm{dest}[e]] \leftarrow \boldsymbol{\alpha}[\mathrm{dest}[e]] + \boldsymbol{\alpha}[\mathrm{src}[e]]\,\mathrm{weight}[e]$
        if $\mathrm{lab}[e] \neq \varphi$ then
            $\tilde{q} = \mathrm{src}[e]$; $w_\varphi \leftarrow 1$
            while $\exists e_\varphi \in E[\tilde{q}]$ with $\mathrm{lab}[e_\varphi] = \varphi$ do
                $w_\varphi \leftarrow w_\varphi\,\mathrm{weight}[e_\varphi]$
                if $\exists e' \in E[\mathrm{dest}[e_\varphi]]$ with $\mathrm{lab}[e'] = \mathrm{lab}[e]$ then
                    $\alpha[\mathrm{dest}[e']] \leftarrow \alpha[\mathrm{dest}[e']] - w_\varphi \mathrm{weight}[e']$
                    BREAK
                else
                    $\tilde{q} \leftarrow \mathrm{dest}[e_\varphi]$
    $\mathsf{p}_{t+1} \leftarrow \frac{\mathbf{w}}{Z}$
```

E.3 Update of $\boldsymbol{\alpha}$ using a modified shortest-distance algorithm

The other key ingredient of the AWM algorithm is the update of $\boldsymbol{\alpha}$ using the shortest-distance algorithm for WFA. However, updating $\boldsymbol{\alpha}$ as we did in AWM may result in summing over 'obsolete φ-transitions'. For example, if at a given state q, there is a transition labeled with a to q' and a φ-transition whose destination state has a single outgoing transition also labeled with a to q', the second path should not be considered.

To account for these types of situations, we use the fact that the semiring $(\mathbb{R}_+, +, \times, 0, 1)$ admits a natural extension to a ring structure under the standard additive inverse -1. Specifically, upon encountering a transition e labeled with a leaving state q, we will check for φ-transitions with destination states that admit further transitions e' labeled with a. Any such transition should not be considered under the semantic of the φ-transition and thus should not contribute any weight to the distance to $\alpha[\mathrm{dest}[e']]$. To correctly account for these paths, we will *preemptively* subtract the weight of e' from its destination state. When the algorithm processes the φ-transition directly, it will add this weight back so that the total contribution of this path is zero.

E.4 φ-AWM algorithm

With the addition of the φ-filter and the modified $\boldsymbol{\alpha}$ update described above, we can present φ-AWM, an extension of AWM that can handle φ-automata. Given an input automaton (not necessarily with φ-transitions), the algorithm first calls φ-CONVERT to determine whether it is beneficial to introduce φ-transitions. The algorithm then composes the output with Σ^T (using the φ-filter) to compute the set of sequences of length T that are accepted by $\mathcal{C}$. Then, the algorithm updates the weights of the automaton in a similar manner as in AWM with the additional adjustment of preemptively accounting for φ-transitions. Algorithm 5 presents the pseudocode for φ-AWM.

Since the update of p_t in φ-AWM is mathematically equivalent to the one in AWM we obtain the same regret guarantees as in Theorem 1. Moreover, if we denote by $N_\varphi(Q_{\mathcal{C}_T})$ the maximum number of consecutive φ-transitions leaving states in $Q_{\mathcal{C}_T}$, the total computational cost of the algorithm is in $O\left(\sum_{t=1}^T N_\varphi(Q_{\mathcal{C}_T,t-1})|E_\mathcal{A}^{t\to t+1}|\right)$.

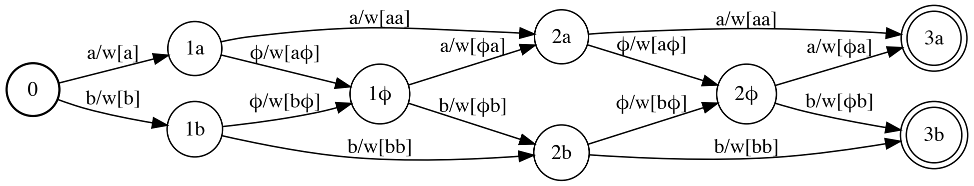

Figure 7: An illustration of a bigram model approximating the k-shifting automaton composed with $\mathcal{S}$. φ-CONVERT has been applied to the bigram model, making it smaller than a standard bigram model.

For the k-shifting automaton, the per-iteration computational complexity of φ-AWM is now $O(Nk)$, since there is at most one consecutive φ-transition in the output of φ-Convert, and we now aggregate transitions at each time using failure transitions. This is a factor of N better than that of AWM, and only a factor of k worse than the FIXED-SHARE algorithm of Herbster and Warmuth [1998]. If we intersect the k-shifting automaton with Σ^T, approximate the result with a bigram model, and then convert this model into a φ-automaton, we obtain an algorithm that runs in $O(N)$, which is the same as that of FIXED-SHARE. See Figure 7 for an illustration.

Algorithm 6: PROD-EG.

```
Algorithm: PROD-EG($\mathsf{q}_1 \in (\Delta_N)^m$, $\eta$)
for $s = 1, 2, \ldots, \tau$ do
    PLAY($\mathsf{q}_s$)
    RECEIVE($\nabla f(\mathsf{q}_s)$)
    for $j = 1, 2, \ldots, m$ do
        for $i = 1, 2, \ldots, N$ do
            $\mathsf{q}_{s+1,j}(i) = \mathsf{q}_{s,j} e^{-\eta \frac{\partial f}{\partial \mathsf{q}_j(i)}(\mathsf{q}_{s,j})}$
```

F PROD-EG

The pseudocode of the PROD-EG algorithm, which is based on a simple multiplicative update, is given in Algorithm 6. The following provides a general guarantee for the convergence of the algorithm.

Theorem 5 (PRODUCT-EXPONENTIATED GRADIENT (PROD-EG))**.**

Let $(\Delta_N)^m$ be the product of m $(N-1)$-dimensional simplices, and let $f\colon (\Delta_N)^m \to \mathbb{R}$ be a convex function whose partial subgradients have absolute values all bounded by L. Let $\mathsf{q}_{1,j}(i) = \frac{1}{N}$ for $i \in [N]$ and $j \in [m]$. Then, PROD-EG *benefits from the following guarantee:*

$$f\left(\frac{1}{\tau}\sum_{s=1}^{\tau} \mathsf{q}_s\right) - f(\mathsf{q}^*) \leq \frac{1}{\eta\tau} m \log(N) + 2\eta L.$$

Proof. Consider the mirror map $\psi\colon (\Delta_N)^m \to \mathbb{R}$ defined by $\psi(\mathsf{q}) = \sum_{j=1}^m \sum_{i=1}^N \mathsf{q}_j(i) \log \mathsf{q}_j(i)$. This induces the Bregman divergence:

$$B_\psi(\mathsf{q}, \mathsf{q}') = \sum_{j=1}^m \sum_{i=1}^N \mathsf{q}_j(i) \log\left(\frac{\mathsf{q}_j(i)}{\mathsf{q}'_j(i)}\right).$$

Since each relative entropy is 1-strongly convex with respect to the l_1 norm over a single simplex, the additivity of strong convexity implies that B_ψ is 1-strongly convex with respect to the l_1 norm defined over$(\Delta_N)^m$.

The update described in the theorem statement corresponds to the mirror descent update based on B_ψ:

$$\mathsf{q}_{s+1} = \operatorname*{argmin}_{\mathsf{q} \in (\Delta_N)^m} \langle g_s, \mathsf{q}\rangle + B_\psi(\mathsf{q}, \mathsf{q}_s).$$

where $g_s \in \partial(f(\mathsf{q}_s))$ is an element of the subgradient of f at q_s. Thus, the standard mirror descent regret bound (e.g. [Bubeck et al., 2015]) implies that

$$\frac{1}{\tau}\sum_{s=1}^{\tau} f(\mathsf{q}_s) - f(\mathsf{q}^*) \leq \frac{1}{\eta\tau} B_\psi(\mathsf{q}^*, \mathsf{q}_1) + \eta 2L.$$

The result now follows from the fact the observation that $B_\psi(\mathsf{q}^*, \mathsf{q}_1) \leq m \log(N)$. □

For the minimum Rényi divergence optimization problem (8), we can apply PROD-EG to the product of $m = \sum_{j=1}^n |\Sigma|^{n-j}$ simplices, each one corresponding to a conditional probability with a specific history. First, we remark that the subgradient of the maximum of a family of convex functions at a point can always be chosen from the subgradient of the maximizing function at that point. Specifically, let $\{f_\alpha\}_{\alpha \in \mathcal{A}}$ be a family of convex functions, and let $\alpha(x) = \operatorname{argmax}_\alpha f_\alpha(x)$. Then, it follows that

$$\max_\alpha f_\alpha(x) - \max_\alpha f_\alpha(y) \geq f_{\alpha(y)}(x) - f_{\alpha(y)}(y) \geq \langle \nabla f_{\alpha(y)}(y), x - y\rangle.$$

Let $\mathbf{x}$ be the maximizing path of the minimum Rényi divergence objective. We can then write

$$\begin{aligned}\log\left[\frac{\mathsf{q}[\mathbf{x}]}{\widehat{\mathsf{q}}_{\mathsf{w}}[\mathbf{x}]}\right] &= \mathsf{q}[\mathbf{x}] - \sum_{t=1}^{T}\log \mathsf{w}\left[\mathbf{x}[t]\middle|\mathbf{x}_{\max(t-n+1,1)}^{t-1}\right]\\ &= \mathsf{q}[\mathbf{x}] - \sum_{j=1}^{n}\sum_{\mathbf{z}_1^j\in\Sigma^j}\sum_{t=1}^{T} 1_{j=\min(t,n)} 1_{\mathbf{x}_{\max(t-n+1,1)}^{t}=\mathbf{z}_1^j}\log \mathsf{w}\left[\mathbf{z}[j]\middle|\mathbf{z}_1^{j-1}\right].\end{aligned}$$

Thus, its partial derivative with respect to $\mathsf{w}\left[\mathbf{z}[j]\middle|\mathbf{z}_1^{j-1}\right]$ is:

$$\frac{\partial}{\partial \mathsf{w}\left[\mathbf{z}[j]\middle|\mathbf{z}_1^{j-1}\right]}\log\left[\frac{\mathsf{q}[\mathbf{x}]}{\widehat{\mathsf{q}}_{\mathsf{w}}[\mathbf{x}]}\right] = -\sum_{t=1}^{T}\frac{1_{j=\min(t,n)} 1_{\mathbf{x}_{\max(t-n+1,1)}^{t}=\mathbf{z}_1^j}}{\mathsf{w}\left[\mathbf{z}[j]\middle|\mathbf{z}_1^{j-1}\right]}.$$

Thus, by tuning PROD-EG with an adaptive learning rate

$$\eta_t \propto \frac{1}{\sqrt{\sum_{s=1}^{t}\left\|\nabla \log\left[\frac{\mathsf{q}[\mathbf{x}(s)]}{\widehat{\mathsf{q}}_{\mathsf{w}_s}[\mathbf{x}(s)]}\right]\right\|_{\infty}^{2}}},$$

where $\mathbf{x}(s) = \operatorname{argmax}_{\mathbf{x}\in\mathcal{C}_T}\log\left[\frac{\mathsf{q}[\mathbf{x}]}{\widehat{\mathsf{q}}_{\mathsf{w}_s}[\mathbf{x}]}\right]$, we can derive the following guarantee for PROD-EG applied to the n-gram approximation problem.

Corollary 1 (n-gram approximation guarantee)**.** *There exists an optimization algorithm outputting a sequence of conditional probabilities* $(\mathsf{q}_t)_{t=1}^{\infty}$ *such that* $\left(\frac{1}{T}\sum_{t=1}^{T}\mathsf{q}_t\right)$ *approximates the* ∞*-Rényi optimal* n*-gram solution with the following guarantee:*

$$F\left(\frac{1}{\tau}\sum_{s=1}^{\tau}\mathsf{q}_t\right) - F(\mathsf{q}^*) \leq \sqrt{\frac{2N^n\log(N)\sum_{s=1}^{\tau}\max_{\substack{j\in[n]\\ \mathbf{z}_1^j\in\Sigma^j}}\left|\sum_{t=1}^{T}\frac{1_{j=\min(t,n)} 1_{\mathbf{x}_{\max(t-n+1,1)}^{t}=\mathbf{z}_1^j}}{\mathsf{w}_s\left[\mathbf{z}[j]\middle|\mathbf{z}_1^{j-1}\right]}\right|}{(N-1)T^2}}.$$

Each iteration of PROD-EG admits a computational complexity that is linear in the dimension of the feature space. Since we have specified an n-gram model as the product of $\frac{N^n-1}{N-1}$ simplices, the total per-iteration cost of solving the convex optimization problem is in $O\left(\frac{N(N^n-1)}{N-1}\right) = O(N^n)$. Since the minimum Rényi divergence is not Lipschitz, the maximizing ratio in the convergence guarantee may also become large when the choice of n is too small. In all cases, observe that this approximation problem can be solved offline.

G Minimum Rényi divergence unigram models

Theorem 3. *Assume that $\mathcal{C}_T$ admits uniform weights over all paths and $\Sigma = \{a_1, a_2\}$. For $j \in \{1, 2\}$, let $n(a_j)$ be the smallest number of occurrences of a_j in a path of $\mathcal{C}_T$. For any $j \in \{1, 2\}$, define*

$$\mathsf{q}[a_j] = \frac{\max\left\{1, \frac{n(a_j)}{T-n(a_j)}\right\}}{1 + \max\left\{1, \frac{n(a_j)}{T-n(a_j)}\right\}}.$$

Then, the unigram model $\mathsf{w} \in \mathcal{W}_1$ solution of ∞-Rényi divergence optimization problem is defined by $\mathsf{w}[a_{j^*}] = \mathsf{q}[a_{j*}]$, $\mathsf{w}[a_{j'}] = 1 - \mathsf{w}[a_{j^*}]$, with $j^* = \operatorname{argmax}_{j\in\{1,2\}}\ n(a_j) \log \mathsf{q}[a_j] + [T - n(a_j)] \log \big[1 - \mathsf{q}[a_j]\big]$.

Proof. We seek a unigram distribution q_w that is a solution of:

$$\min_{\mathsf{w}\in\mathcal{W}_1} \sup_{\mathbf{x}\in\mathcal{C}_T} \log\left(\frac{\mathsf{q}[\mathbf{x}]}{\mathsf{q}_\mathsf{w}[\mathbf{x}]}\right).$$

Since $\mathcal{C}_T$ admits uniform weights, $\mathsf{q}[\mathbf{x}] = \frac{1}{|\mathcal{C}_T|}$, and since q_w is the distribution induced by a unigram model, $\log \mathsf{q}_\mathsf{w}[\mathbf{x}]$ can be expressed as follows:

$$\log \mathsf{q}_\mathsf{w}[\mathbf{x}] = n_\mathbf{x}(a_1) \log p(a_1) + [T - n_\mathbf{x}(a_1)] \log(1 - p(a_1)),$$

where $p(a_j)$ is the automaton's weight on transitions labeled with a_j and $n_\mathbf{x}(a_j)$ is the count of a_j in the sequence $\mathbf{x}$. Thus, the optimization problem is equivalent to the following problem:

$$-\max_{p(a_1)\in[0,1]} \min_{\mathbf{x}\in\mathcal{C}_T} n_\mathbf{x}(a_1) \log p(a_1) + [T - n_\mathbf{x}(a_1)] \log(1 - p(a_1)).$$

Denote the objective by $F(p(a_1), n_\mathbf{x}(a_1))$. Then, the partial derivatives with respect to the label counts are given by

$$\frac{\partial F}{\partial n_\mathbf{x}(a_1)} = \log p(a_1) - \log\Big(1 - p(a_1)\Big).$$

Thus, $\frac{\partial F}{\partial n_\mathbf{x}(a_1)} \geq 0$ if and only if $p(a_1) \geq 1 - p(a_1)$. Furthermore, if $p(a_1) \geq 1 - p(a_1)$, then the sequence $\mathbf{x}$ chosen in the optimization problem is the sequence with the minimal count of symbol a_1. Similarly, if $p(a_2) \geq 1 - p(a_2)$, then the sequence $\mathbf{x}$ chosen in the optimization problem is the one with minimal count of a_2.

Since we have either $p(a_1) \geq p(a_2)$ or vice versa (potentially both), we can write the optimization problem as:

$$-\max_{k\in\{1,2\}} \max_{\substack{p(a_j)\geq 1-p(a_j)\\ j\neq k}} \min_{\{n_\mathbf{x}(a_j)\}_{j\neq k}:\ \mathbf{x}\in\mathcal{C}_T} n_\mathbf{x}(a_j) \log p(a_j) + [T - n_\mathbf{x}(a_j)] \log(1 - p(a_j)).$$

Given $k \in \{1, 2\}$, let $\mathbf{x}(k)$ be the sequence that minimizes $n_\mathbf{x}(a_j)$ over all $\mathbf{x}$ for $j \neq k$. Denote these counts $n_{\mathbf{x}(k)}(a_j)$ by $n(a_j)$. Then we can rewrite the objective as:

$$-\max_{k=1,2,\ldots,N} \max_{\substack{p(a_j)\geq 1-p(a_j)\\ j\neq k}} n(a_j) \log p(a_j) + [T - n(a_j)] \log(1 - p(a_j)).$$

Denote the objective for this new term by $\tilde{F}_k$, which is a function of $p(a_j)$. The partial derivative of $\tilde{F}_k$ with respect to $p(a_j)$ is:

$$\frac{\partial \tilde{F}_k}{\partial p(a_j)} = \frac{n(a_j)}{p(a_j)} - \frac{T - n(a_j)}{1 - p(a_j)},$$

which is equal to 0 if and only if

$$p(a_j) = \frac{n(a_j)}{T - n(a_j)}(1 - p(a_j)) = \max\left\{1, \frac{n(a_j)}{T - n(a_j)}\right\}(1 - p(a_j)).$$

The last equality follows from our assumption that $p(a_j) \geq 1 - p(a_j)$. Now, let $\mathsf{q}(a_j)$ denote the probabilities that we have just computed. Then, we can write the optimization problem of $\tilde{F}_k$ as:

$$-\max_{k\in\{1,2\}, j\in\{1,2\}\setminus\{k\}} n(a_j) \log \mathsf{q}(a_j) + [T - n(a_j)] \log(1 - \mathsf{q}(a_j)).$$

□

H Maximum likelihood n-gram models

Theorem 4. *Let $\mathcal{C}_T$ be the k-shifting automaton for some k. Then, the bigram model w_2 obtained by minimizing relative entropy is defined for all $a_1, a_2 \in \Sigma$ by*

$$\mathsf{q}_{\mathsf{w}_2}[a_1 a_2] = \frac{1}{N}\Big[1 - \frac{k}{(T-1)}\Big] 1_{a_1 = a_2} + \frac{1}{N}\Big[\frac{k}{(T-1)(N-1)}\Big] 1_{a_1 \neq a_2}.$$

Moreover, its approximation error can be bounded by a constant (independent of T):

$$D_\infty(\mathsf{q} \| \mathsf{q}_{\mathsf{w}_2}) \leq -\log\left[1 - 2e^{-\frac{1}{12k}}\right].$$

Proof. Let $a_1, a_2 \in \Sigma$. Then, we can write

$$\mathsf{q}_{\mathsf{w}_2}[a_2|a_1] = \mathsf{q}_{\mathsf{w}_2}[a_2|a_1, a_2 = a_1]\,\mathsf{q}_{\mathsf{w}_2}[a_2 = a_1] + \mathsf{q}_{\mathsf{w}_2}[a_2|a_1, a_2 \neq a_1]\,\mathsf{q}_{\mathsf{w}_2}[a_2 \neq a_1].$$

Consider first the case where $a_2 = a_1$. Then, $\mathsf{q}_{\mathsf{w}_2}[a_2|a_1, a_2 = a_1] = 1$, and $\mathsf{q}_{\mathsf{w}_2}[a_2 = a_1]$ is the expected number of times that we see label a_2 agreeing with label a_1. Since q is uniform for the k-shifting automaton, the expected counts are pure counts, and the probability that we see two consecutive labels agreeing is $1 - \frac{k}{T-1}$. Now, consider the case where $a_2 \neq a_1$. By symmetry, $\mathsf{q}_{\mathsf{w}_2}[a_2|a_1, a_2 \neq a_1] = \frac{1}{N-1}$, since a_2 is equally likely to be any of the other $N-1$ labels. Moreover, $\mathsf{q}_{\mathsf{w}_2}[a_2 \neq a_1] = \frac{k}{T-1}$. Thus, the following holds:

$$\mathsf{q}_{\mathsf{w}_2}[a_2|a_1] = \frac{1}{N-1}\frac{k}{T-1} 1_{a_1 \neq a_2} + \left[1 - \frac{k}{T-1}\right] 1_{a_1 = a_2}.$$

By symmetry, we can write $\mathsf{q}_{\mathsf{w}_2}[a_1] = \frac{1}{N}$, therefore,

$$\mathsf{q}_{\mathsf{w}_2}[a_1 a_2] = \mathsf{q}_{\mathsf{w}_2}[a_2|a_1]\mathsf{q}_{\mathsf{w}_2}[a_1] = \frac{k}{N(N-1)(T-1)} 1_{a_1 \neq a_2} + \left[\frac{T-1-k}{N(T-1)}\right] 1_{a_1 = a_2}.$$

Since the k-shifting automaton has uniform weights and $\mathsf{q}_{\mathsf{w}_2}$ is uniform on $\mathcal{C}_T$, we can write for any string $\mathbf{x}$ accepted by $\mathcal{C}_T$:

$$\begin{aligned}
\log\left[\frac{\mathsf{q}[\mathbf{x}]}{\mathsf{q}_{\mathsf{w}_2}[\mathbf{x}]}\right] &= \log\left[\frac{1}{\mathsf{q}_{\mathsf{w}_2}[\mathbf{x}]}\right] - \log(|\mathcal{C}_T|)\\
&= \log\left[\frac{1}{\mathsf{q}_{\mathsf{w}_2}[\mathbf{z} = \mathbf{x}|\mathbf{z} \in \mathcal{C}_T]\mathsf{q}_{\mathsf{w}_2}[\mathbf{z} \in \mathcal{C}_T] + \mathsf{q}_{\mathsf{w}_2}[\mathbf{z} = \mathbf{x}|\mathbf{z} \notin \mathcal{C}_T]\mathsf{q}_{\mathsf{w}_2}[\mathbf{z} \notin \mathcal{C}_T]}\right] - \log(|\mathcal{C}_T|)\\
&= \log\left[\frac{1}{\frac{1}{|\mathcal{C}_T|}\mathsf{q}_{\mathsf{w}_2}[\mathbf{z} \in \mathcal{C}_T] + \mathsf{q}_{\mathsf{w}_2}[\mathbf{z} = \mathbf{x}|\mathbf{z} \notin \mathcal{C}_T]\mathsf{q}_{\mathsf{w}_2}[\mathbf{z} \notin \mathcal{C}_T]}\right] - \log(|\mathcal{C}_T|)\\
&\leq \log\left[\frac{|\mathcal{C}_T|}{\mathsf{q}_{\mathsf{w}_2}[\mathbf{z} \in \mathcal{C}_T]}\right] - \log(|\mathcal{C}_T|) = \log\left[\frac{1}{\mathsf{q}_{\mathsf{w}_2}[\mathbf{z} \in \mathcal{C}_T]}\right].
\end{aligned}$$

The probability that a string $\mathbf{z}$ is accepted by $\mathcal{C}_T$ (under the distribution $\mathsf{q}_{\mathcal{A}_2}$) is equal to the probability that it admits exactly k shifts. Let $\xi_t = 1_{\{\mathbf{z}\text{ shifts from } t-1 \text{ to } t\}}$ be a random variable indicating whether there is a shift at the t-th symbol in sequence $\mathbf{z}$. This is a Bernoulli random variable bounded by 1 with mean $\frac{k}{T-1}$ and variance $\frac{k}{T-1}(1 - \frac{k}{T-1})$. Since each shift occurs with probability $\frac{k}{T-1}$, we can use Sanov's theorem to write the following bound:

$$\mathsf{q}_{\mathsf{w}_2}[\mathbf{z} \notin \mathcal{C}_T] = \mathsf{q}_{\mathsf{w}_2}\left[\left|\sum_{t=2}^{T} \xi_t - k\right| > \frac{1}{2}\right] \leq 2e^{-(T-1)u},$$

where $u = (T-1)\min\left\{D\left(\frac{k+\frac{1}{2}}{T-1}\middle\|\frac{k}{T-1}\right), D\left(\frac{k-\frac{1}{2}}{T-1}\middle\|\frac{k}{T-1}\right)\right\}$. We now give lower bounds on the relative entropy terms arguments of the minimum operator. For the first term, using the inequalities $\log(1+x) \geq \frac{x}{1+\frac{x}{2}}$ and $\log(1+x) < x$, we can write

$$\begin{aligned}
&-D\left(\frac{k+\frac{1}{2}}{T-1}\middle\|\frac{k}{T-1}\right)\\
&=\left(1+\frac{1}{2k}\right)\frac{k}{T-1}\log\frac{1}{1+\frac{1}{2k}}+\left(1-\frac{k}{T-1}-\frac{1}{2k}\frac{k}{T-1}\right)\log\left(1+\frac{\frac{1}{2k}\frac{k}{T-1}}{1-\frac{k}{T-1}-\frac{1}{2k}\frac{k}{T-1}}\right)\\
&\leq\left(1+\frac{1}{2k}\right)\frac{k}{T-1}\frac{-\frac{1}{2k}}{1+\frac{1}{4k}}+\left(1-\frac{k}{T-1}-\frac{1}{2k}\frac{k}{T-1}\right)\frac{\frac{1}{2k}\frac{k}{T-1}}{1-\frac{k}{T-1}-\frac{1}{2k}\frac{k}{T-1}}\\
&=\frac{1}{2k}\frac{k}{T-1}\left(1-\frac{1+\frac{1}{2k}}{1+\frac{1}{4k}}\right)=\frac{-\frac{1}{8k^2}\frac{k}{T-1}}{1+\frac{1}{4k}}=\frac{-\frac{1}{4k^2}\frac{k}{T-1}}{2+\frac{1}{4k}}\leq-\frac{1}{12k(T-1)}.
\end{aligned}$$

Similarly, we can write:

$$\begin{aligned}
&-D\left(\left(1-\frac{1}{2k}\right)\frac{k}{T-1}\middle\|\frac{k}{T-1}\right)\\
&=\left(1-\frac{1}{2k}\right)\frac{k}{T-1}\log\frac{1}{1-\frac{1}{2k}}+\left[1-\frac{k}{T-1}+\frac{1}{2k}\frac{k}{T-1}\right]\log\left[1-\frac{\frac{1}{2k}\frac{k}{T-1}}{1-\frac{k}{T-1}+\frac{1}{2k}\frac{k}{T-1}}\right]\\
&\leq\left[\frac{1}{2k}-\frac{1}{8k^2}\right]\frac{k}{T-1}+\left[1-\frac{k}{T-1}+\frac{1}{2k}\frac{k}{T-1}\right]\frac{-\frac{1}{2k}\frac{k}{T-1}}{1-\frac{k}{T-1}+\frac{1}{2k}\frac{k}{T-1}}\\
&=-\frac{\frac{1}{4k^2}\frac{k}{T-1}}{2}=-\frac{1}{8k(T-1)}.
\end{aligned}$$

Using these inequalities, we can further bound the approximation error in the regret bound by:

$$\log\left[\frac{1}{\mathsf{q}_{\mathsf{w}_2}[\mathbf{z}\in\mathcal{C}_T]}\right]\leq\log\left[\frac{1}{1-2e^{-\frac{1}{12k}}}\right]=-\log\left(1-2e^{-\frac{1}{12k}}\right),$$

which completes the proof. □

I Extension to sleeping experts

In this section, we present AWAKEAWM, a path-based weighted majority algorithm that generalizes the algorithms in [Freund et al., 1997] to arbitrary families of expert sequences. Like AWM, AWAKEAWM maintains a set of weights over all the paths in the input automaton. At each round t, the algorithm performs a weighted majority-type update. However, it normalizes the weights so that the total weight of the awake set remains unchanged. This prevents the algorithm from "overfitting" to experts that have been asleep for many rounds. The pseudocode of AWAKEAWM is provided as Algorithm 7, and it is accompanied by the following theoretical guarantee.

Theorem 6 (Regret Bound for AWAKEAWM)**.** *Let K denote the number of accepting paths of $\mathcal{C}_T = \mathcal{C} \cap \mathcal{S}_T$, and for each $t \in [T]$, let $A_t \subseteq \Sigma$ denote the set of experts that are awake at time t. Then for any distribution $\mathsf{u} \in \Delta_K$,* AWAKEAWM *admits the following unweighted regret guarantee:*

$$\begin{aligned}&\sum_{t=1}^{T} \sum_{\mathbf{x} \in \mathcal{C}_T \cap A_t} \mathsf{u}[\mathbf{x}] \mathop{\mathbb{E}}_{a \sim \mathsf{p}_t^{A_t}} [l_t[a]] - \sum_{t=1}^{T} \sum_{\mathbf{x} \in \mathcal{C}_T \cap A_t} \mathsf{u}[\mathbf{x}] l_t[\mathbf{x}[t]] \\ &\leq \frac{\eta}{8} \sum_{t=1}^{T} \mathsf{u}(A_t) + \frac{1}{\eta} \log(K).\end{aligned}$$

Proof. As in the proof of Theorem 1, for every $t \in [T]$ and $\mathbf{x} \in \Sigma^T$, let $w_t[\mathbf{x}]$ denote the sequence weight defining q_t via normalization, $\mathsf{q}_t[\mathbf{x}] = \frac{w_t[\mathbf{x}]}{\sum_{\mathbf{x}} w_t[\mathbf{x}]}$. Moreover, let $\mathsf{q}_t^{A_t}$ be the distribution induced over sequences in with labels that awake at time t, so that for every sequence $\mathbf{x} \in \mathcal{C}_T$ with $\mathbf{x}[t] \in A_t$, $\mathsf{q}_t^{A_t}[\mathbf{x}] = \frac{\mathsf{q}_t[\mathbf{x}]}{\sum_{\mathbf{x} \in \mathcal{C}_T \colon \mathbf{x}[t] \in A_t} \mathsf{q}_t[\mathbf{x}]}$, and for every sequence $\mathbf{x} \in \mathcal{C}_T$ with $\mathbf{x}[t] \notin A_t$, $\mathsf{q}_t^{A_t}[\mathbf{x}] = 0$.

Notice that by design, if a sequence $\mathbf{x} \in \mathcal{C}_T$ has a label that isn't awake at time t, $x[t] \notin A_t$, then $\mathsf{q}_{t+1}[\mathbf{x}] = \mathsf{q}_t[\mathbf{x}]$, since we do not update that edge.

Moreover, by the normalization scheme, $\sum_{\mathbf{x} \in \mathcal{C}_T \colon \mathbf{x}[t] \notin A_t} \mathsf{q}_{t+1}[\mathbf{x}] = \sum_{\mathbf{x} \in \mathcal{C}_T \colon \mathbf{x}[t] \notin A_t} \mathsf{q}_t[x]$.

Now let $\mathsf{u} \in \Delta_K$. Then we can write

$$\begin{aligned}&D(\mathsf{u} \| \mathsf{q}_t) - D(\mathsf{u} \| \mathsf{q}_{t+1}) \\ &= \sum_{\mathbf{x} \in \mathcal{C}_T} \mathsf{u}[\mathbf{x}] \log \frac{\mathsf{q}_{t+1}[\mathbf{x}]}{\mathsf{q}_t[\mathbf{x}]} \\ &= \sum_{x \in \mathcal{C}_T \colon \mathbf{x}[t] \in A_t} \mathsf{u}[\mathbf{x}] \log \frac{\mathsf{q}_{t+1}[\mathbf{x}]}{\mathsf{q}_t[\mathbf{x}]} \\ &= \sum_{x \in \mathcal{C}_T \cap A_t} \mathsf{u}[\mathbf{x}] \log \frac{\mathsf{q}_{t+1}^{A_t}[\mathbf{x}]}{\mathsf{q}_t^{A_t}[\mathbf{x}]} \\ &= \sum_{x \in \mathcal{C}_T \cap A_t} \mathsf{u}[\mathbf{x}] \log \frac{\mathsf{q}_t^{A_t}[\mathbf{x}] e^{-\eta l_t[\mathbf{x}[t]]}}{\mathsf{q}_t^{A_t}[\mathbf{x}] \sum_{\mathbf{y} \in \mathcal{C}_T \colon \mathbf{y}[t] \in A_t} \mathsf{q}_t^{A_t}[\mathbf{y}] e^{-\eta l_t[\mathbf{y}[t]]}} \\ &= \sum_{\mathbf{x} \in \mathcal{C}_T \cap A_t} \mathsf{u}[\mathbf{x}](-\eta l_t[\mathbf{x}[t]]) - \sum_{\mathbf{x} \in \mathcal{C}_T \colon \mathbf{x}[t] \in A_t} \mathsf{u}[\mathbf{x}] \log \left(\sum_{\mathbf{y} \in \mathcal{C}_T \colon \mathbf{y}[t] \in A_t} \mathsf{q}_t^{A_t}[\mathbf{y}] e^{-\eta l_t[\mathbf{y}[t]]} \right) \\ &\leq -\eta \sum_{\mathbf{x} \in \mathcal{C}_T \cap A_t} \mathsf{u}[\mathbf{x}] l_t[\mathbf{x}[t]] - \sum_{\mathbf{x} \in \mathcal{C}_T \colon \mathbf{x}[t] \in A_t} \mathsf{u}[\mathbf{x}] \left(\mathop{\mathbb{E}}_{\mathbf{y} \sim \mathsf{q}_t^{A_t}} [-\eta l_t[\mathbf{y}[t]]] + \frac{\eta^2}{8} \right) \\ &= -\eta \sum_{\mathbf{x} \in \mathcal{C}_T \cap A_t} \mathsf{u}[\mathbf{x}] l_t[\mathbf{x}[t]] + \eta \sum_{\mathbf{x} \in \mathcal{C}_T \colon \mathbf{x}[t] \in A_t} \mathsf{u}[\mathbf{x}] \mathop{\mathbb{E}}_{a \sim \mathsf{p}_t^{A_t}} [l_t[a]] - \mathsf{u}(A_t) \frac{\eta^2}{8}.\end{aligned}$$

Thus, by rearranging terms and summing over t, it follows that

$$\sum_{t=1}^{T} \sum_{\mathbf{x} \in \mathcal{C}_T \cap A_t} \mathsf{u}[\mathbf{x}] \mathop{\mathbb{E}}_{a \in \mathsf{p}_t^{A_t}} [\eta l_t[a]] - \sum_{t=1}^{T} \sum_{\mathbf{x} \in \mathcal{C}_T \cap A_t} \mathsf{u}[\mathbf{x}] l_t[\mathbf{x}[t]] \leq \sum_{t=1}^{T} \mathsf{u}(A_t) \frac{\eta}{8} + D(\mathsf{u} \| \mathsf{q}_1),$$

Algorithm 7: AWAKEAUTOMATAWEIGHTEDMAJORITY(AwakeAWM).

```
Algorithm: AWAKEAWM($\mathcal{C}$, $\eta$)
$\mathcal{B} \leftarrow \mathcal{C} \cap \mathcal{S}_T$
$\mathcal{A} \leftarrow$ WEIGHT-PUSHING($\mathcal{B}^\eta$)
$\boldsymbol{\beta} \leftarrow$ BWDDIST($\mathcal{A}$)
$\boldsymbol{\alpha} \leftarrow 0$; $\boldsymbol{\alpha}[I_\mathcal{A}] \leftarrow 1$
for each $e \in E_\mathcal{A}^{0\to 1}$ do
    $\mathsf{p}_1[\mathrm{lab}[e]] \leftarrow \mathrm{weight}[e]$.
for $t \leftarrow 1$ to $T$ do
    RECEIVE($A_t$)
    for each $a \in A_t$ do
        $\mathsf{p}_t^A[a] \leftarrow \mathsf{p}_t[a]/\mathsf{p}_t(A_t)$
    $i_t \leftarrow$SAMPLE($\mathsf{p}_t^A$); PLAY($i_t$); RECEIVE($\mathbf{l}_t$)
    $Z \leftarrow 0$; $\mathbf{w} \leftarrow 0$; $Z^A \leftarrow 0$
    for each $e \in E_\mathcal{A}^{t\to t+1}$ do
        if $\mathrm{lab}[e] \in A_t$ then
            $\mathrm{weight}[e] \leftarrow \mathrm{weight}[e]\, e^{-\eta l_t[\mathrm{lab}[e]]}$
        $\mathbf{w}[\mathrm{lab}[e]] \leftarrow \mathbf{w}[\mathrm{lab}[e]] + \boldsymbol{\alpha}[\mathrm{src}[e]]\,\mathrm{weight}[e]\,\boldsymbol{\beta}[\mathrm{dest}[e]]$
        $\boldsymbol{\alpha}[\mathrm{dest}[e]] \leftarrow \boldsymbol{\alpha}[\mathrm{dest}[e]] + \boldsymbol{\alpha}[\mathrm{src}[e]]\,\mathrm{weight}[e]$
        if $\mathrm{lab}[e] \in A_t$ then
            $Z^A \leftarrow Z^A + \mathbf{w}[\mathrm{lab}[e]]$
    $\mathsf{p}_{t+1} \leftarrow \mathbf{w}\frac{\mathsf{p}_t(A_t)}{Z^A}$
```

and since for the unweighted regret, $\mathsf{q}_1 = \frac{1}{K}$, $D(\mathsf{u}\|\mathsf{q}_1) \leq \log(K)$, which completes the proof. □

As with AWM, AWAKEAWM is an efficient algorithm with a total computational cost that is linear in the number of transitions of $\mathcal{A}$ (or equivalently, $\mathcal{C}_T$). Moreover, as in the non-sleeping expert setting, we can further improve the computational complexity by applying φ-conversion to arrive at a or n-gram approximation and then φ-conversion. All other improvements in the sleeping expert setting will similarly mirror those for the non-sleeping expert algorithms.