Proceedings of the Remembering Who We Are – Human Memory for Artificial Agents Symposium, Mei Yii Lim and Wan Ching Ho (Eds.), at the AISB 2010 convention, 29 March – 1 April 2010, De Montfort University, Leicester, UK

I can (almost) remember what *you* are doing: from actions to tasks

Rudolf Kadlec[1], **Cyril Brom**[1]

Abstract. Recently there has been an increased attention to the episodic memory research in the community studying intelligent virtual agents. An episodic memory can exploit a hierarchical representation, in which higher layers present increasingly more abstract tasks. During storage, an agent with episodic memory can directly access these abstractions from its own behavioral representations or from behavioural representations of agents being observed, provided the representations in question are hierarchical. However, the abstractions are lacking when observing behaviour of human users. In this paper we propose a way in which our already existing computational model of episodic memory can be extended by an ability to incorporate knowledge about abstract tasks of human users. We have tested ID3 classification algorithm and hidden Markov model for reconstructing these tasks from observations. Our approach was tested in a simple scenario with one agent situated in a 3D environment of the game Unreal Tournament that is being observed and its tasks are then inferred.

1 INTRODUCTION

Episodic memory [1] (EM) is a memory for personal history of an entity. It was proposed that it enhances believability of life-like intelligent virtual agents (IVAs) [2].

Compared to other published computational models of EM (see [2] for overview), our model [3] has some unique features capitalizing on the fact that IVAs' behaviour is often expressed in a hierarchical fashion. Today, models like hierarchical behaviour trees or hierarchical finite state automata are *de facto* standards in computer games industry [4]. The behavioural hierarchies present in the program controlling the IVA can be at the same time used for storing the IVA's history in EM, making the EM representation also hierarchical.

We are developing our model in an incremental way. First, we have implemented a central module for remembering history of one IVA [3] in a hierarchical tree-like structure. An IVA's behaviour is represented by a forest of AND-OR trees where each tree corresponds to one top-level task. The inner nodes of the tree represent tasks or sets of alternative tasks that the IVA has to complete in order to satisfy the top-level task. Leafs of the tree are atomic actions that can be executed in the environment. Figure 1 shows an example of such tree.

In the central module of our EM, the IVA's remembered history is represented as a sequence of executed tasks preserving their hierarchical structure. Upon this central module we have implemented gradual forgetting [2, 3] that is driven by emotional response to the remembered event and also by its age. Lower levels of the hierarchy can be forgotten first, thus preserving at least an abstract summary of agent's actions. Other implemented extensions of the central module include abilities to use socially established time patterns [5] and more plausible spatial representation [2].

So far the model was able to store only an IVA's own actions. Theoretically this approach can be also used for other computer controlled agents because the behavioural hierarchy is represented in their "minds" and can be disclosed to an observing IVA. However, human users cannot be treated this way since their abstract tasks are not directly accessible by the program, an observer can see only a stream of atomic actions. The hierarchy has to be inferred from the available data – that is from that stream of atomic actions and from a state of the world. In other words, we have to reconstruct the AND-OR tree from observations to store it in the episodic memory.

In order to deduce the actual AND-OR tree, we have to: 1) Find a suitable representation of an IVA's state. 2) Find out the concrete task that most probably produced the observed state (we mean by a *concrete task* the tasks from the first layer upon the atomic actions – see Fig. 1 – and, for brevity, when the context makes it clear, we will use the term *task* for *concrete tasks*). 3) Then reveal more abstract levels in the tree based on already known lower level tasks (layers 2 and 3 in Fig. 1).

We have taken the first step towards this goal by inferring the concrete tasks from streams of atomic actions. To this end we have tested two machine learning algorithms. First of the selected approaches has taken a naïve assumption that only one observation of an agent's current state is needed in order to deduce the concrete task. ID3 [6] classifier was used in this stage. The other approach used hidden Markov model (HMM) [7] and operated with sequences of observations. ID3 was taken as a simple solution that served as a baseline for comparison with the HMM that is better suited for temporal inference.

Performance of the algorithms was tested in a scenario with an IVA whose behaviour was classified by both algorithms. The IVA was used for generating training data that consisted of atomic actions, state of the agent's surrounding and the agent's current concrete task. Then, testing data, only atomic actions and the agent's surrounding, were generated by the same agent. The agent's program was not deterministic. Behaviour of human users was not classified yet. Both the agent and the classification mechanism were implemented in Java. Agent was coded using Pogamut [8], which is a middleware for interfacing with the virtual environment of Unreal Tournament 2004 [9].

The rest of the paper continues as follows: Sect. 2 presents possible alternative solutions, Sect. 3 describes our IVA's architecture, Sect. 4 formalizes the problem, Sect. 5 presents our scenario, Sect. 6 describes the experiments, and Sect. 7 discusses the results and possible future directions. The last section concludes the paper.

[1] Dept. of Software and Computing Science Education, Charles University in Prague, Ke Karlovu 3, 12118, Prague, Czech Republic.

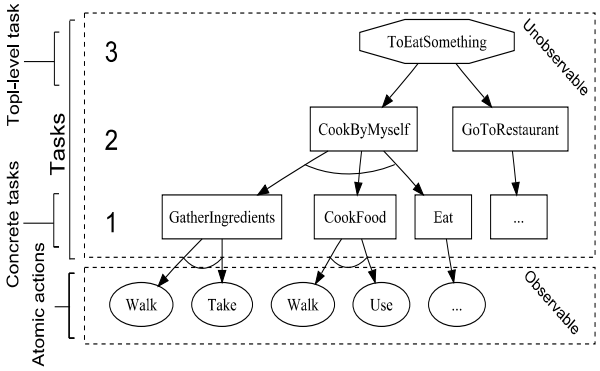

Figure 1. An example of a tree representing behaviour of an IVA. The top-level task *ToEatSomething* on the highest level can be realized by the task *CookByMyself* or by the task *GoToRestaurant*. Each of these tasks is further decomposed into several concrete tasks. Atomic actions, shown on the lowest level, are the only output that can be observed when behaviour is generated by a human user's avatar.

2 POSSIBLE ALTERNATIVES

In general the problem of determining an agent's motivation, based on the observed state is referred to as a "keyhole plan recognition problem". There exist many approaches to solving this issue; here is a non exhaustive list of these methods.

Case based plan recognition (CBPR) used for example in [10] originates from the case based reasoning (CBR) paradigm. CBPR works with a library of cases/plans, each case is a sequence of observed states. Another possibility is to use mental state abduction [11]. This technique supposes that the decision mechanism can be described by a set of implications. Switching antecedents and consequents then makes it possible to reason what rules may have caused current state of the IVA. Both the abduction and the CBPR are symbolic approaches and it is questionable how they would deal with noise and uncertainty.

The other branch of techniques follows the probabilistic approach. Majority of these algorithms use a form of HMM, for example [12]. Algorithm for hierarchical probabilistic plan recognition is proposed in [13]. For HMM's ability to cope well with uncertainty, we decided to employ this approach to our purpose as well. To our knowledge, HMM was not used previously for the problem of high level task recognition in domain of computer games.

3 OUR AGENT'S ARCHITECTURE

Our IVA's architecture is a reminiscence of a classical cognitive AI architecture, by which many virtual agents have been inspired. Our IVA is driven by *hierarchical reactive planning* with behaviour represented by *AND-OR trees*. The AND-OR tree metaphor works with abstract *goals* representing what shall be achieved, and *tasks* representing how to achieve the goals. Typically, every goal can be accomplished by several tasks, while every task can be achieved by adopting some sub-goals. In Figure 1, goals were intentionally omitted for parsimony. As said in Introduction, in this paper, we are interested only in concrete tasks; thus, the higher layers of the hierarchy, including goals, are of no importance here.

The tasks that cannot be decomposed are *atomic actions*, i.e. action primitives. Every task may need several resources to be performed, i.e. *objects*. Every top-level goal has its *activity level* based on drives, external events, and a schedule. The goals compete among themselves and the winning goal chooses the most appropriate task (e.g. "to eat" goal can chose "take something from the fridge") and passes its template to the *task field* of the *visual short term memory*. From the AI standpoint, this mechanism capitalises on the BDI framework [14].

The visual short-term memory holds templates of objects seen that passed through a simple threshold-based *attention filter*. Every object is regarded as a tool for action, i.e. it *is* a set of "affordances" [15], meaning it possesses pointers to the tasks it can be used as a resource for. These pointers are perceived directly by the IVA when observing his environment. Objects in experiments we have been running are state-less for the sake of simplification, though our simulations allow the objects to have states as well.

4 FORMALISATION

In this section, we will formalize previously presented agent architecture in the context of task recognition problem.

Suppose that the IVA is controlled by a decision making system (DMS), which can be formalised by function $DMS: P \times S \rightarrow A \times S$ where P is the IVA's perception, A actions that can be performed in the environment and perceived by other IVAs and S is the agent's internal state. A part of S is also the IVA's *motivation* M. M provides an explanation of the agent's actions; it can be for example currently running procedure or a plan speaking in terms of BDI [14] or a trace in AND-OR tree going from last issued atomic action to the root of the active tree or an emotion state. For purposes of our experiments, M will contain only concrete tasks. In case of IVAs, both A and S can usually be disclosed for observing agents (with some exceptions such as an agent going to raid a bank). In case of human users, S and thus M is usually unobservable.

The representation of the set P can be fairly complicated. Considering domain of computer games, there can be positions and states of all observed objects, information about geometry of the level and virtually any feature of the simulation engine that can be accessed by the *DMS*. To reduce the dimensionality of P we have mapped it to a new space $P' = \{0, 1\}^n$ where each vector component means presence of a salient feature in an agent's surrounding. Detected features have to be selected by a designer with respect to the agent's *DMS*. P' also has to contain only features that can be observed by an external observer because it will be used for external task inference.

With this notation, the classification of IVA's motivation, i.e. the inference of concrete tasks, is a job of constructing the sequence $m_1, m_2, \ldots m_T$; $m_t \in M$ given the sequence $x_1, x_2, \ldots x_T$; $x_t \in P' \times A$. This means inferring motivation based only on the external observation of actions issued by the agent and on current state of the environment.

5 PROBLEM DOMAIN

In our example scenario the IVA have 5 top-level tasks: *Cooking, Training, GoToToilet, PlayComputerGames, Hygiene*.. Those were decomposed into 20 concrete tasks. All the concrete tasks and their associated top-level tasks are listed in Table 1. As can be seen there are generally two types of concrete tasks: tasks that usually control the IVA's interaction with some world objects (e.g. *WashHands*) and tasks executed when the IVA is looking for some objects/places with which/where he can execute another desired tasks (eg. *Search_PlaceToWashHands)*.

Concrete task names	Top-level intention the concrete task belongs to
Idle	*NONE*
Search_InPocket	All top-level tasks
Search_PlaceToWashUp	*Cooking, GoToToilet, Hygiene*
WashHands	*Cooking, GoToToilet,*
Search_Ingredients, GatherIngredients, Search_ToCookIn, Cook, Search_PlaceToEat, EatTheFood	*Cooking*
Search_Toilet, UseToilet	*GoToToilet*
Search_InternetAccess, ConfirmTraining, Search_SportGround, Play	*Training*
Search_Computer, PlayVideogames	*PlayComputerGames*
BrushTeeth, WashFace	*Hygiene*

Table 1. List of all concrete tasks and their top-level tasks.

In our experiments, M was a set of all concrete tasks from Table 1. Set A consisted of atomic actions *{USE, EAT, TAKE, IDLE, WALK}*. In simplified perception P', each component of the vector was associated with one affordance. If an object with this affordance was near the agent, the component was assigned value 1, otherwise it was set to 0. Affordances specified in the scenario were *COOK_IN, EAT_IN, STUDY_AT, WASH_UP, FOOD_STORE, PLAY_ON, HAS_INTERNET, GO_ON_TOILET, WORK_OUT_AT*. As you can see, P' has 9 dimensions. Actions were coded similarly as perception. Each atomic action has been associated with a dimension: if the agent was performing an action, the vector component corresponding to this action was assigned value 1; otherwise it was set to 0.

6 EXPERIMENTS

In this section, we describe two classification methods we have used for inferring motivations of IVA's in our scenario.

In the first instance we tested a naïve classification method that have no access to the observed history – ID3 tree classifier. Hidden Markov model that was tested in the second experiment was chosen because it is well suited for temporal reasoning.

In the training phase the IVA was executed for approximately 12 hours of the simulated time. This resulted in 1200 data samples. Then we tested the algorithms on 650 samples obtained from the second simulation run.

6.1 Reasoning about one observation - Tree based classification

Motivation. In our first experiment we have used tree based ID3 [6] classifier to serve us as a baseline in assessing performance of the HMM based classification. We choose ID3 because it is one of the well known algorithms suitable for classification of vectors with nominal attributes without many parameters that has to be fine tuned to get the best performance.

Method. Input to the ID3 was just the current snapshot of the agent's perception and its action ($P' \times A$), not the previous time steps. The target class was the current task. We used ID3 implementation from Weka 3.6 [16].

Results. Error of the classification was 47%. The classification was successful in cases where the perceivable context implied the task, as in the cooking example (error in all non *Search_** tasks was only 9%), but failed when the context was insufficient (error in *Search_** tasks was 78%). This happened because P' contained neither the current location of the IVA nor its past actions, all states when the agent was walking were perceived as the same. There are more ways of how to overcome this problem. For instance, we can aggregate several last states into one vector and run the ID3 on that vector, but this does not seem as a scalable solution. Instead, we have adopted a solution employing hidden Markov model.

6.2 Reasoning about sequence of observations - Hidden Markov model based classification

Motivation. HMM is well suited for reasoning under uncertainty that is why we decided to test it also in our domain. We hypothesize that HMM will outperform ID3 results from the previous experiment.

Method. HMM works with an assumption that the observed process is driven by some probabilistic finite automata whose current state $S_i \in S$ can be observed only indirectly via symbols from an observation alphabet $V=\{v_1, v_2, .. v_m\}$. The HMM λ is then defined by $\lambda = \langle A, B, \pi \rangle$ where A is a matrix of probabilities of transition between every two states; B is an observation symbol probability distribution, that is, probability of observing symbol v_j given the automata is in state S_i; and π is an initial state probability.

If we have a sequence of observations $O=O_1O_2\ldots O_t$, $O_i \in V$ we can compute the most probable sequence of inner states $Q=q_1q_2\ldots q_t$, $q_i \in S$ that generated this observation. This sequence can be found efficiently by the Viterbi algorithm [7].

If we put S equal to set of all tasks and V equal to $P' \times A$ then we have translated our plan recognition problem into finding the most probable sequence Q given the set O and HMM λ. For practical reasons, the set of all observations V was simplified by k-means clustering to just 20 possible symbols. Observation space clustering is a standard technique used in HMM [7].

In the training phase, matrices A and B were estimated from the data logged through the IVA's execution, π was set to equal value for all states since we do not know which state will be the initial. When some conditional probability was 0 according to the data we set it to 0.001, this expresses the fact that even events that were not observed in the training data may happen.

In the testing phase, the Viterbi algorithm was executed with the whole testing sequence as an input. Implementation of the Viterbi algorithm was taken from the Jahmm[3] library.

Results. Resulting error in prediction was 20%. Error in *Search_** tasks was 30%, error in non *Search_** tasks was only 7%. Compared to the ID3 the improvement was in sequences when the IVA was mostly walking between locations. The information that was missing in the context of the current state was compensated by context of previous states. Most of the remaining error in *Search_** tasks is caused by situations when the IVA was looking for some object *A* but then changed its mind and started looking for some other object *B*. When the *B* object was later used in some recognized task then the whole previous searching sequence was classified as *Search_B* instead of *Search_A* and then *Search_B*.

7 DISCUSSION AND FUTURE WORKS

A positive result of this paper is that the experiments with HMM from previous section suggest that this technique can be used for classification of concrete tasks. Thus the first level of the behavioural hierarchy upon atomic actions can be reconstructed and then stored in EM. However, the current model of task inference is only a proof of concept. We have used the same agent for both training and testing. The next step will be to extend the scenario and explain actions of one IVA by another IVA controlled by different tasks which in turn determine the HMM λ used for inference. This means that an IVA with different "vocabulary" of known tasks determined by its own behavioural representation will try to explain behaviour of other IVA. However, our major focus is on testing the performance of our model on explaining tasks of human players. We will have to obtain sequences of users' actions in a virtual scenario of similar complexity to the current scenario. Then we will annotate these sequences by a hierarchy of tasks and finally test the HMM model on inferring this hierarchy.

The current experiments were focused on determining only the concrete tasks, i.e. the first level upon leaves of the behavioural hierarchy. To infer higher levels of the tree, we aim at trying to construct a second HMM that will have already discovered concrete tasks on its input and the top-level tasks will be its states.

8 CONCLUSION

In this paper we have presented our existing computational model of episodic memory, proposed how it can be extended to represent actions of other IVAs and humans whose internal state cannot be accessed by the EM module. Finally we have proposed, implemented and tested two solutions in a simple virtual scenario. One solution used the ID3 classifier and one used the HMM. As we have expected the HMM outperformed the ID3 in revealing the task hierarchy.

Acknowledgement. This work was partially supported by the student research grant GA UK 21809 and by the grant 201/09/H057 and by the research project MSM0021620838 of the Ministry of Education of the Czech Republic.

[3] Jahmm: http://code.google.com/p/jahmm/ [19.1.2010]

REFERENCES

[1]E. Tulving, W. Donaldson. *Organization of memory*. New York: Academic Press. (1972).

[2]C. Brom, J. Lukavský. Towards virtual characters with episodic memory II: Episodic memory strikes back. In: *Proc. Empathic Agents, AAMAS workshop*. 1 - 9. (2009).

[3]C. Brom, K. Pešková, J. Lukavský. What does your actor remember? Towards characters with a full episodic memory. In: *Proc. ICVS*, LNCS 4871, Springer-Verlag. (2007).

[4]D. Isla. Handling Complexity in the Halo 2 AI. (2004). URL: http://www.gamasutra.com/gdc2005/features/20050311/isla_01.shtml [21.1.2010]

[5]O. Burkert. Connectionist Model of Episodic Memory for Virtual Humans. Master thesis. Charles University in Prague. (2009).

[6]J. R. Quinlan. Induction of Decision Trees. In: *Machine Learning 1*, pp. 81-106. (1986).

[7]L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*. pp. 257—286. (1989).

[8]J. Gemrot, R. Kadlec, M. Bida, O. Burkert, R. Pibil, J. Havlicek, L. Zemcak, J. Simlovic, R. Vansa, M. Stolba, T. Plch, C. Brom. *Pogamut 3 Can Assist Developers in Building AI (Not Only) for Their Videogame Agents*. In: *Agents for Games and Simulations*, LNCS 5920, Springer, pp. 1-15. (2009).

[9]Epic Games Inc. Unreal Tournament 2004. (2004). URL: http://www.unrealtournament2004.com [21.1.2010]

[10]M. Fagan, P. Cunningham. Case-Based Plan Recognition in Computer Games. In: *Case-Based Reasoning Research and Development*. Springer Berlin. pp. 161-170. (2003).

[11]M.P. Sindlar, M.M. Dastani, F. Dignum & J.-J.Ch. Meyer. Mental State Abduction of BDI-Based Agents. *Proceedings of DALT 2008*, Springer Verlag, pp. 161-178. (2008).

[12]K. Han, M. Veloso. Automated robot behavior recognition applied to robotic soccer. In: *Robotics Research: the Ninth International Symposium*, John Hollerbach and Dan Koditschek, editors, pp. 199-204. Springer-Verlag, London. (2000).

[13]N. Blaylock, J. Allen. Hierarchical instantiated goal recognition. In: *AAAI Workshop on Modeling Others from Observations*, Boston, (2006).

[14]M. E. Bratman. *Intentions, Plans, and Practical Reason*. Harvard University Press: Cambridge, MA. (1987).

[15]J. J. Gibson. The Theory of Affordances. In: *Perceiving, Acting, and Knowing*, Eds. Robert Shaw and John Bransford. (1977).

[16]M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten. The WEKA Data Mining Software: An Update. In: *SIGKDD Explorations*, Volume 11, Issue 1. (2009).