Performance Analysis of Online Anticipatory Algorithms for Large Multistage Stochastic Integer Programs

Luc Mercier and Pascal Van Hentenryck
Brown University
{mercier,pvh}@cs.brown.edu

Abstract

Despite significant algorithmic advances in recent years, finding optimal policies for large-scale, multistage stochastic combinatorial optimization problems remains far beyond the reach of existing methods. This paper studies a complementary approach, online anticipatory algorithms, that make decisions at each step by solving the anticipatory relaxation for a polynomial number of scenarios. Online anticipatory algorithms have exhibited surprisingly good results on a variety of applications and this paper aims at understanding their success. In particular, the paper derives sufficient conditions under which online anticipatory algorithms achieve good expected utility and studies the various types of errors arising in the algorithms including the anticipativity and sampling errors. The sampling error is shown to be negligible with a logarithmic number of scenarios. The anticipativity error is harder to bound and is shown to be low, both theoretically and experimentally, for the existing applications.

1 Introduction

Online stochastic algorithms for solving large multistage stochastic integer programs have attracted increasing interest in recent years. They are motivated by applications in which different types of requests arrive dynamically, and it is the role of the algorithm to decide which requests to serve and how. Unlike traditional online algorithms, these applications assume that the uncertainty is stochastic and that distributions of the requests are given.

Consider the packet scheduling problem from [Chang *et al.*, 2000]. A router receives a set of packets at each time step and must choose which packet to serve. Packets can be served only for a limited time and they are characterized by a value. The goal is to maximize the values of the served packets. The packet distributions are specified by Markov models whose states specify arrival frequencies for the packet type.

Online reservation systems [Benoist *et al.*, 2001] are another such application. Customers place requests in real time for some service at a fixed date. The resources are modeled by a multiknapsack constraint. (Think of tour operators requesting rooms in hotels for a group: the choice of a specific hotel is not pertinent for the group but all group members must be allocated to the same hotel). Customers must be immediately notified of acceptance or rejection of their requests, and accepted requests must be satisfied. Accepted requests must also be assigned to a specific resource at reservation time and this choice cannot be reconsidered. The goal is to maximize the profit of the served requests which come from different types with different characteristics and arrival frequencies.

Online multiple vehicle routing with time windows [Bent and Van Hentenryck, 2003] captures an important class of applications arising in transportation and distribution systems. In these problems, a fleet of vehicles serve clients which are located in many different locations and place request for service in real-time in specific time windows. Clients must be immediately notified of acceptance or rejection of their requests, and all accepted requests must be satisfied. Routing decisions however can be delayed if necessary. The goal is to maximize the number of satisfied requests.

All these problems share several characteristics. First, they can be modeled as multistage integer stochastic programs. Second, the number of stages is large. In packet scheduling, time is discrete by nature, and experiments were made with 200,000 stages. For the two other applications, time is continuous but a reasonable discretization of time would require 200 stages. Third, the set of feasible decisions at each stage is finite. Finally, these applications require fast decision-making. These characteristics prohibit the use of a priori methods for (Partially Observable) Markov Decision Processes and for Stochastic Programs. Indeed, [Chang *et al.*, 2000] and [Benoist *et al.*, 2001] have shown that (PO)MDPs do not scale on these applications. Moreover, successful algorithms for 2-stage stochastic optimization, such as the Sample Average Approximation method, are shown to require a number of samples exponential in the number of stages [Shapiro, 2006], precluding their use on these applications.

Interestingly, high-quality solutions to these applications have been obtained by online algorithms that relax the non-anticipativity constraints in the stochastic programs. These *online anticipatory algorithms* make decisions online at a time t in three steps:

1. sample the distribution to obtain scenarios of the future;
2. optimize each scenario for each possible decision;
3. select the best decision over all scenarios.

It is clear that this strategy is necessarily suboptimal, even with many scenarios. However, experimental results have been surprisingly good, especially with the *Regret* algorithm [Bent and Van Hentenryck, 2004; Hentenryck *et al.*, 2006] which is an efficient way of implementing step 2. Our goal in this paper is to demystify these results by providing a theoretical analysis of these algorithms. Section 2 describes the model and the algorithm. Section 3 analyses the performance of the online anticipatory algorithm and isolates two fundamental sources of error: a sampling error and a quantity called the *global anticipatory gap* which is inherent to the problem. Section 4 shows how to bound the anticipatory gap theoretically and experimentally. Section 5 analyzes the effect of approximating the optimization problem. Section 6 compares the anticipatory gap to the expected value of perfect information. Section 7 presents directions for future research.

2 Model and Algorithm

We consider finite stochastic integer programs of the form

$$Q = \max_{x_1 \in \mathcal{X}(s_1)} \mathbb{E}\left[\max_{x_2 \in \mathcal{X}(s_2)} \mathbb{E}\left[\dots \max_{x_T \in \mathcal{X}(s_T)} f(x,\xi)\right]\right],$$

where ξ is a stochastic process, with ξ_t being the *observation* at time t, (with ξ_1 being deterministic), $s_t = (x_{1..t-1}, \xi_{1..t})$ is the *state* at time t, $\mathcal{X}$ maps states to non-empty subsets of a finite set X of *decisions* (so the max's are well-defined), and f is the utility function bounded by F_{max}. We denote respectively x and ξ the vectors $x_{1..T}$ and $\xi_{1..T}$.

A *decision process* is a stochastic process x such that $\forall t$: $x_t \in \mathcal{X}(s_t)$. We can assume that the computation of each x_t requires exactly one random variable γ_t. These variables are independent and independent of ξ.

In practice, decisions cannot be made based on future observations. A decision process x is *non-anticipative* if x_t is a deterministic function of $\gamma_{1..t}$ and $\xi_{1..t}$ (that is, if x is adapted to the filtration $\mathcal{F}_t = \sigma(\gamma_{1..t}, \xi_{1..t})$). We can rewrite the stochastic program as

$$Q = \max \{\mathbb{E}[f(x,\xi)] \mid x \text{ non-anticip. dec. proc.}\}.$$

A *scenario* is a realization of the process ξ. The *offline problem* is the problem a decision maker would face if, in a given state s_t, the future observations are revealed; we define

$$\mathcal{O}(s_t, x_t, \xi) = \max\{f(y,\xi) \mid y \text{ dec. proc.}, y_{1..t} = x_{1..t}\},$$
$$\mathcal{O}(s_t,\xi) = \max\{f(y,\xi) \mid y \text{ dec. proc.}, y_{1..t-1} = x_{1..t-1}\} = \max_{x \in \mathcal{X}(s_t)} \mathcal{O}(s_t, x, \xi).$$

Note that these two problems are deterministic.

Finally, the *expected value of the clairvoyant* (EVC) is defined as the expected utility of a clairvoyant decision maker, that is, $EVC = \mathbb{E}[\mathcal{O}(s_1,\xi)]$. The problems discussed in the introduction all fit in this model: in particular, the utility is bounded thanks to capacity constraints. The model can also be generalized to the case in which $f(x,\xi)$ has finite first and second moments for every x.

The *anticipatory algorithm* studied here is Algorithm `MakeDecision`, parametrized by the number of scenarios m, whose successive decisions form a non-anticipative process:

Function `MakeDecision`(s_t, γ_t)

```
Use $\gamma_t$ to compute scenarios $\xi^1 \dots \xi^m$ where $\xi^i_{1..t} = \xi_{1..t}$
foreach $x \in \mathcal{X}(s_t)$ do
    $g(x) \leftarrow \frac{1}{m}\sum_{i=1}^m \mathcal{O}(s_t, x, \xi^i)$
$x_t \leftarrow \text{argmax}_{x \in \mathcal{X}(s_t)} g(x)$
```

3 Analysis of the Anticipatory Algorithm

We compare the performance of the anticipatory algorithm with the offline, a posteriori solution in the expected sense, as is typically done in online algorithms [Borodin and El-Yaniv, 1998]. In other words, for the decision process x produced by the anticipatory algorithm, we bound $EVC - \mathbb{E}[f(x,\xi)]$, which we call the *expected global loss* (EGL).

3.1 Local and Global Losses

We first show that the EGL is the sum of the expected losses of the stages.

Definition 1 *Let s_t be a state. The* expected local loss *of decision $x \in \mathcal{X}(s_t)$ is defined as*

$$\Delta(s_t, x) = \mathbb{E}[\mathcal{O}(s_t,\xi) - \mathcal{O}(s_t,x,\xi) \,|s_t].$$

Note that conditioning on a state s_t does not provide any information on γ_t: when reading an expression of the form $\mathbb{E}[\dots|s_t]$, keep in mind that there is uncertainty on $\xi_{t+1},\dots,\xi_T$ and on $\gamma_t,\dots,\gamma_T$.

Lemma 1 (Global Loss = Sum of Local Losses) *For any decision process x,*

$$EVC - \mathbb{E}[f(x,\xi)] = \sum_{t=1}^{T} \mathbb{E}[\Delta(s_t,x_t)]$$

Proof. Let C_t be the random variable $\mathcal{O}(s_t,x_t,\xi)$ and $A_t = \mathbb{E}[C_t - f(x,\xi)]$. Then $A_T = 0$ and, for $t < T$,

$$\begin{aligned} A_t &= \mathbb{E}[C_t - C_{t+1} + C_{t+1} - f(x,\xi)] \\ &= \mathbb{E}[C_t - C_{t+1}] + A_{t+1} \\ &= \mathbb{E}[\Delta(s_{t+1}, x_{t+1})] + A_{t+1}. \end{aligned}$$

The last equality comes from decomposing and re-assembling amongst all possible values of x_t. Finally $EVC - \mathbb{E}[f(x,\xi)] = \mathbb{E}[C_0 - f(x,\xi)] = A_0$. □

3.2 Decomposition of the Local Loss

We now show that the local loss at a state s_t consists of a sampling error and the anticipatory gap.

Definition 2 *The* anticipatory gap *of a state s_t is defined as*

$$\Delta_g(s_t) = \min_{x \in \mathcal{X}(s_t)} \Delta(s_t,x).$$

The choice error *of x wrt s_t is defined as*

$$\Delta_c(s_t,x) = \Delta(s_t,x) - \Delta_g(s_t).$$

The anticipatory gap is inherent to the problem and independent of the decision process x. An equivalent definition is

$$\max_{x \in \mathcal{X}(s_t)} \mathbb{E}[\mathcal{O}(s_t,x,\xi)|st] - \mathbb{E}\left[\max_{x \in \mathcal{X}(s_t)} \mathcal{O}(s_t,x,\xi)\middle| st\right].$$

This expression shows that this gap can be interpreted as the cost of commuting of $\mathbb{E}$ and $\max$. We now bound $\Delta_c(s_t,x)$.

Lemma 2 (Sampling Error) *Let x_t be computed by the anticipatory algorithm using m samples per decision. Let s_t be a state and $x^\star$ be* $\operatorname{argmax} \mathbb{E}\left[\mathcal{O}(s_t, x, \xi) \,|s_t\right]$ *(break ties arbitrarily). Then*

$$\mathbb{E}\left[\Delta_c(s_t, x_t) \,|s_t\right] \leq \sum_{x \in \mathcal{X}(s_t)} \Delta_c(s_t, x) \exp\left(\frac{-m\Delta_c(s_t, x)^2}{2\sigma(s_t, x)^2}\right),$$

where $\sigma(s_t, x)$ is the standard deviation of $\mathcal{O}(s_t, x, \xi) - \mathcal{O}(s_t, x^\star, \xi)$ given s_t.

Proof. Here all probabilities and expectations are implicitly conditional on s_t. The left-hand side can be decomposed as

$$\mathbb{E}\left[\Delta_c(s_t, x_t)\right] = \sum_{x \in \mathcal{X}(s_t)} \Delta_c(s_t, x)\mathbb{P}(x_t = x).$$

Due to the argmax in `MakeDecision`, the event $x_t = x$ implies $\forall x' \in \mathcal{X}(s_t),\ g(x') \leq g(x)$. Therefore $\mathbb{P}\left(x_t = x\right) \leq \mathbb{P}\left(g(x) \geq g(x^\star)\right)$. Since f is bounded, $\mathcal{O}(s_t, x, \xi) - \mathcal{O}(s_t, x^\star, \xi)$ has a finite expectation and variance. Now,

$$g(x) - g(x^\star) = \frac{1}{m}\sum_{i=1}^{m}\Big(\mathcal{O}(s_t, x, \xi^i) - \mathcal{O}(s_t, x^\star, \xi^i)\Big)$$

and, by the central limit theorem, this difference is normally distributed for m large enough, with mean $-\Delta_c(s_t, x)$ and variance $\frac{1}{m}\sigma(s_t, x)^2$. Finally, if $X \sim \mathcal{N}(\mu, \sigma^2)$ with $\mu < 0$, then $\mathbb{P}\left(X \geq 0\right) \leq \exp\left(-\frac{\mu^2}{2\sigma^2}\right)$ (Chernoff bound). $\square$

3.3 Performance of the Algorithm

We now assemble the previous results.

Definition 3 *The* Global Anticipatory Gap *of the problem is*

$$GAG = \mathbb{E}\left[\max_{\substack{x_{1..T} \\ x_i \in \mathcal{X}(s_i)}} \sum_{t=1}^{T} \mathit{\Delta}_g(s_t)\right].$$

Once again, this quantity is inherent to the problem.

Theorem 1 *The expected global loss of the anticipatory algorithm satisfies*

$$EGL \leq GAG + O\left(e^{-Km}\right)$$

where m is the number of samples per decision and

$$K = \min_{\substack{s_t, x \in \mathcal{X}(s_t) \\ \Delta_c(s_t, x) > 0}} \frac{\Delta_c(s_t, x)^2}{2\sigma(s_t, x)^2}.$$

Proof. We have

$$EGL = \sum_{t=1}^{T} \mathbb{E}\left[\mathit{\Delta}(s_t, x_t)\right] \leq \sum_{t=1}^{T}\Big(\mathbb{E}\left[\mathit{\Delta}_g(s_t)\right] + \mathbb{E}\left[\mathit{\Delta}_c(s_t, x_t)\right]\Big).$$

The term GAG comes from

$$\sum_{t=1}^{T} \mathbb{E}\left[\Delta_g(s_t)\right] = \mathbb{E}\left[\sum_{t=1}^{T} \Delta_g(s_t)\right] \leq \mathbb{E}\left[\max_{x_{1..T}} \sum_{t=1}^{T} \Delta_g(s_t)\right],$$

and the global sampling error satisfies

$$\sum_{t=1}^{T} \mathbb{E}\left[\Delta_c(s_t, x_t)\right] \leq 2T\left|X\right| F_{max} e^{-Km}.$$

$\square$

An important consequence of this theorem is that the sampling error can be made smaller than some constant a by choosing $m \geq {}^1\!/_K \log\left({}^2\!/_a T\left|X\right| F_{max}\right)$. [Shapiro, 2006] argues that the SAA method does not scale to multistage problems, because the number of samples to achieve a given accuracy grows *exponentially* with T. The anticipatory algorithm only requires m to grow *logarithmically* with $T|X|$, which makes it highly scalable. Of course, it only produces high-quality decisions when the anticipatory gap is small.

4 Bounding the Global Anticipatory Gap

This section provides theoretical and experimental results on the anticipatory gap, explaining why anticipatory algorithms are effective in the applications mentioned in the introduction.

4.1 Theoretical Proof on Packet Scheduling

We first show how to compute an upper bound on GAG for a simplified version of the packet scheduling problem. Suppose that there are k types of packets whose values are $v_1 < \ldots < v_k$ respectively. At each step from 1 to $T-1$, a packet of type i arrives with probability p_i. All these random variables are independent. Each packet has a lifetime of 2, meaning a packet received at time t can be scheduled either at time t or at time $t+1$. The utility is the sum of scheduled packets over the T stages. All packets take a single time step to serve. For convenience, we introduce a packet type 0 with value $v_0 = 0$ and probability $p_0 = 1$. It should be clear that this problem satisfies all assumptions above. In particular, the utility is bounded ($0 \leq f \leq Tv_k$).

Why is the GAG small on this problem? We show that Δ_g is rarely high, inducing a small GAG. For s_t a state and $x, y \in \mathcal{X}(s_t)$, we say that x dominates y if $\mathcal{O}(s_t, x, \xi) \geq \mathcal{O}(s_t, y, \xi)$ almost surely given s_t. Studying $\Delta_g(s_t)$ only requires to focus on non-dominated decisions: there are at most two non-dominated decisions for a given state, which consists of scheduling

- the most valuable packet, of type i, received at time $t-1$ and not already scheduled; or
- the most valuable packet, of type j, received at time t.

Moreover, if $i \geq j$, then choosing j is dominated, since i is more valuable and will be lost if not chosen now. Also, if $i < j$ but the second most valuable packet received at t is of type $k \geq i$, then choosing i is dominated. If one of these two conditions holds, a decision dominates all the other ones, and thus $\Delta_g(s_t) = 0$.

Suppose now that s_t does not satisfy any of them. By the dominance property, scenarios can be partitioned into those where scheduling i (resp. j) is the unique offline, optimal decision and those on which there is a tie. Introduce the random variable y_t, taking values i, j or $\perp$ in these three respective cases. We then have

$$\begin{aligned}\Delta(s_t, i) &= \mathbb{E}\left[\mathcal{O}(s_t, \xi) - \mathcal{O}(s_t, i, \xi) \,|s_t\right] \\ &= \mathbb{E}\left[\mathcal{O}(s_t, \xi) - \mathcal{O}(s_t, i, \xi) \,|s_t, y_t = j\right] \mathbb{P}\left(y_t = j\right)\end{aligned}$$

and symmetrically

$$\begin{aligned}\Delta(s_t, j) &= \mathbb{E}\left[\mathcal{O}(s_t, \xi) - \mathcal{O}(s_t, j, \xi) \,|s_t\right] \\ &= \mathbb{E}\left[\mathcal{O}(s_t, \xi) - \mathcal{O}(s_t, j, \xi) \,|s_t, y_t = i\right] \mathbb{P}\left(y_t = i\right).\end{aligned}$$

Now, if i is scheduled and the optimal offline solution was to schedule j, then the loss cannot exceed $v_j - v_i$, since the rest of the optimal offline schedule is still feasible. Hence $\mathbb{E}\left[\mathcal{O}(s_t,\xi) - \mathcal{O}(s_t,i,\xi)\,|s_t, y_t = j\right] \le v_j - v_i$. Moreover, for the optimal offline schedule to choose j at time t, it is necessary to have a packet of value greater than v_i arriving at $t+1$ and thus $\mathbb{P}\left(y_t = j\right) \le 1 - \prod_{k>i} q_k$ where $q_k = 1 - p_k$. Finally, we find

$$\Delta(s_t, i) \le (v_j - v_i)\left(1 - \prod_{k>i} q_k\right).$$

The other case is harder to study, but a trivial upper bound is $\Delta(s_t, j) \le v_i$. Now it remains to bound the expectation of $\Delta_g(s_t) = \min(\Delta(s_t,i), \Delta(s_t,j))$ by enumerating the possible values of i and j and weighting each case with its probability of occurrence. This weight is, in fact, bounded by the product of the probabilities of the 3 following events:

- a packet of type i arrived at time $t-1$;
- the most valuable packet arrived at t is of type j;
- no packet of type $i \le k < j$ arrived at time t.

Here is a numerical example. Suppose there are 4 types of packets, with the following values and emission probabilities:

type	0	1	2	3	4
value	0	1	2	4	8
prob	1.	.60	.30	.20	.10

The upper bound on Δ_g depending on i and j, is

	1				
(j)	2	.496			
	3	1.00	.560		
	4	1.00	1.68	.400	
		1	2	3	4 (i)

We find that $\mathbb{E}\left[\Delta_g(s_t)\right] \le .125$. On the other hand, a simple lower bound on the EVC is the expectation of the value of the most valuable packet arriving at each stage multiplied by the number of stages. In this case, that leads to $EVC \ge 2.25T$. As a result, the ratio of GAG over EVC on this problem is less than 5.55%. Because this analysis is not tight – especially the lower bound of the EVC–, the anticipatory algorithm is likely to have an even better expected global loss. This analysis also hints at why online anticipatory algorithms are so effective on packet scheduling and why they outperform competitive algorithms on these instances.

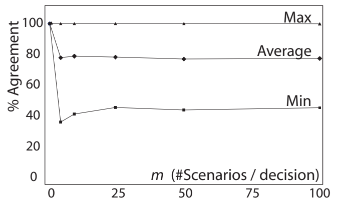
4.2 Discussion on Practical Problems

The previous section shows how to bound the GAG on a particular problem: study dominance properties between the decisions, bound the loss of making a non-optimal (in the offline sense) decision, and bound the probability of this event. We are currently applying this method on more complex problems but proofs quickly become very cumbersome. As an alternative, we discuss another, empirical way to argue that the GAG of a problem is small.

We have emphasized in the theoretical discussion the importance of bounding the probability that the chosen decision is not a posteriori optimal. We call $\mathbb{P}\left(\mathcal{O}(s_t,x_t,\xi) = \mathcal{O}(s_t,\xi)\,|s_t\right)$ the *consensus rate*. This quantity can be estimated easily during the computation of an anticipatory algorithm. It suffices to count how many scenarios make the same decision, i.e.,

$$\frac{1}{m}\left|\left\{i \in \{1,\ldots,m\}\,\middle|\,\mathcal{O}(s_t,x,\xi^i) = \mathcal{O}(s_t,x_t,\xi^i)\right\}\right|.$$

[Hentenryck *et al.*, 2006] kindly gave us some of these statistics on online reservation systems: they depict the consensus rate (min/max/avg) as a function of the number of scenarios.



On this class of instances, there are 6 possible decisions in each state. Therefore, one could expect an average consensus rate of 20%, but it is actually much higher, at about 80%. Moreover the maximal offline loss of a bad decision can easily be bounded in this problem. By Markov inequality, the GAG is low. Similar observations were made in the packet scheduling problem, where the measured average consensus was about 90%, and in the vehicle routing problem, where the rate varies among the stages, exhibiting an increasing trend from 65 to 100%. This argument, however, would be useless when $F_{\max}$ is high, e.g. problems with high penalty states.

More generally, the following theorem gives a way to measure the anticipatory gap of a state.

Theorem 2 *Let s_t be a state. Define $\widehat{\Delta}_g(s_t)$ as*

$$\frac{1}{m}\left(\sum_{i=1}^{m} \max_{x\in\mathcal{X}(s_t)} \mathcal{O}(s_t,x,\xi^i) - \max_{x\in\mathcal{X}(s_t)} \sum_{i=1}^{m} \mathcal{O}(s_t,x,\xi^i)\right),$$

then this is a strongly consistent estimator of $\Delta_g(s_t)$, i.e.,

$$\mathbb{P}\left(\lim_{m\to+\infty} \widehat{\Delta}_g(s_t) \;=\; \Delta_g(s_t) \;\middle|\; s_t\right) = 1.$$

Proof. Apply the strong law of large numbers to $\mathcal{O}(s_t,x,\xi)$ for all x and conclude with the finiteness of $\mathcal{X}(s_t)$. $\square$

$\widehat{\Delta}_g$ can be computed by the online anticipatory algorithm at no additional cost.

5 Approximating the Offline Problem

Theorem 1 explains why the anticipatory algorithm provides good results when the GAG is small. However, practical implementations use a variant of Algorithm 1 in which $\mathcal{O}$ is replaced by a fast approximation $\widetilde{\mathcal{O}}$. This is the case for the three applications mentioned earlier which use an approximating technique called *Regret* [Bent and Van Hentenryck, 2004; Hentenryck *et al.*, 2006]. The *Regret* algorithm can be seen as a discrete version of sensitivity analysis: instead of computing $\mathcal{O}(s_t,x,\xi)$ for each for each $x \in \mathcal{X}(s_t)$, the idea is to compute $x^* = \operatorname{argmax}_x(\mathcal{O}(s_t,x,\xi^i))$ first and then to compute a vector of approximations $\left(\widetilde{\mathcal{O}}(s_t,x,\xi^i)\right)_{x\in\mathcal{X}(s_t)}$

using x^*. Each entry in this vector is derived by approximating the loss of selecting a specific x in $\mathcal{X}(s_t)$ instead of the optimal decision x^*. As a result, the *Regret* algorithm ensures (i) $\widetilde{\mathcal{O}} \leq \mathcal{O}$ and (ii) $\max_x(\widetilde{\mathcal{O}}(s_t, x, \xi^i)) = \max_x(\mathcal{O}(s_t, x, \xi^i))$. See [Bent *et al.*, 2005] for a discussion on the complexity of computing this approximated vector.

It is not easy to provide tight theoretical bounds on the expected global loss for the *Regret* approximation. We thus measured the empirical distribution of $\mathcal{O} - \widetilde{\mathcal{O}}$ on online stochastic reservation systems from [Hentenryck *et al.*, 2006]. The difference $\mathcal{O} - \widetilde{\mathcal{O}}$ is zero in 80% of the cases and its mean is very small (around .2 while the typical values of $\mathcal{O}$ are in the range [400,500]), although it can occasionally be large. This intuitively justifies the quality of *Regret*, whose expected global loss is not significatively different from the anticipatory algorithm for the same sample size.

Finally, recall that, on online reservation systems, the consensus rate is very high on average. Let $x^\star = \text{argmax}_x \sum_{i=1}^{m} \mathcal{O}(s_t, x, \xi^i)$ and let the consensus rate be α. By properties (i) and (ii) of *Regret*, the approximated "score" $\sum_{i=1}^{m} \widetilde{\mathcal{O}}(s_t, x^\star, \xi^i)$ of decision $x^\star$ may only exhibit errors in $(1-\alpha)m$ scenarios and hence will be very close to $\sum_{i=1}^{\alpha m} \mathcal{O}(s_t, x^\star, \xi^i)$. Moreover, other decisions have an approximated score where (almost all) the terms of the sum has a negative error. Therefore the approximated decision is biased toward consensual decisions and a high consensus rate tends to hide the approximation errors $\mathcal{O} - \widetilde{\mathcal{O}}$ of *Regret*.

In summary, a high consensus rate not only makes the AG small but also allows *Regret* to produce decisions close in quality to the exact anticipatory algorithm. This does not mean that a brutal *Regret* approximation, e.g., assigning zero to each non-optimal decision, would be as effective [Bent and Van Hentenryck, 2004].

6 *GAG* Versus *EVPI*

This section studies the relationships between the anticipatory gap and the expected value of perfect information ($EVPI$). Since these concepts are seemingly close, it is useful to explain how they differ and why we chose to introduce the notion of anticipatory gap.

Consider the following two maps assigning values to states: the *offline value* and the *online value* of state s_t, respectively denoted by $\phi(s_t)$ and $\pi(s_t)$ and defined by

$$\phi(s_t) = \max\{\mathbb{E}[f(x,\xi)|s_t] \ \mid \ x \text{ dec. proc.}\}$$
$$\pi(s_t) = \max\{\mathbb{E}[f(x,\xi)|s_t] \ \mid \ x \text{ non-anticip. dec. proc.}\}.$$

Note that $\phi(s_t) \geq \pi(s_t)$ for all state s_t. The difference

$$\eta(s_t) = \phi(s_t) - \pi(s_t).$$

is the (local) *expected value of perfect information* ($EVPI$). The expected value of perfect information of the problem is $\eta(s_1)$, that is, the advantage, in the expected sense, of a clairvoyant over a non-clairvoyant (both with infinite computational resources). The next lemma relates the offline problem and ϕ and shows that the operators max and $\mathbb{E}$ commute for clairvoyant decision processes.

Lemma 3 *For any state s_t, $\phi(s_t) = \mathbb{E}[\mathcal{O}(s_t, \xi)|s_t]$.*

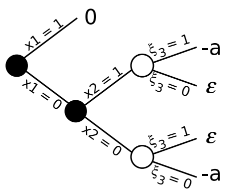


Figure 1: low $EVPI$ but high Global Anticipatory Gap

Proof. Let $x^\star$ be a decision process maximizing $\mathbb{E}[f(x,\xi)|s_t]$. Then for all ξ, $\mathcal{O}(s_t, \xi) \geq f(x^\star, \xi)$, and, as $\mathbb{E}$ is non-decreasing, $\mathbb{E}[\mathcal{O}(s_t,\xi)|s_t] \geq \mathbb{E}[f(x^\star,\xi)|s_t)] = \phi(s_t)$. Inversely, let x^ξ be a decision process maximizing $f(x,\xi)$ with $x^\xi_{1..t-1} = x_{1..t-1}$. Define $x^\star$ by aggregation: $x^\star$ does the same thing as x^ξ on the scenario ξ. Then $\phi(s_t) \geq \mathbb{E}[f(x^\star,\xi)|s_t] = \mathbb{E}[\mathcal{O}(s_t,\xi)|s_t]$. □

In two-stage stochastic programming, a low $EVPI$ makes the problem much easier because an optimal decision is also a good one for each specific scenario [Birge and Louveaux, 1997, ch. 4]. However, this is no longer true in the multistage case. Consider the three-stage problem depicted in Figure 1. Black dots represent decisions variables x_1 and x_2. Stochastic variables ξ_1 and ξ_2 have no influence and are not represented. The white dot represents ξ_3 which take values 0 and 1 with equal probability. Leaves are tagged with their utilities and a is large positive number. The value of the $EVPI$ and the anticipatory gap Δ_g for each state are the following:

state	root state	$x_1 = 0$
Δ_g	0	$1/2(\varepsilon + a)$
η	ε	$1/2(\varepsilon + a)$

On this problem, the $EVPI$ is ε: an optimal solution has a score of ε, whatever the scenario. The expected value of the optimal policy is zero. However, the online anticipatory algorithm always chooses $x_1 = 0$ and thus has an expected utility of $1/2(\varepsilon - a)$. Therefore anticipatory algorithms may behave poorly even with a low $EVPI$. Moreover, in this case, the inequality of Theorem 1 is tight when m converges to $+\infty$, since the GAG equals $1/2(\varepsilon + a)$.

The phenomenon comes from the fact that the $EVPI$ of the problem is low although the $EVPI$ of the node $(x_1 = 0)$ is $\varepsilon - 1/2(\varepsilon - a) = 1/2(\varepsilon + a)$ and thus much larger. This does not contradict the super-martingale property of [Dempster, 1998] because Dempster considers optimal decision processes, which is not the case of anticipatory algorithms. As a result, the expected global loss of the anticipatory algorithm cannot be bounded by the root $EVPI$. The example may suggest that the maximum of the *EVPI*s at each node of the tree gives an upper bound of the *EGL*, but this is not true either. Figure 2 presents a stochastic program, where 'Sub' are clones of the problem in Figure 1, with variables indices shifted. On this problem, the optimal solutions to the scenarios have an expected expected utility of ε, and those of the anticipatory algorithm (with $m = \infty$) have expected utility $1/4(-3a+\varepsilon)$; the EGL thus equals $3/4(a+\varepsilon)$. By Theorem 1, the GAG is not smaller than the EGL ($m = \infty$: no sampling error). As a result, the GAG is greater than the maximum of the $EVPI$ over all nodes, which is equal to $1/2(\varepsilon + a)$.

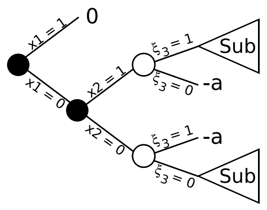


Figure 2: GAG higher than max of the *EVPI*s

Finally, the following theorem gives one more reason why the concept of anticipatory gap is of interest.

Theorem 3 *For any state s_t, we have $\eta(s_t) \geq \Delta_g(s_t)$ and there exist cases in which the inequality is strict.*

Proof. $\eta(s_t) = \phi(s_t) - \pi(s_t)$. Recall that $\pi(s_t)$ is the optimal expected utility of a non-anticipative decision process given s_t. Because of non-anticipativity and because $\mathcal{X}(s_t)$ is finite, there exists an optimal decision $x^\star \in \mathcal{X}(s_t)$ such that $\pi(s_t) = \mathbb{E}\left[\pi(s_{t+1})\,|s_t, x_t = x^\star\right]$. Now

$$\max_{x \in \mathcal{X}(s_t)} \mathbb{E}\left[\mathcal{O}(s_t, x, \xi)\,|s_t\right] \geq \mathbb{E}\left[\mathcal{O}(s_t, x^\star, \xi)\,|s_t\right] \\ \geq \mathbb{E}\left[\pi(s_{t+1})\,|s_t, x_t = x^\star\right].$$

and thus, using lemma 3,

$$\eta(s_t) \geq \mathbb{E}\left[\mathcal{O}(s_t, \xi)\,|s_t\right] - \max_{x \in \mathcal{X}(s_t)} \mathbb{E}\left[\mathcal{O}(s_t, x, \xi)\,|s_t\right] \\ \geq \Delta_g(s_t).$$

This proves the inequality. The second part of the theorem is proven by the example of Figure 1, on which the root node s_1 satisfies $\eta(s_1) = \varepsilon > 0 = \Delta_g(s_1)$. □

7 Conclusion and Research Perspectives

Anticipatory algorithms have been shown experimentally to be successful in tackling a variety of large multistage stochastic integer programs which were outside the scope of a priori methods such as (PO)MDPs and multistage stochastic programming. This paper studied the performance of anticipatory algorithms in terms of their sampling error and the anticipatory gap of the problems. It showed that, whenever the anticipatory gap is small, anticipatory algorithms are scalable and provide high-quality solutions in the expected sense with a logarithmic number of samples in the problem size. The paper also studied how to bound the anticipatory gap both theoretically and experimentally, showing that a simple packet scheduling problem admits a small anticipatory gap and providing experimental evidence on several large multistage stochastic programs. Finally, the paper indicated that the anticipatory gap is an important concept and studied its relationships with the expected value of perfect information.

There are many research directions opened by this research. First, It is desirable to to deepen the understanding of the problem features (both combinatorial and statistical) which lead to small (or large) anticipatory gaps. Second, it is also important to study novel anticipatory algorithms for applications with non-negligible anticipatory gaps. Here an interesting direction is to borrow ideas from Real-Time Dynamic Programming [Barto *et al.*, 1995; Paquet *et al.*, 2005]. Indeed, because the estimation of $\phi(s_t)$ obtained by relaxing non-anticipativity constraints is an upper bound of its online value $\pi(s_t)$ with high probability, a RTDP approach can produce increasingly tighter approximations of the optimal policy until decision time. Despite negative complexity results ([Kearns *et al.*, 1999]), we believe that, if the GAG is not too large, high-quality decisions could be obtained in reasonable time. Indeed, since the far future is unlikely to be as important as the near future for the current decision, we may hope that small trees will be sufficient for many applications.

Acknowledgments

This research is partly supported by NSF Award DMI-0600384 and ONR DEPSCOR Award N000140610607. Thanks to the reviewers for their interesting suggestions.

References

[Barto *et al.*, 1995] A.G. Barto, S.J. Bradtke, and S.P. Singh. Learning to act using real-time dynamic programming. *Artificial Intelligence*, 72(1):81–138, 1995.

[Benoist *et al.*, 2001] T. Benoist, E. Bourreau, Y. Caseau, and B. Rottembourg. Towards stochastic constraint programming: A study of online multi-choice knapsack with deadlines. In *CP'01*, 2001.

[Bent and Van Hentenryck, 2003] R. Bent and P. Van Hentenryck. Dynamic Vehicle Routing with Stochastic requests. *IJCAI'03*.

[Bent and Van Hentenryck, 2004] R. Bent and P. Van Hentenryck. Regrets only! online stochastic optimization under time constraints. In *AAAI'04*, 2004.

[Bent *et al.*, 2005] R. Bent, I. Katriel, and P. Van Hentenryck. Sub-Optimality Approximation. In *CP'05*, 2005.

[Birge and Louveaux, 1997] J.R. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer Verlag.

[Borodin and El-Yaniv, 1998] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998.

[Chang *et al.*, 2000] H. Chang, R. Givan, and E. Chong. Online Scheduling Via Sampling. *Artificial Intelligence Planning and Scheduling (AIPS'00)*, pages 62–71, 2000.

[Dempster, 1998] M. A. H. Dempster. Sequential importance sampling algorithms for dynamic stochastic programming. *Annals of Operations Research*, 84:153–184, 1998.

[Hentenryck *et al.*, 2006] P. Van Hentenryck, R. Bent, and Y. Vergados. Online stochastic reservation systems. In *CPAIOR'06*, Springer, 2006.

[Kearns *et al.*, 1999] M. Kearns, Y. Mansour, and A. Ng. A Sparse Sampling Algorithm for Near-Optimal Planning in Large Markov Decision Processes. In *IJCAI'99*, 1999.

[Paquet *et al.*, 2005] S. Paquet, L. Tobin, and B. Chaib-draa. Real-Time Decision Making for Large POMDPs. In *18th Canadian Conference on Artificial Intelligence*, 2005.

[Shapiro, 2006] A. Shapiro. On complexity of multistage stochastic programs. *Oper. Res. Lett*, 34(1):1–8, 2006.