Adapting Content Delivery to Observable Resources and Semi-Observable User Interest

Cezar Pleşca, Vincent Charvillat and Romulus Grigoraş

***Abstract*—This paper discusses adaptation policies for information systems that are subject to dynamic and stochastic contexts such as mobile access to multimedia web sites. In our approach adaptation agents apply sequential decisional policies under uncertainty. We focus on the modeling of such decisional processes depending on whether the context is fully or partially observable. Our case study is a movie browsing service in a mobile environment that we model by using Markov Decision Processes (MDP) and Partially Observable MDP (POMDP).We derive adaptation policies for this service that take into account the limited (and observable) resources such as the network bandwidth. We further refine these policies according to the (partially observable) users' interest level estimated from implicit feedback. Our theoretical models are validated through numerous simulations.**

***Index Terms*—adaptation, Markov decision process, partially observable context, user interest, implicit feedback.**

I. Introduction

ACCESS alternatives to computer services continue to progress, facilitating our interaction with family, friends or workplace. These new access alternatives encompass a wide range of mobile and distributed devices that our technological environment becomes truly pervasive. The execution contexts in which these devices operate are naturally heterogeneous. The resources offered by wireless networks vary with the number and the position of connected users. The available memory and the processing power also fluctuate dynamically. Last but not least, the needs and expectations of users can change at any instant. As a consequence, there are numerous research projects that aim to provide modern information systems with adaptation capabilities according to context variability.

In order to handle highly dynamic contexts, the approach that we propose in this paper is based on an adaptation agent. The agent perceives the successive states of the context thanks to observations and carries out adaptation actions. Often, the adaptations mechanisms proposed in literature suppose that the contextual data is easy to perceive or at least that there is no possible ambiguity to identify the state of the current context. One calls this an **observable context**. In this work we relax this hypothesis and therefore deal with **partially observable contexts**.

Our case study is an information system for browsing multimedia descriptions of movies on mobile devices. The key idea is to show how a given adaptation strategy can be refined according to the estimation of user interest. User interest is clearly not directly observable by the system. We build upon research on ”implicit feedback” in order to allow the adaptation agent to estimate the user interest level while interacting with the context [1], [2].

The first section of this paper reviews important elements of the state of the art and details our adaptation approach. Next, we introduce the two formalisms used by our model: the Markov Decision Processes (MDP) and the Partially Observable MDP (POMDP). The following section presents our case study and establishes the operational principles of this information system. Thanks to a MDP, we formalize an adaptation policy for our information system seen as an observable context. Then we show how to refine this policy according to user interest using a POMDP (refined itself from a MDP). Various experiments validate this approach and give a practical view of the behaviour of an adaptation agent. We conclude this paper with some perspectives on this work.

II. Related Work

This section introduces useful current literature in the field of adaptation to dynamic execution contexts which helps to position our adaptation approach. Adaptive systems commonly provide adaptation capabilities and therefore these systems can be categorized according to available resources, user preferences or more generally, to the context.

A. Resource-based Adaptation

Given the heterogeneous nature of modern networks and mobile devices, there is an obvious need for adaptation to limited resources [3]. Networks' QoS parameters vary in terms of available bandwidth, loss rate or latency. The capabilities of the terminal are also very heterogeneous since its memory, processing or display limitations can be used to maintain (or hand over) a service initially offered by a workstation. To manage these limitations, one can adapt the content to be displayed or the access/distribution modalities. When considering content adaptation, several authors propose classifications [4], [5] where the elementary components of the content (a media for example) or the entire document's structure is to be transformed. A media can thus be transcoded [3], [6], converted into another modality [5] or summarized [7]. The distribution or the access can also be adapted, for example, by optimizing the streaming [8] or by modifying the degree of interactivity of the service.

B. User-aware Adaptation

In addition to adaptation capabilities to the available resources, one should also consider an application's adaptation according to human factors which are a matter of user preferences and satisfaction. Henceforth we describe three main research directions as given by the literature.

The first one consists of switching the adaptation mechanisms for maximizing the quality of the service perceived by the user. A typical scenario is the choice of the strategy to transcode a stream (for example, a video stream), for maximizing the perceptual quality given a limited bandwidth [9]. What is the best parameter to adapt: the size of the video, its chromatic resolution or the frame-rate? For this line of research the key factor for consideration is how variation in objective multimedia quality impacts on user perception [10].

An active second direction is related to user modeling. Here, the idea is to customize an application by modeling user profiles in order to recognize them later. For example, adaptive hypermedia contents or services [11], [12] provide a user with navigation support for ”easier/better learning using an on-line educational service” or support for ”more efficient selling on a e-commerce site” according to the user profile. Very often, these systems analyze the access patterns in order to recognize profiles.

The third research direction finds its motivation in the first two. In order to learn a user model or to evaluate the perceptual impact of a content adaptation solution, it is necessary to either explicitly ask users for evaluations or to obtain implicit feedback information. Research aiming to evaluate ”implicit feedback” (IF) is experiencing a growing interest, since it avoids bringing together significant collections of explicit returns (which is intrusive and expensive) [1]. These IF methods are used in particular to decode user reactions in information search systems [2]. The idea is to measure the user interest for a list of query results, in order to adapt the search function. Among the studied implicit feedback signals one can consider are: the total browsing time, the number of clicks, the scrolling interactions and some characteristic sequences of interactions. In our work, we estimate user interest using IF by interpreting interaction sequences [2], [13].

C. *Mixing Resources and User-aware Adaptation*

More general adaptation mechanisms can be obtained by combining resource-based with user-based adaptation. The characteristics of users and resources are mixed to form an adaptation to the *context*. For mobile and pervasive systems, the link between available resources and users starts by taking into account the geo-localization of the user, that can be traced in time and even predicted [14]. The context becomes richer by integrating these elements. In the MPEG-21 DIA (Digital Item Adaptation) standard, the context descriptors group together the user's preferences, the network's and the terminal's capabilities, the authors' recommendations to adapt their multimedia productions, the perceptual characteristics of the user's environment, etc. Given this complexity, the normative works only propose tools simply for describing the running context as a set of carefully chosen and extensible descriptors [15]. This is an approach by metadata that leaves free the conception of adaptation components while authorizing a high level of interoperability [16].

Naturally, the elements of the context vary in time. Therefore one speaks of a dynamic context and, by extension, of a dynamic adaptation. It is important to note that static adaptation to static context elements is possible as well: one can negotiate once for all and always in the same manner the favorite language of a user at the moment of access to a multilingual service. On the contrary, the adaptation algorithm itself and/or its parameters can be dynamically changed according to the context state [17]. Our adaptation approach is compatible with the latter case.

An important element of research in context adaptation is also the distinction between the adaptation decision and its effective implementation [16]. In a pervasive system, one can decide that a document must be transcoded into another format, but some questions still need to be answered. Is a transcoding component available ? Where can it be found ? Should one compose the transcoding service ? In order to find solutions to these questions, many authors propose to use artificial learning techniques to select the right decision and/or the appropriate implementation of adaptation mechanisms (see [18] for a review). In this case, a description of the running context is given as input to a decision-making agent that predicts the best adaptation actions according to what it has previously learned. We extend this idea in line with a reinforcement learning principle.

We model the context dynamics by a Markov Decision Process whose states are completely or partially observable. This approach provides means to find the optimal decision (adaptation action) according to the current context. Next section introduces our MDP-based adaptation approach.

III. Markov Decision Processes - Our Formal Approach

Figure 1 summarizes our adaptation approach, that has been introduced in [19] and is further refined in this article. In this paper, an adaptation strategy for dynamic contexts is applied by an adaptation agent. This agent perceives sequentially, over a discrete temporal axis, the variations of the context through observations.

From its observations, the agent will compute the context state in order to apply an adaptation policy. Such a policy is simply a function that maps context states to adaptation decisions. Therefore the agent acts on the context while deciding an adaptation action: it consumes bandwidth, influences the future user's interactions, increases or reduces the user's interest... It is therefore useful to measure its effect by associating a reward (immediate or delayed) with the adaptation action decided in a given context state. The agent can thus learn from its interaction with the context and perform a ”trial-and-error” learning also called reinforcement learning [20]. It attempts to reinforce the actions resulting in a good accumulation of rewards and, conversely, avoids renewing fruitless decisions. This process actually represents a continuous improvement of its ”decision policy”.

This dynamic adaptation approach is common to frameworks of *sequential decisional policies under uncertainty*. In these frameworks, the uncertainty comes from two sources. On the one hand, the dynamic of the context can be random as a consequence of available resources' variability (for example the bandwidth); on the other hand, the effect of an agent's

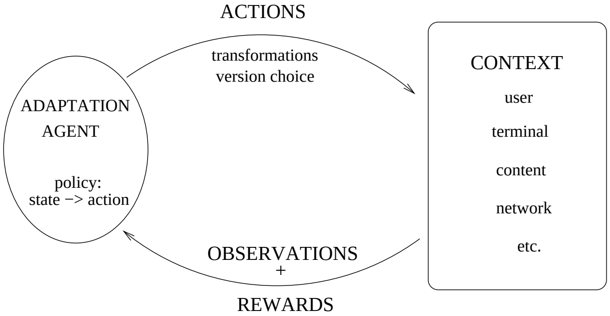


Fig. 1. Context-based adaptation agent

decision can be itself random. For example, if an adaptation action aims to anticipate user interactions, the prediction quality is obviously uncertain and subject to the user's behavior variations.

In this situation, by adopting a Markov definition of the context state, the agent's dynamics can be modeled as a Markov Decision Process (MDP). This section introduces this formalism. We initially assume that context state variables are observable by the agent which makes it a sufficient condition to identify the decision state without any ambiguity. This paper takes a step forward by refining adaptation policies according to user interest. We estimate sequentially this hidden information through user behavior as suggested by research on the evaluation of "implicit feedback". Therefore, the new decision-making state contains at the same time observable variables as well as a hidden element associated with user interest. We then move on from a MDP to a POMDP (Partially Observable Markov Decision Process). To the best of our knowledge, the application of the POMDP to the adaptation problem in partially observable contexts has not been studied before. To give concrete expression to this original idea, a case study will be presented in section IV.

Markov Decision Processes (MDP) are briefly introduced along with their extension for semi-observable contexts, Partially Observable Markov Decision Processes (POMDP).

A. MDP Definition

A MDP is a stochastic controlled process that assigns rewards to transitions between states [21]. It is defined as a quintuple $(S; A; T; p; r_t)$ where S is the state space, A is the action space, T is the discrete temporal axis of instants when actions are taken, $p()$ are the probability distributions of the transitions between states and $r_t()$ is a function of reward on the transitions. We rediscover in a formal way the ingredients necessary to understand the figure 1: at each instant $t \in T$, the agent observes its state $\sigma \in S$, applies the action $a \in A$ that brings the system (randomly, according to $p(\sigma'|\sigma, a)$) to a new state σ', and receives a reward $r_t(\sigma, a)$.

As previously mentioned, we are looking for the best policy with respect to the accumulated rewards. A policy is a function π that associates an action $a \in A$ with each state $\sigma \in S$. Our aim is to find the best one: π^*.

The MDP theoretical framework assigns a *value function* V_π to each policy π. This value function associates each state $\sigma \in S$ with a global reward $V_\pi(\sigma)$, obtained by applying π beginning with σ. Such a value function allows to compare policies. A policy π outperforms another policy π' if

$$\forall \sigma \in S, \quad V_\pi(\sigma) \geq V_{\pi'}(\sigma)$$

The expected sum of rewards is weighted by a parameter γ in order to limit the influence of infinitely distant rewards (especially in the case when S is infinite):

$$\forall \sigma \in S, \quad V_\pi(\sigma) = E\left[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = \sigma\right]$$

In brief, for each state, this value function gives the expected sum of future rewards that can be obtained if the policy π is applied from this state on. This value function allows to formalize the research of the optimal policy π^* which is the one associated with the best value function $V^* = V_{\pi^*}$.

Bellman's optimality equations characterize the optimal value function V^* and an optimal policy π^* that can be obtained from it. In the case of the γ-weighted criterion and stationnary rewards, they can be written:

$$V^*(\sigma) = \max_{a \in A}\left(r(\sigma, a) + \gamma \sum_{\sigma' \in S} p(\sigma'|\sigma, a) V^*(\sigma')\right)$$

$$\forall \sigma \in S, \quad \pi^*(\sigma) = \operatorname*{argmax}_{a \in A}\left(r(\sigma, a) + \gamma \sum_{\sigma' \in S} p(\sigma'|\sigma, a) V^*(\sigma')\right)$$

B. Resolution and Reinforcement Learning

When considering solving an MDP, we can distinguish between two cases, according to whether the model is known or unknown. When the model ($p()$) and the rewards are known, a dynamic programming solution can be found.

The operator L verifying $V_{n+1} = L.V_n$ according to

$$V_{n+1}(\sigma) = \max_{a}\left(r(\sigma, a) + \gamma \sum_{\sigma'} p(\sigma'|\sigma, a) V_n(\sigma')\right)$$

is a contraction. Therefore the Bellman equation in $V^*(\sigma)$ can be solved by using a fixed point iterative method while choosing randomly V_0, then applying repeatedly the operator L that improves the current policy associated to V_n. If the rewards are bounded, the sequence converges to V^* and allows to compute π^*.

If the model (the probabilities) is unknown, we can solve the MDP using a reinforcement learning algorithm [20]. The reinforcement learning approach aims to find an optimal policy through iterative estimations of the optimal value function. The *Q-learning* algorithm is a reinforcement learning method that is able to solve the Bellman equations for the γ-weighted criterion. It uses simulations to iteratively estimate the value function V^*, based on the observations of instantaneous transitions and their associated reward. For this purpose, Watkins [21] introduced a function Q, that carries a significance similar to that of V but makes it easier to extract the associated policy because it does not need transition probabilities any more.

We now express the "Q-value" as a function of a given policy π and its value function V_π:

$$\forall \sigma \in S, a \in A, \quad Q_\pi(\sigma, a) = r(\sigma, a) + \gamma \sum_{\sigma'} p(\sigma'|\sigma, a) V_\pi(\sigma')$$

Therefore it is easy to see that, in spite of the lack of transition probabilities, we can trace back to the optimal policy:

$$\forall \sigma \in S, \quad V^*(\sigma) = \max_a Q^*(\sigma, a) \quad \pi^*(\sigma) = \operatorname*{argmax}_a Q^*(\sigma, a)$$

The principle of the *Q-learning* algorithm (figure 2) says that after each observed transition $(\sigma_n, a_n, \sigma_{n+1}, r_n)$ the current value function Q_n for the couple (σ_n, a_n) is updated, where σ_n represents the current state, a_n the chosen and applied action, σ_{n+1} the resulted state and r_n the immediate reward.

```
Initialize $Q_0$
**for** $n = 0$ **to** $N_{\text{tot}} - 1$ **do**
    $\sigma_n =$chooseState
    $a_n =$chooseAction
    $(\sigma'_n, r_n) =$simulate$(\sigma_n, a_n)$
    /* update $Q_{n+1}$ */
    $Q_{n+1} \leftarrow Q_n$
    $d_n = r_n + (\gamma \max_b Q_n(\sigma'_n, b)) - Q_n(\sigma_n, a_n)$
    $Q_{n+1}(\sigma_n, a_n) \leftarrow Q_n(\sigma_n, a_n) + \alpha_n(\sigma_n, a_n) d_n$
**end for**
**return** $Q_{N_{\text{tot}}}$
```

Fig. 2. The *Q-learning* algorithm.

In this algorithm, N_{tot} is an initial parameter that represents the number of iterations. The *learning rate* $\alpha_n(\sigma, a)$ is particular to each pair state-action, and decreases toward 0 at each iteration. The function `simulate` returns a new state and its associated reward according to the dynamics of the system. The choice of the current state and of the action to execute is made by the functions `chooseState` and `chooseAction`. The function `initialize` is used most of the time to initialize the values Q_0 to 0.

The convergence of this algorithm has been thoroughly studied and is now well established. We assume:

- S and A are finite, $\gamma \leq 1$.
- Each pair (σ, a) is visited an infinite number of times.
- $\sum_n \alpha_n(\sigma, a) = \infty$, $\sum_n \alpha_n(\sigma, a)^2 < \infty$.

Under these hypothesis, the function Q_n converges almost surely to Q^*. Let us recall that the almost-sure convergence means that $\forall \sigma, a$, the sequence $Q_n(\sigma, a)$ converges to $Q^*(\sigma, a)$ with a probability equal to 1. Practically, the sequence $\alpha_n(\sigma, a)$ is often defined as follows:

$$\alpha_n(\sigma, a) = \frac{1}{n_{\sigma,a}}$$

where $n_{\sigma,a}$ represents the number of times the state σ was visited and the decision a was made.

C. Partial Observation and POMDP Definition

In many cases, the observations that a decision agent is able to capture (figure 1) are only partial and do not allow the identification of the context state without ambiguity. Therefore a new class of problems needs to be solved: Partially Observable Markov Decision Processes. The states of the underlying MDP are hidden and only the observation process will help to rediscover the running state of the process.

A Partially Observable Markov Decision Process (POMDP) is defined by:

- $(S; A; T; p; r_t)$ the underlying MDP;
- $\mathcal{O}$ a set of observations;
- $O : S \rightarrow \Pi(\mathcal{O})$ an observation function that maps every state s to a probability distribution on the observations' space. The probability to observe o knowing the agent's state s will be referred as: $O(s, o) = P(o_t = o | s_t = s)$.

Non-Markovian behavior. It is worth to note that, in this new model, we loose a widely used property for the resolution of the MDPs, namely that the observation process is Markovian. The probability of the next observation o_{t+1} may depend not only on the current observation and action taken, but also on previous observations and actions:

$$P(o_{t+1} | o_t, a_t) \neq P(o_{t+1} | o_t, a_t, o_{t-1}, a_{t-1}, ...)$$

Stochastic policy. It has been proved that the results obtained for the V and Q convergence using MDP resolution algorithms are not applicable anymore. The POMDPs will need the use of stochastic policies and not deterministic ones, as in the case of MDP [22].

D. Resolution

The POMDP classic methods attempt to bring back the resolution problem to the underlying MDP. Two situations are possible. If the MDP model is known, one can not determine the exact state of the system but a distribution probability on the set of the possible states (a *belief state*). In the second situation, without knowing the model parameters, the agent attempts to construct the MDP model relying only on observations' history.

Our experimental test bed uses the resolution software package provided by Cassandra [23] that works in the potentially infinite continuous space of belief states using linear programming methods.

IV. Case Study: A Movie Presentation System for Mobile Terminals

We introduce here a system for browsing movie descriptions on mobile devices. This case study is intended to be both simple and pedagogical, while integrating a degree of realistic interactivity.

A. Interactive Access to a Movie Database

Figure 3 introduces an information system accessible from mobile terminals such as PDAs. A keyword search allows the user to obtain an ordered list of links to various movie descriptions. Among this list, the user can follow a link towards an interesting movie (the associated interaction will be referred to as *clickMovie*); then, he or she can consult details regarding the movie in question. This consultation will call on a full screen interactive presentation and a navigation scenario detailed below. Having browsed the details for one movie, the user is able to come back to the list of query results

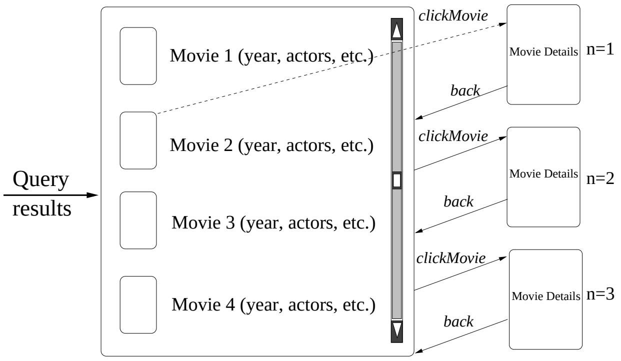

clickMovie
Movie 1 (year, actors, etc.)
Movie Details
n=1
back
Movie 2 (year, actors, etc.)
clickMovie
Query results
Movie Details
n=2
back
Movie 3 (year, actors, etc.)
clickMovie
Movie 4 (year, actors, etc.)
Movie Details
n=3
back

Fig. 3. Information system of movie descriptions

(interaction *back* on the figure 3). It is then possible to access the description of a second interesting film. The index of the accessed movie description will be referred as n.

To simplify the context modeling, we choose to consider the browsing sequence indexed by n. Our problem now becomes one that aims at adapting the content (movie descriptions) presented during this sequence. Our execution environment is dynamic because of the bandwidth's (bw) variability in a wireless network. As we consider the browsing session at a high level, we do not need to provide special specifications for the final goal of the service that can be renting/buying a DVD, downloading a media, etc.

B. From the Simplest to the Richest Descriptions

To present the details of a movie, three forms of descriptions are possible (figure 4). The poor "Textual" version (referred as T) groups together a small poster image, a short text description and links pointing to more production photos as well as a link to the video trailer. The intermediary version (I) provides a slideshow of still photos and a link to the trailer. The richest version (V) includes, in addition, the video trailer.

As the available bandwidth (bw) is variable, the usage of the three versions is not equivalent. The bandwidth required to download the content increases with the complexity of the versions ($T \rightarrow I \rightarrow V$). In other words, for a given bandwidth, the latencies perceived by the user during the download of the different versions grow proportionally with the size of the content.

More precisely, we now point out two problems generated by the inexistence of dynamic adaptation of the content when the available bandwidth varies. The adaptation strategy could systematically select only one of the three possible alternatives mentioned above. If it always selects the richest version (V) this impacts the behavior of the user who experiences bad network conditions (low bandwidth). Although strong latencies could be tolerated while browsing the first query results (small index n), it becomes quickly unacceptable if n grows. If the adaptation strategy selects systematically the simplest version (T) this would also have a harmful impact on the behavior of the user. Despite the links towards the other resources (I)mages and (V)ideo, the lack of these visual components which normally stimulate interest, will not encourage further browsing. An important and a legitimate question to be raised is what can be called an ”appropriate” adaptation policy.

C. Properties of Appropriate Adaptation Policies

The aforementioned two examples of policies (one too "ambitious", the other "too modest") show how complex is the relationship among the versions, the number of browsed films, the time spent on the service, the quality of service, the available bandwidth and the user interest. An in-depth analysis of these relationships can represent a research project in itself. We do not claim to deliver such an analysis in this paper, but we simply want to show how a policy and an adaptation agent can be generated automatically from a model where the context state is observable or partially observable.

Three properties of a good adaptation policy can be identified:

1) The version chosen for presenting the content must be simplified if the available bandwidth bw decreases [1].
2) The version must be simplified if n increases: it is straightforward to choose rich versions for the first browsed movie descriptions that are normally the most pertinent ones [2].
3) The version must be enriched if the user shows a high interest for the query results. The simple underlying idea is that a very interested user is more likely to be patient and to tolerate more easily large downloading latencies.

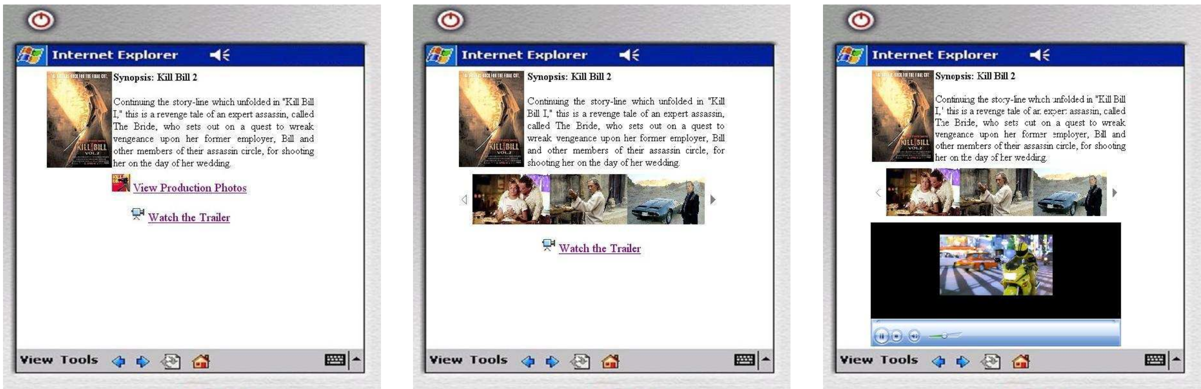
The first two properties are related to the variation of the context parameters, that we consider observable (n and bw), while the third one is related to a hidden element, namely user interest. At this stage, given these three properties, an adaptation policy for our case study can be expressed: the selection of the version (T, I or V) knowing n and bw and having a way to estimate the interest.

D. On Navigation Scenarios

This paragraph introduces by examples some possible navigation scenarios. Figure 5 illustrates different possible steps during navigation and introduces different events that are tracked. In this figure, the user chooses a film (event *clickMovie*), the presentation in version T is downloaded (event *pageLoad*) without the user interrupting this download. Interested in this film, the user requests the production photos, following the link towards the pictures (event *linkI*). In the one case, the downloading seems too long and the user interrupts it (event *stopDwl* means stopDownload) then returns to the movie list (event *back*). In the other case, the user waits for the downloading of the pictures to finish, then starts viewing the slideshow (event *startSlide*). Either this slideshow is shown completely and then an event *EI* (short for EndImages) is raised, or the visualization is incomplete, leading to the event *stopSlide* (not represented in the figure). Next, the link to the trailer can be followed (event *linkV*); here again an impatient user can interrupt the downloading (*stopDwl*) or start playing

[1] T is simpler than I, itself simpler than V

[2] as we have already mentioned, we should avoid large latencies for big values of n and small bw



Synopsis: Kill Bill 2

Continuing the story-line which unfolded in "Kill Bill I," this is a revenge tale of an expert assassin, called The Bride, who sets out on a quest to wreak vengeance upon her former employer, Bill and other members of their assassin circle, for shooting her on the day of her wedding.

View Production Photos

Watch the Trailer

Synopsis: Kill Bill 2

Continuing the story-line which unfolded in "Kill Bill I," this is a revenge tale of an expert assassin, called The Bride, who sets out on a quest to wreak vengeance upon her former employer, Bill and other members of their assassin circle, for shooting her on the day of her wedding.

Watch the Trailer

Synopsis: Kill Bill 2

Continuing the story-line which unfolded in "Kill Bill I," this is a revenge tale of an expert assassin, called The Bride, who sets out on a quest to wreak vengeance upon her former employer, Bill and other members of their assassin circle, for shooting her on the day of her wedding.

Fig. 4. Basic (T), intermediary (I) and rich (V) versions of movie details

the video (*play*). Then the video can be watched completely (event *EV* for EndVideo) or stopped (*stopVideo*), before a return (event *back*). Obviously, this example does not introduce all the possibilities. For instance the user may choose not to interact with the proposed media: we introduce a sequence of events *pageLoad*, *noInt* (no interaction), *back*. Similarly, a *back* is possible just after a *pageLoad*, a *stopDwl* may occur immediately after the event *clickMovie*, watching the video before the pictures is also possible.

V. Problem Statement

A. Rewards for well chosen adaptation policies

From the previous example and the definitions of associated interactions, it is possible to propose a simple mechanism aiming at rewarding a pertinent adaptation policy. A version (T, I or V) is considered well chosen in a given context if it is not questioned by the user. The reassessment of a version T as being too simple is suggested, for example, by the full consumption of the pictures. In the same way, the reassessment of a version V as being too rich, is indicated by a partial consumption of the downloaded video. Four simple principles that guide our rewarding system are as follows:

- We reward the event *EI* for versions I and V.
- We reward the event *EV* if the chosen version was V.
- We penalize upon arrival of interruption events (”stops”).
- We favor the simpler versions for no or little interaction.

Thus, a version T is sufficient if the user does not request (or at least not completely consume) the pictures. A version I is preferable if the user is interested enough and has access to enough resources to download and view the set of pictures (rewards *EI*). Similarly, a version I is adopted if the user views all the pictures (reward *EI*) and, trying to download the video, is forced to interrupt it because of limited bandwidth. Finally, a rich version V is adopted if the user is in good condition to consume the video completely (reward *EV*). The following decision-making models formalize these principles.

B. Towards an Implicit Measure of the Interest

The previously introduced navigations and interactions, make it possible to estimate the interest of the user. We proceed by evaluating ”implicit feedback” and use the sequences of events to estimate the user’s interest level. Our approach is inspired by [13] and is based on the two following ideas.

The first idea is to identify two types of interactions according to what they suggest: either an increasing interest (*linkI*, *linkV*, *startSlide*, *play*, *EI*, *EV*) or a decreasing interest (*stopSlide*, *stopVideo*, *stopDwl*, *noInt*). Therefore the event distribution (seen as the probability of occurrence) depends on the user’s interest state. An event *EV* is more likely to occur if the user is very interested in the browsed movie.

The second idea is to consider not only **a single running event** to update the estimation of user interest but also to regard **an entire sequence of events** as being more significant. In fact, it has been recently established that the user actions on a response page to a search (on Google for example) depend not only on the relevance of the current response but also on the global relevance of the set of the query results [2].

Following the work of [13], it is natural to model the sequences of events or observations produced by a Hidden Markov Model (HMM) for which we do not detail here the definition (see for example [24]). One can simply translate the two previous ideas by using a HMM model with several (hidden) states of interest. The three states of interest shown in figure 6 are referred as S, M, and B respectively for a Small, Medium or Big interest. The three distributions of observable events in every state are different as stressed in the first idea mentioned above. These differences explain the occurrences of different sequences of observations in terms of sequential interest evolutions(second idea). These evolutions are encoded thanks to transition probabilities (stippled) between hidden states of interest. Given a sequence of observations, an HMM can thus provide the most likely underlying sequence of hidden states or the most likely running hidden state. At this point, the characteristics of our information system are rich enough to define an adaptation agent applying decision policies under uncertainty. These policies can be formalized in the theoretical framework presented in the section III.

VI. Modeling Content Delivery Policies

In this section, we model the dynamic context of our browsing system (section IV) in order to obtain the appropriate adaptation agents. Our goal is to characterize the adaptation policies in terms of Markov Decision Processes (MDP or POMDP).

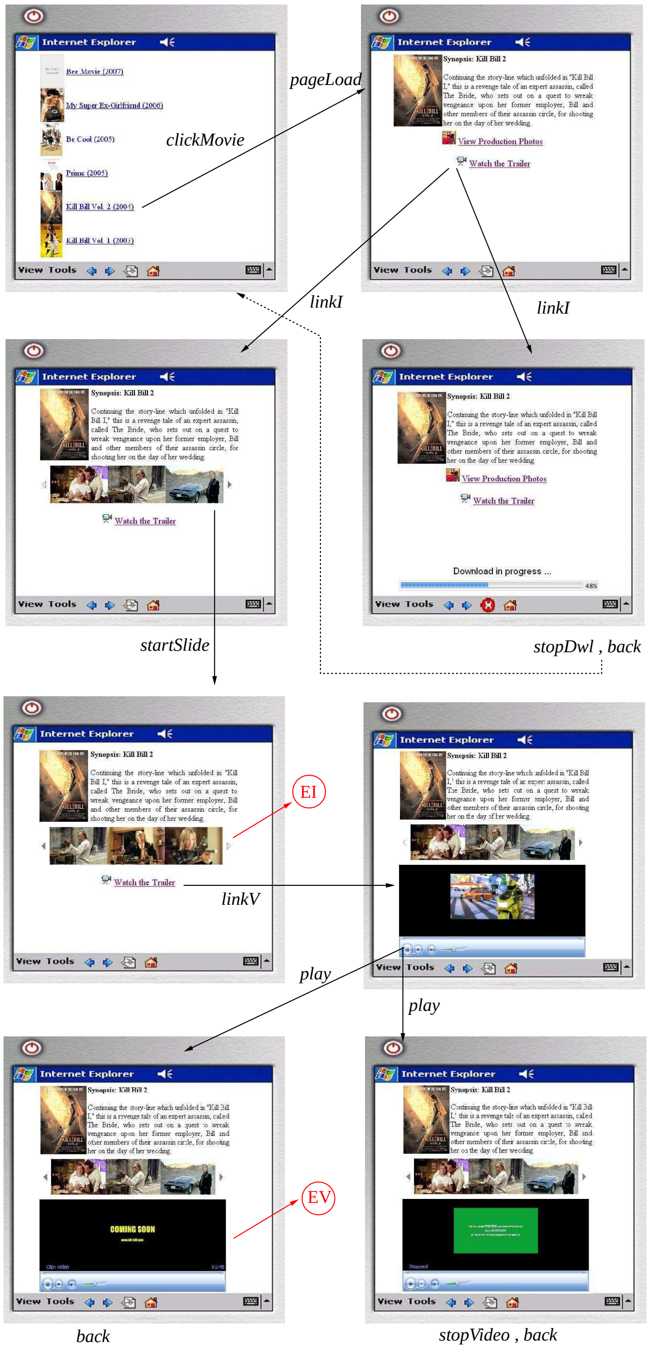


Fig. 5. Example of navigations and interactions

A. MDP modeling

Firstly, an observable context is considered. Let us introduce the proposed MDP that models it. The aim is to characterize adaptation policies which verify properties 1. and 2. described in IV-C: the presented movie description must be simplified if the bandwidth available bw decreases or if n increases.

A state (observable) of the context is a tuple s $=<$ $n, bw, v, evt >$ with n being the rank of the film consulted, bw the bandwidth available, v the version proposed by the adaptation agent and evt the running event (figure 7). With $n \in [1, 2, ...N_{max}]$, $bw \in [bw_{min}; bw_{max}]$, $v \in \{T, I, V\}$ and[3] $evt \in E$.

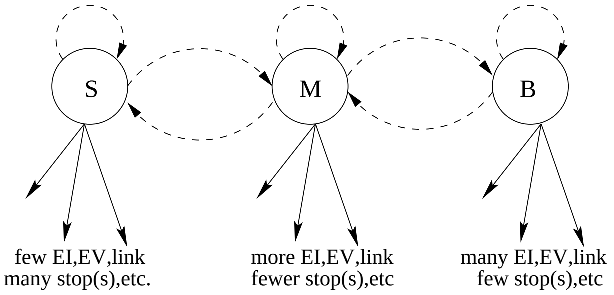


Fig. 6. A Hidden Markov Model

To obtain a finite and reasonable number of such states (limiting thus the MDP size), we will quantize the variables according to our needs. Thus n (resp. bw) can be quantized according to three levels $n \in \{B, M, E\}$ meaning Begin, Middle, End (resp. $bw \in \{L, A, H\}$ for Low, Average, High) while segmenting in three regions the interval $[1..N_{max}]$ (resp. $[bw_{min}; bw_{max}]$).

The temporal axis of MDP is naturally represented by the sequence of the observed events, every event implying a change of state.

The dynamics of our MDP is constrained by the dynamics of the context, especially by the user navigation. Thus, a transition from a movie index n to $n-1$ is not possible. Similarly, every *back* is followed by an event $clickMovie$. The bandwidth's own dynamics will have also an impact (according to quantized levels) on the dynamics between the states of the MDP.

The choice of the movie description version (T, I or V) proposed by the adaptation agent is done when the user follows the link to the film. This is encoded in the model by the event $clickMovie$. The states of the MDP can be classified in:

- decision states (ds) in which the agent executes a real action (it effectively chooses among T, I or V);
- non-decision or intermediary (is) states where the agent does not execute any action.

In a MDP framework, the agent decides an action in every single state. Therefore the model needs to be enriched with an artificial action (ϕ) as well as an absorbent state of strong penalty income ($-\infty$). Thus, all valid action $a \in \mathcal{A} = \{T, I, V\}$ chosen in an intermediary state brings the agent in the absorbent state where it will be strongly penalized. Similarly, the agent will avoid deciding ϕ in a decision-making state where a valid action is desired. Thus, the valid actions mark out the visit of the decision states while the dynamics of the context (subject to user navigation and bandwidth variability) are captured by the transitions between intermediary states for which the action ϕ (the non-action) is carried out. These properties are clearly illustrated in figure 7.

In other words, there is no change of version during the transitions between intermediary states. The action a (representing the proposed version) chosen in a decision-making

[3] $E = \{clickMovie, stopDwl, pageLoad, noInt, linkI, startSlide... stopSlide, EI, linkV, play, stopVideo, EV, back\}$

state is therefore memorized ($v_n \leftarrow a$) in all the following intermediary states, until the next decision state. Thus, the MDP captures the variation of the context dynamics *according to the chosen version*. Therefore it will be able to identify which are the good choices of versions (to reproduce later in similar conditions) if it is rewarded for them.

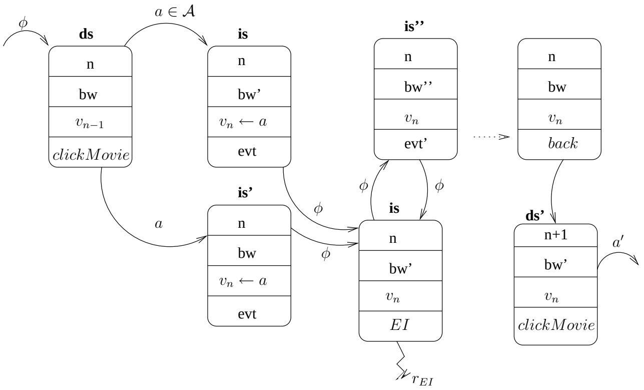


Fig. 7. MDP dynamics illustration

The rewards are associated with the decision states according to the chosen action. Intermediary states corresponding to the occurrences of the events *EI* and *EV* are rewarded as well, according to subsection V-A. The rewards[4] are defined as follows:

$r(<n, bw, v, clickMovie>, a) = r_a, a \in \{T, I, V\}$
$r(<n, bw, v, EI>, \phi) = r_{EI}$ iff $v \in \{I, V\}$
$r(<n, bw, v, EV>, \phi) = r_{EV}$ iff $v = V$

To favour simpler versions for users who do not interact with the content and do not view any media (cf. subsection V-A), let us choose $r_T > r_I > r_V$. To summarize, the model behaves in the following manner: the agent starts with a decision state ds where it decides a valid action a for which it receives an "initial" reward r_a: the simpler the version, the bigger is the reward. According to the transitions probabilities based on context dynamics, the model goes through intermediary states where it can receive new rewards r_{EI} or r_{EV} at the time of the occurrences of *EI* (resp. *EV*), if the taken action a was I or V, (resp. V). As these occurrences are more frequent for small n and high bw, while the absence of interactions is more likely if n is big and bw low, then the MDP:

- will favor the richest version for small n and high bw;
- will favor the simplest version for big n and low bw;
- will establish a tradeoff (optimum according to the rewards) for all the other cases.

The best policy given by the model is obviously related to the chosen values for $r_T, r_I, r_V, r_{EI}, r_{EV}$. In order to control this choice in the experimental section, a simplified version of the MDP will be defined.

A simplified MDP can be obtained by memorizing the occurrence of the events EI and EV during the navigation between two events $clickMovie$. Thus, we can delay the rewards r_{EI} or r_{EV}. This simplified model does not contain non decision-making states, if two booleans (ei and ev) are added to the state structure (figure 8). The boolean ei (resp. ev) passes to 1 if the event EI (resp. EV) is observed between two decision-making states.

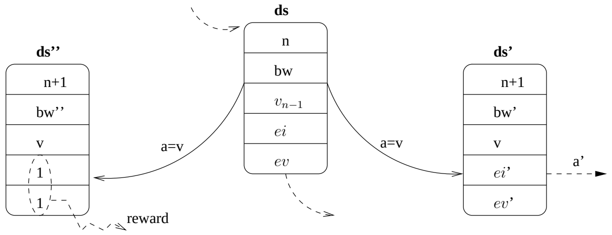


Fig. 8. Simplified MDP

The simplified MDP is defined by its states ($s =< n, bw, v, ei, ev >$), the actions $a \in \{T, I, V\}$, the temporal axis given by the sequence of events $clickMovie$ and the rewards r redefined as:

$r(<*, *, T, *, *>, *) = r_T$
$r(<*, *, I, ei, *>, *) = r_I + ei \cdot r_{EI}$
$r(<*, *, V, ei, ev>, *) = r_V + ei \cdot r_{EI} + ev \cdot r_{EV}$

This ends the presentation of our observable model and we continue by integrating user interest in a richer POMDP model.

B. POMDP Modeling

The new partially observable model adds a hidden variable (*It*) to the state. The value of *It* represents the user's interest quantized on three levels ($Small$, $Average$, Big). To be able to estimate user interest, we follow the principles described in section V-B and figure 6. The events (interactions) are taken out from the previous MDP state to become observations in the POMDP model. These observations are distributed according to *It* (the interest level). A sequence of observations provides an implicit measure of *It*, following the same principle described for the HMM in figure 6. Therefore it becomes possible for the adaptation agent to refine its decisions according to the probability of the running user's interest: $Small$, $Average$, Big. In other words, this refinement is done according to a belief state. The principle of this POMDP is illustrated in figure 9.

A hidden state of our POMDP becomes a tuple $s =< n, bw, v, ei, ev, It >$. The notations are unchanged including the booleans ei and ev.

The temporal axis and the actions $\{T, I, V, \phi\}$ are unchanged.

The dynamics of the model. When an event $clickMovie$ occurs, the adaptation agent is in a decision state ds. It chooses a valid action a and moves, according to the model's random transitions, to an intermediary state is where ei and ev are equal to 0. The version proposed by the agent is memorized in the intermediary states is during the browsing of the current film. The booleans ei and ev become 1 if the events EI or respectively EV are observed and preserve this value until the next decision state ds'. During the browsing of the running film, n and v remain constant while the other factors (bw, It and the booleans) can change.

The observations o are the occurred events: $o \in E$. They are distributed according to the states. In figure 9, the event

[4] Other formulations are possible as well including, for example, negative rewards for interruption events.

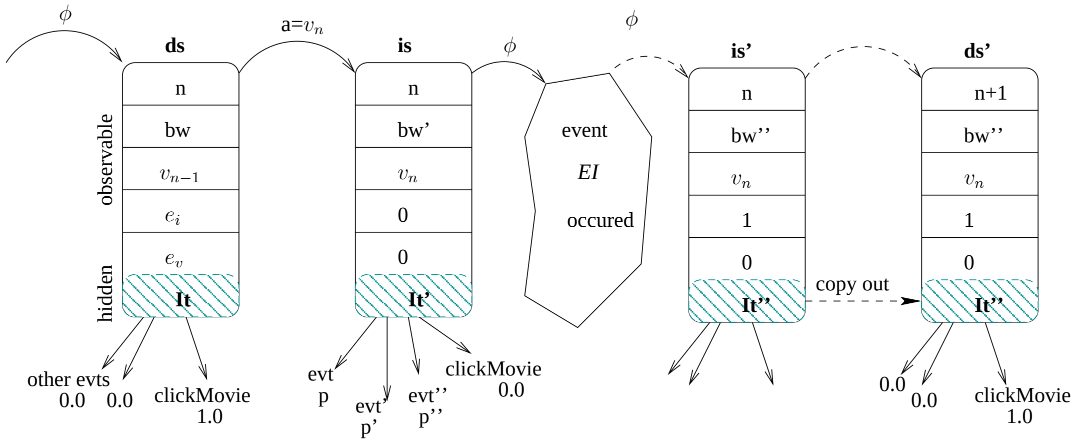


Fig. 9. POMDP dynamics between hidden states

$clickMovie$ can be observed in ds and ds' (probability 1.0) and cannot be observed elsewhere (is and is').

In every intermediary state, the event distribution characterizes the value of the interest. Thus, just as the HMM of the figure 6, the POMDP will know how to evaluate, from the sequence of events, the current belief state. The most likely interest value will evolve therefore, along with the events occurred: increase if $linkI$, EI, $linkV$, EV, ..., decrease in case of $stopDwl, stopSlide, stopVideo$. To preserve the interest level throughout the decision states, the interest of the current ds receives the value corresponding to the last is (figure 9).

The rewards associated with the actions taken in a decision-making state ds are collected in the following decision-making state ds' where all necessary information is present: v, ei and ev.

$r(< *, *, T, *, *, * >, *) = r_T$
$r(< *, *, I, ei, *, * >, *) = r_I + ei \cdot r_{EI}$
$r(< *, *, V, ei, ev, * >, *) = r_V + ei \cdot r_{EI} + ev \cdot r_{EV}$

VII. Experimental Results

Simulations are used in order to experimentally validate the models. The developed software simulates navigations such as the one illustrated in figure 5. Every transition probability between two successive states of navigation is a stochastic function of three parameters: bw, $It(n)$ and v. The bandwidth bw is simulated as a random variable uniformly distributed in a realistic interval. $It(n)$ represents a family of random variables, whose expectation decreases with n. The parameter v is the movie version proposed to the user.

A. MDP validation for observable contexts

To validate the MDP model of section VI-A let us choose a problem with $N_{max} = 12$ and $[bw_{min}; bw_{max}] = [0; 128Kbps]$. Initially, the intervals of n and bw are quantized on 2 granularity levels: $[1..N_{max}] = N_S \cup N_L$, $[bw_{min}; bw_{max}] = BW_S \cup BW_L$. Rather than proceeding to an arbitrary choice of values r_T, r_I, r_V, r_{EI}, r_{EV} that define the rewards, we can look for the ones driving to the optimal policy shown in table I. In fact, this policy π_p^* respects the principles formulated in the section IV-C and could be proposed beforehand by an expert[5].

The value functions Q corresponding to the simplified MDP, estimated over on a 1-length horizon, (between two decision-making states ds and ds') can be written as:

$$Q_1(ds, a) = r(ds) + \gamma \sum_{ds'} p(ds'|ds, a) r(ds')$$

because, $\forall ds, r(ds, a)$ does not depend on action a.

$$Q_1(ds, T) = r(ds) + \gamma \cdot r_T (\sum_{ds'} p(ds'|ds, T)) = r(ds) + \gamma \cdot r_T$$

$$Q_1(ds, I) = r(ds) + \gamma(r_I + p_{EI|I} \cdot r_{EI})$$

$$Q_1(ds, V) = r(ds) + \gamma(r_V + p_{EI|V} \cdot r_{EI} + p_{EV|V} \cdot r_{EV})$$

where $p_{EI|a}$ and $p_{EV|a}$ represent the probabilities to observe the events EI, respectively EV, knowing the version a.

	$N_S(\{1, 2, ..., 6\})$	$N_L(\{7, 8, ..., 12\})$
$BW_S[0 - 64]\ Kbps$	I	T
$BW_L[64 - 128]\ Kbps$	V	I

TABLE I. Policy π_p^* stated for two-level granularity (n and bw)

For every pair (n, bw) we have computed, based on simulations, the probabilities $p_{EI|I}$, $p_{EI|V}$, $p_{EV|V}$. The respect of the policy π_p^* is assured if and only if:

$$\forall a \in \{T, I, V\}, \forall ds =< n, bw, ... >, Q(ds, \pi_p^*(ds)) \geq Q(ds, a)$$

Writing these inequalities for the 4 pairs (n, bw) from table I and using the estimations $Q_1(ds, a)$ for Q, we obtain a 12-linear inequations system in the variables r_T, r_I, r_V, r_{EI}, r_{EV}. Two solutions of the system among an infinity are:

$R_1 : r_T = 2$, $r_I = 1$, $r_V = 0$, $r_{EI} = 6$, $r_{EV} = 6$.
$R_2 : r_T = 2$, $r_I = 1$, $r_V = 0$, $r_{EI} = 7$, $r_{EV} = 7$.

Starting from these values, we can experimentally check the correct behavior of our MDP model. Table II shows the

[5] Table I gives π_p^* only for the pairs n, bw since $\pi_p^*(< n, bw, *, ... >) = \pi_p^*(n, bw)$.

	N_1	N_2	N_3	N_4
BW_1	I	$\underline{T}$	T	T
BW_2	I	I	I	T
BW_3	V	$\underline{I}$	I	$\underline{T}$
BW_4	V	V	$\underline{I}$	I

TABLE II. Policy π_p^* refinement for R_1 rewards

	N_1	N_2	N_3	N_4
BW_1	I	$\underline{I}$	T	T
BW_2	I	I	I	T
BW_3	V	$\underline{V}$	I	$\underline{I}$
BW_4	V	V	$\underline{V}$	I

TABLE III. Policy π_p^* refinement for R_2 rewards

policy obtained automatically by dynamic programming or Q-learning algorithm, with 4 granularity levels for n and bw and the rewards R_1. This table refines the previous coarse-grained policy; this is not a simple copy of π_p^* actions (see for example the pairs (N_2, BW_1): change from I to T, (N_2, BW_3): change from V to I, etc.). This new policy is optimal with respect to the rewards R_1, for this finer granularity level.

Resolving the MDP for the second set of rewards (R_2), gives a different refinement (table III) that shows richer versions (underlined) comparing to R_1. The explanation stays in the growth of the rewards associated to the events EI, EV that induces the choice of a more complex versions, for a long time (V lasts for 3 classes of n, when $bw = BW_4$).

B. POMDP Validation: Interest-refined Policies

Once MDPs are calibrated and return appropriate adaptation policies, their rewards can be reused to solve the POMDP models. The goal is to refine the MDP policies for the observable case by estimating user interest.

Two experimental steps are necessary. The first step consists of learning the POMDP model and the second in solving the decision-making problem.

For the learning process, the simpler method consists of empirically estimating the transitions and observations probabilities directly from the simulator's traces. Starting from these traces, the probabilities are obtained from the frequencies' computation:

$$p(o|s) = \frac{\#\ emissions\ of\ o\ from\ s}{\#visits\ of\ s}$$

$$p(s'|s,a) = \frac{\#\ transitions\ from\ s\ to\ s'\ taking\ action\ a}{\#visits\ of\ couple\ (s,a)}$$

Having a POMDP model, the resolution is the next step. Solving a POMDP is notoriously delicate and computationally intensive (see, for example, the tutorial proposed at www.pomdp.org). We used the software package *pomdp-solve 5.3* in combination with *CPLEX* (with the more recent strategy called finite grid).

The results returned by pomdp-solve is an automaton that implements a "near optimal" deterministic policy, represented by a decision-making graph (*policy graph*). The nodes of the graph contain the actions ($\{T, I, V, \phi\}$) while the transitions are done according to the observations. Only the transitions made possible by the navigation process are to be exploited. To illustrate this form of result, let us show one of the automata that is small enough to be displayed on a A4 page (figure 10). We choose **a single granularity level for n and bw and three levels for It**. Additionally we consider that the consumption of the slideshow precedes the consumption of the video. The obtained adaptation policy therefore takes into account only the variation of the estimated user interest (n and bw do not play any role). Figure 10 shows that the POMDP agent learns to react in a coherent way. For example, starting from a version T, and observing *pageLoad, linkI, startSlide, EI, noInt, back* the following version decided by the POMDP agent is I, which translates the sequence into an interest rise. This rise is even stronger if, after the event *EI*, the user follows the link *linkV*. This is enough to make the agent select the version V further.

Conversely, starting from version V, an important decrease in interest can be observed on the sequence *startSlide, stopSlide, play, stopVideo, back*, so the system decides T. A smaller decrease in interest can be associated with the sequence *startSlide, stopSlide, play, EV, back*, the next version selected being I. These examples show that there exists a natural correlation between the wealth of the selected versions and the implicit user interest. For this problem, where n and bw are not involved, the version given by the *policy graph* translates the estimation of the running interest (growing with $T \rightarrow I \rightarrow V$). For each movie, the choice of version is therefore based only on the events observed while browsing the previous movies.

Other sequences cause the decisions to be less intuitive or harder to interpret. For example, the sequence *pageLoad, linkI, startSlide, stopSlide, noInt, back* leaving T leads to the decision I. In this sequence, a compromise between interest rise (suggested by *linkI, startSlide*) and decrease (suggested by *stopSlide, noInt*) must be established. Thus, a decision T would not be illegitimate. The POMDP trades off this decision according to its dynamics and its rewards. To obtain a modified graph leading to a decision T for this sequence, it would be sufficient that the product $r_{EI}\xi$ decreases, where ξ represents the probability to observe *EI* in the version I, for a medium interest. In this case, *stopSlide*, instead of provoking a loopback on the node 5, would bring the agent to the node 1. Then the agent would decide T since the expectation of the gains associated to I would be smaller.

In general, the decision-making automaton depends on n and bw. When n, bw and It vary, the automaton becomes too complex to be displayed. The results of the POMDP require a different presentation. Henceforth working with **3 granularity levels on n, 2 on bw, 3 on It** and the set of rewards R_1 leads to a *policy graph* of more than 100 nodes. We apply it during numerous sequences of simulated navigations. Table IV gives the statistics on the decisions that have been taken. For every triplet (n,bw,It) the decisions - **the agent not knowing It** - are counted and translated into percentages.

We notice that the proposed content becomes statistically richer when the interest increases, proving again that the interest estimation from the previous observations is as expected. Let's take an example and consider the bottom-right part of table IV (corresponding to BW_L and N_3). The probability of the policy proposing version V increases with the interest:

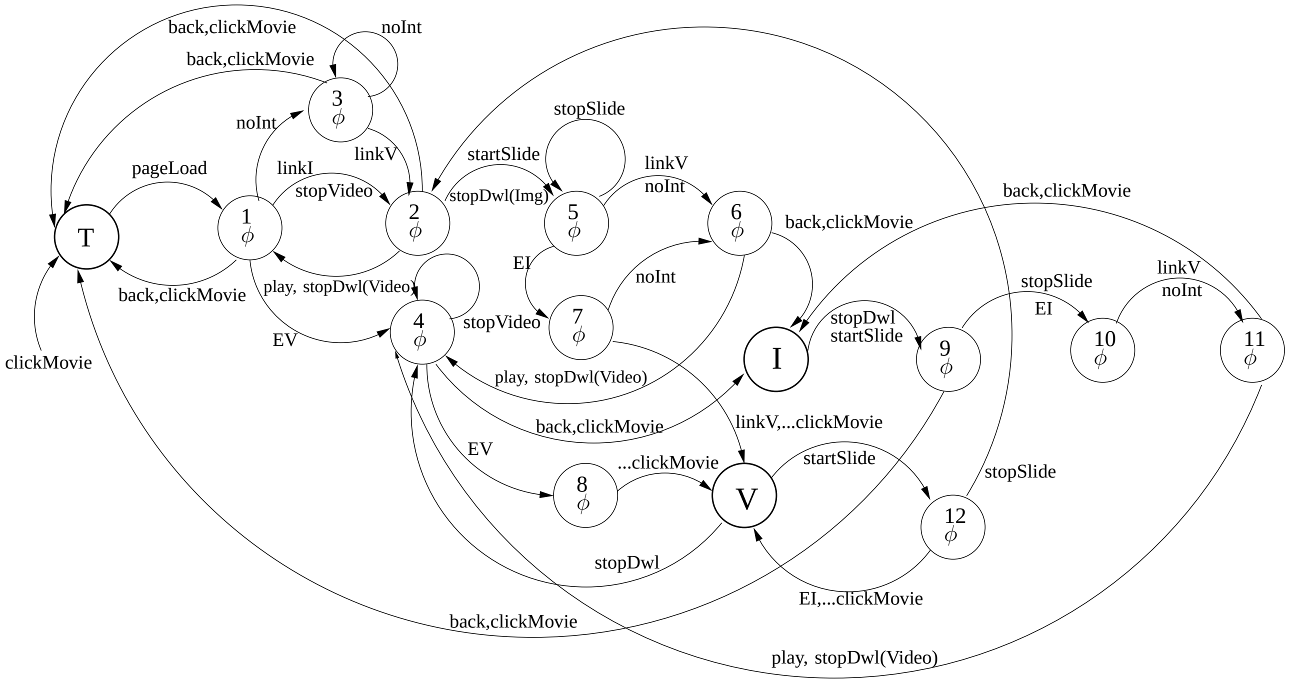


Fig. 10. Decision-making automaton (*policy graph*), POMDP solution. Please note the different *stopDwl*, *stopDwl(Img)* and *stopDwl(Video)*.

	$Interest$	$N_1\{1,2,3,4\}$			$N_2\{5,6,7,8\}$			$N_3\{9,10,11,12\}$		
		T	I	V	T	I	V	T	I	V
BW_S	$Small$		100%		5%	95%		96%	4%	
	$Average$		100%		1%	99%		86%	14%	
	Big		100%			100%		76%	24%	
BW_L	$Small$		4%	96%		73%	27%	67%	33%	
	$Average$		2%	98%		51%	49%	30%	68%	2%
	Big			100%		33%	67%	9%	81%	10%

TABLE IV. Actions' distribution for the POMDP solution policy

from 0% (small interest) to 2% (average interest) then 10% (big interest).

Moreover, when n and/or bw increase, the interest trend is correct. For example, for a given set of It and n ($It = Average$ and $n = N_2$), the proposed version becomes richer with the bandwidth's increase from (1%T, 99%I, 0%V) to (0%T, 51%I, 49%V). Additionally, from one set of rewards to another, these trends are always respected, although the values of the percentages are different.

The POMDP capacity to refine adaptation policies according to the user interest is thus validated. Once the POMDP model is solved (**offline** resolution), the obtained automaton is easily put into practice **online** by encoding it into an adaptation agent.

VIII. Conclusion

This paper has shown that sequential decision processes under uncertainty are well suited for defining adaptation mechanisms for dynamic contexts. According to the type of the context state (observable or partially observable), we have shown how to characterize adaptation policies by solving Markov Decision Processes (MDP) or Partially Observable MDP (POMDP). These ideas have been applied to adapt a movie browsing service. In particular, we have proposed a method for refining a given adaptation policy according to user interest. The perspectives of this work are manifold. Our approach can be applied to cases where rewards are explicitly related to the service (e.g. to maximize the number of rented DVDs). It will also be interesting to extend our model by coupling it with functionalities from recommendation systems and/or from multimedia search systems. In the latter case, we would benefit a lot from a collection of real data (navigation logs). These are the research directions that will guide our future work.

References

[1] D. Kelly and J. Teevan, "Implicit feedback for inferring user preference : A bibliography," in *SIGIR Forum*, vol. 37(2), 2003, pp. 18–28.

[2] T. Joachims, L. Granka, and B. Pan, "Accurately interpreting clickthrough data as implicit feedback," in *SIGIR'05*, August 2005.

[3] M. Margaritidis and G. C. Polyzos, "Adaptation techniques for ubiquitous internet multimedia," *J. on Wireless Communications and Mobile Computing*, vol. 1, no. 2, pp. 141–163, 2001.

[4] T. Lemlouma and N. Layaida, "Media resources adaptation for limited devices," in *Proc. ICCC/IFIP International Conference on Electronic Publishing*, June 2003, pp. 209–218.

[5] M. K. Asadi, "Multimedia content adaptation with mpeg-21," Ph.D. dissertation, ENST Paris, 2005.

[6] T. Lemlouma and N. Layaida, "Encoding multimedia presentation for user preferences and limited environments," in *IEEE ICME*, July 2003, pp. 165–168.

[7] A. Divakaran, K. A. Peker, R. Radhakrishnan, Z. Xiong, and R. Cabasson, *Video Summarization Using MPEG-7 Motion Activity and Audio Descriptors in Video Mining*. Kluwer, 2003.

[8] B. Girod, M. Kalman, Y. J. Liang, and R. Zhang, "Advances in channel-adaptive video streaming," in *IEEE ICIP (invited paper)*, September 2002, pp. 1–8.

[9] G. Ghinea and G. Magoulas, "Quality of service for perceptual considerations: an integrated perspective," in *ICME*, 2001, p. 146.

[10] S. R. Gulliver, T. Serif, and G. Ghinea, "Pervasive computing: the perceptual effects of variable multimedia quality," *Intl. J. of Human-Computer Studies*, vol. 60, no. 5-6, pp. 640–665, 2004.

[11] P. Brusilovsky, "Adaptive hypermedia," *User Modeling and User-Adapted Interaction*, vol. 11, no. 1-2, 2001.

[12] P. Brusilovsky and M. T. Maybury, "From adaptive hypermedia to the adaptive web," *Communications of the ACM*, vol. 45, no. 5, pp. 30–33, 2002.

[13] T. Syeda-Mahmood, "Learning and tracking browsing behavior of users using hidden markov models," in *IBM Make It Easy Conference*, 2001.

[14] G. Yavas, D. Katsaros, Ö. Ulusoy, and Y. Manolopoulos, "A data mining approach for location prediction in mobile environments." *Data Knowl. Eng.*, vol. 54, no. 2, pp. 121–146, 2005.

[15] H. Kosch, L. Boszormenyi, M. Doller, M. Libsie, and P. Schojer, "The life cycle of multimedia metadata," *IEEE Multimedia*, vol. 12, no. 1, pp. 80–86, 2005.

[16] C. Timmerer and H. Hellwagner, "Interoperable adaptive multimedia communication," *IEEE Multimedia*, vol. 12, no. 1, pp. 74–79, 2005.

[17] O. Layaida, S. Atallah, and D. Hagimont, "A framework for the dynamic configuration and reconfiguration of network-based media adaptation," *J. of Internet Techno.*, vol. 5, 2004.

[18] P. Ruiz, J. Botia, and A. Gomez, "Providing qos through machine learning driven adaptive multimedia applications," *IEEE Trans on Sys., Man and Cybernetics*, vol. 34, p. 1398, 2004.

[19] V. Charvillat and R. Grigoraş , "Reinforcement learning for dynamic multimedia adaptation," *to appear in Journal of Network and Computer Applications*, vol. in press, 2006.

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

[21] M. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. Wiley-Interscience, 1994.

[22] S. P. Singh, T. Jaakkola, and M. I. Jordan, "Learning without state-estimation in partially observable markovian decision processes," in *International Conference on Machine Learning*, 1994, pp. 284–292.

[23] A. R. Cassandra, L. P. Kaelbling, and M. L. Littman, "Acting optimally in partially observable stochastic domains," in *Proc. of AAAI 94*. Seattle: MIT Press, 1994, pp. 1023–1028.

[24] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.