Efficient Forecasters for Large Classes of Experts

5.1 Introduction

The results presented in Chapters 2, 3, and 4 show that it is possible to construct algorithms for online forecasting that predict an arbitrary sequence of outcomes almost as well as the best of \( N \) experts. Namely, the per-round cumulative loss of the predictor is at most as large as that of the best expert plus a term proportional to \( \sqrt{\ln N/n} \) for any bounded loss function, where \( n \) is the number of rounds in the prediction game. The logarithmic dependence on the number of experts makes it possible to obtain meaningful bounds even if the pool of experts is very large. However, the basic prediction algorithms, such as weighted average forecasters, have a computational complexity proportional to the number of experts, and they are therefore infeasible when the number of experts is very large.

On the other hand, in many applications the set of experts has a certain structure that may be exploited to construct efficient prediction algorithms. Perhaps the best known such example is the problem of tracking the best expert, in which there is a small number of “base” experts and the goal of the forecaster is to predict as well as the best “compound” expert. This expert is defined by a sequence consisting of at most \( m + 1 \) blocks of base experts so that in each block the compound expert predicts according to a fixed base expert. If there are \( N \) base experts and the length of the prediction game is \( n \), then the total number of compound experts is \( \Theta(N(nN/m)^m) \), exponentially large in \( m \). In Section 5.2 we develop a forecasting algorithm able to track the best expert on any sequence of outcomes while requiring only \( O(N) \) computations in each time period.

Prototypical examples of structured classes of experts for which efficient algorithms have been constructed include classes that can be represented by discrete structures such as lists, trees, and paths in graphs. It turns out that many of the forecasters we analyze in this book can be efficiently calculated over large classes of such “combinatorial experts.” Computational efficiency is achieved because combinatorial experts are generated via manipulation of a simple base structure (i.e., a class might include experts associated with all sublists of a given list or all subtrees of a given tree), and for this reason their predictions are tightly related.

Most of these algorithms are based on efficient implementations of the exponentially weighted average forecaster. We describe two important examples in Section 5.3, concerned with experts defined on binary trees, and in Section 5.4, devoted to experts defined by paths in a given graph. A different approach uses follow-the-perturbed-leader predictors (see Section 4.3) that may be used to obtain efficient algorithms for a large class of problems, including the shortest path problem. This application is described in Section 5.4. The
purpose of Section 5.5 is to develop efficient algorithms to track the best expert in the case when the class of “base” experts is already very large and has some structure. This is, in a sense, a combination of the two types of problems described above.

For simplicity, we analyze combinatorial experts in the framework of randomized prediction (see Chapter 4). Most results of this chapter (with the exception of the algorithms using the follow-the-perturbed-leader forecaster) can also be presented in the deterministic framework developed in Chapters 2 and 3. Following the model and convention introduced in Chapter 4, we sometimes use the term action instead of expert. The two are synonymous in the context of this chapter.

5.2 Tracking the Best Expert

In the basic model of randomized prediction the forecaster’s predictions are evaluated against the best performing single action. This criterion is formalized by the usual notion of regret

\[ \sum_{t=1}^{n} \ell(I_t, Y_t) - \min_{i=1, \ldots, N} \sum_{t=1}^{n} \ell(i, Y_t), \]

which expresses the excess cumulative loss of the forecaster when compared with strategies that use the same action all the time. In this section we are interested in comparing the cumulative loss of the forecaster with more flexible strategies that are allowed to switch actions a limited number of times. (Recall that in Section 4.2 we briefly addressed such comparison classes of “dynamic” strategies.) This is clearly a significantly richer class, because there may be many outcome sequences \(Y_1, \ldots, Y_n\) such that

\[ \min_{i=1, \ldots, N} \sum_{t=1}^{n} \ell(i, Y_t) \]

is large, but there is a partition of the sequence in blocks \(Y_1^{l_1-1}, Y_2^{l_2-1}, \ldots, Y_n^{l_m}\) of consecutive outcomes so that on each block \(Y_i^{l_{k+1}-1} = (Y_{l_k}, \ldots, Y_{l_{k+1}-1})\) some action \(i_k\) performs very well.

Forecasting strategies that are able to “track” the sequence \(i_1, i_2, \ldots, i_{m+1}\) of good actions would then perform substantially better than any individual action. This motivates the following generalization of regret. Fix a horizon \(n\). Given any sequence \(i_1, \ldots, i_n\) of actions from \(\{1, \ldots, N\}\), define the tracking regret by

\[ R(i_1, \ldots, i_n) = \sum_{t=1}^{n} \ell(I_t, Y_t) - \sum_{t=1}^{n} \ell(i_t, Y_t), \]

where, as usual, \(I_1, \ldots, I_n\) is the sequence of randomized actions drawn by the forecaster. To simplify the arguments, throughout this chapter we study the “expected” regret

\[ \bar{R}(i_1, \ldots, i_n) = \sum_{t=1}^{n} \ell(p_t, Y_t) - \sum_{t=1}^{n} \ell(i_t, Y_t), \]

where, following the notation introduced in Chapter 4, \(p_t = (p_{1,t}, \ldots, p_{N,t})\) denotes the distribution according to which the random action \(I_t\) is drawn at time \(t\), and
5.2 Tracking the Best Expert

\[ \ell(p_t, Y_t) = \sum_{i=1}^{N} p_{i,t} \ell(I_t, Y_t) \]

is the expected loss of the forecaster at time \( t \). Recall from Section 4.1 that, with probability at least \( 1 - \delta \),

\[ \sum_{t=1}^{n} \ell(I_t, Y_t) \leq \sum_{t=1}^{n} \ell(p_t, Y_t) + \sqrt{\frac{n}{2} \ln \frac{1}{\delta}} \]

and even tighter bounds can be established if \( \sum_{t=1}^{n} \ell(p_t, Y_t) \) is small.

Clearly, it is unreasonable to require that a forecaster perform well against the best sequence of actions for any given sequence of outcomes. Ideally, the tracking regret should scale, with some measure of complexity, penalizing action sequences that are, in a certain sense, harder to track. To this purpose, introduce

\[ \text{size } (i_1, \ldots, i_n) = \sum_{t=2}^{n} \mathbb{I}_{i_{t-1} \neq i_t} \]

counting how many switches \((i_t, i_{t+1})\) with \( i_t \neq i_{t+1} \) occur in the sequence. Note that

\[ \sum_{t=1}^{n} \ell(p_t, Y_t) - \min_{(i_1, \ldots, i_n) : \text{size } (i_1, \ldots, i_n) = 0} \sum_{t=1}^{n} \ell(i_t, Y_t) \]

corresponds to the usual (nontracking) regret.

It is not difficult to modify the randomized forecasting strategies of Chapter 4 in order to achieve a good tracking regret against any sequence of actions with a bounded number of switches. We may simply associate a compound action with each action sequence \( i_1, \ldots, i_n \) so that \( \text{size } (i_1, \ldots, i_n) \leq m \) for some \( m \) and fixed horizon \( n \). We then run our randomized forecaster over the set of compound actions: at any time \( t \) the randomized forecaster draws a compound action \( (I_1, \ldots, I_n) \) and plays action \( I_t \). Denote by \( M \) the number of all compound actions with size bounded by \( m \). If we use the randomized exponentially weighted forecaster over this set of all compound actions, then Corollary 4.2 implies that the tracking regret is bounded by \( \sqrt{n \ln M / 2} \). Hence, it suffices to count the number of compound actions: for each \( k = 0, \ldots, m \) there are \( \binom{n-1}{k} \) ways to pick \( k \) time steps \( t = 1, \ldots, n - 1 \) where a switch \( i_t \neq i_{t+1} \) occurs, and there are \( N(N - 1)^k \) ways to assign a distinct action to each of the \( k + 1 \) resulting blocks. This gives

\[ M = \sum_{k=0}^{m} \binom{n-1}{k} N(N - 1)^k \leq N^{m+1} \exp \left( (n - 1)H \left( \frac{m}{n - 1} \right) \right), \]

where \( H(x) = -x \ln x - (1 - x) \ln(1 - x) \) is the binary entropy function defined for \( x \in [0, 1] \). Substituting this bound in the above expression, we find that the tracking regret of the randomized exponentially weighted forecaster for compound actions satisfies

\[ \bar{R}(i_1, \ldots, i_n) \leq \sqrt{\frac{n}{2} \left( (m + 1) \ln N + (n - 1)H \left( \frac{m}{n - 1} \right) \right)} \]

on any action sequence \( i_1, \ldots, i_n \) such that \( \text{size } (i_1, \ldots, i_n) \leq m \).

The Fixed Share Forecaster

In its straightforward implementation, the exponentially weighted average forecaster requires to explicitly manage an exponential number of compound actions. We now show how to efficiently implement a generalized version of this forecasting strategy that achieves
the same performance bound for the tracking regret. This efficient forecaster is derived from a variant of the exponentially weighted forecaster where the initial weight distribution is not uniform. The basis of this argument is the next simple general result. Consider the randomized exponentially weighted average forecaster defined in Section 4.2, with the only difference that the initial weights \( w_{1,0} \) assigned to the \( N \) actions are not necessarily uniform. Via a straightforward combination of the proof of Theorem 2.2 (see also Exercise 2.5) with the results of Section 4.2, we obtain the following result.

**Lemma 5.1.** For all \( n \geq 1 \), if the randomized exponentially weighted forecaster is run using initial weights \( w_{1,0}, \ldots, w_{N,0} \geq 0 \) such that \( W_0 = w_{1,0} + \cdots + w_{N,0} \leq 1 \), then
\[
\sum_{t=1}^{n} \bar{I}(p_t, Y_t) \leq \frac{1}{\eta} \ln \frac{1}{W_n} + \frac{\eta}{8} n,
\]
where \( W_n = \sum_{i=1}^{N} w_{i,n} = \sum_{i=1}^{N} w_{i,0} e^{-\eta \sum_{i=0}^{n-1} \ell(i, Y_t)} \) is the sum of the weights after \( n \) rounds.

Nonuniform initial weights may be interpreted to assign prior importance to the different actions. The weighted average forecaster gives more importance to actions with larger initial weight and is guaranteed to achieve a smaller regret with respect to these experts.

For the tracking application, we choose the initial weights of compound actions \( (i_1, \ldots, i_n) \) so that their values correspond to a probability distribution parameterized by a real number \( \alpha \in (0, 1) \). We show that our efficient forecaster achieves the same tracking regret bound as the (nonefficient) exponentially weighted forecaster run with uniform weights over all compound actions whose number of switches is bounded by a function of \( \alpha \).

We start by defining the initial weight assignment. Throughout the whole section we assume that the horizon \( n \) is fixed and known in advance. We write \( w'_0(i_1, \ldots, i_n) \) to denote the weight assigned at time \( t \) by the exponentially weighted forecaster to the compound action \( (i_1, \ldots, i_n) \). For any fixed choice of the parameter \( \alpha \in (0, 1) \), the initial weights of the compound actions are defined by
\[
w'_0(i_1, \ldots, i_n) = \frac{1}{N} \left( \frac{\alpha}{N} \right)^{\text{size}(i_1, \ldots, i_n)} \left( 1 - \alpha + \frac{\alpha}{N} \right)^{n - \text{size}(i_1, \ldots, i_n)}.
\]
Introducing the “marginalized” weights
\[
w'_0(i_1, \ldots, i_t) = \sum_{i_{t+1}, \ldots, i_n} w'_0(i_1, \ldots, i_t, i_{t+1}, \ldots, i_n)
\]
for all \( t = 1, \ldots, n \), it is easy to see that the initial weights are recursively computed as follows:
\[
w'_0(i_t) = 1/N
\]
\[
w'_0(i_1, \ldots, i_{t+1}) = w'_0(i_1, \ldots, i_t) \left( \frac{\alpha}{N} + (1 - \alpha) \mathbb{1}_{i_{t+1} = i_t} \right).
\]
Note that, for any \( n \), this assignment corresponds to a probability distribution over \( \{1, \ldots, N\}^n \), the set of all compound actions of length \( n \). In particular, the ratio \( w'_0(i_1, \ldots, i_{t+1})/w'_0(i_1, \ldots, i_t) \) can be viewed as the conditional probability that a random compound action \( (I_1, \ldots, I_n) \), drawn according to the distribution \( w'_0 \), has \( I_{t+1} = i_{t+1} \) given that \( I_t = i_t \). Hence, \( w'_0 \) is the joint distribution of a Markov process over the set
5.2 Tracking the Best Expert

\[ \{1, \ldots, N\} \text{ such that } I_1 \text{ is drawn uniformly at random, and each next action } I_{t+1} \text{ is equal to the previous action } I_t \text{ with probability } 1 - \alpha + \alpha/N, \text{ and is equal to a different action } j \neq I_t \text{ with probability } \alpha/N. \text{ Thus, choosing } \alpha \text{ small amounts to assigning a small initial weight to compound actions with a large number of switches.} \]

At any time \( t \), a generic weight of the exponentially weighted forecaster has the form

\[ w'_t(i_1, \ldots, i_n) = w'_0(i_1, \ldots, i_n) \exp\left(-\eta \sum_{s=1}^{t} \ell(i_s, Y_s)\right) \]

and the forecaster draws action \( i \) at time \( t + 1 \) with probability \( w'_t(i)/W'_t \), where \( W'_t = w'_{1,t} + \cdots + w'_{N,t} \) and

\[ w'_{i,t} = \sum_{i_1, \ldots, i_{t+2}, \ldots, i_n} w'_t(i_1, \ldots, i_t, i, i_{t+2}, \ldots, i_n) \quad \text{for } t \geq 1 \text{ and } w'_i,0 = \frac{1}{N}. \]

We now define a general forecasting strategy for running efficiently the exponentially weighted forecaster with this choice of initial weights.

THE FIXED SHARE FORECASTER

**Parameters:** Real numbers \( \eta > 0 \) and \( 0 \leq \alpha \leq 1 \).

**Initialization:** \( w_0 = (1/N, \ldots, 1/N) \).

For each round \( t = 1, 2, \ldots \)

1. draw an action \( I_t \) from \( \{1, \ldots, N\} \) according to the distribution

\[ p_{i,t} = \frac{w_{i,t-1}}{\sum_{j=1}^{N} w_{j,t-1}}, \quad i = 1, \ldots, N. \]

2. obtain \( Y_t \) and compute

\[ v_{i,t} = w_{i,t-1} e^{-\eta \ell(i, Y_t)} \quad \text{for each } i = 1, \ldots, N. \]

3. let

\[ w_{i,t} = \alpha \frac{W_t}{N} + (1 - \alpha)v_{i,t} \quad \text{for each } i = 1, \ldots, N, \]

where \( W_t = v_{1,t} + \cdots + v_{N,t} \).

Note that with \( \alpha = 0 \) the fixed share forecaster reduces to the simple exponentially weighted average forecaster over \( N \) base actions. A positive value of \( \alpha \) forces the weights to stay above a minimal level, which allows to track the best compound action. We will see shortly that if the goal is to track the best compound action with at most \( m \) switches, then the right choice of \( \alpha \) is about \( m/n \).

The following result shows that the fixed share forecaster is indeed an efficient version of the exponentially weighted forecaster.

**Theorem 5.1.** For all \( \alpha \in [0, 1] \), for any sequence of \( n \) outcomes, and for all \( t = 1, \ldots, n \), the conditional (given the past) distribution of the action \( I_t \), drawn at time \( t \) by the fixed share forecaster with input parameter \( \alpha \), is the same as the conditional distribution of
action $I'_t$ drawn at time $t$ by the exponentially weighted forecaster run over the compound actions $(i_1, \ldots, i_n)$ using initial weights $w'_0(i_1, \ldots, i_n)$ set with the same value of $\alpha$.

**Proof.** It is enough to show that, for all $i$ and $t$, $w_{i,t} = w'_{i,t}$. We proceed by induction on $t$. For $t = 0$, $w_{i,0} = w'_{i,0} = 1/N$ for all $i$. For the induction step, assume that $w_{i,s} = w'_{i,s}$ for all $i$ and $s < t$. We have

$$w'_{i,t} = \sum_{i_1, \ldots, i_{t-1}} w'_0(i_1, \ldots, i_{t-1}, i, i_{t+1}, \ldots, i_n)$$

$$= \sum_{i_1, \ldots, i_{t-1}} \exp \left( -\eta \sum_{s=1}^{t} \ell(i_s, Y_s) \right) w'_0(i_1, \ldots, i_{t-1}, i)$$

$$= \sum_{i_1, \ldots, i_{t-1}} \exp \left( -\eta \sum_{s=1}^{t} \ell(i_s, Y_s) \right) w'_0(i_1, \ldots, i_{t-1}) \frac{w'_0(i_1, \ldots, i_{t-1}, i)}{w'_0(i_1, \ldots, i_{t-1})}$$

$$= \sum_{i_1, \ldots, i_{t-1}} \exp \left( -\eta \sum_{s=1}^{t} \ell(i_s, Y_s) \right) w'_0(i_1, \ldots, i_{t-1}) \left( \frac{\alpha}{N} + (1 - \alpha)I_{i_{t-1} = i} \right)$$

(using the recursive definition of $w'_0$)

$$= \sum_{i_t} e^{-\eta \ell(i_t, Y_t)} w'_{i, t-1} \left( \frac{\alpha}{N} + (1 - \alpha)I_{i_{t-1} = i} \right)$$

$$= \sum_{i_t} e^{-\eta \ell(i_t, Y_t)} w'_{i, t-1} \left( \frac{\alpha}{N} + (1 - \alpha)I_{i_{t-1} = i} \right)$$

(by the induction hypothesis)

$$= \sum_{i_t} v_{i, t} \left( \frac{\alpha}{N} + (1 - \alpha)I_{i_{t-1} = i} \right)$$

(using step 2 of fixed share)

$$= w_{i,t}$$

(using step 3 of fixed share). □

Note that $n$ does not appear in the proof, and the choice of $n$ is thus immaterial in the statement of the theorem. Hence, unless $\alpha$ and $\eta$ are chosen in terms of $n$, the prediction at time $t$ of the exponentially weighted forecaster can be computed without knowing the length of the compound actions.

We are now ready to state the tracking regret bound for the fixed share forecaster.

**Theorem 5.2.** For all $n \geq 1$, the tracking regret of the fixed share forecaster satisfies

$$\overline{R}(i_1, \ldots, i_n) \leq \frac{m + 1}{\eta} \ln N + \frac{1}{\eta} \ln \frac{1}{(\alpha/N)^m(1 - \alpha)^{n-1}} + \frac{\eta}{8} n$$

for all action sequences $i_1, \ldots, i_n$, where $m = \text{size } (i_1, \ldots, i_n)$.

We emphasize that the bound of Theorem 5.2 is true for all sequences $i_1, \ldots, i_n$, and the bound on the regret depends on size $(i_1, \ldots, i_n)$, the complexity of the sequence. If the objective is to minimize the tracking regret for all sequences with size bounded by $m$, then the parameters $\alpha$ and $\eta$ can be tuned to minimize the right-hand side. This is shown
in the next result. We see that by tuning the parameters $\alpha$ and $\eta$ we obtain a tracking regret bound exactly equal to the one proven for the exponentially weighted forecaster run with uniform weights over all compound actions of complexity bounded by $m$. The good choice of $\alpha$ turns out to be $m/(n-1)$. Observe that with this choice the compound actions with $m$ switches have the largest initial weight. In fact, it is easy to see that the initial weight distribution is concentrated on the set of compound actions with about $m$ switches. This intuitively explains why the next performance bound matches the one obtained for the exponentially weighted average algorithm run over the full set of compound actions.

**Corollary 5.1.** For all $n, m$ such that $0 \leq m < n$, if the fixed share forecaster is run with parameters $\alpha = m/(n-1)$, where for $m = 0$ we let $\alpha = 0$, and

$$\eta = \sqrt{\frac{8}{n} \left( (m+1) \ln N + (n-1)H\left(\frac{m}{n-1}\right) \right)},$$

then

$$\overline{R}(i_1, \ldots, i_n) \leq \sqrt{\frac{n}{2} \left( (m+1) \ln N + (n-1)H\left(\frac{m}{n-1}\right) \right)}$$

for all action sequences $i_1, \ldots, i_n$ such that size $(i_1, \ldots, i_n) \leq m$.

**Proof.** First of all, note that for $\alpha = m/(n-1)$

$$\ln \frac{1}{\alpha^m(1-\alpha)^{n-m-1}} \leq -m \ln \frac{m}{n-1} - (n-m-1) \ln \frac{n-m-1}{n-1} = (n-1)H\left(\frac{m}{n-1}\right).$$

Using our choice for $\eta$ in the bound of Theorem 5.2 concludes the proof. 

In the special case $m = 0$, when the tracking regret reduces to the usual regret, the bound of Corollary 5.1 is $\sqrt{(n/2)\ln N}$, which is the bound for the exponentially weighted forecaster proven in Theorem 2.2.

**Proof of Theorem 5.2.** Recall that for an arbitrary compound action $i_1, \ldots, i_n$ we have

$$\ln w'_n(i_1, \ldots, i_n) = \ln w'_0(i_1, \ldots, i_n) - \eta \sum_{t=1}^{n} \ell(i_t, Y_t).$$

By definition of $w'_0$, if $m = \text{size } (i_1, \ldots, i_n)$,

$$w'_0(i_1, \ldots, i_n) = \frac{1}{N} \left( \frac{\alpha}{N} \right)^m \left( \frac{\alpha}{N} + (1-\alpha) \right)^{n-m-1} \geq \frac{1}{N} \left( \frac{\alpha}{N} \right)^m (1-\alpha)^{n-m-1}. $$
Therefore, using this in the bound of Lemma 5.1 we get, for any sequence \((i_1, \ldots, i_n)\) with size \((i_1, \ldots, i_n) = m,
\[
\sum_{t=1}^{n} \ell(p_t, Y_t)
\leq \frac{1}{\eta} \ln \frac{1}{W_n'} + \frac{\eta}{8} n
\leq \frac{1}{\eta} \ln \frac{1}{w_n'(i_1, \ldots, i_n)} + \frac{\eta}{8} n
\leq \sum_{t=1}^{n} \ell(i_t, Y_t) + \frac{1}{\eta} \ln N + \frac{m}{\eta} \ln N - \frac{n - m - 1}{\eta} \ln(1 - \alpha) + \frac{\eta}{8} n,
\]
which concludes the proof. ■

**Hannan Consistency**

A natural question is under what conditions Hannan consistency for the tracking regret can be achieved. In particular, if we assume that at time \(n\) the tracking regret is measured against the best compound action with at most \(\mu(n)\) switches, we may ask what is the fastest rate of growth of \(\mu(n)\) a Hannan consistent forecaster can tolerate. The following result shows that a growth slightly slower than linear is already sufficient.

**Corollary 5.2.** Let \(\mu : \mathbb{N} \to \mathbb{N}\) be any nondecreasing integer-valued function such that
\[
\mu(n) = o\left(\frac{n}{\log(n) \log \log(n)}\right).
\]
Then there exists a randomized forecaster such that
\[
\limsup_{n \to \infty} \frac{1}{n} \left(\sum_{t=1}^{n} \ell(I_t, Y_t) - \min_{\mathcal{F}_n} \sum_{t=1}^{n} \ell(i_t, Y_t)\right) = 0 \quad \text{with probability 1},
\]
where \(\mathcal{F}_n\) is the set of compound actions \((i_1, \ldots, i_n)\) whose size is at most \(\mu(n)\).

The proof of Corollary 5.2 goes along the same lines as the proof of Corollary 6.1, and we leave it as an exercise.

**The Variable Share Forecaster**

Recall that if the best action has a small cumulative loss, then improved regret bounds may be achieved that involve the loss of the best action (see Section 2.4). Next we show how this can be done in the tracking framework such that the resulting forecaster is still computationally feasible. This is achieved by a modification of the fixed share forecaster. All we have to do is to change the initial weight assignment appropriately.

A crucial feature of the choice of initial weights \(w_0'\) when forecasting compound actions is that for any \(i\) and \(t\), the weight \(w_{i,t}'\) depends only on \(w_0'(i_1, \ldots, i_t, i)\) and on the realized losses up to time \(t\). That is, for computing the prediction at time \(t + 1\) there is no need to know how \(w_0'(i_1, \ldots, i_t, i)\) is split into \(w_0'(i_1, \ldots, i_t, i, i_{t+2}, \ldots, i_n)\) for each continuation \(i_{t+2}, \ldots, i_n \in \{1, \ldots, N\}\). This fact, which is the key to the proof of Theorem 5.1, can be exploited to define \(w_0'(i_1, \ldots, i_t, i_{t+1})\) using information that is only made available at
time $t$. For example, consider the following recursive definition

$$w'_0(i_1, \ldots, i_{t+1}) = w'_0(i_1, \ldots, i_t) \left( \frac{1 - (1 - \alpha) \ell(i_t, Y_t)}{N - 1} \mathbb{I}_{[i_t \neq i_{t+1}]} + (1 - \alpha) \ell(i_t, Y_t) \mathbb{I}_{[i_t = i_{t+1}]} \right).$$

This is similar to the definition of the Markov process associated with the initial weight distribution used by the fixed share forecaster. The difference is that here we see that, given $I_t = i_t$, the probability of $I_{t+1} \neq i_t$ grows with $\ell(i_t, Y_t)$, the loss incurred by action $i_t$ at time $t$. Hence, this new distribution assigns a further penalty to compound actions that switch to a new action when the old one incurs a small loss.

On the basis of this new distribution, we introduce the variable share forecaster, replacing step 3 of the fixed share forecaster with

$$w_{i,t} = \frac{1}{N - 1} \sum_{j \neq i} (1 - (1 - \alpha) \ell(j, Y_t)) v_{j,t} + (1 - \alpha) \ell(i, Y_t) v_{i,t}.$$  

Mimicking the proof of Theorem 5.1, it is easy to check that the weights $w'_{i,t}$ of the variable share forecaster satisfy

$$w_{i,t} = \sum_{i_1, \ldots, i_t, i_{t+2}, \ldots, i_n} w'_{i}(i_1, \ldots, i_t, i_{t+2}, \ldots, i_n).$$

Hence, the computation carried out by the variable share forecaster efficiently updates the weights $w'_{i,t}$.

Note that this initial weight distribution assigns a negligible weight $w'_0$ to all compound actions $(i_1, \ldots, i_n)$ such that $i_{t+1} \neq i_t$ and $\ell(i_t, Y_t)$ is close to 0 for some $t$. In Theorem 5.3 we analyze the performance of the variable share forecaster under the simplifying assumption of binary losses $\ell \in \{0, 1\}$. Via a similar, but somewhat more complicated, argument, a regret bound slightly larger than the bound of Theorem 5.3 can be proven in the general setup where $\ell \in [0, 1]$ (see Exercise 5.4).

**Theorem 5.3.** Fix a time horizon $n$. Under the assumption $\ell \in \{0, 1\}$, for all $\eta > 0$ and for all $\alpha \leq (N - 1)/N$ the tracking regret of the variable share forecaster satisfies

$$\sum_{t=1}^{n} \ell(p_t, Y_t) - \sum_{t=1}^{n} \ell(i_t, Y_t) \leq m + \frac{m + 1}{\eta} \ln N + \frac{m}{\eta} \ln \frac{1}{1 - \alpha} + \frac{1}{\eta} \left( \sum_{t=1}^{n} \ell(i_t, Y_t) \right) \ln \frac{1}{1 - \alpha} + \frac{\eta}{8} n$$

for all action sequences $i_1, \ldots, i_n$, where $m = \text{size} (i_1, \ldots, i_n)$.

Observe that the performance bound guaranteed by the theorem is similar to that of Theorem 5.2 with the only exception that

$$\frac{n - m - 1}{\eta} \ln \frac{1}{1 - \alpha}$$

is replaced by

$$\frac{1}{\eta} \left( \sum_{t=1}^{n} \ell(i_t, Y_t) \right) \ln \frac{1}{1 - \alpha} + m.$$  

Because the inequality holds for all sequences with size $(i_1, \ldots, i_n) \leq m$, this is a significant improvement if there exists a sequence $(i_1, \ldots, i_n)$ with at most $m$ switches, with $m$ not
too large, that has a small cumulative loss. Of course, if the goal of the forecaster is to minimize the cumulative regret with respect to the class of all compound actions with size \( (i_1, \ldots, i_n) \leq m \), then the optimal choice of \( \alpha \) and \( \eta \) depends on the smallest such cumulative loss of the compound actions. In the lack of prior knowledge of the minimal loss, these parameters may be chosen adaptively to achieve a regret bound of the desired order. The details are left to the reader.

**Proof of Theorem 5.3.** In its current form, Lemma 5.1 cannot be applied to the initial weights \( w_0 \) of the variable share distribution because these weights depend on the outcome sequence \( Y_1, \ldots, Y_n \) that, in turn, may depend on the actions of the forecaster. (Recall that in the model of randomized prediction we allow nonoblivious opponents.) However, because the draw \( I_t \) of the variable share forecaster is conditionally independent (given the past outcomes \( Y_1, \ldots, Y_{t-1} \) of the past random draws \( I_1, \ldots, I_{t-1} \)), Lemma 4.1 may be used, which states that, without loss of generality, we may assume that the opponent is oblivious. In other words, it suffices to prove our result for any fixed (nonrandom) outcome sequence \( y_1, \ldots, y_n \). Once the outcome sequence is fixed, the initial weights are well defined and we can apply Lemma 5.1.

Introduce the notation \( L(j_1, \ldots, j_n) = \ell(j_1, y_1) + \ldots + \ell(j_n, y_n) \). Fix any compound action \((i_1, \ldots, i_n)\). Let \( m = \text{size} \ (i_1, \ldots, i_n) \) and \( L^* = L(i_1, \ldots, i_n) \). If \( m = 0 \), then

\[
\ln W'_n \geq \ln w'_0(i_1, \ldots, i_n) = \ln \left( \frac{1}{N} e^{-\eta L^*} (1 - \alpha)^{L^*} \right)
\]

and the theorem follows from Lemma 5.1. Assume then \( m \geq 1 \). Denote by \( F_{L^*+m} \) the set of compound actions \((j_1, \ldots, j_n)\) with cumulative loss \( L(j_1, \ldots, j_n) \leq L^* + m \). Then

\[
\ln W'_n = \ln \left( \sum_{(j_1, \ldots, j_n) \in F_{L^*+m}} w'_0(j_1, \ldots, j_n) e^{-\eta (L(j_1, \ldots, j_n))} \right)
\]

\[
\geq \ln \left( \sum_{(j_1, \ldots, j_n) \in F_{L^*+m}} w'_0(j_1, \ldots, j_n) e^{-\eta (L^*+m)} \right)
\]

\[
= -\eta (L^* + m) + \ln \left( \sum_{(j_1, \ldots, j_n) \in F_{L^*+m}} w'_0(j_1, \ldots, j_n) \right).
\]

We now show that \( F_{L^*+m} \) contains at least a compound action \((j_1, \ldots, j_n)\) with a large weight \( w'_0(j_1, \ldots, j_n) \). This compound action \((j_1, \ldots, j_n)\) mimics \((i_1, \ldots, i_n)\) until the latter makes a switch. If the switch is made right after a step \( t \) where \( \ell(i_t, y_t) = 0 \), then \((j_1, \ldots, j_n)\) delays the switch (which would imply \( w'_0(j_1, \ldots, j_n) = 0 \)) and keeps repeating action \( i_t \) until some later time step \( t' \) where \( \ell(i_{t'}, y_{t'}) = 1 \). Then, from time \( t' + 1 \) onward, \((j_1, \ldots, j_n)\) mimics \((i_1, \ldots, i_n)\) again until the next switch occurs.

We formalize the above argument as follows. Let \( t_1 \) be the time step where the first switch \( i_{t_1} \neq i_{t_1+1} \) occurs. Set \( j_t = i_t \) for all \( t = 1, \ldots, t_1 \). Then set \( j_t = i_t \) for all \( t = t_1 + 1, \ldots, t'_1 \), where \( t'_1 \) is the first step after \( t_1 \) such that \( \ell(i_{t'_1}, y_{t'_1}) = 1 \). If no such \( t'_1 \) exists, then let \( t'_1 = n \).

Proceed by setting \( j_t = i_t \) for all \( t = t'_1 + 1, t'_1 + 2, \ldots \), until a new switch \( i_{t_2} \neq i_{t_2+1} \) occurs for some \( t_2 > t'_1 \) (if there are no more switches after \( t'_1 \), then set \( j_t = i_t \) for all \( t = t'_1 + 1, \ldots, n \)). Repeat the procedure described above until the end of the sequence is reached.

Call a sequence of steps \( t'_k + 1, \ldots, t_{k+1} - 1 \) (where \( t'_k \) may be 1 and \( t_{k+1} - 1 = n \)) an A-block, and a sequence of steps \( t_k, \ldots, t'_k \) (where \( t'_k \) may be \( n \)) a B-block. Note that
which concludes the proof. Not make any assumptions on the source of this side information. In the case of tree

diction round, a piece of “side information” is made available to the forecaster. We do

An important example of structured classes of experts is obtained by representing experts

\(\{\}

within each \(B\)-block. Also, \(L(j_1, \ldots, j_n) \leq L^* + m\), because \(j_i = i\) within A-blocks, \(L(j_1, \ldots, j_n) \leq 1\) within each B-block, and the number of B-blocks is at most \(m\).

Introduce the notation

\[
Q_t = \frac{1 - (1 - \alpha)^{\ell(j_t, y_t)}}{N - 1} \mathbb{I}_{[i_t \neq i_{t+1}]} + (1 - \alpha)^{\ell(j_t, y_t)} \mathbb{I}_{[i_t = i_{t+1}]},
\]

By definition of \(w_0^{'}\) we have

\[
w_0'(j_1, \ldots, j_n) = w_0'(j_1) \prod_{t=1}^{n-1} Q_t.
\]

For all \(t\) in an A-block, \(Q_t = (1 - \alpha)^{\ell(j_t, y_t)}\). Now fix any B-block \(t_k, \ldots, t'_k\). Then \(\ell(j_t, y_t) = 0\) for all \(t = t_k, \ldots, t'_k - 1\) and \(\ell(j_{t'_k}, y_{t'_k}) \leq 1\), as \(\ell(j_{t'_k}, y_{t'_k})\) might be 0 when \(t'_k = n\). We thus have

\[
\prod_{t \in \text{B-block}} Q_t \geq (1 - \alpha)^0 \times \cdots \times (1 - \alpha)^0 \times \frac{1 - (1 - \alpha)^1}{N - 1} = \frac{\alpha}{N - 1}.
\]

The factor \((1 - (1 - \alpha)^1)/(N - 1)\) appears under the assumption that \(j_{t'_k} < n\) and \(j_{t'_k+1} \neq j_{t'_k}\). If \(j_{t'_k+1} = i_t\), implying \(j_{t'_k+1} = j_{t'_k}\), then \(Q_{t'_k} = (1 - \alpha)^1\) and the above inequality still holds since \(\alpha/(N - 1) \leq 1 - \alpha\) is implied by the assumption \(\alpha \leq (N - 1)/N\).

Now, as explained earlier,

\[
\sum_t \ell(j_t, y_t) \leq L^* \quad \text{and} \quad \sum_{t'} \ell(j_{t'}, y_{t'}) \leq m,
\]

where the first sum is over all \(t\) in A-blocks and the second sum is over all \(t'\) in B-blocks (recall that there are at most \(m\) B-blocks). Thus, there exists a \((j_1, \ldots, j_n) \in \mathcal{F}_{L^*+m}\) with

\[
w_0'(j_1, \ldots, j_n) = w_0'(j_1) \prod_{t=1}^{n-1} Q_t \geq \frac{1}{N} (1 - \alpha)^{L^*} \left(\frac{\alpha}{N - 1}\right)^m.
\]

Hence,

\[
\ln \sum_{\mathcal{F}_{L^*+m}} w_0'(j_1, \ldots, j_n) \geq \ln \left(\frac{1}{N} (1 - \alpha)^{L^*} \left(\frac{\alpha}{N - 1}\right)^m\right),
\]

which concludes the proof.  

### 5.3 Tree Experts

An important example of structured classes of experts is obtained by representing experts (i.e., actions) by binary trees. We call such structured classes of experts tree experts. A tree expert \(E\) is a finite ordered binary tree in which each node has either 2 child nodes (a left child and a right child) or no child nodes (if it is a leaf). The leaves of \(E\) are labeled with actions chosen from \(\{1, \ldots, N\}\). We use \(\lambda\) to indicate the root of \(E\) and 0 and 1 to indicate the left and right children of \(\lambda\). In general, \((x_1, \ldots, x_d) \in \{0, 1\}^d\) denotes the left (if \(x_d = 0\)) or right (if \(x_d = 1\)) child of the node \((x_1, \ldots, x_{d-1})\).

Like the frameworks studied in Chapters 9, 11, and 12, we assume that, at each prediction round, a piece of “side information” is made available to the forecaster. We do not make any assumptions on the source of this side information. In the case of tree
A binary tree (a) and a tree expert based on it (b). Given any side information vector with prefix \((1, 0, \ldots)\), this tree expert chooses action 4, the label of leaf \((1, 0)\).

experts the side information is represented, at each time step \(t\), by an infinite binary vector \(x_t = (x_{1,t}, x_{2,t}, \ldots)\). In the example described in this section, however, we only make use of a finite number of components in the side information vector.

An expert \(E\) uses the side information to select a leaf in its tree and outputs the action labeling that leaf. Given side information \(x = (x_1, x_2, \ldots)\), let \(v = (v_1, \ldots, v_d)\) be the unique leaf of \(E\) such that \(v_1 = x_1, \ldots, v_d = x_d\). The label of this leaf, denoted by \(i_E(x) \in \{1, \ldots, N\}\), is the action chosen by expert \(E\) upon observation of \(x\) (see Figure 5.1).

**Example 5.1 (Context trees).** A simple application where unbounded side information occurs is the following: consider the set of actions \(\{0, 1\}\), let \(Y = \{0, 1\}\), and define \(\ell\) by \(\ell(i, Y) = \mathbb{I}_{i \neq Y}\). This setup models a randomized binary prediction problem in which the forecaster is scored with the number of prediction mistakes. Define the side information at time \(t\) by \(x_t = (Y_{t-1}, Y_{t-2}, \ldots)\), where \(Y_t\) for \(t \leq 0\) is defined arbitrarily, say, as \(Y_t = 0\). Hence, the leaf of a tree expert \(E\) determining the expert’s prediction \(i_E(x_t)\) is selected according to a suffix of the outcome sequence. However, depending on \(E\), the length of the suffix used to determine \(i_E(x_t)\) may vary. This model of interaction between the outcome sequence and the expert predictions was originated in the area of information theory. Suffix-based tree experts are also known as prediction suffix trees, context trees, or variable-length Markov models. □

We define the (expected) regret of a randomized forecaster against the tree expert \(E\) by

\[
\overline{R}_{E,n} = \sum_{t=1}^{n} \ell(p_t, Y_t) - \sum_{t=1}^{n} \ell(i_E(x_t), Y_t).
\]

In this section we derive bounds for the maximal regret against any tree expert such that the depth of the corresponding binary tree (i.e., the depth of the node of maximum distance from the root) is bounded. As in Section 5.2, this is achieved by a version of the exponentially weighted average forecaster. Furthermore, we will see that the forecaster is easy to compute.

Before moving on to the analysis of regret for tree experts, we state and prove a general result about sums of functions associated with the leaves of a binary tree. We use \(T\) to denote a finite binary tree. Thus, a tree expert \(E\) is a binary tree \(T\) with an action labeling each
Figure 5.2. A fragment of a binary tree. The dashed line shows a subtree $T_v$ rooted at $v$.

leaf. The size $\|T\|$ of a finite binary tree $T$ is the number of nodes in $T$. In the following, we often consider binary subtrees rooted at arbitrary nodes $v$ (see Figure 5.2).

**Lemma 5.2.** Let $g : \{0, 1\}^* \to \mathbb{R}$ be any nonnegative function defined over the set of all binary sequences of finite length, and introduce the function $G : \{0, 1\}^* \to \mathbb{R}$ by

$$G(v) = \sum_{T_v} 2^{-\|T_v\|} \prod_{x \in \text{leaves}(T_v)} g(x),$$

where the sum is over all finite binary subtrees $T_v$ rooted at $v = (v_1, \ldots, v_d)$. Then

$$G(v) = \frac{g(v)}{2} + \frac{1}{2} G(v_1, \ldots, v_d, 0) G(v_1, \ldots, v_d, 1).$$

**Proof.** Fix any node $v = (v_1, \ldots, v_d)$ and let $T_0$ and $T_1$ range over all the finite binary subtrees rooted, respectively, at $(v_1, \ldots, v_d, 0)$ and $(v_1, \ldots, v_d, 1)$. The leaves of any subtree rooted at $v$ can be split in three subsets: those belonging to the left subtree $T_0$ of $v$, those belonging to the right subtree $T_1$ of $v$, and the singleton set $\{v\}$ belonging to the subtree containing only the leaf $v$. Thus we have

$$G(v) = \frac{g(v)}{2} + \sum_{T_0} \sum_{T_1} 2^{-\|T_0\| - \|T_1\|} \left( \prod_{x \in \text{leaves}(T_0)} g(x) \right) \left( \prod_{x' \in \text{leaves}(T_1)} g(x') \right)$$

$$= \frac{g(v)}{2} + \frac{1}{2} \left( \sum_{T_0} 2^{-\|T_0\|} \prod_{x \in \text{leaves}(T_0)} g(x) \right) \left( \sum_{T_1} 2^{-\|T_1\|} \prod_{x' \in \text{leaves}(T_1)} g(x') \right)$$

$$= \frac{g(v)}{2} + \frac{1}{2} G(v_1, \ldots, v_d, 0) G(v_1, \ldots, v_d, 1).$$
Our next goal is to define the randomized exponentially weighted forecaster for the set of tree experts. To deal with finitely many tree experts, we assume a fixed bound \( D \geq 0 \) on the maximum depth of a tree expert (see Exercise 5.8 for a more general analysis). Hence, the first \( D \) bits of the side information \( x = (x_1, x_2, \ldots) \) are sufficient to determine the predictions of all such tree experts. We use \( \text{depth}(v) \) to denote the depth \( d \) of a node \( v = (v_1, \ldots, v_d) \) in a binary tree \( T \), and \( \text{depth}(T) \) to denote the maximum depth of a leaf in \( T \). (We define \( \text{depth}(\lambda) = 0 \).) Because the number of tree experts based on binary trees of depth bounded by \( D \) is \( N^{2^D} \) (see Exercise 5.10), computing a weight separately for each tree expert is computationally infeasible even for moderate values of \( D \). To obtain an easily calculable approximation, we use, just as in Section 5.2, the exponentially weighted average forecaster with nonuniform initial weights. In view of using Lemma 5.1, we just have to assign initial weights to all tree experts so that the sum of these weights is at most 1. This can be done as follows.

For \( D \geq 0 \) fixed, define the \( D \)-size of a binary tree \( T \) of depth at most \( D \) as the number of nodes in \( T \) minus the number of leaves at depth \( D \):

\[
\| T \|_D = \| T \| - \left| \{ v \in \text{leaves}(T) : \text{depth}(v) = D \} \right|.
\]

**Lemma 5.3.** For any \( D \geq 0 \),

\[
\sum_{T : \text{depth}(T) \leq D} 2^{-\| T \|_D} = 1,
\]

where the summation is over all trees rooted at \( \lambda \), of depth at most \( D \).

**Proof.** We proceed by induction on \( D \). For \( D = 0 \) the lemma holds because \( \| T \|_0 = 0 \) for the single-node tree \( \{ \lambda \} \) of depth 0. For the induction step, fix \( D \geq 1 \) and define

\[
g(v) = \begin{cases}
0 & \text{if depth}(v) > D \\
2 & \text{if depth}(v) = D \\
1 & \text{otherwise}.
\end{cases}
\]

Then, for any tree \( T \),

\[
2^{-\| T \|} \prod_{x \in \text{leaves}(T)} g(x) = \begin{cases}
0 & \text{if depth}(T) > D \\
2^{-\| T \|_D} & \text{if depth}(T) \leq D.
\end{cases}
\]

Applying Lemma 5.2 with \( v = \lambda \) we get

\[
\sum_{T : \text{depth}(T) \leq D} 2^{-\| T \|_D} \prod_{x \in \text{leaves}(T)} g(x) = \sum_{T} 2^{-\| T \|} \prod_{x \in \text{leaves}(T)} g(x) = \frac{g(\lambda)}{2} + \frac{1}{2} G(0)G(1).
\]

Since \( \lambda \) has depth 0 and \( D \geq 1 \), \( g(\lambda) = 1 \). Furthermore, letting \( T_0 \) and \( T_1 \) range over the trees rooted at node 0 (the left child of \( \lambda \)) and node 1 (the right child of \( \lambda \)), for each \( b \in \{0, 1\} \) we have

\[
G(b) = \sum_{T_b : \text{depth}(T_b) \leq D-1} 2^{-\| T_b \|} \prod_{x \in \text{leaves}(T_b)} g(x) = \sum_{T : \text{depth}(T) \leq D-1} 2^{-\| T \|_{D-1}} = 1
\]

by induction hypothesis. \qed
Using Lemma 5.3, we can define an initial weight over all tree experts $E$ of depth at most $D$ as follows:

$$w_{E,0} = 2^{-\|E\|_D} N^{-|\text{leaves}(E)|}$$

where, in order to simplify notation, we write $\|E\|_D$ and $\text{leaves}(E)$ for $\|T\|_D$ and $\text{leaves}(T)$ whenever the tree expert $E$ has $T$ as the underlying binary tree. (In general, when we speak about a leaf, node, or depth of a tree expert, we always mean a leaf, node, or depth of the underlying tree $T$.) The factor $N^{-|\text{leaves}(E)|}$ in the definition of $w_{E,0}$ accounts for the fact that there are $N^{|\text{leaves}(E)|}$ tree experts $E$ for each binary tree $T$.

If $u = (u_1, \ldots, u_d)$ is a prefix of $v = (u_1, \ldots, u_d, \ldots, u_D)$, we write $u \subseteq v$. In particular, $u \sqsubset v$ if $u \not\subseteq v$ and $v$ has at least one extra component $v_{d+1}$ with respect to $u$.

Define the weight $w_{E,t-1}$ of a tree expert $E$ at time $t$ by

$$w_{E,t-1} = w_{E,0} \prod_{v \in \text{leaves}(E)} w_{E,v,t-1},$$

where $w_{E,v,t-1}$, the weight of leaf $v$ in $E$, is defined as follows: $w_{E,v,0} = 1$ and

$$w_{E,v,t} = \begin{cases} w_{E,v,t-1} e^{-\eta \ell(i_E(v), Y_t)} & \text{if } v \subseteq x_t \\ w_{E,v,t-1} & \text{otherwise,} \end{cases}$$

where $i_E(v) = i_E(x_t)$ is the action labeling the leaf $v \subseteq x_t$ of $E$. (Note that $v$ is unique.) As no other leaf $v' \neq v$ of $E$ is updated at time $t$, we have $w_{E,v',t} = w_{E,v',t-1}$ for all such $v'$, and therefore

$$w_{E,t} = w_{E,t-1} e^{-\eta \ell(i(x_t), Y_t)}$$

for all $t = 1, 2, \ldots$

as $v \subseteq x_t$ always holds for exactly one leaf of $E$. At time $t$, the randomized exponentially weighted forecaster draws action $k$ with probability

$$P_{k,t} = \frac{\sum_E \mathbb{1}_{i_E(x_t)=k} w_{E,t-1}}{\sum_E w_{E',t-1}},$$

where the sums are over tree experts $E$ with $\text{depth}(E) \leq D$. Using Lemma 5.1, we immediately get the following regret bound.

**Theorem 5.4.** For all $n \geq 1$, the regret of the randomized exponentially weighted forecaster run over the set of tree experts of depth at most $D$ satisfies

$$\overline{R}_{E,n} \leq \frac{\|E\|_D}{\eta} \ln 2 + \frac{|\text{leaves}(E)|}{\eta} \ln N + \frac{\eta}{8} n$$

for all such tree experts $E$ and for all sequences $x_1, \ldots, x_n$ of side information.

Observe that if the depth of the tree is at most $D$, then $\|E\|_D \leq 2^D - 1$ and $|\text{leaves}(E)| \leq 2^D$, leading to the regret bound

$$\max_{E : \text{depth}(E) \leq D} \overline{R}_{E,n} \leq \frac{2^D}{\eta} \ln(2N) + \frac{\eta}{8} n.$$

Thus, choosing $\eta$ to minimize the upper bound yields a forecaster with

$$\max_{E : \text{depth}(E) \leq D} \overline{R}_{E,n} \leq \sqrt{n 2^{D-1} \ln(2N)}.$$
We now show how to implement this forecaster using $N$ weights for each node of the complete binary tree of depth $D$; thus $N(2^D+1) - 1$ weights in total. This is a substantial improvement with respect to using a weight for each tree expert because there are $N^{2^D}$ tree experts with corresponding tree of depth at most $D$ (see Exercise 5.10). The efficient forecaster, which we call the \textit{tree expert forecaster}, is described in the following. Because we only consider tree experts of depth $D$ at most, we may assume without loss of generality that the side information $x_t$ is a string of $D$ bits, that is, $x_t \in \{0, 1\}^D$. This convention simplifies the notation that follows.

\textbf{THE TREE EXPERT FORECASTER}

\textbf{Parameters:} Real number $\eta > 0$, integer $D \geq 0$.

\textbf{Initialization:} $w_{i,x,0} = 1$, $w_{i,0,0} = 1$ for each $i = 1, \ldots, N$ and for each node $v = (v_1, \ldots, v_d)$ with $d \leq D$.

For each round $t = 1, 2, \ldots$

1. draw an action $I_t$ from $\{1, \ldots, N\}$ according to the distribution $p_{i,t} = \frac{w_{i,\lambda,t-1}}{\sum_{j=1}^{N} w_{j,\lambda,t-1}}$, $i = 1, \ldots, N$;

2. obtain $Y_t$ and compute, for each $v$ and for each $i = 1, \ldots, N$,
\[
    w_{i,v,t} = \begin{cases} 
        w_{i,v,t-1} e^{-\eta \ell(i,Y_t)} & \text{if } v \subseteq x_t \\
        w_{i,v,t-1} & \text{otherwise} 
    \end{cases}
\]

3. recursively update each node $v = (v_1, \ldots, v_d)$ with $d = D, D-1, \ldots, 0$
\[
    w_{i,v,t} = \begin{cases} 
        \frac{1}{2N} w_{i,v,t} & \text{if } v = x_t \\
        \frac{1}{2N} \sum_{j=1}^{N} w_{j,v,t} & \text{if depth}(v) = D \\
        \frac{1}{2N} w_{i,v,t} + \frac{1}{2N} (\overline{w}_{i,v_0,t} + \overline{w}_{i,v_1,t}) & \text{if } v \sqsubset x_t \\
        \overline{w}_{i,v,t-1} & \text{if depth}(v) < D \\
        \overline{w}_{i,v,t-1} & \text{and } v \not\sqsubseteq x_t 
    \end{cases}
\]

where $v_0 = (v_1, \ldots, v_d, 0)$ and $v_1 = (v_1, \ldots, v_d, 1)$.

With each weight $w_{i,v,t}$ the tree expert forecaster associates an auxiliary weight $\overline{w}_{i,v,t}$. As shown in the next theorem, the auxiliary weight $\overline{w}_{i,\lambda,t}$ equals
\[
    \sum_{E} \mathbb{1}_{\{I_t(x_t) = i\}} W_{E,t}.
\]

The computation of $w_{i,v,t}$ and $\overline{w}_{i,v,t}$ given $w_{i,v,t-1}$ and $\overline{w}_{i,v,t-1}$ for each $v, i$ can be carried out in $O(D)$ time using a simple dynamic programming scheme. Indeed, only the weights associated with the $D$ nodes $v \sqsubseteq x_t$ change their values from time $t-1$ to time $t$.

\textbf{Theorem 5.5}. Fix a nonnegative integer $D \geq 0$. For any sequence of outcomes and for all $t \geq 1$, the conditional distribution of the action $I_t$ drawn at time $t$ by the tree expert
Proof. We show that for each $E$ where in the last step we split the sum over tree experts $T$ where

$$
\sum_E \mathbb{I}_{[i_E(x_t) = k]} w_{E,t-1} = \overline{w}_{k,t-1},
$$

where the sum is over all tree experts $E$ such that depth($E$) $\leq$ $D$. We start by rewriting the left-hand side of the this expression as

$$
\sum_E \mathbb{I}_{[i_E(x_t) = k]} w_{E,t-1} = \sum_E 2^{-\|E\|_D} N^{-\text{leaves}(E)} \mathbb{I}_{[i_E(x_t) = k]} \prod_{v \in \text{leaves}(E)} w_{E,v,t-1}
$$

$$
= \sum_E 2^{-\|E\|_D} \mathbb{I}_{[i_E(x_t) = k]} \prod_{v \in \text{leaves}(E)} \frac{w_{E,v,t-1}}{N}
$$

$$
= \sum_T 2^{-\|T\|_D} \sum_{(i_1, \ldots, i_d)} \prod_{j=1}^d \frac{w_{i_j,v_{i_j},t-1}}{N} \mathbb{I}_{\{v_j \sqsubseteq x_t \Rightarrow i_j = k\}}
$$

where in the last step we split the sum over tree experts $E$ in a sum over trees $T$ and a sum over all assignments $(i_1, \ldots, i_d) \in \{1, \ldots, N\}^d$ of actions to the leaves $v_1, \ldots, v_d$ of $T$. The indicator function selects those assignments $(i_1, \ldots, i_d)$ such that the unique leaf $v_j$ satisfying $v_j \sqsubseteq x_t$ has the desired label $k$. We now proceed the above derivation by exchanging $\sum_{(i_1, \ldots, i_d)}$ with $\prod_{j=1}^d$. This gives

$$
\sum_E \mathbb{I}_{[i_E(x_t) = k]} w_{E,t-1} = \sum_T 2^{-\|T\|_D} \prod_{v \in \text{leaves}(T)} g_{t-1}(v) \text{ for } g_{t-1}(v) = \sum_{i=1}^N \frac{w_{i,v,t-1}}{N} \mathbb{I}_{\{v \sqsubseteq x_t \Rightarrow i = k\}}.
$$

Note that this last expression is of the form

$$
\sum_T 2^{-\|T\|_D} \prod_{v \in \text{leaves}(T)} g_{t-1}(v) \text{ for } g_{t-1}(v) = \sum_{i=1}^N \frac{w_{i,v,t-1}}{N} \mathbb{I}_{\{v \sqsubseteq x_t \Rightarrow i = k\}}.
$$

Let

$$
G_{t-1}(v) = \sum_{T_v} 2^{-\|T_v\|_D} \prod_{x \in \text{leaves}(T_v)} g_{t-1}(x),
$$

where $T_v$ ranges over all trees rooted at $v$. By Lemma 5.2 (noting that the lemma remains true if $\|T_v\|$ is replaced by $\|T_v\|_D$), and using the definition of $g_{t-1}$, for all $v = (v_1, \ldots, v_d)$,

$$
G_{t-1}(v) = \begin{cases} 
\frac{1}{2N} w_{k,v,t-1} & \text{if } v = x_t, \\
\frac{1}{2N} \sum_{i=1}^N w_{i,v,t-1} & \text{if depth(v) = } D \text{ and } v \neq x_t, \\
\frac{1}{2N} w_{k,v,t-1} + \frac{1}{2} (G_{t-1}(v_0) + G_{t-1}(v_1)) & \text{otherwise}
\end{cases}
$$
where \( v_0 = (v_1, \ldots, v_d, 0) \) and \( v_1 = (v_1, \ldots, v_d, 1) \). We now prove, by induction on \( t \geq 1 \), that \( G_{t-1}(\lambda) = \overline{w}_{k,\lambda,t-1} \) for all \( t \). For \( t = 1 \), \( \overline{w}_{k,\lambda,0} = 1 \). Also,

\[
G_0(\lambda) = \sum_T 2^{-\|T\|_D} \prod_{v \in \text{leaves}(T)} \frac{1}{N} \sum_{i=1}^{N} w_{i,v,0} = 1
\]

because \( w_{i,v,0} = 1 \) for all \( i \) and \( v \), and we used Lemma 5.3 to sum \( 2^{-\|T\|_D} \). Assuming the claim holds for \( t - 1 \), note that the definition of \( G_t(v) \) and \( \overline{w}_{k,v,t} \) is the same for the cases \( v = x_i, v \neq x_i \), and \( v \subseteq x_i \). For the case \( v \not\subseteq x_i \) with depth\((v) < D \), we have \( \overline{w}_{k,v,t} = \overline{w}_{k,v,t-1} \). Furthermore, \( w_{i,v,t} = w_{i,v,t-1} \) for all \( i \). Thus \( G_t(v) = G_{t-1}(v) \) as all further \( v' \) involved in the recursion for \( G_t(v) \) also satisfy \( v' \not\subseteq x_i \). By induction hypothesis, \( G_{t-1}(v) = \overline{w}_{k,v,t-1} \) and the proof is concluded.

### 5.4 The Shortest Path Problem

In this section we discuss a representative example of structured expert classes that has received attention in the literature for its many applications. Our purpose is to describe the main ideas in the simplest form rather than to offer an exhaustive account of online prediction problems for which computationally efficient solutions exist. The shortest path problem is the ideal guinea pig for our purposes.

Consider a network represented by a set of nodes connected by edges, and assume that we have to send a stream of packets from a source node to a destination node. At each time instance a packet is sent along a chosen route connecting source and destination. Depending on traffic, each edge in the network may have a different delay, and the total delay the packet suffers on the chosen route is the sum of delays of the edges composing the route. The delays may vary in each time instance in an arbitrary way, and our goal is to find a way of choosing the route in each time instance such that the sum of the total delays over time is not much more than that of the best fixed route in the network.

Of course, this problem may be cast as a sequential prediction problem in which each possible route is represented by an expert. However, the number of routes is typically exponentially large in the number of edges, and therefore computationally efficient predictors are called for. In this section we describe two solutions of very different flavor. One of them is based on the follow-the-perturbed-leader forecaster discussed in Section 4.3, whereas the other is based on an efficient computation of the exponentially weighted average forecaster. Both solutions have different advantages and may be generalized in different directions. The key for both solutions is the additive structure of the loss, that is; the fact that the delay corresponding to each route may be computed as the sum of the delays of the edges composing the route.

To formalize the problem, consider a (finite) directed acyclic graph with a set of edges \( E = \{ e_1, \ldots, e_{|E|} \} \) and set of vertices \( V \). Thus, each edge \( e \in E \) is an ordered pair of vertices \( (v_1, v_2) \). Let \( u \) and \( v \) be two distinguished vertices in \( V \). A path from \( u \) to \( v \) is a sequence of edges \( e^{(1)}, \ldots, e^{(k)} \) such that \( e^{(1)} = (u, v_1), e^{(j)} = (v_{j-1}, v_j) \) for all \( j = 2, \ldots, k - 1 \), and \( e^{(k)} = (v_{k-1}, v) \). We identify a path with a binary vector \( i \in \{0, 1\}^{|E|} \) such that the \( j \)th component of \( i \) equals 1 if and only if the edge \( e_j \) is in the path. For simplicity, we assume that every edge in \( E \) is on some path from \( u \) to \( v \) and every vertex in \( V \) is an endpoint of an edge (see Figure 5.3 for examples).
5.4 The Shortest Path Problem

Figure 5.3. Two examples of directed acyclic graphs for the shortest path problem.

In each round \( t = 1, \ldots, n \) of the forecasting game, the forecaster chooses a path \( I_t \) among all paths from \( u \) to \( v \). Then a loss \( \ell_{e,t} \in [0, 1] \) is assigned to each edge \( e \in E \). Formally, we may identify the outcome \( Y_t \) with the vector \( \ell_t \in [0, 1]^{|E|} \) of losses whose \( j \)th component is \( \ell_{e_j,t} \). The loss of a path \( i \) at time \( t \) equals the sum of the losses of the edges on the path, that is,

\[
\ell(i, Y_t) = i \cdot \ell_t.
\]

Just as before, the forecaster is allowed to randomize and to choose \( I_t \) according to the distribution \( p_t \) over all paths from \( u \) to \( v \). We study the “expected” regret

\[
\sum_{t=1}^n \ell(p_t, Y_t) - \min_{i} \sum_{t=1}^n \ell(i, Y_t),
\]

where the minimum is taken over all paths \( i \) from \( u \) to \( v \) and \( \ell(p_t, Y_t) = \sum_i p_i \ell(i, Y_t) \).

Note that the loss \( \ell(i, Y_t) \) is not bounded by 1 but rather by the length \( K \) of the longest path from \( u \) to \( v \). Thus, by the Hoeffding–Azuma inequality, with probability at least \( 1 - \delta \), the difference between the actual cumulative loss \( \sum_{t=1}^n \ell(I_t, Y_t) \) and \( \sum_{t=1}^n \ell(p_t, Y_t) \) is bounded by \( K \sqrt{(n/2) \ln(1/\delta)} \).

We describe two different computationally efficient forecasters with similar performance guarantees.

Follow the Perturbed Leader

The first solution is based on selecting, at each time \( t \), the path that minimizes the “perturbed” cumulative loss up to time \( t - 1 \). This is the idea of the forecaster analyzed in Section 4.3 that may be adapted to the shortest path problem easily as follows.

Let \( Z_1, \ldots, Z_n \) be independent, identically distributed random vectors taking values in \( \mathbb{R}^{|E|} \). The follow-the-perturbed-leader forecaster chooses the path

\[
I_t = \arg\min_i \left( \sum_{s=1}^{t-1} \ell_s + Z_t \right).
\]

Thus, at time \( t \), the cumulative loss of each edge \( e_j \) is “perturbed” by the random quantity \( Z_{j,t} \) and \( I_t \) is the path that minimizes the perturbed cumulative loss over all paths. Since efficient algorithms exist for finding the shortest path in a directed acyclic graph (for acyclic graphs linear-time algorithms are known), the forecaster may be computed efficiently. The
following theorem bounds the regret of this simple algorithm when the perturbation vectors $Z_t$ are uniformly distributed. An improved performance bound may be obtained by using two-sided exponential distribution instead (just as in Corollary 4.5). The straightforward details are left as an exercise (see Exercises 5.11 and 5.12).

**Theorem 5.6.** Consider the follow-the-perturbed-leader forecaster for the shortest path problem such that the vectors $Z_t$ are distributed uniformly in $[0, \Delta|E|]$. Then, with probability at least $1 - \delta$,

$$\sum_{t=1}^{n} \ell(I_t, Y_t) - \min_{i} \sum_{t=1}^{n} \ell(i, Y_t) \leq K \Delta + \frac{nK|E|}{\Delta} + K \sqrt{\frac{n}{2} \ln \frac{1}{\delta}},$$

where $K$ is the length of the longest path from $u$ to $v$. With the choice $\Delta = \sqrt{n|E|}$ the upper bound becomes $2K \sqrt{n|E|} + K \sqrt{n/2} \ln(1/\delta)$.

**Proof.** The proof mimics the arguments of Theorem 4.2 and Corollary 4.4: by Lemma 4.1, it suffices to consider an oblivious opponent and to show that in that case the expected regret satisfies

$$E \sum_{t=1}^{n} \ell(I_t, y_t) - \min_{i} \sum_{t=1}^{n} \ell(i, y_t) \leq K \Delta + \frac{nK|E|}{\Delta}.$$

As in the proof of Theorem 4.2, we define the fictitious forecaster

$$\hat{I}_t = \arg\min_{i} i \cdot \left( \sum_{s=1}^{t} \ell_s + Z_t \right),$$

which differs from $I_t$ in that the cumulative loss is now calculated up to time $t$ (rather than $t - 1$). Of course, $\hat{I}_t$ cannot be calculated by the forecaster because he uses information not available at time $t$: it is defined merely for the purpose of the proof. Exactly as in Theorem 4.2, one has, for the expected cumulative loss of the fictitious forecaster,

$$E \sum_{t=1}^{n} \ell(\hat{I}_t, y_t) \leq \min_{i} \sum_{t=1}^{n} \ell(i, y_t) + E \max_{i} (i \cdot Z_n) + E \max_{i} (-i \cdot Z_1).$$

Since the components of $Z_1$ are assumed to be nonnegative, the last term on the right-hand side may be dropped. On the other hand,

$$E \max_{i} (i \cdot Z_n) \leq \max_{i} \|i\|_1 \|Z_n\|_\infty \leq K \Delta,$$

where $K = \max_{i} \|i\|_1$ is the length of the longest path from $u$ to $v$. It remains to compare the cumulative loss of $I_1$ with that of $\hat{I}_1$. Once again, this may be done just as in Theorem 4.2. Define, for any $z \in \mathbb{R}^{|E|}$, the optimal path

$$i^*(z) = \arg\min_{i} i \cdot z$$

and the function

$$F_t(z) = i^* \left( \sum_{s=1}^{t-1} \ell_s + z \right) \cdot \ell_t.$$
5.4 The Shortest Path Problem

Denoting the density function of the random vector $Z_t$ by $f(z)$, we clearly have

$$
\mathbb{E} \ell(I_t, y_t) = \int_{\mathbb{R}^{|E|}} F_i(z) f(z) \, dz \quad \text{and} \quad \mathbb{E} \hat{\ell}(\hat{I}_t, y_t) = \int_{\mathbb{R}^{|E|}} F_i(z - \ell_t) f(z) \, dz.
$$

Therefore, for each $t = 1, \ldots, n$,

$$
\mathbb{E} \ell(I_t, y_t) - \mathbb{E} \hat{\ell}(\hat{I}_t, y_t)
$$

$$
\leq \ell_t \cdot \int_{\{z : f(z) > f(z - \ell_t)\}} \mathbb{I}(\sum_{s=1}^{t-1} \ell_s + z) \left( f(z) - f(z - \ell_t) \right) f(z) \, dz
$$

$$
\leq \|\ell_t\|_{\infty} \int_{\{z : f(z) > f(z - \ell_t)\}} \mathbb{I}(\sum_{s=1}^{t-1} \ell_s + z) f(z) \, dz
$$

$$
\leq K \int_{\{z : f(z) > f(z - \ell_t)\}} f(z) \, dz
$$

(since $\|\ell_t\|_{\infty} \leq 1$ and all paths have length at most $K$)

$$
\leq \frac{K|E|}{\Delta},
$$

where the last inequality follows from the argument in Corollary 4.4. ■

Theorem 5.6 asserts that the follow-the-perturbed-leader forecaster used with uniformly distributed perturbations has a cumulative regret of the order of $K \sqrt{n|E|}$. By using two-sided exponentially distributed perturbation, an alternative bound of the order of $\sqrt{L^*|E|K \ln |E|} \leq K \sqrt{n|E| \ln |E|}$ may be achieved (Exercise 5.11) or even $\sqrt{L^*|E| \ln M} \leq \sqrt{nK|E| \ln M}$, where $M$ is the number of all paths in the graph leading from $u$ to $v$ (Exercise 5.12). Clearly, $M \leq \binom{|E|}{K}$, but in concrete cases (e.g., in the two examples of Figure 5.3) $M$ may be significantly smaller than this upper bound. These bounds can be improved to $K \sqrt{n \ln M}$ by a fundamentally different solution, which we describe next.

**Efficient Computation of the Weighted Average Forecaster**

A conceptually different way of approaching the shortest path problem is to consider each path as an action (expert) and look for an efficient algorithm that computes the exponentially weighted average predictor over the set of these experts.

The exponentially weighted average forecaster, calculated over the set of experts given by all paths from $u$ to $v$, selects, at time $t$, a path $I_t$ randomly, according to the probability distribution

$$
\mathbb{P}_t[I_t = i] = p_{i,t} = \frac{e^{-\eta \sum_{s=1}^{t-1} i \ell_s}}{\sum_i e^{-\eta \sum_{s=1}^{t-1} i \ell_s}},
$$

where $\eta > 0$ is a parameter of the forecaster and $\mathbb{P}_t$ denotes the conditional probability given the past actions.

In the remaining part of this section we show that it is possible to draw the random path $I_t$ in an efficiently computable way. The main idea is that we select the edges of the path one
by one, according to the appropriate conditional distributions generated by the distribution over the set of paths given above.

With an abuse of notation, we write $e \in i$ if the edge $e \in E$ belongs to the path represented by the binary vector $i$. Then observe that for any $t = 1, \ldots, n$ and path $i$,

$$i \cdot \ell_t = \sum_{e \in i} \ell_{e,t},$$

and therefore the cumulative loss of each expert $i$ takes the additive form

$$\sum_{s=1}^{t} i \cdot \ell_s = \sum_{e \in i} L_{e,t},$$

where $L_{e,t} = \sum_{s=1}^{t} \ell_{e,s}$ is the loss accumulated by edge $e$ during the first $t$ rounds of the game.

For any vertex $w \in V$, let $P_w$ denote the set of paths from $w$ to $v$. To each vertex $w$ and $t = 1, \ldots, n$, we assign the value

$$G_t(w) = \sum_{i \in P_w} e^{-\eta \sum_{e \in i} L_{e,t}}.$$

First observe that the function $G_t(w)$ may be computed efficiently. To this end, assume that the vertices $v_1, \ldots, v_{|V|}$ of the graph are labeled such that $u = v_1, v = v_{|V|}$, and if $i < j$, then there is no edge from $z_j$ to $z_i$. (Note that such a labeling can be found in time $O(|E|)$). Then $G_t(v) = 1$, and once $G_t(v_j)$ has been calculated for all $v_i$ with $i = |V|, |V - 1|, \ldots, j + 1$, then $G_t(v_j)$ is obtained, recursively, by

$$G_t(v_j) = \sum_{w : (v_j, w) \in E} G_t(w) e^{-\eta L_{(v_j, w)} |v_j|}. $$

Thus, at time $t$, all values of $G_t(w)$ may be calculated in time $O(|E|)$.

It remains to see how the values $G_{t-1}(w)$ may be used to generate a random path $I_t$ according to the distribution prescribed by the exponentially weighted average forecaster. We may draw the path $I_t$ by drawing its edges successively. Denote the $k$th vertex along a path $i \in P_u$ by $v_{i,k}$ for $k = 0, 1, \ldots, K_i$, where $v_{i,0} = u$, $v_{i,K_i} = v$, and $K_i = ||i||_1$ denotes the length of the path $i$. For each path $i$, we may write

$$p_{I_t} = \mathbb{P}_t[I_t = i] = \prod_{k=1}^{K_i} \mathbb{P}_t[v_{I_t,k} = v_{i,k} \mid v_{I_t,k-1} = v_{i,k-1}, \ldots, v_{I_t,0} = v_{i,0}].$$

To generate a random path $I_t$ with this distribution, it suffices to generate the vertices $v_{i,k}$ successively for $k = 1, 2, \ldots$ such that the product of the conditional probabilities is just $p_{I_t}$.

Next we show that, for any $k$, the probability that the $k$th vertex in the path $I_t$ is $v_{i,k}$, given that the previous vertices in the graph are $v_{i,0}, \ldots, v_{i,k-1}$, is

$$\mathbb{P}_t[v_{I_t,k} = v_{i,k} \mid v_{I_t,k-1} = v_{i,k-1}, \ldots, v_{I_t,0} = v_{i,0}] = \begin{cases} \frac{G_{t-1}(v_{i,k})}{G_{t-1}(v_{i,k-1})} & \text{if } (v_{i,k-1}, v_{i,k}) \in E \\ 0 & \text{otherwise.} \end{cases}$$
To see this, just observe that if \((v_{i,k-1}, v_{i,k}) \in E\),
\[
P_t[v_{i,k} = v_{i,k} \mid v_{i,k-1} = v_{i,k-1}, \ldots, v_{i,0} = v_{i,0}] = \frac{\sum_{j \in P_{v_{i,k}}} e^{-\eta \sum_{s=t}^{t-1} L_{s,t-1}}}{\sum_{j' \in P_{v_{i,k-1}}} e^{-\eta \sum_{s=t}^{t-1} L_{s,t-1}}} = \frac{G_{t-1}(v_{i,k})}{G_{t-1}(v_{i,k-1})}
\]
as desired. Summarizing, we have the following.

**Theorem 5.7.** The algorithm described above computes the exponentially weighted average forecaster over all paths between vertices \(u\) and \(v\) in a directed acyclic graph such that, at each time instance, the algorithm requires \(O(|E|)\) operations. The regret is bounded, with probability at least \(1 - \delta\), by
\[
\sum_{t=1}^{n} \ell(I_t, Y_t) - \min_{i} \sum_{t=1}^{n} \ell(i, Y_t) \leq K \left( \frac{\ln M}{\eta} + \frac{n \eta}{8} + \sqrt{\frac{n}{2} \ln \frac{1}{\delta}} \right),
\]
where \(M\) is the total number of paths from \(u\) to \(v\) in the graph and \(K\) is the length of the longest path.

### 5.5 Tracking the Best of Many Actions

The purpose of this section is to develop efficient algorithms to track the best action in the case when the class of “base” experts is already very large but has some structure. Thus, in a sense, we consider a combination of the problem of tracking the best action described in Section 5.2 with predicting as well as the best in a large class of experts with a certain structure, such as the examples described in Sections 5.3 and 5.4.

Our approach is based on a reformulation of the fixed share tracking algorithm that allows one to apply it, in a computationally efficient way, over some classes of large and structured experts. We will illustrate the method on the problem of “tracking the shortest path.”

The main step to this direction is an alternative expression of the weights of the fixed share forecaster.

**Lemma 5.4.** Consider the fixed share forecaster of Section 5.2. For any \(t = 2, \ldots, n\), the probability \(p_{i,t}\) and the corresponding normalization factor \(W_{t-1}\) can be obtained as
\[
p_{i,t} = \frac{(1-\alpha)^{t-1} e^{-\eta \sum_{s=t}^{t-1} \ell(i,Y_s)}}{NW_{t-1}}
+ \frac{\alpha}{NW_{t-1}} \sum_{t'=2}^{t-1} (1-\alpha)^{t-t'} W_{t'-1} e^{-\eta \sum_{s=t'}^{t-1} \ell(i,Y_s)}
+ \frac{\alpha}{N} W_{t-1} = \frac{\alpha}{N} \sum_{t'=2}^{t-1} (1-\alpha)^{t-1-t'} W_{t'-1} Z_{t',t-1} + \frac{(1-\alpha)^{-2}}{N} Z_{1,t-1},
\]
where \(Z_{t',t-1} = \sum_{i=1}^{N} e^{-\eta \sum_{s=t'}^{t-1} \ell(i,Y_s)}\) is the sum of the (unnormalized) weights assigned to the experts by the exponentially weighted average forecaster method based on the partial past outcome sequence \((Y_{t'}, \ldots, Y_{t-1})\).
**Proof.** The expressions in the lemma follow directly from the recursive definition of the weights $w_{i,t-1}$. First we show that, for $t = 1, \ldots, n$,

$$v_{i,t} = \frac{\alpha}{N} \sum_{t' = 2}^{t} (1 - \alpha)^{t-t'} W_{t'-1} e^{-\eta \sum_{i=1}^{t'} \ell(i,Y_{i})} + \frac{(1 - \alpha)^{t-1}}{N} e^{-\eta \sum_{i=1}^{t} \ell(i,Y_{i})}$$

and

$$w_{i,t} = \frac{\alpha}{N} W_{t} + \frac{\alpha}{N} \sum_{t' = 2}^{t} (1 - \alpha)^{t+1-t'} W_{t'-1} e^{-\eta \sum_{i=1}^{t'+1} \ell(i,Y_{i})} + \frac{(1 - \alpha)^{t}}{N} e^{-\eta \sum_{i=1}^{t} \ell(i,Y_{i})}.$$  

(5.1)

Clearly, for a given $t$, (5.1) implies (5.2) by the definition of the fixed share forecaster. Since $w_{i,0} = 1/N$ for every expert $i$, (5.1) and (5.2) hold for $t = 1$ and $t = 2$ (for $t = 1$ the summations are 0 in both equations). Now assume that they hold for some $t \geq 2$. We show that then (5.1) holds for $t + 1$. By definition,

$$v_{i,t+1} = w_{i,t} e^{-\eta \ell(i,Y_{t+1})}$$

$$= \frac{\alpha}{N} W_{t} e^{-\eta \ell(i,Y_{t+1})} + \frac{\alpha}{N} \sum_{t' = 2}^{t} (1 - \alpha)^{t+1-t'} W_{t'-1} e^{-\eta \sum_{i=1}^{t'+1} \ell(i,Y_{i})} + \frac{(1 - \alpha)^{t}}{N} e^{-\eta \sum_{i=1}^{t} \ell(i,Y_{i})}$$

and therefore (5.1) and (5.2) hold for all $t = 1, \ldots, n$. Now the expression for $p_{i,t}$ follows from (5.2) by normalization for $t = 2, \ldots, n + 1$. Finally, the recursive formula for $W_{t-1}$ can easily be proved from (5.1). Indeed, recalling that $\sum_{i} w_{i,t} = \sum_{i} v_{i,t}$, we have that for any $t = 2, \ldots, n$,

$$W_{t-1} = \sum_{i=1}^{N} w_{i,t-1}$$

$$= \sum_{i=1}^{N} \left( \frac{\alpha}{N} \sum_{t' = 2}^{t-1} (1 - \alpha)^{t-t'} W_{t'-1} e^{-\eta \sum_{i=1}^{t'-1} \ell(i,Y_{i})} + \frac{(1 - \alpha)^{t-2}}{N} e^{-\eta \sum_{i=1}^{t-1} \ell(i,Y_{i})} \right)$$

$$= \frac{\alpha}{N} \sum_{t' = 2}^{t-1} (1 - \alpha)^{t-t'} W_{t'-1} \sum_{i=1}^{N} e^{-\eta \sum_{i=1}^{t'-1} \ell(i,Y_{i})} + \frac{(1 - \alpha)^{t-2}}{N} \sum_{i=1}^{N} e^{-\eta \sum_{i=1}^{t-1} \ell(i,Y_{i})}$$

$$= \frac{\alpha}{N} \sum_{t' = 2}^{t-1} (1 - \alpha)^{t-t'} W_{t'-1} Z_{t',t-1} + \frac{(1 - \alpha)^{t-2}}{N} Z_{1,t-1}. \quad \blacksquare$$

Examining the formula for $p_{i,t} = \mathbb{P}[I_{t} = i]$ given by Lemma 5.4, one may realize that $I_{t}$ may be drawn by the following two-step procedure.
5.5 Tracking the Best of Many Actions

THE ALTERNATIVE FIXED SHARE FORECASTER

**Parameters:** Real numbers $\eta > 0$ and $0 \leq \alpha \leq 1$.

**Initialization:** For $t = 1$, choose $I_1$ uniformly from the set $\{1, \ldots, N\}$.

For each round $t = 2, 3 \ldots$

1. draw $\tau_t$ randomly according to the distribution

$$
P_t[\tau_t = t'] = \begin{cases} 
\frac{(1-\alpha)^{t'\ell}(N W_{t-1})}{NW_t} & \text{for } t' = 1 \\
\frac{\alpha(1-\alpha)^{t'\ell}W_{t-1}Z_{t',t-1}}{NW_t} & \text{for } t' = 2, \ldots, t,
\end{cases}
$$

where we define $Z_{t',t-1} = N$;

2. given $\tau_t = t'$, choose $I_t$ randomly according to the probabilities

$$
P_t[I_t = i | \tau_t = t'] = \begin{cases} 
\frac{\exp(-\sum_{s=1}^{t'} (\ell i, Y_{s,t}))}{Z_{t',t-1}} & \text{for } t' = 1, \ldots, t-1 \\
1/N & \text{for } t' = t.
\end{cases}
$$

Indeed, Lemma 5.4 immediately implies that

$$p_{i,t} = \sum_{t'=1}^{t} P_t[I_t = i | \tau = t'] P_t[\tau = t']$$

and therefore the alternative fixed share forecaster provides an equivalent implementation of the fixed share forecaster.

**Theorem 5.8.** The fixed share forecaster and the alternative fixed share forecaster are equivalent in the sense that the generated forecaster has the same distribution. More precisely, the sequence $(I_1, \ldots, I_n)$ generated by the alternative fixed share forecaster satisfies

$$P_t[I_t = i] = p_{i,t},$$

for all $t$ and $i$, where $p_{i,t}$ is the probability of drawing action $i$ at time $t$ computed by the fixed share forecaster.

Observe that once the value $\tau = t'$ is determined, the conditional probability of $I_t = i$ is equivalent to the weight assigned to expert $i$ by the exponentially weighted average forecaster computed for the last $t - t'$ outcomes of the sequence, that is, for $(Y_{t', \ldots, Y_{t-1}})$. Therefore, for $t \geq 2$, the randomized prediction $I_t$ of the fixed share forecaster can be determined in two steps. First we choose a random time $\tau_t$, which specifies how many of the most recent outcomes we use for the prediction. Then we choose $I_t$ according to the exponentially weighted average forecaster based only on these outcomes.

It is not immediately obvious why the alternative implementation of the fixed share forecaster is more efficient. However, in many cases the probabilities $P_t[I_t = i | \tau_t = t']$ and normalization factors $Z_{t',t-1}$ may be computed efficiently, and in all those cases, since $W_{t-1}$ can be obtained via the recursion formula of Lemma 5.4, the alternative fixed share forecaster becomes feasible. Theorem 5.8 offers a general tool for obtaining such algorithms.
Rather than isolating a single theorem that summarizes conditions under which such an efficient computation is possible, we illustrate the use of this algorithm on the problem of “tracking the shortest path,” that is, when the base experts are defined by paths in a directed acyclic graph between two fixed vertices. Thus, we are interested in an efficient forecaster that is able to choose a path at each time instant such that the cumulative loss is not much larger than the best forecaster that is allowed to switch paths \( m \) times. In other words, the class of (base) experts is the one defined in Section 5.4, and the goal is to track the best such expert. Because the number of paths is typically exponentially large in the number of edges of the graph, a direct implementation of the fixed share forecaster is infeasible. However, the alternative fixed share forecaster may be combined with the efficient implementation of the exponentially weighted average forecaster for the shortest path problem (see Section 5.4) to obtain a computationally efficient way of tracking the shortest path. In particular, we obtain the following result. Its straightforward, though somewhat technical, proof is left as an exercise (see Exercise 5.13).

**Theorem 5.9.** Consider the problem of tracking the shortest path between two fixed vertices in a directed acyclic graph described above. The alternative fixed share algorithm can be implemented such that, at time \( t \), computing the prediction \( I_t \) requires time \( O(tK|E| + t^2) \). The expected tracking regret of the algorithm satisfies

\[
\overline{R}(i_1, \ldots, i_n) \leq K \sqrt{\frac{n}{2} \left( (m+1) \ln M + m \ln \frac{e(n-1)}{m} \right)}
\]

for all sequences of paths \( i_1, \ldots, i_n \) such that size \( (i_1, \ldots, i_n) \) \( \leq m \), where \( M \) is the number of all paths in the graph between vertices \( u \) and \( v \) and \( K \) is the length of the longest path.

Note that we extended, in the obvious way, the definition of size \( (\cdot) \) to sequences \( (i_1, \ldots, i_n) \).

### 5.6 Bibliographic Remarks

The notion of tracking regret, the fixed share forecaster, and the variable share forecaster were introduced by Herbster and Warmuth [159]. The tracking regret bounds stated in Theorem 5.2, Corollary 5.1, and Theorem 5.3 were originally proven in [159]. Vovk [299] has shown that the fixed and variable share forecasters correspond to efficient implementations of the exponentially weighted average forecaster run over the set of compound actions with a specific choice of the initial weights. Our proofs follow Vovk’s analysis; The work [299] also provides an elegant solution to the problem of tuning the parameter \( \alpha \) optimally (see Exercise 5.6).

Minimization of tracking regret is similar to the sequential allocation problems studied within the competitive analysis model (see Borodin and El-Yaniv [36]). Indeed, Blum and Burch [32] use tracking regret and the exponentially weighted average forecaster to solve a certain class of sequential allocation problems.

Bousquet and Warmuth [40] consider the tracking regret measured against all compound actions with at most \( m \) switches and including at most \( k \) distinct actions (out of the \( N \) available actions). They prove that a variant of the fixed share forecaster achieves, with high probability, the bound of Theorem 5.2 in which the factor \( m \ln N \) is replaced by
5.7 Exercises

5.1 (Tracking a subset of actions) Consider the problem of tracking a small unknown subset of $k$ actions chosen from the set of all $N$ actions. That is, we want to bound the tracking regret

$$\sum_{i=1}^{n} \ell(p_i, Y_i) - \sum_{i=1}^{n} \ell(i_i, Y_i),$$

where $p_i$ is the action chosen by the forecaster and $Y_i$ is the true action. We also refer to Auer and Warmuth [15] and Herbster and Warmuth [160] for various extensions and powerful variants of the problem.

Tree-based experts are natural evolutions of tree sources, a probabilistic model intensively studied in information theory. Lemma 5.2 is due to Willems, Shtarkov, and Tjalkens [311], who were the first to show how to efficiently compute averages over all trees of bounded depth. Theorem 5.4 was proven by Helmbold and Schapire [156]. An alternative efficient forecaster for tree experts, also based on dynamic programming, is proposed and analyzed by Takimoto, Maruoka, and Vovk [283]. Freund Schapire, Singer, and Warmuth [115] propose a more general expert framework in which experts may occasionally abstain from predicting (see also Section 4.8). Their algorithm can be efficiently applied to tree experts, even though the resulting bound apparently has a quadratic (rather than linear) dependence on the number of leaves of the tree. Other closely related references include Pereira and Singer [233] and Takimoto and Warmuth [285], which consider planar decision graphs. The dynamic programming implementations of the forecasters for tree experts and for the shortest path problem, both involving computations of sums of products over the nodes of a directed acyclic graph, are special cases of the general sum–product algorithm of Kschischang, Frey, and Loeliger [187].

Kalai and Vempala [174] were the first to promote the use of follow-the-perturbed-leader type forecasters for efficiently computable prediction, described in Section 5.4. Their framework is more general, and the shortest path problem discussed here is just an example of a family of online optimization problems. The required property is that the loss has an additive form. Awerbuch and Kleinberg [19] and McMahan and Blum [211] extend the follow-the-perturbed-leader forecaster to the bandit setting. For a general framework of linear-time algorithms to find the shortest path in a directed acyclic graph, we refer to [219]. Takimoto and Warmuth [286] consider a family of algorithms based on the efficient computation of weighted average predictors for the shortest path problem. György, Linder, and Lugosi [137, 138] apply similar techniques to the ones described for the shortest path problem in online lossy data compression, where the experts correspond to scalar quantizers. The material in Section 5.5 is based on György, Linder, and Lugosi [139].

A further example of efficient forecasters for exponentially many experts is provided by Maass and Warmuth [207]. Their technique is used to learn the class of indicator functions of axis-parallel boxes when all “instances” $x_i$ (i.e., side information elements) belong to the $d$-dimensional grid $\{1, \ldots, N\}^d$. Note that the complement of any such box is represented as the union of $2d$ axis-parallel halfspaces. This union can be learned by the Winnow algorithm (see Section 12.2) by mapping each original instance $x_i$ to a transformed boolean instance whose components are the indicator functions of all $2N d$ axis-parallel halfspaces in the grid. Maass an Warmuth show that Winnow can be run over the transformed instances in time polynomial in $d$ and log $N$, thus achieving an exponential speedup in $N$. In addition, they show that the resulting mistake bound is essentially the best possible, even if computational issues are disregarded.

$$k \ln N + m \ln k$$ (see Exercise 5.1). We also refer to Auer and Warmuth [15] and Herbster and Warmuth [160] for various extensions and powerful variants of the problem.
where size \((i_1, \ldots, i_n) \leq m\) and the compound action \((i_1, \ldots, i_n)\) contains at most \(k\) distinct actions, \(k\) being typically much smaller than \(m\) and \(N\). Prove that by running the fixed share forecaster over a set of at most \(N^k\) meta-actions, and by tuning the parameter \(\alpha\) in a suitable way, a tracking regret bound of order

\[
\sqrt{n(k \ln N + m \ln k + m \ln n)}
\]

is achieved (Bousquet and Warmuth [40].)

5.2 Prove Corollary 5.2.

5.3 (Lower bound for tracking regret) Show that there exists a loss function such that for any forecaster there exists a sequence of outcomes on which the tracking regret

\[
\sup_{(i_1, \ldots, i_n) \text{ : size } (i_1, \ldots, i_n) = m} \overline{R}(i_1, \ldots, i_n)
\]

is lower bounded by a quantity of the order of \(\sqrt{nm \ln N}\).

5.4 Prove that a slightly weaker version of the tracking regret bound established by Theorem 5.3 holds when \(\ell \in [0, 1]\). Hint: Fix an arbitrary compound action \((i_1, \ldots, i_n)\) with size\((i_1, \ldots, i_n) = m\) and cumulative loss \(L^*\). Prove a lower bound on the sum of the weights \(w'_0(i_1, \ldots, j_b)\) of all compound actions \((j_1, \ldots, j_b)\) with cumulative loss bounded by \(L^* + 2m\) (Herbster and Warmuth [159], Vovk [299]).

5.5 (Tracking experts that switch very often) In Section 5.2 we focused on tracking compound actions that can switch no more than \(m\) times, where \(m \ll n\). This exercise shows that it is possible to track the best compound action that switches almost all times. Consider the problem of tracking the best action when \(N = 2\). Construct a computationally efficient forecaster such that, for all action sequences \(i_1, \ldots, i_n\) such that size \((i_1, \ldots, i_n) \geq n - m - 1\), the tracking regret is bounded by

\[
\overline{R}(i_1, \ldots, i_n) \leq \sqrt{\frac{n}{2} \left( (m + 1) \ln 2 + m \ln \frac{e(n - 1)}{m} \right)}.
\]

Hint: Use the fixed share forecaster with appropriately chosen parameters.

5.6 (Exponential forecaster with average weights) In the problem of tracking the best expert, consider the initial assignment of weights where each compound action \((i_1, \ldots, i_n)\), with \(m = \text{size } (i_1, \ldots, i_n)\), gets a weight

\[
w'_{0,\alpha}(i_1, \ldots, i_n) = \frac{1}{N} \left( \frac{\alpha}{N} \right)^m \left( \frac{\alpha}{N} + (1 - \alpha) \right)^{n-m-1} \epsilon^{\alpha - 1}
\]

for each value of \(\alpha \in (0, 1)\), where \(\epsilon > 0\). Using known facts on the Beta distribution (see Section A.1.9 in the Appendix) and the lower bound \(B(a, b) \geq \Gamma(a)(b + (a - 1))^{-a}\) for all \(a, b > 0\), prove a bound on the tracking regret \(\overline{R}(i_1, \ldots, i_n)\) for the exponentially weighted average forecaster that predicts at time \(t\) based on the average weights

\[
w'_{t-1}(i_1, \ldots, i_n) = \int_0^1 w'_{t-1,\alpha}(i_1, \ldots, i_n) \, d\alpha
\]

(Vovk [299]).

5.7 (Fixed share with automatic tuning) Obtain an efficient implementation of the exponential forecaster introduced in Exercise 5.6 by a suitable modification of the fixed share forecaster. Hint: Each action \(i\) gets an initial weight \(w_{i,0}(\alpha) = (\epsilon^{\alpha - 1})/N\). At time \(t\), predict with

\[
p_{i,t} = \frac{w_{i,t-1}}{\sum_{j=1}^N w_{j,t-1}} i = 1, \ldots, N,
\]
where

\[ w_{i,t-1} = \int_0^1 w_{i,t-1}(\alpha) \, d\alpha \quad \text{for } i = 1, \ldots, N. \]

Use the recursive representations of weights given in Lemma 5.4 and properties of the Beta distribution to show that \( w_{i,t} \) can be updated in time \( O(Nt) \) (Vovk [299].)

5.8 (Arbitrary-depth tree experts) Prove an oracle inequality of the form

\[ \sum_{t=1}^n \ell(p_t, Y_t) \leq \min_k \left( \sum_{t=1}^n \ell(i_E(x_t), Y_t) + \frac{c}{\eta} \|E\| + |\text{leaves}(E)| \ln N \right) + \frac{\eta}{8} n, \]

which holds for the randomized exponentially weighted forecaster run over the (infinite) set of all finite tree experts. Here \( c \) is a positive constant. **Hint:** Use the fact that the number of binary trees with \( 2k + 1 \) nodes is

\[ a_k = \frac{1}{k} \left( \frac{2k}{k - 1} \right) \quad \text{for } k = 1, 2, \ldots \]

(closely related to the Catalan numbers) and that the series \( a_1^{-1} + a_2^{-1} + \cdots \) is convergent. Ignore the computability issues involved in storing and updating an infinite number of weights.

5.9 (Continued) Prove an oracle inequality of the same form as the one stated in Exercise 5.8 when the randomized exponentially weighted forecaster is only allowed to perform a finite amount of computation to select an action. **Hint:** Exploit the exponentially decreasing initial weights to show that, at time \( t \), the probability of picking a tree expert \( E \) whose size \( \|E\| \) is larger than some function of \( t \) is so small that it can be safely ignored irrespective of the performance of the expert.

5.10 Show that the number of tree experts corresponding to the set of all ordered binary trees of depth at most \( D \) is \( N^{2^D} \). Use this to derive a regret bound for the ordinary exponentially weighted average forecaster over this class of experts. Compare the bound with Theorem 5.4.

5.11 (Exponential perturbation) Consider the follow-the-perturbed-leader forecaster for the shortest path problem. Assume that the distribution of \( Z_t \) is such that it has independent components, all distributed according to the two-sided exponential distribution with parameter \( \eta > 0 \) so that the joint density of \( Z_t \) is \( f(z) = (\eta/2)^{|E|} e^{-\eta|z|} \). Show that with a proper tuning of \( \eta \), the regret of the forecaster satisfies, with probability at least \( 1 - \delta \),

\[ \sum_{t=1}^n \ell(I_t, y_t) - L^* \leq c \left( \sqrt{L^* K |E| \ln |E|} + K |E| \ln |E| \right) + K \sqrt{\frac{n}{2} \ln \frac{1}{\delta}}, \]

where \( L^* = \min_i \sum_{t=1}^n \ell(i, y_t) \leq Kn \) and \( c \) is a constant. (Note that \( L^* \) may be as large as \( Kn \) in which case this bound is worse than that of Theorem 5.6.) **Hint:** Combine the proofs of Theorem 5.6 and Corollary 4.5.

5.12 (Continued) Improve the bound of the previous exercise to

\[ \sum_{t=1}^n \ell(I_t, y_t) - L^* \leq c \left( \sqrt{L^* |E| \ln M + |E| \ln M} \right) + K \sqrt{\frac{n}{2} \ln \frac{1}{\delta}}, \]

where \( M \) is the number of all paths from \( u \) to \( v \) in the graph. **Hint:** Bound \( \mathbb{E} \max_i |i \cdot Z_n| \) more carefully. First show that for all \( \lambda \in (0, \eta) \) and path \( i \), \( \mathbb{E} e^{\lambda Z_n} \leq \left( 2\eta^2/(\eta - \lambda)^2 \right)^K \), and then use the technique of Lemma A.13.

5.13 Prove Theorem 5.9. **Hint:** The efficient implementation of the algorithm is obtained by combining the alternative fixed share forecaster with the techniques used to establish the computationally efficient implementation of the exponentially weighted average in Section 5.4. The regret bound is a direct application of Corollary 5.1 (György, Linder, and Lugosi [139].)