Optimal timeouts for power management under renewal or hidden Markov processes for requests

Victor Ciriza, Laurent Donini, Jean-Baptiste Durand, Stephane Girard

Optimal timeouts for power management under renewal or hidden Markov models processes for requests

Victor Ciriza[(1)], Laurent Donini[(1,2)], Jean-Baptiste Durand[(2)]*, Stéphane Girard[(2)]

1: Xerox Research Centre Europe
6, chemin de Maupertuis
38240 Meylan, France.
2: Team Mistis, INRIA Rhône-Alpes and LJK
655, avenue de l'Europe, Montbonnot
38334 Saint-Ismier Cedex, France.

Abstract

This article addresses the optimal choice of the waiting period (or *timeout*) that a device should respect before entering sleep mode, so as to optimize a trade-off between power consumption and user impact. The optimal timeout is inferred by appropriate statistical modeling of the times between user requests. In a first approach, these times are supposed independent, and a constant optimal timeout is inferred accordingly. In a second approach, some dependency is introduced through a hidden Markov chain, which also models specific activity states, like business hours or night periods. This model leads to a statistical framework for computing adaptive optimal timeout values. Different strategies are assessed using real datasets, on the basis of the power consumption, user impact and the frequency of wrong decisions.

1 Introduction

The goal of this study is to determine a policy based on the analysis of user behavior achieving a compromise between low power consumption of devices and limited user impact. We primarily describe this with respect to the behavior of printers, however similar policies could also be applied to other devices such as disk drives and displays. Currently, in most printers the time period to wait before entering sleep mode is either set by the administrator or predefined by the device manufacturer according to Energy Star[1] environmental standards. Today, Energy Star criteria do not take into account observed printer usage patterns. Those criteria rather set power consumption requirements depending on the device features (*e.g.* functionalities, estimated volume) and marking technology type (*e.g.* laser, solid ink, inkjet). In this paper, observed printer usage patterns are taken into account through the sequence of print job submissions (referred to as the print process). We model a device having several modes with different power consumptions. For a printer, these might correspond to:

*Corresponding author – Jean-Baptiste.Durand@inrialpes.fr
[1] http://www.energystar.gov

Print mode The device activates its marking engine, print path and controller and completes any print requests. Power consumption is typically the highest in this mode.

Idle mode The device is ready to print immediately and therefore a certain power consumption is required to maintain the device in a state of readiness.

Sleep modes The device is not ready to print immediately, which induces a delay between the user request and the actual beginning of the print job. These modes are sometimes referred to as *standby* or *power-save modes*. Depending on the printer, one or several such modes are available. Power consumption is typically the lowest in one of these sleep modes.

The difference in power consumption between idle mode and sleep modes is often as large as 40%. Power consumption due to transitions between modes depends on the printing technology (*e.g.* laser, solid ink) and is usually larger than the power consumption in idle mode. In the sequel, the transition from sleep modes to idle mode is referred to as *wake-up* while the transition from idle mode to sleep modes is referred to as *shutdown*. From the consumption point of view, the device features are summarized by the power consumption in each of these modes as well as the energy required to switch between these modes. Therefore, our goal amounts to inferring the optimal inactivity interval (or *timeout period*) before entering into sleep modes, given both the device power consumption model and observed usage patterns.

1.1 Consumption model and notations

Our approach relies on the following assumptions. Firstly, assuming that each print request is processed as soon as possible, the power consumption during print jobs is independent on the timeout period. Secondly, power consumptions during idle and sleep modes are supposed to be constant. Finally, printing, shutdown and wake-up transitions are supposed to be instantaneous. Therefore, the optimization focuses on the power consumption in idle and sleep modes and on the energy consumption of the associated transitions. Two kinds of transition are assumed on the printer:

- Transitions from one mode to the sleep mode with closest lower consumption.
- Transitions from one sleep mode to idle mode.

In reality, power consumption often takes the form of a series of pulses, as it is typically thermostatically controlled. Since the timing of these pulses is hard to predict, we assume a time-average power consumption for each mode. Figure 2 illustrates a real printer consumption (Xerox *Phaser 4500* with a single sleep mode) during 5 minutes of use (between 11 am and 11:05 am) and the corresponding power consumption model.

In this paper, we denote by:

- m the number of sleep modes,
- a the power consumption in idle mode (Watts),
- b_j the power consumption in sleep mode j (Watts),
- c_j the energy required to switch from sleep mode $j-1$ to sleep mode j (Joules),
- d_j the wake-up energy required to switch from sleep mode j to print mode (Joules),

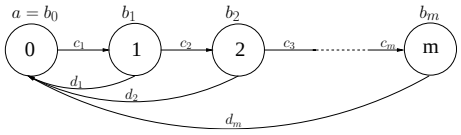


Figure 1: Possible transitions between idle and sleep modes.

with $j = 1, \ldots, m$. Note that, in the above notations, sleep mode 0 corresponds to idle mode, and thus one can define $b_0 = a$. These notations are illustrated in Figure 1.

We limit ourselves to timeout strategies consisting in waiting a duration $\tau^{(j)}$ from the latest print onward, before switching into mode j. Since each print request must be processed immediately, the actual switch only occurs if the time between latest print job completion and the following print request is larger than $\tau^{(j)}$. This requires that the sequence $(\tau^{(1)}, \ldots, \tau^{(m)})$ is increasing. It is also assumed that the sequence $(b_0, \ldots, b_m)$ is decreasing, which implies that mode j is only reachable from mode $j - 1$.

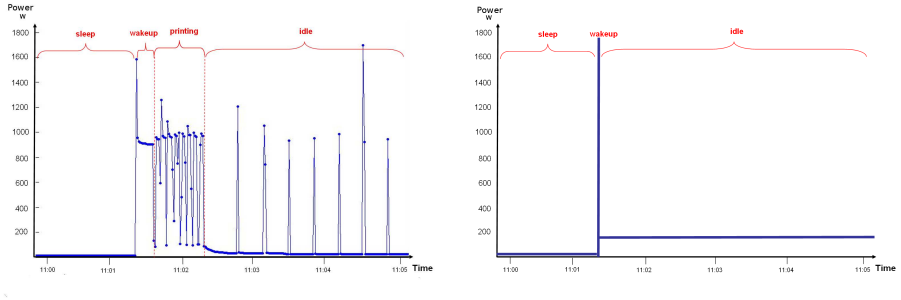


Figure 2: Top: Real power consumption of a printer (between 11 am and 11:05 am). During this period, the device switches between 4 different states: sleep, wake-up, print and idle modes. Bottom: simplified consumption model.

1.2 Related work

The issue of power saving strategies has already been addressed in several works. Although most of them present a very general framework for power management, their applications mainly focus on hardware device (*e.g.* CPU, monitors, hard disk drives). A wide range of approaches are compared in [17], using the following typology of methods:

Timeout A timeout period is fixed either using a quantile of the residual time before next request, or using a parametric function of times between the last two requests and / or request and timeout [8, 12, 17, 4].

L-shape This is a variant of timeout approaches dedicated to request patterns where short busy periods tend to be followed by a long idle period [24].

Exponential average This approach relies on a prediction of next idle period, based on an average of the previous idle periods with exponential weights [14].

Stochastic model These methods aim at finding an optimal probability distribution for the different actions to perform, given the past actions, states of the system and the expected power consumption for each action. The different levels of consumption are related to the notion of state. These approaches mainly rely on the theory of Markov decision processes [25] or their different variants (continuous time, semi-Markov or piecewise homogeneous Markov processes).

Competitive algorithm A c-competitive power saving algorithm is such that the power consumption is less than c times that of an oracle algorithm [16]. An oracle algorithm considers all random variables, including future observations, as known, and achieves the minimal possible power consumption.

Learning tree Adaptive learning trees transform sequences of idle periods into discrete events and store them in tree nodes. They predict idle periods using finite-state machines. This is similar to branch prediction used in microprocessors and selects a path which resembles previous idle periods. At the beginning of an idle period, a learning tree determines an appropriate sleeping state; this algorithm is capable of controlling multiple sleeping states [7].

Our method belongs to the category of stochastic models, and combines the principles of continuous time modeling, piecewise identically distributed times between requests and Markov decision processes.

In [2], a function of a homogeneous Markov process with discrete time and discrete state space is used to model the sequences of requests. The states represent different levels of requests of the device. Markov processes are directly used to model completions of those requests (or *service process*), request queues, and decisions. The system performance is also assessed, based on the waiting time before request completion, and on the number of requests in the queue. In this context, a strategy is optimal if and only if it minimizes a function of the future expected power consumption under a performance constraint (where the notions of power consumption and performance can easily be exchanged). This optimization problem can be seen as a linear programming optimization problem. Thus, an exact optimal solution can be found in polynomial time with respect to input length (the input includes the number of actions, states, the number of requests per time and the queue length).

An extension of this work was proposed in [6] to take into account possible violation of the homogeneity assumption. This is addressed by a piecewise homogeneous Markov request process. This work was further extended in [23] and [3] using semi-Markov processes for modeling the dynamics of the states and events (typically the requests). These extensions are still discrete-time approaches. Continuous-time models were proposed in [19] to represent the requests process (by a homogeneous Poisson process), the service process and its queue (by Markov processes with discrete state space). This paper does not provide details about an algorithm to find a solution to the optimization problem. In [21], the requests process is modeled by a Markov-modulated Poisson process.

In [26], the decision is based on classification algorithms (logistic regression, k-nearest-neighbors or classification trees). A sample of vectors is used to train the system. This vector is composed of characteristics that reflect the state of activity of the user, and in the learning set, the action to perform (turn the device on or off). The time since last request of the device was also added to the vector of characteristics. This approach is compared with a so-called "timeout-based strategy" as defined in [17]. All those approaches take into account the system performance to assess the quality of the method for a given value of power saving, except [19]. In the context of power management for printers, the duration of a CPU cycle is negligible compared to the time between requests. Therefore, a continuous-time model is a natural way to model the request process.

In Section 2, we present a stochastic model for the requests that extends the piecewise homogeneous Markov process presented in [6]. A method is derived for optimally updating the timeout period after each request in the case of one single sleep mode. The results are extended to an arbitrary number of sleep modes. Then a model is proposed for assessing user impact.

In Section 3, particular processes are considered to model the request process: independent models, or hidden Markov models to account for possible temporal heterogeneity. We pay particular attention to parametric models which allow fast update of the timeout. For independent Weibull or Pareto-distributed times between requests, an explicit formula is provided for the optimal timeout. More generally, the optimal timeout is the solution of a non-linear equation that depends on the model parameters, as estimated by maximum likelihood. If the solution has no closed form, it can be approximated by numerical methods. In Section 4, two methods are proposed for the comparison of power management strategies. The former is based on out-of-sample prediction of power consumption under different policies. The latter is based on the number of wrong decisions of both types, and on the numbers of shutdowns. The results highlight the good performance of the method based of a fixed timeout period, computed from a parametric statistical model. Real-world implementation as well as possible extensions or alternatives to our approach are provided in the discussion. Proofs are postponed to the Appendices A and B. In Appendix C, we show how our approach is related to the theory of Markov decision processes.

2 Stochastic model

The print process model is a particular case of a point process, similar to some reliability models, see for instance [20], Chapter 7. In our framework, the failure sequence is replaced by the print request sequence $\{T_i\}_{i\geq 1}$, with the convention $T_0 = 0$ and where i denotes the index of the print request. Equivalently, the print process can be described by

- $\{X_i\}_{i\geq 1}$, where $X_i = T_i - T_{i-1}$ is the time between the $(i-1)^{\text{th}}$ and the i^{th} print request,
- $\{N_t\}_{t\geq 0}$, the counting process of print requests, where $\forall t \in \mathbb{R}^+$, $N_t = \max\{i \in \mathbb{N}; T_i \leq t\}$ is the cumulative number of print requests between 0 and t.

As a consequence of the previous assumptions, the process $\{N_t\}_{t\geq 0}$ is *simple*: there cannot be more than one print request at a time with probability 1. The print process is depicted in Figure 3. In the next section, we limit ourselves to a single-sleep-mode printer. This approach is then extended to several sleep modes in Section 2.2 and to the quantification of the user impact in Section 2.3.

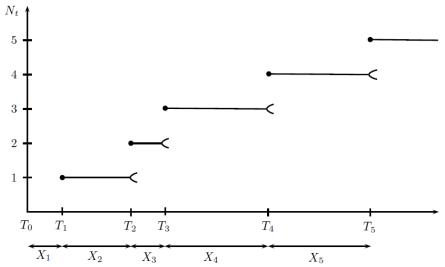
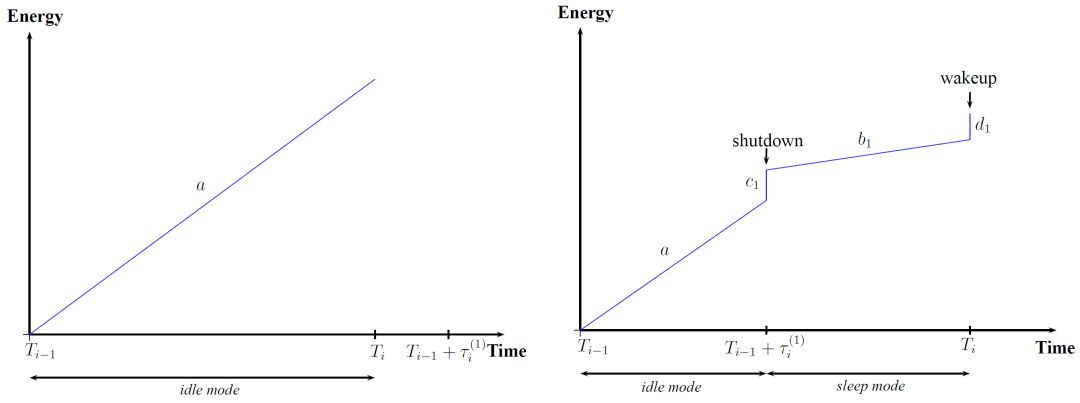


Figure 3: Print Process



(a) When $X_i \leq \tau_i^{(1)}$ (b) When $X_i > \tau_i^{(1)}$

Figure 4: Energy consumption between T_{i-1} and T_i according to the position of T_{i-1}, T_i and $T_{i-1}+\tau_i^{(1)}$

2.1 Single sleep mode

We consider a framework with one sleep mode, so that $m = 1$, where the timeout period $\tau^{(1)}$ may be updated after each print request i. Consequently, this timeout period will be denoted by $\tau_i^{(1)}$. Given a probabilistic model for the print process $\{X_i\}_{i\geq 1}$, we aim at optimizing over $\tau_i^{(1)}$ the expectation of the energy consumption, given the past of the print process $X_{1:i-1} := (X_1, \ldots, X_{i-1})$, between two successive print jobs $i-1$ and i. To compute this energy consumption, two cases arise:

a) Either the time X_i between two successive printings is larger than $\tau_i^{(1)}$. Then the printer stays in idle mode for $\tau_i^{(1)}$ before switching into sleep mode. After a delay $X_i - \tau_i^{(1)}$, the print job is processed and the printer returns to idle mode $j = 0$. Consequently, the energy consumption in this case is $a\tau_i^{(1)} + c_1 + b_1(X_i - \tau_i^{(1)}) + d_1$.

b) Or X_i is smaller than or equal to $\tau_i^{(1)}$. Then the printer stays in idle mode for X_i before processing the job. Consequently, the energy consumption in this case is aX_i. These two cases are illustrated in Figure 4.

The expected consumption between two successive printings is derived in Lemma 1 below. Let $f_{X_i|X_{1:i-1}}$ be the probability density function (pdf) of X_i given $X_{1:i-1}$, $\bar{F}_{X_i|X_{1:i-1}}$ be its survival distribution function, and

$$z_{X_i|X_{1:i-1}}(x) = \frac{f_{X_i|X_{1:i-1}}(x)}{\bar{F}_{X_i|X_{1:i-1}}(x)}$$

be the failure rate function in reliability theory [1], Chapter 2. In our case, it can be interpreted as a *printing rate function*. We also define the asymptotic printing rate $\ell_i = \lim_{x\to+\infty} z_{X_i|X_{1:i-1}}(x)$ and $\Delta t_1 = (c_1 + d_1)/(a - b_1)$. In a static analysis of the printer energy consumption, Δt_1 is the time after which switching into sleep mode is less expensive than staying in idle mode (see Figure 5). This is frequently called the *break-even* time.

Lemma 1 *The expected consumption between two successive printings given $X_{1:i-1}$ is:*

$$\begin{aligned}\mathbb{E}(h(X_i,\tau_i^{(1)})|X_{1:i-1}) &= a\mathbb{E}(X_i|X_{1:i-1})\\ &+ \quad (a-b_1)\bar{F}_{X_i|X_{1:i-1}}(\tau_i^{(1)})(\Delta t_1+\tau_i^{(1)})\\ &- \quad (a-b_1)\int_{\tau_i^{(1)}}^{+\infty} x f_{X_i|X_{1:i-1}}(x)dx.\end{aligned}$$

Formally, an optimal timeout period is defined by:

$$\hat{\tau}_i^{(1)} \in \arg\min_{\tau} \mathbb{E}(h(X_i,\tau)|X_{1:i-1}) \tag{1}$$

and can be computed based on the following result.

Proposition 1 *Two situations are examined, depending on the behavior of the printing rate function.*
a) Suppose that the printing rate function $z_{X_i|X_{1:i-1}}(x)$ is decreasing in x. Three cases occur:

- *If $1/\Delta t_1 < \ell_i$, then $\hat{\tau}_i^{(1)} = +\infty$.*
- *If $\ell_i \leq 1/\Delta t_1 \leq z_{X_i|X_{1:i-1}}(0)$, then $\hat{\tau}_i^{(1)}$ is the unique solution of $z_{X_i|X_{1:i-1}}(\hat{\tau}_i^{(1)}) = 1/\Delta t_1$.*
- *If $z_{X_i|X_{1:i-1}}(0) < 1/\Delta t_1$, then $\hat{\tau}_i^{(1)} = 0$.*

b) Suppose that $z_{X_i|X_{1:i-1}}$ is increasing or constant. Four cases occur:

- *If $1/\Delta t_1 < z_{X_i|X_{1:i-1}}(0)$, then $\hat{\tau}_i^{(1)} = +\infty$.*
- *If $z_{X_i|X_{1:i-1}}(0) \leq 1/\Delta t_1 \leq \min\left(\ell_i, 1/\mathbb{E}(X_i|X_{1:i-1})\right)$, then $\hat{\tau}_i^{(1)} = +\infty$.*
- *If $\max\left(z_{X_i|X_{1:i-1}}(0), 1/\mathbb{E}(X_i|X_{1:i-1})\right) < 1/\Delta t_1 \leq \ell_i$, then $\hat{\tau}_i^{(1)} = 0$.*
- *If $\ell_i < 1/\Delta t_1$, then $\hat{\tau}_i^{(1)} = 0$.*

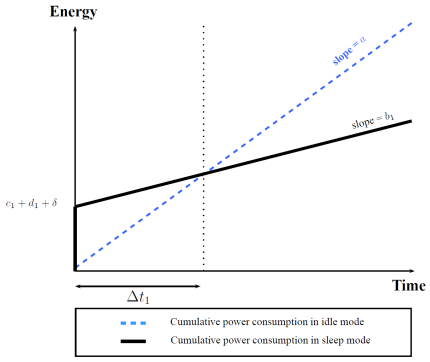


Figure 5: Graphical interpretation of Δt_1

It appears that three situations are possible. Either the times between printings are so small on average that the printer should not enter sleep mode ($\hat{\tau}_i^{(1)} = \infty$), or they are large on average, and the best strategy is to enter sleep mode immediately ($\hat{\tau}_i^{(1)} = 0$). The intermediate case provides non-degenerate optimal timeouts defined by the equation $z_{X_i|X_{1:i-1}}(\hat{\tau}_i^{(1)}) = 1/\Delta t_1$. This result highlights the separate roles of the printer characteristics (summarized by Δt_1) and the user behavior (modeled through the printing rate function $z_{X_i|X_{1:i-1}}$).

2.2 Multiple sleep modes

The aim of this section is to extend to multiple sleep mode printers, the results derived for single-mode printers. As in the previous paragraph, each timeout period $\tau^{(j)}$ may be updated after each print request i and will be denoted by $\tau_i^{(j)}$. To compute the energy consumption, three cases are considered.

a) If the time X_i between two successive printings is smaller than $\tau_i^{(1)}$, then the printer stays in idle mode for X_i before processing the job. Consequently, the energy consumption in this case is aX_i.

b) If X_i is larger than $\tau_i^{(1)}$ and smaller than $\tau_i^{(m)}$, then the printer stays in idle mode for $\tau_i^{(1)}$ before switching into the first sleep mode. Afterward, the printer successively switches into the r following sleep modes where r is such that $\tau_i^{(r)} < X_i < \tau_i^{(r+1)}$. The time spent in each sleep mode is $\tau_i^{(j)} - \tau_i^{(j-1)}$ if $2 \leq j \leq r-1$ and $X_i - \tau_i^{(j)}$ if $j = r$. Consequently, the energy consumption in this case is $a\tau_i^{(1)} + \sum_{j=1}^{r-1}(c_j + b_j(\tau_i^{(j+1)} - \tau_i^{(j)})) + c_r + b_r(X_i - \tau_i^{(r)}) + d_r$.

c) Finally, if X_i is larger than $\tau_i^{(m)}$, then the printer stays in idle mode for $\tau_i^{(1)}$ before switching into the first sleep mode. Afterward, the printer successively switches into the m sleep modes. As in the previous case, the time spent in each sleep mode is $\tau_i^{(j)} - \tau_i^{(j-1)}$ if $2 \leq j \leq m-1$ and $X_i - \tau_i^{(j)}$ if $j = m$. Consequently, the energy consumption is $a\tau_i^{(1)} + \sum_{j=1}^{m-1}(c_j + b_j(\tau_i^{(j+1)} - \tau_i^{(j)})) + c_m + b_m(X_i - \tau_i^{(m)}) + d_m$. Introducing $\Delta t_j = (c_j + d_j - d_{j-1})/(b_{j-1} - b_j)$ for $j = 1, \ldots, m$ with the conventions $b_0 = a$ and $d_0 = 0$, we have:

Lemma 2 *The expected consumption between two successive print requests given $X_{1:i-1}$ is:*

$$\begin{aligned}\mathbb{E}(h(X_i, \tau_i^{(1)}, \ldots, \tau_i^{(m)})|X_{1:i-1}) &= a\mathbb{E}(X_i|X_{1:i-1}) \\ &+ \sum_{j=1}^{m}(b_{j-1} - b_j)\bar{F}_{X_i|X_{1:i-1}}(\tau_i^{(j)})(\Delta t_j + \tau_i^{(j)}) \\ &- \sum_{j=1}^{m}(b_{j-1} - b_j)\int_{\tau_i^{(j)}}^{+\infty} x f_{X_i|X_{1:i-1}}(x)dx.\end{aligned}$$

It is remarkable that the expected energy consumption is expanded as the sum of m terms, each of them depending on one and only one timeout. Thus, the minimization of $\mathbb{E}(h(X_i, \tau_i^{(1)}, \ldots, \tau_i^{(m)})|X_{1:i-1})$ with respect to $(\tau_i^{(1)}, \ldots, \tau_i^{(m)})$ can be split into m optimization problems leading to explicit optimal timeouts.

Proposition 2 *Two situations are examined, depending on the behavior of the printing rate function.*
a) Suppose that the printing rate function $z_{X_i|X_{1:i-1}}(x)$ is decreasing in x. For each $j = 1, \ldots, m$ three cases occur:

- *If $1/\Delta t_j < \ell_i$, then $\hat{\tau}_i^{(j)} = +\infty$.*

- *If* $\ell_i \leq 1/\Delta t_j \leq z_{X_i|X_{1:i-1}}(0)$*, then* $\hat{\tau}_i^{(j)}$ *is the unique solution of* $z_{X_i|X_{1:i-1}}(\hat{\tau}_i^{(j)}) = 1/\Delta t_j$*.*
- *If* $z_{X_i|X_{1:i-1}}(0) < 1/\Delta t_j$*, then* $\hat{\tau}_i^{(j)} = 0$*.*

b) Suppose that $z_{X_i|X_{1:i-1}}$ *is increasing or constant. For each* $j = 1, \ldots, m$ *four cases occur:*

- *If* $1/\Delta t_j < z_{X_i|X_{1:i-1}}(0)$*, then* $\hat{\tau}_i^{(j)} = +\infty$*.*
- *If* $z_{X_i|X_{1:i-1}}(0) \leq 1/\Delta t_j \leq \min\left(\ell_i, 1/\mathbb{E}(X_i|X_{1:i-1})\right)$*, then* $\hat{\tau}_i^{(j)} = +\infty$*.*
- *If* $\max\left(z_{X_i|X_{1:i-1}}(0), 1/\mathbb{E}(X_i|X_{1:i-1})\right) < 1/\Delta t_j \leq \ell_i$*, then* $\hat{\tau}_i^{(j)} = 0$*.*
- *If* $\ell_i < 1/\Delta t_j$*, then* $\hat{\tau}_j^{(1)} = 0$*.*

2.3 Modeling user impact

In reality, transitions between sleep and idle modes may delay printing. The more frequently the system switches between sleep and idle modes, the more the user will be impacted. We thus propose to model this impact by a penalty term in the energy consumption. For the sake of simplicity, let us consider the case of a single-sleep-mode printer. We further assume that the user impact is proportional to the number of shutdown transitions. With such a model, the consumption between two successive print requests $h(X_i, \tau_i^{(1)})$ is replaced by the cost $g(X_i, \tau_i^{(1)}) = h(X_i, \tau_i^{(1)}) + \delta \mathbb{1}_{\{X_i > \tau_i^{(1)}\}}$, where $\delta > 0$ is the weight assigned to user impact in the energy consumption. The expected consumption including this user impact is given in the next Lemma.

Lemma 3 *The expected penalized consumption between two successive print requests given* $X_{1:i-1}$ *is:*

$$\begin{aligned}\mathbb{E}(g(X_i, \tau_i^{(1)})|X_{1:i-1}) = a\mathbb{E}(X_i|X_{1:i-1}) \\ + \quad (a-b_1)\bar{F}_{X_i|X_{1:i-1}}(\tau_i^{(1)})(\tilde{\Delta}t_1 + \tau_i^{(1)}) \\ - \quad (a-b_1)\int_{\tau_i^{(1)}}^{+\infty} x f_{X_i|X_{1:i-1}}(x)dx\end{aligned}$$

with $\tilde{\Delta}t_1 = (c_1 + d_1 + \delta)/(a - b_1)$*.*

It turns out that penalizing the consumption by the number of shutdowns can be interpreted as increasing the transition consumption $c_1 + d_1$ by δ. As a consequence, Proposition 1 still holds with Δt_1 replaced by $\tilde{\Delta}t_1$. In particular, when the printing rate is a decreasing function and there is a non-degenerate optimal timeout period such that $z_{X_i|X_{1:i-1}}(\hat{\tau}_i^{(1)}) = 1/\tilde{\Delta}t_1$, the optimal timeout period is an increasing function of δ. Moreover, this property also allows user impact to be accounted for in the break-even time. In practice, $\tilde{\Delta}t_1$ can be seen as the optimal timeout if X_i follows a particular Pareto distribution – see Section 3.1.

3 Modeling the print process

According to the previous Section, the optimal timeout period depends on the model for the print process through the printing rate function. Four different print process models are proposed hereafter. In the first three approaches, times between printings are supposed independent. In

the last approach, a hidden Markov chain (HMC) is used to model dependencies between printing times. The HMC states can be interpreted as specific states of activity like business hours or night periods. While we only consider a single sleep mode, extension to multiple sleep modes is straightforward.

3.1 Renewal process

In this paragraph, the times between print requests are supposed independent. The point process $\{N_t\}_{t\geq 0}$ is then a particular case of renewal process (see [20], Chapter 7). The random variable modeling the times between printings is denoted by X, since its distribution does not depend on the index i of the print job. Similarly, the optimal timeout period in sleep mode j is denoted by $\hat{\tau}^{(j)}$. In the following, the optimal timeout is studied under the assumptions that X is Weibull, Gamma or Pareto distributed.

3.1.1 Weibull distribution

The pdf of the two-parameter Weibull distribution is parametrized as $f_X(x) = \alpha\lambda^\alpha x^{\alpha-1}\exp[-(\lambda x)^\alpha]$ for $x \geq 0$, where $\lambda > 0$ is a scale parameter and $\alpha > 0$ is referred to as the shape parameter. Let us recall that, in this case the mean time between printings is $\mathbb{E}(X) = \Gamma(1+1/\alpha)/\lambda$ and the printing rate function is $z_X(x) = \alpha\lambda^\alpha x^{\alpha-1}$ for $x \geq 0$. Note that this function can be decreasing if $\alpha \in (0,1)$, increasing if $\alpha > 1$ or constant if $\alpha = 1$ (exponential distribution). As a consequence of Proposition 2, we have

$$\hat{\tau}^{(j)} = \left| \begin{array}{ll} (\alpha\lambda^\alpha \Delta t_j)^{\frac{1}{1-\alpha}} & \text{if } \alpha \in (0,1) \\ 0 & \text{if } \alpha \geq 1 \text{ and } \Delta t_j < \mathbb{E}(X) \\ +\infty & \text{if } \alpha \geq 1 \text{ and } \Delta t_j > \mathbb{E}(X). \end{array} \right.$$

Of course, in practical situations, the parameters α and λ are replaced by their maximum likelihood estimates, see [15], Chapter 21 for their computation and Section 4 for examples.

3.1.2 Gamma distribution

The pdf of the two-parameter Gamma distribution is parametrized as $f_X(x) = \beta^{-\alpha}\Gamma(\alpha)^{-1}x^{\alpha-1}\exp(-x/\beta)$ for $x \geq 0$, where $\beta > 0$ is a scale parameter and $\alpha > 0$ is the shape parameter. In this case, we have $\mathbb{E}(X) = \alpha\beta$ but no closed-form expression for the printing rate function is available for non-integral α. Nevertheless, it can be shown [1] that, similarly to the Weibull case, the printing rate function is decreasing if $\alpha \in (0,1)$, increasing if $\alpha > 1$ or constant if $\alpha = 1$ (exponential distribution). Thus,

$$\hat{\tau}^{(j)} = \left| \begin{array}{ll} z_X^{-1}(1/\Delta t_j) & \text{if } \alpha \in (0,1) \\ 0 & \text{if } \alpha \geq 1 \text{ and } \Delta t_j < \alpha\beta \\ +\infty & \text{if } \alpha \geq 1 \text{ and } \Delta t_j > \alpha\beta, \end{array} \right.$$

the printing rate function $z_X(x)$ being evaluated numerically through the use of the incomplete gamma function. The maximum likelihood estimates of α and β are computed following [15], Chapter 17 and the computation of $\hat{\tau}^{(j)}$ is achieved with a dichotomy procedure.

3.1.3 Pareto distribution

The pdf of the two-parameter Pareto distribution is parametrized as $f_X(x) = \frac{\alpha}{x}\left(\frac{\beta}{x}\right)^\alpha$ for $x \geq \beta$, where β and α are two positive parameters. The associated printing rate function is $z_X(x) = \alpha/x$,

which yields $\hat{\tau}^{(j)} = \alpha\Delta t_j$ if $\Delta t_j \geq \beta/\alpha$ (and $\hat{\tau}^{(j)} = 0$ otherwise). This result is consistent with that in [4]. As a consequence, the break-even time can be seen as the optimal timeout for a Pareto distribution with $\alpha = 1$. The maximum likelihood estimators of the parameters are given in [15], Chapter 20.

3.2 Hidden Markov model

The assumption of a constant printing frequency throughout day and night does not seem realistic *a priori*. It can be expected that during given periods, users will tend to print more often or less often than average. Such periods can be interpreted in terms of activity levels, corresponding to different levels of printing rates: *a)* rush hours with short times between printings, mainly due to users printing a sequence of documents in a short period of time; *b)* normal hours with medium times between printings; *c)* off-peak times with long times between printings, due to night or weekends.

This corresponds to a heterogeneous distribution of the times between printings, such that there exist some homogeneous periods $(i_1, \ldots, i_k)$ where $(X_{i_1}, ..., X_{i_k})$ have the same distribution. Those characteristics can be modeled by a HMC print process [9]. In this model, activity periods are defined by non-visible factors, such as the amount of users at a given time in the printer network (which is related to working hours and can vary with the company), the type of users, country or even site specificities. In HMC modeling, these non-visible factors must be deduced from the observed variables. In our case, one might imagine that periods with comparable times between printings tend to correspond to the same level of activity.

3.2.1 Definition

Formally, an HMC is defined by two processes $X_{1:n} = (X_1, \ldots, X_n)$ (observed process) and $S_{1:n} = (S_1, \ldots, S_n)$ (hidden process), such that:

a) $S_{1:n}$ is a homogeneous Markov chain with finite state space $\{1, \ldots, K\}$, with transition matrix A and a distribution $\pi = (\pi_1, ..., \pi_K)$ for the initial state S_1. Here, $S_{1:n}$ is assumed stationary and ergodic. The marginal distribution of S_i is denoted by π. Here, S_i represents the state of the process at ith printing request, which is not directly observed.

b) Given $S_{1:n} = s_{1:n}$, the X_i are mutually independent, and independent on the $(S_{i'})_{i' \neq i}$, with conditional pdf $f_{\theta_{s_i}}$ (called *emission distributions*), where $(f_\theta)_{\theta \in \Theta}$ is a parametric family of pdf.

The set of parameters of this model is $\eta = (\pi, A, \theta_1, ..., \theta_K)$; this is estimated by likelihood maximization, using the Expectation Maximization (EM) algorithm for HMCs [9].

Since stationarity is assumed for the hidden process, the marginal distribution of X_i has pdf $\sum_j \pi_j f_{\theta_j}$, which does not depend on i; thus the observed process is also stationary. However, the conditional distribution of X_i given $S_{1:n} = s_{1:n}$ has pdf $f_{\theta_{s_i}}$, which depends on i; this is why the above HMC can be used to model changes in the printing rate.

Our HMC model is closely related to the Markov-modulated Poisson process used in [21]. The connection between both models is clarified in [22]. In the particular model considered in [21], transitions between past and current states do not depend on past times between requests, so their model actually is an HMC.

As an alternative, non-stationarity can be modeled by a sliding window approach inspired by [6]. The model parameters of the renewal process in Section 3.1 are reestimated after each request i, using dataset $X_{i-\mathcal{L}+1:i}$, where $\mathcal{L}$ is called the window length.

3.2.2 Parameter estimation

We use the general notation $\mathbb{P}()$ to denote either a probability mass function or a probability density function, the true nature of $\mathbb{P}()$ being obvious from the context. The parameter is estimated by likelihood maximization, using the Expectation Maximization (EM) algorithm for hidden Markov chains [9]. This iterative algorithm starts from an initial value $\eta^{(0)}$ of the parameter and creates a sequence $(\eta^{(m)})_{m\geq 0}$ whose likelihood grows. The sequence $(\eta^{(m)})_{m\geq 0}$ converges to a consistent solution of the likelihood equations when $\eta^{(0)}$ is close to the optimal solution. At each iteration m, it proceeds as follows:

- Expectation (E) step: determination of the $\mathbb{Q}$ function defined by:

$$\begin{aligned}\mathbb{Q}(\eta,\eta^{(m)}) &= \mathbb{E}_{\eta}\left[\log \mathbb{P}_{\eta^{(m)}}(S_1^n = s_1^n, X_1^n = x_1^n)|X_1^n = x_1^n\right] \\ &= \sum_{k=1}^{K} \log \pi_k \mathbb{P}_{\eta^{(m)}}(S_1 = k|X_1^n = x_1^n) \\ &\quad + \sum_{i=1}^{n-1}\sum_{k,l} \log A_{k,l}\mathbb{P}_{\eta^{(m)}}(S_i = k, S_{i+1} = l|X_1^n = x_1^n) \\ &\quad + \sum_{i=1}^{n}\sum_{k=1}^{K} \log f_{\theta_k}(x)\mathbb{P}_{\eta^{(m)}}(S_i = k|X_1^n = x_1^n) \qquad (2)\end{aligned}$$

- Maximization (M) step: maximization of $\mathbb{Q}(\eta,\eta^{(m)})$ with respect to η:

$$\eta^{(m+1)} = \arg\max_{\eta} \mathbb{Q}(\eta,\eta^{(m)}) \qquad (3)$$

3.2.3 Adaptive timeout period using HMCs

Since the case of printers with multiple sleep modes is not considered, the timeout period will be denoted by τ rather than $\tau^{(j)}$. This Section details how to exploit the hidden state values to propose adaptive timeout periods $\hat{\tau}_i$ that are updated after each printing job i. Those strategies basically consist in predicting the time to the next print request X_i, from the past observed values $X_{1:i-1}$. We propose three approaches to dynamically re-estimate τ_i. The first two approaches are based on a prediction $\hat{S}_i$ of the next state value from the past of the process $X_{1:i-1}$ (using two variants for the prediction). The predicted distribution for X_i is then $f_{\theta_{\hat{S}_i}}$. The third approach considers all possible values of S_i, and thus takes into account the uncertainty about its value.

Viterbi-based approach In this approach the next state value $\hat{S}_i$ is predicted as

$$\arg\max_{k}(\max_{s_{1:i-1}} \mathbb{P}(S_{1:i-1} = s_{1:i-1}, S_i = k|X_{1:i-1})).$$

This value is deduced from the Viterbi algorithm [9].

Filtering-based approach This approach consists in predicting the next state value S_i as

$$\tilde{S}_i = \arg\max_{k} \beta_i(k). \qquad (4)$$

where $\beta_i(k) = \mathbb{P}(S_i = k|X_{1:i-1})$ is the filtered probability. This quantity is deduced from $\mathbb{P}(S_{i-1} = j|X_{1:i-1})$ (forward recursion in [9]).

Approach based on full conditional distribution This approach consists in computing the printing rate function of X_i given $X_{1:i-1}$. The pdf of this distribution is given by:

$$f_{X_i|X_{1:i-1}}(x) = \sum_{k=1}^{K} f_{\theta_k}(x)\beta_i(k). \tag{5}$$

Thus, the survival distribution function is:

$$\bar{F}_{X_i|X_{1:i-1}}(x) = \sum_{k=1}^{K} \bar{F}_{\theta_k}(x)\beta_i(k) \tag{6}$$

and the printing rate function follows from (5) and (6).

Each of the three approaches results into an estimated pdf f_i for the predictive distribution of X_i, namely $f_{\theta_{\hat{S}_i}}$ in a) and b), and $f_{X_i|X_{1:i-1}}$ in c). Each pdf is associated with a printing rate function z_i. The optimal timeout period $\hat{\tau}_i$ is given by Proposition 1, replacing $\hat{\tau}_i^{(1)}$ by $\hat{\tau}_i$ and $z_{X_i|X_{1:i-1}}$ by z_i.

In the approach based on full conditional distributions, even in the case of Weibull, Gamma or Pareto observation distribution families $(f_\theta)_{\theta\in\Theta}$, we could not derive general conditions on the parameters $(\theta_k)_{k=1,\dots,K}$, under which equation $z_{X_i|X_{1:i-1}}(\hat{\tau}_i) = 1/\Delta t_1$ has a unique solution. Thus, numerical methods have to be used, to determine whether the optimal timeout period is null, positive or infinite.

4 Experiments

Our methodology is illustrated, in the sequel, by experiments on two real datasets. The efficiencies of the timeout strategies introduced in Section 3 are compared in terms of energy consumption. These strategies are also compared with four alternatives called *Energy star method*, *oracle method*, *c-competitive algorithm* and *exhaustive search method*. In the *Energy star method*, the timeout period is fixed so as to comply with the Energy Star standard, depending on the printer features. In the *oracle method*, the future of the print process is supposed to be known. The printer switches into sleep mode j before print job i if $X_i > \Delta t_j$. Let us highlight that this reference method provides a lower bound on the consumption but cannot be used in practice. The strategy consisting in setting the timeout at Δt_j is referred to in [17] and [4] as a *c-competitive* algorithm (in the sense of [16]). Since this is a deterministic algorithm, here $c = 2$. Finally, the *exhaustive search method* consists in finding the timeout that minimizes the actual consumption thanks to an exhaustive search. Moreover, the method mentioned in Section 3.1 will be called *static method*, while the HMC-based methods described in Section 3.2.3 will be referred to as the *Viterbi method*, *filtering method* and *conditional method*. The *sliding window method*, also described in Section 3.2.3, is also considered in the experiments.

In both datasets, times between printings are deduced from the print logs, recorded during the whole of the year 2006 on XRCE print infrastructure which is composed of 14 printers and involves 155 users. These data have been collected by the *Xerox Job Tracking Agent*[2]. It is an office print tracking tool that allows the capture and recording of information about end users' printing behaviors. Additionally, real power consumption has been measured on two different printer models with a specific power metering unit[3]. The first printer is a *Xerox WorkCentre 238* model with two sleep modes. Its power consumption is $a_1 = 270W$ in *idle mode*, $b_1 = 150W$

[2] http://www.consulting.xerox.com/print-tracking/

[3] Fluke 43B Power Quality Analyzer (http://us.fluke.com)

in the *first sleep mode* and $b_2 = 50W$ in the *second sleep mode*. The energy required to switch from the first and second sleep modes to idle mode are respectively $d_1 = 40kJ$ and $d_2 = 200kJ$. Energies to enter sleep modes are negligible, $c_1 = c_2 = 0$. The second printer is a *Phaser 4500* model with one single sleep mode. Its power consumption is $a_1 = 80W$ in *idle mode*, $b_1 = 16W$ in *sleep mode*. The energy required to switch from sleep mode to idle mode is $d_1 = 25.3kJ$ and the energy to enter sleep mode is negligible, $c_1 = 0$.

Static, Viterbi, filtering, conditional and sliding window methods require the selection of a distribution for the times between print requests. This choice has been made using a goodness-of-fit χ^2 test, whose results are summed up in Table 1.

Distribution	P-value	
	Phaser 4500	WorkCentre 238
Gamma	8e-02	8e-04
Weibull	1e-07	7e-04
Lognormal	2e-32	2e-04
Pareto	3e-07	3e-04
Normal	5e-75	2e-11
Cauchy	7e-110	2e-25

Table 1: P-values obtained with the χ^2 goodness of fit test for the selection of a distribution for the times between printings.

It appears that Weibull and Gamma are the most appropriate distributions. In the following, a Weibull distribution is adopted to model the distribution of the times between printings, when those are assumed independent. A Gamma distribution would also be suitable but would not allow the derivation of an explicit timeout (see paragraph 3.1.2). The considered HMC model has also Weibull emission distributions, and three states, which can be interpreted as rush, normal and calm periods, from the point of view of the print requests. Note that the number of states or the family of emission distributions could also be selected using penalized likelihood criteria [11] or cross-validation [5]. The M step for parameter estimation by the EM algorithm is given in Appendix B. Section 4.1 is dedicated to the analysis of the out-of-sample performance of the above-mentioned methods without taking user impact into account. In paragraph 4.2, the influence of the penalty term is examined under several points of view: consumption, number of shutdowns, number of wrong wake-ups and wrong shutdowns.

4.1 Cross-validated assessment of the strategies

The goal of this experiment is to investigate the methods' performance on future data, and thus to assess their generalization capacities. We focus on the *Xerox WorkCentre 238* dataset ($n = 3910$ print jobs) and user impact is not considered. Its predefined timeouts according to Energy Star environmental standards is $900s$ for the first sleep mode and $1{,}800s$ for the second. The test procedure is multi-fold cross-validation [27], as follows: the dataset is divided into L contiguous sub-samples of equal size. Then for each sub-sample $\ell \leq L-1$, the method parameters are estimated on this sub-sample while the consumption is computed on sub-sample $\ell + 1$. The length of the sliding window $\mathcal{L}_\ell$ is one of the parameters; this is also estimated on sub-sample ℓ only, by minimizing the consumption over $\mathcal{L}_\ell$. Three cases are considered: $L = 10$ sub-samples of size 361, $L = 30$ sub-samples of size 121 and $L = 60$ sub-samples of size 61. Results are summarized in Tables 2 and 3. The computation times include the computation of the actual consumption for the dataset.

It appears on Table 2 that exhaustive search, static, filtering, conditional and sliding window methods are the most efficient ones in terms of consumption. The consumption associated to these methods is about 12% larger than the lower bound given by the Oracle method. This slight increase of the optimal consumption indicates that the Weibull distribution fits the $(x_i)_{1\leq i\leq n}$ well. Besides, exhaustive search, static, conditional and sliding window methods are quite robust since they yield a constant consumption, whatever the subdivision. Moreover, the standard deviation of the consumption represents less than 2% of the total consumption. Among these four methods, the static one is at least thousand time faster than the other ones. Experiments were conducted in Matlab on an Intel Pentium Dual Core running at 2.5GHz.

Focusing on Table 3, it appears that the timeout periods provided by the static and exhaustive search methods are approximately independent on the subdivision of the sample, for both methods. Let us emphasize that static timeouts benefit from small standard deviation whereas exhaustive search timeouts suffer from a high variability. As a conclusion, static method seems to be an accurate, reliable and fast method to select the optimal timeouts. A decrease of about 12% of power consumption can be achieved with regard to the Energy Star method. The gain with regard to the competitive algorithm is about 6%.

	Total consumption (kWh)			Standard deviation of consumption			Mean computation time by sample (ms)		
Sample size	361	121	61	361	121	61	361	121	61
Energy Star	500	500	500	7.99	4.73	3.04	1.2e+00	1.0e+00	2.0e+00
$\tau^{(1)} = \tau^{(2)} = 0$	498	498	498	6.77	3.76	2.55	2.0e+00	1.0e+00	1.0e+00
Exhaustive search	446	446	447	6.86	4.17	2.76	6.6e+04	1.5e+05	2.7e+05
Oracle	399	399	399	7.13	4.11	2.72	2.0e+00	2.0e+00	2.0e+00
c-competitive	471	471	471	7.66	4.54	2.94	5.0e-01	1.0e+00	2.0e+00
Static	446	446	446	7.02	4.13	2.77	2.0e+01	5.0e+01	9.0e+01
Sliding window	445	445	444	7.02	4.18	2.77	5.2e+05	1.5e+05	6.6e+04
Viterbi	471	464	462	8.07	3.97	2.64	1.0e+04	4.3e+03	2.8e+03
Filtering	472	462	462	8.29	3.93	2.62	1.3e+03	1.7e+02	1.9e+02
Conditional	456	454	450	5.62	4.04	2.72	5.8e+04	5.4e+04	6.0e+04

Table 2: Energy consumption and mean computation time associated to the different strategies.

		Mean timeouts (s)			Standard deviation of timeouts		
Sample size		361	121	61	361	121	61
Exhaustive search	$\tau^{(1)}$	26	25	24	13	17	21
	$\tau^{(2)}$	203	208	218	55	108	121
Static	$\tau^{(1)}$	11	12	12	2	5	7
	$\tau^{(2)}$	179	188	192	33	64	84

Table 3: Timeout associated to the different strategies.

4.2 Assessment of user impact

In what follows, the behavior of the methods is compared when taking user impact into account on the Phaser 4500 printer. Our test procedure is the following: The dataset ($n = 2,320$) is divided into 2 sub-samples with the same size. Parameters of each method are estimated on the first sub-sample, while the total consumption is computed on the second one as the penalty δ varies (see Section 2.3).

The variations of the number of shutdowns (or equivalently wake-ups) as a function of the penalty δ are depicted in Figure 6. Given a delay of 8 s caused by each transition on this printer model, the y-axis in Figure 6 also corresponds to an upper bound of the total delay, from the users' point of view. Even though, for a fixed penalty δ, the different methods yield different numbers of shutdowns and different consumptions, it appears on Figure 7 that, for a fixed number of shutdowns, the consumption is about the same whatever the method used. To compare the different methods, we propose a graphical comparison based on an adaption of ROC (Receiver Operating Characteristic) curves [13] to our framework. To this end, let us denote by $\alpha = \mathbb{P}(X_i > \Delta t | X_i < \tau_i)$ the probability of a type I error at i^{th} print request. In this context, this error occurs when a printer stays in idle mode, whereas it should enter into sleep mode. Similarly, the probability of a type II error is given by $\beta = \mathbb{P}(X_i < \Delta t | X_i > \tau_i)$. In the case of type II errors, the printer enters into sleep mode, whereas it should stay idle. Note that the number N_{wd} of errors of type II is used in [17] as a measure of performance of several algorithms for power management. In practice, α and β are estimated by their empirical counterparts. The ROC curve (Figures 8 and 9) is built for each method by drawing $1 - \beta$ as a function of α by letting the penalty δ vary. The best method is the one whose ROC curve is the closest to the vertical line $\alpha = 0$ and to the horizontal line $1 - \beta = 0$. At the opposite, the worst method is the one whose ROC curve is the closest to the diagonal line $1 - \beta = \alpha$. Here, it appears that Viterbi and static methods (including the c-competitive method) are the best ones, since they yield the best compromise between the two risks. Keeping in mind the conclusions of the previous paragraph, it seems that the static methods should be preferred since they are the simplest and most robust ones.

In the case where no penalty due to user impact is applied ($\delta = 0$), the consumptions corresponding to the Energy Star and to the c-competitive timeouts are 20 % and 7 % higher, respectively, than that corresponding to the static method. Moreover, the optimal timeout period provided by the static method is $\tau = 10s$. The associated value of the consumption ($78.1kWh$) is only 0.5% lower that the consumption achieved by setting $\tau = 0s$, which corresponds to the so-called *eager* policy in [2]. The eager policy, considered as "often unacceptable" by the authors in their context, is actually nearly optimal for our dataset. However, this result is specific to the considered printer and user behavior: in paragraph 4.1, the eager policy yields a waste of 5.2% in the energy consumption, compared to the static method (see Table 2).

5 Conclusion and discussion

In this paper, we have proposed a statistical cost-based analysis to determine optimal timeout period for devices. This method takes into account the real usage patterns in order to optimize power consumption. In a first approach, times between requests were supposed independent and the timeout period inferred accordingly. We also proposed three approaches to dynamically re-estimate an optimal timeout period using an HMC to model print events. Finally, a methodology to take into account user impact due to power saving exit transitions is also included. It allows the dynamic timeout period methods to achieve a trade-off between user impact and power consumption, depending on the user's priorities.

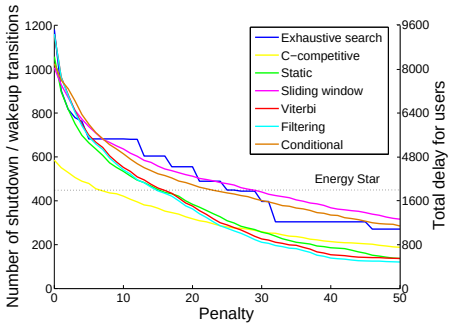


Figure 6: Number of shutdown transitions (left vertical axis) and total delay for users (right vertical axis) as the penalty δ increases.

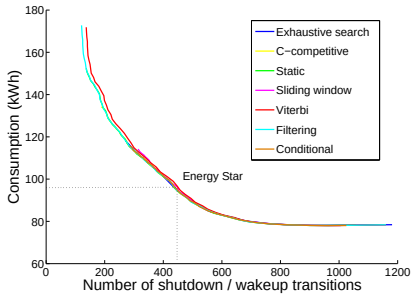


Figure 7: Consumption as a function of the number of shutdown transitions obtained with the different methods.

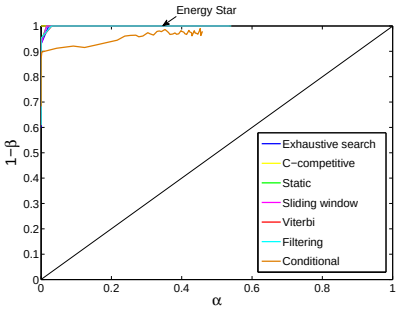


Figure 8: ROC curves obtained by varying the penalty δ. Horizontally: type I error, vertically: 1− type II error.

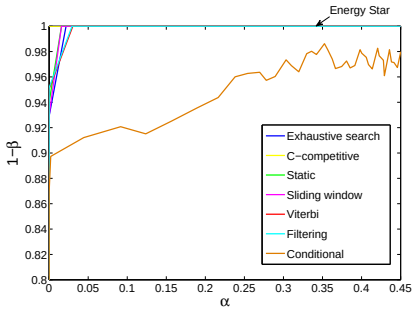


Figure 9: Close-up of upper left part of Figure 8 (ROC curves obtained by varying the penalty δ).

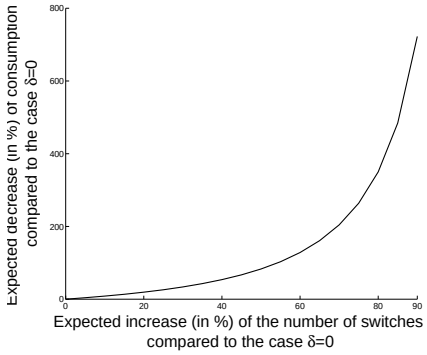


Figure 10: Expected decrease (in %) of the consumption as a function of the expected increase (in %) of the number of transitions. The reference model is a renewal process without penalty.

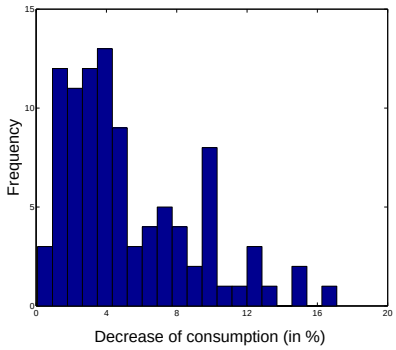


Figure 11: Histogram of consumption gains (on each of the 100 printers) obtained with a number of switches twice larger than the Energy Star standard.

From the application of our methods to two real datasets composed of printing requests, it can be remarked that the static method is the fastest and most accurate method to select the optimal timeout periods. Moreover it is rather simple to set up in practice. Indeed, it only learns the parameters of the distribution of the times between printings by the Maximum likelihood estimation method, and deduces the optimal timeout period according to the estimated value of the printing rate.

The theoretical formulation of power consumption in terms of a print process can be considered as a stepping stone for more complex models that will allow the model to progressively gain completeness in the consideration of other several cost factors, as for example device aging due to increased transitions from power saving mode due to a more dynamic power saving policy. We have also established the foundations to develop in the future a power saving strategy capable of performing accurate prediction of power saving entry as described in this article, but also of optimal power saving exit.

5.1 Real-world implementation

Large device fleets accounting for two or three thousand printers can be managed by dedicated servers in charge of several tasks. For instance, some device management servers (*e.g.* Xerox Device Manager, Xerox CentreWare Web) include status retrieval for monitoring purposes, page meter and job accounting retrieval for billing purposes, configuration checking for security and compliance purposes... One can propose to extend these capabilities of device management software by performing the timeout optimization on the servers dedicated to these tasks. The typical power consumption of this type of servers ranges from 140 Watts to 400 Watts. The algorithm production execution environment will be different from Matlab and probably implemented in .NET or Java environments. In the case of Xerox, the production prototype has implemented the static method in .NET using a high performance mathematical library. The execution time of the .NET implemented prototype with respect to Matlab is on average 10 times slower. It should be noticed that the prototype implementation has shown that auxiliary tasks as database access, remote device timeout setup or power consumption estimations attached to the execution of the method are notably more processing intensive. To summarize, one run of the static method on a 400 Watts server represents 400 milliseconds of CPU time, *i.e.* $4.4e-5$ kWh which is negligible with respect to the energy consumption of a printer. A real-world experimentation has been conducted on 100 Xerox Phaser 4500 printers where 47,000 print jobs have been collected. On this basis, a predictive model has been built, based on the renewal process approach. Using this model, the probability of shutdown of the printer can be computed, as well the associated consumption and timeout, for any value of the penalty. Figure 10 shows the expected decrease of the consumption as a function of the expected increase of the number of transitions. The reference model is a renewal process without penalty (using the estimated optimal timeout for $\delta = 0$). To illustrate the variability of these gains, their values have been represented on a histogram (Figure 11), corresponding to the gain on each of the 100 printers. Consequently, the users can now specify which increase (in %) in the number of shutdowns they are ready to accept (or equivalently which increase of an upper bound of the time they are ready to wait). Then the model deduces the corresponding penalty, timeout and consumption.

5.2 Extensions

Refinements of our approach may include the use of covariates (*e.g.* hour, day of the week), to adapt the timeout period to different conditions identified *a priori* (contrary to HMCs where the conditions can be interpreted *a posteriori*). These conditions may be related to the print

process, as well as the penalty associated with user impact. The transition delays can also be used as covariates to model user impact.

Instead of a continuous-time framework where the decision corresponds to the value of a timeout period, a discrete-time framework could be considered. After each time step, the decision would consist in switching or not the printer into sleep mode. This is the approach chosen in [26]. The authors claim that the timeout-based policies "not only waste power during the periods of inactivity, but also needlessly annoy the user when they turn off components at inappropriate times". This is only true when the times of future requests are considered as known (oracle point of view). In the case of unknown times of future requests, the power used to maintain a device into idle mode cannot be considered *a priori* as wasted. However, some generalization of our classical timeout-based approach could be achieved, with potentially better results, by replacing the assumption of a deterministic timeout period by a stochastic one, whose optimal distribution would have to be determined.

A further extension of this work is the challenging issue of optimal redirection of print jobs and power saving policy within a network of printers managed by a server. Given a printing request, this consists in determining on which printer the job has to be processed, and after what delay each printer has to be turned into sleep mode, so as to minimize the global consumption. Modeling this problem should take into account constraints due to user impact, that are partially related to network connectivity.

Finally, our approach deals separately with model identification (parameter estimation from trajectories of user requests) and computation of the optimal timeout periods (in a framework with fixed parameters). As an alternative, a unified model for handling both model identification and decision taking would be provided by the Bayesian Partially-Observed Markov Decision Processes (POMDPs) in [18]. Here the non-observed part of the MDP would consist in, firstly, the unknown parameters, considered as stochastic in a Bayesian framework, and secondly, potential unknown states as in the HMC models. The benefit of Bayesian POMDPs to our application would come from taking into account simultaneously the different sources of uncertainty: states of the printer and of the user, value of the parameter and of the reward.

A Appendix: Proofs.

Proof of Lemma 1

In view of Section II.A, the consumption between two successive printings is given by

$$\begin{aligned} h(X_i, \tau_i^{(1)}) &= (aX_i)(1 - \mathbb{1}_{\{X_i > \tau_i^{(1)}\}}) \\ &+ (a\tau_i^{(1)} + c_1 + b_1(X_i - \tau_i^{(1)}) + d_1)\mathbb{1}_{\{X_i > \tau_i^{(1)}\}}. \end{aligned} \tag{7}$$

Introducing $\Delta t_1 = (c_1 + d_1)/(a - b_1)$, the consumption can be rewritten as

$$\begin{aligned} h(X_i, \tau_i^{(1)}) &= aX_i \\ &+ (a - b_1)(\Delta t_1 + \tau_i^{(1)} - X_i)\mathbb{1}_{\{X_i > \tau_i^{(1)}\}}. \end{aligned}$$

As a consequence, the expected consumption between two successive printings given $X_{1:i-1}$ is

$$\begin{aligned}\mathbb{E}(h(X_i,\tau_i^{(1)})|X_{1:i-1}) &= a\mathbb{E}(X_i|X_{1:i-1})\\ &+ (a-b_1)\bar{F}_{X_i|X_{1:i-1}}(\tau_i^{(1)})(\Delta t_1+\tau_i^{(1)})\\ &- (a-b_1)\int_{\tau_i^{(1)}}^{+\infty} x f_{X_i|X_{1:i-1}}(x)dx.\end{aligned}$$

and the result is proved. ■

Proof of Proposition 1

Differentiating the above expected consumption with respect to $\tau_i^{(1)}$ yields

$$\begin{aligned}&\frac{d\mathbb{E}(h(X_i,\tau_i^{(1)})|X_{1:i-1})}{d\tau_i^{(1)}} =\\ &(a-b_1)\bar{F}_{X_i|X_{1:i-1}}(\tau_i^{(1)})(1-\Delta t_1 z_{X_i|X_{1:i-1}}(\tau_i^{(1)})).\end{aligned} \tag{8}$$

Let us recall that $a > b_1$. Two main cases are considered:
(i) Suppose that $z_{X_i|X_{1:i-1}}$ is decreasing. Three situations occur:
- If $1/\Delta t_1 < \lim_{x\to+\infty} z_{X_i|X_{1:i-1}}(x)$, then the derivative (8) is negative, the expected consumption is a decreasing function of $\tau_i^{(1)}$ and thus $\hat{\tau}_i^{(1)} = +\infty$.
- If $\lim_{x\to+\infty} z_{X_i|X_{1:i-1}}(x) \leq 1/\Delta t_1 \leq z_{X_i|X_{1:i-1}}(0)$, then the following equation

$$z_{X_i|X_{1:i-1}}(\tau_i^{(1)}) = 1/\Delta t_1 \tag{9}$$

has an unique root $\tau_i^{(1)}$ in $(0,+\infty)$ which is the unique minimum of the expected consumption.
- Finally, if $z_{X_i|X_{1:i-1}}(0) < 1/\Delta t_1$, then the derivative (8) is positive, the expected consumption is an increasing function of $\tau_i^{(1)}$ and thus $\hat{\tau}_i^{(1)} = 0$.
(ii) Suppose that $z_{X_i|X_{1:i-1}}$ is increasing or constant. Three situations occur:
- If $1/\Delta t_1 \leq z_{X_i|X_{1:i-1}}(0)$, then the derivative (8) is non-positive, the expected consumption is a non-increasing function of $\tau_i^{(1)}$ and thus $\hat{\tau}_i^{(1)} = +\infty$.
- If $z_{X_i|X_{1:i-1}}(0) < 1/\Delta t_1 < \lim_{x\to+\infty} z_{X_i|X_{1:i-1}}(x)$ then equation (9) has an unique root in $(0,+\infty)$ and the expected consumption is a concave function of $\tau_i^{(1)}$. As a consequence, $\hat{\tau}_i^{(1)} = 0$ if $\mathbb{E}\left(h(X_i,0)|X_{1:i-1}\right) < \lim_{x\to\infty} \mathbb{E}\left(h(X_i,x)|X_{1:i-1}\right)$ and $\hat{\tau}_i^{(1)} = +\infty$ otherwise. Since

$$\begin{aligned}&\mathbb{E}\left(h(X_i,0)|X_{1:i-1}\right) - \lim_{x\to\infty} \mathbb{E}\left(h(X_i,x)|X_{1:i-1}\right)\\ &= (a-b_1)(\Delta t_1 - \mathbb{E}(X_i|X_{1:i-1})),\end{aligned}$$

the conclusion follows.
- Finally, if $\lim_{x\to+\infty} z_{X_i|X_{1:i-1}}(x) \leq 1/\Delta t_1$, then the derivative (8) is non-negative, the expected consumption is a non-decreasing function of $\tau_i^{(1)}$ and thus $\hat{\tau}_i^{(1)} = 0$. ■

Proof of Lemma 2. As a consequence from the equations in Section 2.2, the consumption h between two successive print requests is given by:

$$\begin{aligned}
h(X_i, \tau_i^{(1)}, \ldots, \tau_i^{(m)}) &= aX_i \mathbb{1}_{\{X_i \leq \tau_i^{(1)}\}} \\
&+ \textstyle\sum_{r=1}^{m-1} \left(a\tau_i^{(1)} + \sum_{j=1}^{r-1} \left(c_j + b_j(\tau_i^{(j+1)} - \tau_i^{(j)}) \right) + c_r + b_r(X_i - \tau_i^{(r)}) + d_r \right) \mathbb{1}_{\{\tau_i^{(r)} < X_i \leq \tau_i^{(r+1)}\}} \\
&+ \left(a\tau_i^{(1)} + \textstyle\sum_{j=1}^{m-1} \left(c_j + b_j(\tau_i^{(j+1)} - \tau_i^{(j)}) \right) + c_m + b_m(X_i - \tau_i^{(m)}) + d_m \right) \mathbb{1}_{\{X_i > \tau_i^{(m)}\}}. \quad (10)
\end{aligned}$$

Letting $a = b_0$ yields

$$\begin{aligned}
&\mathbb{E}\left(h(X_i, \tau_i^{(1)}, \ldots, \tau_i^{(m)}) | X_{1:i-1} \right) \\
&= b_0 \mathbb{E}\left(X_i \mathbb{1}_{\{X_i < \tau_i^{(1)}\}} | X_{1:i-1} \right) + \sum_{r=1}^{m-1} b_r \mathbb{E}\left(X_i \mathbb{1}_{\{\tau_i^{(r)} < X_i \leq \tau_i^{(r+1)}\}} | X_{1:i-1} \right) + b_m \mathbb{E}\left(X_i \mathbb{1}_{\{X_i > \tau_i^{(m)}\}} | X_{1:i-1} \right) \\
&\quad + \sum_{r=1}^{m-1} \mathbb{P}\left(\tau_i^{(r)} < X_i \leq \tau_i^{(r+1)} | X_{1:i-1} \right) \left(\sum_{j=1}^{r} \left(\tau_i^{(j)}(b_{j-1} - b_j) + c_j \right) + d_r \right) \\
&\quad + \mathbb{P}\left(X_i > \tau_i^{(m)} | X_{1:i-1} \right) \left(\sum_{j=1}^{m} \left(\tau_i^{(j)}(b_{j-1} - b_j) + c_j \right) + d_m \right).
\end{aligned}$$

Taking account of

$$\begin{aligned}
\mathbb{E}\left(X_i \mathbb{1}_{\{\tau_i^{(r)} < X_i \leq \tau_i^{(r+1)}\}} | X_{1:i-1} \right) &= \mathbb{E}\left(X_i \mathbb{1}_{\{X_i > \tau_i^{(r)}\}} | X_{1:i-1} \right) - \mathbb{E}\left(X_i \mathbb{1}_{\{X_i > \tau_i^{(r+1)}\}} | X_{1:i-1} \right), \\
\mathbb{E}\left(X_i \mathbb{1}_{\{X_i < \tau_i^{(1)}\}} | X_{1:i-1} \right) &= \mathbb{E}\left(X_i | X_{1:i-1} \right) - \mathbb{E}\left(X_i \mathbb{1}_{\{X_i > \tau_i^{(1)}\}} | X_{1:i-1} \right)
\end{aligned}$$

the expected consumption can be rewritten as

$$\begin{aligned}
&\mathbb{E}\left(h(X_i, \tau_i^{(1)}, \ldots, \tau_i^{(m)}) | X_{1:i-1} \right) \\
&= -\sum_{j=1}^{m} (b_{j-1} - b_j) \mathbb{E}\left(X_i \mathbb{1}_{\{X_i > \tau_i^{(j)}\}} | X_{1:i-1} \right) + \left(1 - F_{X_i | X_{1:i-1}}(\tau_i^{(m)}) \right) \left(\sum_{j=1}^{m} \left(\tau_i^{(j)}(b_{j-1} - b_j) + c_j \right) + d_m \right) \\
&\quad + \sum_{r=1}^{m-1} \left(\left(F_{X_i | X_{1:i-1}}(\tau_i^{(r+1)}) - F_{X_i | X_{1:i-1}}(\tau_i^{(r)}) \right) \left(\sum_{j=1}^{r} \left(\tau_i^{(j)}(b_{j-1} - b_j) + c_j \right) + d_r \right) \right) + b_0 \mathbb{E}(X_i | X_{1:i-1}).
\end{aligned}$$

Splitting the second right-hand term into two parts yields

$$\begin{aligned}
&\mathbb{E}\left(h(X_i, \tau_i^{(1)}, \ldots, \tau_i^{(m)}) | X_{1:i-1} \right) \\
&= -\sum_{j=1}^{m} (b_{j-1} - b_j) \mathbb{E}\left(X_i \mathbb{1}_{\{X_i > \tau_i^{(j)}\}} | X_{1:i-1} \right) + \sum_{j=1}^{m} \left(\tau_i^{(j)}(b_{j-1} - b_j) + c_j \right) + d_m \\
&\quad + b_0 \mathbb{E}(X_i | X_{1:i-1}) + \sum_{r=2}^{m} F_{X_i | X_{1:i-1}}(\tau_i^{(r)}) \left(\sum_{j=1}^{r-1} \left(\tau_i^{(j)}(b_{j-1} - b_j) + c_j \right) + d_{r-1} \right) \\
&\quad - \sum_{r=1}^{m} F_{X_i | X_{1:i-1}}(\tau_i^{(r)}) \left(\sum_{j=1}^{r} \left(\tau_i^{(j)}(b_{j-1} - b_j) + c_j \right) + d_r \right)
\end{aligned}$$

and collecting the two last right-hand terms we obtain

$$\begin{aligned}
&\mathbb{E}\left(h(X_i,\tau_i^{(1)},\ldots,\tau_i^{(m)})|X_{1:i-1}\right)\\
&=-\sum_{j=1}^{m}(b_{j-1}-b_j)\mathbb{E}\left(X_i\mathbb{1}_{\{X_i>\tau_i^{(j)}\}}|X_{1:i-1}\right)+b_0\mathbb{E}(X_i|X_{1:i-1})+\left(\sum_{j=1}^{m}\left(\tau_i^{(j)}(b_{j-1}-b_j)+c_j\right)+d_m\right)\\
&\quad-F_{X_i|X_{1:i-1}}(\tau_i^{(1)})\left(\tau_i^{(1)}(b_0-b_1)+d_1\right)-\sum_{j=2}^{m}F_{X_i|X_{1:i-1}}(\tau_i^{(j)})\left(\tau_i^{(j)}(b_{j-1}-b_j)+c_j+d_j-d_{j-1}\right).
\end{aligned}$$

Finally, letting $\Delta t_j=(c_j+d_j-d_{j-1})/(b_{j-1}-b_j)$ with the convention $d_0=0$, we have

$$\begin{aligned}
&\mathbb{E}\left(h(X_i,\tau_i^{(1)},\ldots,\tau_i^{(m)})|X_{1:i-1}\right)\\
&=-\sum_{j=1}^{m}(b_{j-1}-b_j)\mathbb{E}\left(X_i\mathbb{1}_{\{X_i>\tau_i^{(j)}\}}|X_{1:i-1}\right)+\sum_{j=1}^{m}(\tau_i^{(j)}+\Delta t_j)(b_{j-1}-b_j)\\
&\quad-\sum_{j=1}^{m}(b_{j-1}-b_j)\,F_{X_i|X_{1:i-1}}(\tau_i^{(r)})(\tau_i^{(j)}-\Delta t_j)+b_0\mathbb{E}(X_i|X_{1:i-1})\\
&=\sum_{j=1}^{m}(b_{j-1}-b_j)\left[\left(1-F_{X_i|X_{1:i-1}}(\tau_i^{(j)})\right)\left(\Delta t_j+\tau_i^{(j)}\right)-\int_{\tau_i^{(j)}}^{+\infty}xf_{X_i|X_{1:i-1}}(x)dx\right]\\
&\quad+b_0\mathbb{E}(X_i|X_{1:i-1}),
\end{aligned}\tag{11}$$

and the conclusion follows. ■

Proof of Proposition 2 is quite similar to that of Proposition 1, and thus is omitted in this report. This is also the case for lemma 3, which proof is similar to that of lemma 1.

B Appendix: M step of the EM algorithm for Weibull HMCs.

This paragraph describes the M step of EM algorithm, dedicated to parameter re-estimation in HMCs, in the case of Weibull emission distributions.
It can be seen from equations (3) and (2), that the re-estimation procedure for the π_k and $A_{k,l}$ parameters is not specific to Weibull distributions. Consequently the usual formulae (6.14) and (6.15) in [9] hold. For all $k=1,..,K$ the new values of parameters $(\lambda_k^{(m+1)},\alpha_k^{(m+1)})$ after m iterations of the EM algorithm cancel the partial derivatives of the $\mathbb{Q}$ function, and thus satisfy the system:

$$\begin{cases}\sum_t \mathbb{P}_{\eta^{(m)}}(S_t=k|X_1^n=x_1^n)\dfrac{\partial}{\partial\lambda_k}\log f_{\lambda_k,\alpha_k}(x)=0\\ \sum_t \mathbb{P}_{\eta^{(m)}}(S_t=k|X_1^n=x_1^n)\dfrac{\partial}{\partial\alpha_k}\log f_{\lambda_k,\alpha_k}(x)=0.\end{cases}\tag{12}$$

Let $\xi_k^{(t)}=\mathbb{P}_{\eta^{(m)}}(S_t=k|X_1^n=x_1^n)$. Since for Weibull emission distributions,

$$\log f_{\lambda_k,\alpha_k}(x)=\log(\alpha_k)+\alpha_k\log(\lambda_k)+(\alpha_k-1)\log(x)-(\lambda_k x)^{\alpha_k},$$

we have

$$\frac{\partial\log f_{\lambda_k,\alpha_k}(x)}{\partial\lambda_k}=\frac{\alpha_k}{\lambda_k}-\alpha_k x^{\alpha_k}\lambda_k^{\alpha_k-1}$$

and

$$\frac{\partial \log f_{\lambda_k,\alpha_k}(x)}{\partial \alpha_k} = \frac{1}{\alpha_k} + \log(\lambda_k) + \log x - (\log(\lambda_k x))(\lambda_k x)^{\alpha_k}.$$

The first equation of the system (12) can be rewritten as

$$\begin{aligned}
&\sum_t \xi_k^{(t)} \frac{\partial}{\partial \lambda_k} \log f_{\lambda_k,\alpha_k}(x_t) = \sum_t \xi_k^{(t)} \left[\frac{\alpha_k}{\lambda_k} - \alpha_k x_t^{\alpha_k} \lambda_k^{\alpha_k - 1} \right] = 0 \\
&\quad \Leftrightarrow \frac{\alpha_k}{\lambda_k} \sum_t \xi_k^{(t)} - \alpha_k \lambda_k^{\alpha_k - 1} \sum_t \xi_k^{(t)} x_t^{\alpha_k} = 0 \Leftrightarrow \sum_t \xi_k^{(t)} = \lambda_k^{\alpha_k} \sum_t \xi_k^{(t)} x_t^{\alpha_k} \qquad (13) \\
&\quad \Leftrightarrow \lambda_k = \left[\frac{\sum_t \xi_k^{(t)} x_t^{\alpha_k}}{\sum_t \xi_k^{(t)}} \right]^{-\frac{1}{\alpha_k}}. \qquad (14)
\end{aligned}$$

Replacing the expression of λ_k obtained in equation (14) into the second equation of the system (12) yields

$$\begin{aligned}
0 = \sum_t \xi_k^{(t)} \frac{\partial}{\partial \alpha_k} \log f_{\lambda_k,\alpha_k}(x_t) &= \sum_t \xi_k^{(t)} \left[\frac{1}{\alpha_k} + \log \lambda_k + \log x_t - (\log(\lambda_k x_t))(\lambda_k x_t)^{\alpha_k} \right] \\
&= \sum_t \xi_k^{(t)} \left(\frac{1}{\alpha_k} + \log x_t + \log \lambda_k \right) - \lambda_k^{\alpha_k} \sum_t \xi_k^{(t)} x_t^{\alpha_k} (\log \lambda_k + \log x_t) \\
&= \sum_t \xi_k^{(t)} \left(\frac{1}{\alpha_k} + \log x_t \right) - \lambda_k^{\alpha_k} \sum_t \xi_k^{(t)} x_t^{\alpha_k} \log x_t \\
&\quad + \log \lambda_k \left[\sum_t \xi_k^{(t)} - \lambda_k^{\alpha_k} \sum_t \xi_k^{(t)} {x_t}^{\alpha_k} \right]. \qquad (15)
\end{aligned}$$

Using equations (13) and (14) in (15) yields

$$\begin{aligned}
0 &= \sum_t \xi_k^{(t)} \log x_t - \left[\frac{\sum_t \xi_k^{(t)}}{\sum_t \xi_k^{(t)} x_t^{\alpha_k}} \right] \sum_t \xi_k^{(t)} x_t^{\alpha_k} \log x_t + \frac{1}{\alpha_k} \sum_t \xi_k^{(t)} \\
\Leftrightarrow \quad 0 &= \alpha_k \left[\frac{\sum_t \xi_k^{(t)} \log x_t}{\sum_t \xi_k^{(t)}} - \frac{\sum_t \xi_k^{(t)} x_t^{\alpha_k} \log x_t}{\sum_t \xi_k^{(t)} x_t^{\alpha_k}} \right] + 1. \qquad (16)
\end{aligned}$$

Equation (16) has no known solution; hence it has to be solved numerically, by the algorithm described in [10] in the circumstances.

C Appendix: Markov decision processes

In this Section, a connection between our approach and the theory of Markov decision processes (MDPs) is established. More specifically, the problem of determining the optimal timeout period by minimizing the expected consumption up to following request, defined by equation (1), is shown to be a particular case of an MDP with a continuous action space, if the times between printings are independent random variables. The value function with (finite) horizon 1 of the corresponding MDP is shown to be the opposite of the expected future cost. Moreover, this MDP has a single possible state, which explains why an explicit solution of this problem could be derived in Section 2.1.

C.1 General principle

Markov decision processes are a class of optimization problems for controlling the temporal evolution of an agent in a given environment characterized by a set of states $\mathcal{S}$. At each time step t, given the state S_t of the environment, the agent is allowed to perform an action A_t chosen from a set $\mathcal{A}$. The chosen action may modify the next state S_{t+1} of the environment, and brings a scalar reward R_{t+1} to the agent. All the quantities A_t, S_t and R_t constitute a homogeneous random process. The problem is to determine the distribution for A_t that maximizes the expected future rewards given the current state S_t.

The process $(A_t, S_t, R_t)_{t\in\mathbb{N}}$ is supposed to obey the *Markov property*. Moreover, under the three following assumptions:

1. the action A_{t+1} is independent on the past of the three processes up to time t, and on R_{t+1}, given state S_{t+1};
2. the reward R_{t+1} is independent on the past of the three processes up to time t given the states S_t and S_{t+1}, and given A_t;
3. S_{t+1} is independent on the past of the three processes up to time t given S_t and A_t;

an MDP is totally specified by the following distributions:

- the transition probabilities $\mathcal{P}^a_{ss'} = \mathbb{P}(S_{t+1} = s'|S_t = s, A_t = a)$, which define how next state is affected by current state s and the chosen action a;
- the policy function $\pi(s, a) = \mathbb{P}(A_t = a|S_t = s)$, which defines what action to choose given current state s;
- the reward distribution, *i.e.* the distribution of R_{t+1} given $S_t = s$, $A_t = a$ and $S_{t+1} = s'$.

The optimization problem associated with this MDP consists in finding the policy $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$ that maximizes the expected future rewards (under the constraints $\sum_a \pi(s, a) = 1$ and $\forall(a, s), \pi(a, s) \geq 0$). The future rewards are modelled through the random variable $\mathfrak{R}_t = \sum_{k=0}^{\infty} \gamma_k R_{t+k+1}$, where $\forall k$, γ_k represents the weight of the reward after $k + 1$ time steps. The sequence $(\gamma_k)_{k\in\mathbb{N}}$ is referred to the *discount* sequence. The function to be maximized is called the *value function* and is denoted by V^π; it corresponds to the expectation of $\mathfrak{R}_t$ given the value s of current state. This leads to the following formal definitions:

$$V^\pi(s) = \mathbb{E}(\mathfrak{R}_t|S_t = s) = \sum_{k=0}^{\infty} \gamma_k \mathbb{E}(R_{t+k+1}|S_t = s) \tag{17}$$

$$\text{and } \hat{\pi}(s, .) = \arg\max_{\pi(s,.)} V^\pi(s). \tag{18}$$

The reward distribution $\mathbb{P}(R_{t+1}|S_t = s, A_t = a, S_{t+1} = s')$ is only involved through its expectation $\mathcal{R}^a_{ss'} = \mathbb{E}(R_{t+1}|S_t = s, A_t = a, S_{t+1} = s')$; consequently, only $\mathcal{R}^a_{ss'}$ needs to be defined explicitly.

There are two particular cases of interest for the sequence $(\gamma_k)_{k\in\mathbb{N}}$:

- $\forall k$, $\gamma_k = \gamma^k$, where $0 \leq \gamma < 1$. In this case, $V^\pi(s)$ satisfies a fixed point equation known as the *Bellman equation*. Generally, no closed form is available for the optimal policy.

- $\forall k,\ \gamma_k = \delta_0(k) = \begin{cases} 1 & \text{if } k = 0 \\ 0 & \text{otherwise} \end{cases}$, where δ denotes the Kronecker symbol. Then, it is easily shown that:

$$V^\pi(s) = \sum_a \pi(s,a) \sum_{s'} \mathcal{P}^a_{ss'} \mathcal{R}^a_{ss'}. \tag{19}$$

Lemma 4 *In the case where the sequence $(\gamma_k)_{k\in\mathbb{N}}$ is defined by $\gamma_k = \delta_0(k)$, the optimal policy is:*

$$\hat{\pi}(s,a') = \begin{cases} 1 & \text{if } a' = \arg\max_a \sum_{s'} \mathcal{R}^a_{ss'} \mathcal{P}^a_{ss'} \\ 0 & \text{otherwise.} \end{cases}$$

In the case where the state space is reduced to a singleton $\mathcal{S} = \{1\}$ and where $\forall k,\ \gamma_k = \gamma^k$, the optimal policy is:

$$\hat{\pi}(1,a') = \begin{cases} 1 & \text{if } a' = \arg\max_a \mathcal{R}^a_{1,1} \\ 0 & \text{otherwise.} \end{cases}$$

Proof of Lemma 4. In the case where $\gamma_k = \begin{cases} 1 & \text{if } k = 0 \\ 0 & \text{else} \end{cases}$, then

$$\begin{aligned} V^\pi(s) &= \mathbb{E}(R_{t+1}|S_t = s) \\ &= \sum_a \pi(s,a) \sum_{s'} \mathbb{E}(R_{t+1}|S_t = s, S_{t+1} = s', A_t = a) \mathbb{P}(S_{t+1} = s'|S_t = s, A_t = a) \\ &= \sum_a \pi(s,a) \sum_{s'} \mathcal{R}^a_{ss'} \mathcal{P}^a_{ss'}. \end{aligned} \tag{20}$$

The optimal policy is the policy π maximizing the value function V^π:

$$\hat{\pi} = \arg\max_\pi \left(\sum_a \pi(s,a) \sum_{s'} \mathcal{R}^a_{ss'} \mathcal{P}^a_{ss'} \right)$$

with $\sum_a \pi(s,a) = 1$ and $\pi(s,a) \geq 0\ \forall (s,a)$.

In the case where the state space is reduced to a singleton $\mathcal{S} = \{1\}$, and if $\gamma_k = \gamma^k \quad \forall k$, the optimal policy is, from Bellman equation (see [25]):

$$\begin{aligned} V^\pi(1) &= \sum_a \pi(s,a) \sum_{s'} \mathcal{P}^a_{ss'} \left[\mathcal{R}^a_{ss'} + \gamma V^\pi(s') \right] \\ &= \sum_a \pi(1,a) \mathcal{R}^a_{1,1} + \gamma V^\pi(1) \sum_a \pi(1,a) \end{aligned}$$

since $\mathcal{S} = \{1\}$ and $\forall a,\ \mathcal{P}^a_{ss'} = 1$. Remarking that $\sum_a \pi(1,a) = 1$, we have

$$\begin{aligned} (1-\gamma) V^\pi(1) &= \sum_a \pi(1,a) \mathcal{R}^a_{1,1} \\ \Leftrightarrow \quad V^\pi(1) &= \frac{1}{(1-\gamma)} \sum_a \pi(1,a) \mathcal{R}^a_{1,1}. \end{aligned} \tag{21}$$

In both equations (20) and (21), for each s, $V^\pi(s)$ is a linear function with respect to $\pi(s,a)$. Thus, a classical result of the optimization theory states that the maximum of $V^\pi(s)$ is achieved on an endpoint of an edge of the simplex $\left\{\pi(s,a)|a\in\mathbb{R}, \pi(s,a)\geq 0 \text{ and } \sum_a \pi(s,a)=1\right\}$.
As a consequence, in the case where the sequence $(\gamma_k)_{k\in\mathbb{N}}$ is defined by $\gamma_k=\delta_0(k)$, the optimal policy is:

$$\hat{\pi}(s,a')=\begin{cases}1 & \text{if } a'=\arg\max_a \sum_{s'}\mathcal{R}^a_{ss'}\mathcal{P}^a_{ss'}\\ 0 & \text{otherwise.}\end{cases}$$

In the case where the state space is reduced to a singleton $\mathcal{S}=\{1\}$ and where $\forall k$, $\gamma_k=\gamma^k$, the optimal policy is:

$$\hat{\pi}(1,a')=\begin{cases}1 & \text{if } a'=\arg\max_a \mathcal{R}^a_{1,1}\\ 0 & \text{otherwise.}\end{cases}$$

■

Thus, the optimal policy corresponds to a deterministic strategy. Given current state s, the chosen action systematically is the one that maximizes $\sum_{s'}\mathcal{R}^a_{ss'}\mathcal{P}^a_{ss'}$ or $\mathcal{R}^a_{1,1}$ (depending on the sequence $(\gamma_k)_{k\in\mathbb{N}}$), with respect to a.

C.2 Connection of our approach with MDPs

In this paragraph, our approach is shown to be a particular case of MDP. The proof is derived in the case of one single sleep mode printer for the sake of simplicity, but can easily be extended to an arbitrary number of sleep modes.

In our context, the set of actions for the printer is the timeout period $a\in\mathcal{A}=\mathbb{R}^+$, which corresponds to $\tau^{(1)}$ in previous paragraphs. The decision is taken after each print job, after which the printer is necessarily in *idle* mode. Consequently, the state space is $\mathcal{S}=\{idle\}$. In our problem, the reward is minus the cost between two successive printings. It only depends on the time between printings X_i and on the action a, *i.e.* the timeout period.

The time index $i\in\mathbb{N}$ represents the number of past printings requests. Hence, even if the times of requests T_i take continuous values, the MDP is essentially a discrete-time problem, where decisions are taken after each print job only.

Let the expected reward be defined as $\mathcal{R}^a=-\mathbb{E}(h(X_i,a))$, where h is the cost between two successive print jobs defined by equation (7) (or by equation (10) in the case of multiple sleep modes). The transition probabilities are $\mathcal{P}^a_{ss'}=1,\ \forall\ a$, s and s', since the printer is always in *idle* state when a decision is taken.
As a consequence, from equation (19), the value function $V^\pi(s)$ for $\gamma_k=\delta_0(k)$ is:

$$V^\pi(s)=-\sum_a \pi(s,a)\mathbb{E}(h(X_i,a)),$$

and according to Lemma 4, the optimal policy is:

$$\pi(s,a')=\begin{cases}1 & \text{if } a'=\arg\max_a -\mathbb{E}(h(X_i,a))=\arg\min_a \mathbb{E}(h(X_i,a))\\ 0 & \text{otherwise,}\end{cases}$$

which corresponds to the optimization problem in equation (1) in the case where the times between printings X_i are independent random variables.

To conclude, our approach is a degenerate case of an MDP problem with a continuous action space and with one single state. Using the particular discount sequence $\gamma_k = \delta_0(k)$, the expected future cost coincides with the value function of the MDP with (finite) horizon 1. Since the state space is reduced to a single state, an explicit solution of this problem can be derived. This solution is given in Proposition 1.

Acknowledgment

The authors are grateful to Guillaume Bouchard for his significant contribution to the MDP modeling and for fruitful discussions. The authors are indebted to Christopher Dance for his helpful comments and suggestions.

References

[1] R.E. Barlow and F. Proschan. *Statistical theory of reliability and life testing; Probability models*. To Begin With, Silver Spring, 1981.

[2] L. Benini, A. Bogliolo, G. Paleologo, and G. De Micheli. Policy Optimization for Dynamic Power Management. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 18(6):813–833, June 1999.

[3] A. Bogliolo, L. Benini, E. Lattanzi, and G. De Micheli. Specification and Analysis of Power–Managed systems. *Proceedings of the IEEE*, 92(8):1308–1346, August 2004.

[4] Le Cai and Yung-Hsiang Lu. Joint power management of memory and disk. In *DATE '05: Proceedings of the conference on Design, automation and test in Europe*, pages 86–91, Washington, DC, USA, 2005. IEEE Computer Society.

[5] G. Celeux and J.-B. Durand. Selecting Hidden Markov Model State Number with Cross-Validated Likelihood. *Computational Statistics*, 23:541–564, 2008.

[6] E.-Y. Chung, L. Benini, A. Bogliolo, Y.-H. Lu, and G. De Micheli. Dynamic Power Management for Nonstationary Service Requests. *IEEE Transactions on Computers*, 11(51):1345–1361, November 2002.

[7] E.-Y. Chung, L. Benini, and G. De Micheli. Dynamic Power Management Using Adaptive Learning Tree. In *International Conference on Computer-Aided Design (ICCAD '99)*, pages 274–279, 1999.

[8] F. Douglis, P. Krishnan, and B. Bershad. Adaptive disk spin-down policies for mobile computers. In *Proc. 2nd USENIX Symp. on Mobile and Location-Independent Computing*, 1995.

[9] Y. Ephraim and N. Merhav. Hidden Markov processes. *IEEE Transactions on Information Theory*, 48:1518–1569, June 2002.

[10] G. E. Forsythe, M. A. Malcolm, and C. B. Moler. *Computer Methods for Mathematical Computations*. Prentice-Hall, 1976.

[11] E. Gassiat. Likelihood ratio inequalities with application to various mixtures. *Annales de l'Institut Henri Poincaré*, 38:897–906, 2002.

[12] R. Golding, P. Bosch, C. Staelin, T. Sullivan, and J. Wilkes. Idleness is not sloth. In *TCON'95: Proceedings of the USENIX 1995 Technical Conference Proceedings on USENIX 1995 Technical Conference Proceedings*, pages 17–17, Berkeley, CA, USA, 1995. USENIX Association.

[13] J. A. Hanley and B. J. Mcneil. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143(1):29–36, April 1982.

[14] C.-H. Hwang and A. C. Wu. A predictive system shutdown method for energy saving of event-driven computation. *ACM Trans. Des. Autom. Electron. Syst.*, 5:pages 226–241, 2000.

[15] N. Johnson, S. Kotz, and N. Balakrishnan. *Continuous Univariate Distributions, 2nd edition, vol.1.* Wiley Series in Probability and Statistics, 1995.

[16] A. R. Karlin, M. S. Manasse, L. A. McGeoch, and S. Owicki. Competitive randomized algorithms for nonuniform problems. *Algorithmica*, 11:542–571, 1994.

[17] Y.-H. Lu, E.-Y. Chung, T. Šimunić, L. Benini, and G. De Micheli. Quantitative Comparison of Power Management Algorithms. In *DATE '00: Proceedings of the conference on Design, automation and test in Europe*, pages 20–26. IEEE Computer Society, March 2000.

[18] P. Poupart and N. Vlassis. Model-based Bayesian Reinforcement Learning in Partially Observable Domains. In *International Symposium on Artificial Intelligence and Mathematics (ISAIM), Fort Lauderdale, Florida, USA*, 2008.

[19] Q. Qiu and M. Pedram. Dynamic Power Management Based on Continuous-Time Markov Decision Processes. In *DAC '99: Proceedings of the 36th ACM/IEEE conference on Design Automation, New Orleans, Louisiana (USA)*, pages 555–561, New York, NY, USA, 1999. ACM.

[20] M. Rausand and A. Høyland. *System Reliability Theory: Models, Statistical Methods, and Applications, 2nd Edition.* Wiley–Interscience, 2004.

[21] Z. Ren, B. Krogh, and R. Marculescu. Hierarchical adaptive dynamic power management. *IEEE Transactions on Computers*, 54(4):409–420, 2005.

[22] S.L. Scott and P. Smyth. *The Markov modulated Poisson process and Markov Poisson cascade with applications to Web traffic data*, pages 671–680. M. J. Bayarri et al., eds.: Oxford University Press, Oxford, UK, 2003.

[23] T. Šimunić. *Dynamic management of power consumption*, pages 102–125. Graybill, R., Melhem, R., eds.: Power Aware Computing. Kluwer Academic, 2002.

[24] M. B. Srivastava, A. P. Chandrakasan, and R. W. Brodersen. Predictive system shutdown and other architectural techniques for energy efficient programmable computation. *IEEE Trans. Very Large Scale Integr. Syst.*, 4(1):42–55, 1996.

[25] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction.* MIT Press, Cambridge, Massachusetts, 1998.

[26] G. Theocharous, S. Mannor, N. Shah, P. Gandhi, B. Kveton, S. Siddiqi, and C.-H. Yu. Machine Learning for Adaptive Power Management. *Intel Technology Journal*, 10(4):298–311, November 2006.

[27] P. Zhang. Model selection via multifold cross validation. *The Annals of Statistics*, 21(1):299–313, 1993.