ACTA TECHNICA CORVINIENSIS – Bulletin of Engineering
Tome VI (Year 2013) – FASCICULE 3 [July–September]
ISSN 2067–3809

[1.] Mamun Bin Ibne REAZ, [2.] Mohd. MARUFUZZAMAN

PATTERN MATCHING AND REINFORCEMENT LEARNING TO PREDICT THE USER NEXT ACTION OF SMART HOME DEVICE USAGE

[1-2.] DEPARTMENT OF ELECTRICAL, ELECTRONIC AND SYSTEMS ENGINEERING, UNIVERSITI KEBANGSAAN MALAYSIA, 43600, UKM, BANGI, SELANGOR, MALAYSIA

ABSTRACT: Future Smart-Home device usage prediction is a very important module in artificial intelligence. The technique involves analyzing the user performed actions history and apply mathematical methods to predict the most feasible next user action. This paper presents a new algorithm of user action prediction based on pattern matching and reinforcement learning techniques. Synthetic data has been used to test the algorithm and the result shows that the accuracy of the proposed algorithm is 87%, which is better than ONSI, SHIP and IPAM algorithms from other researchers.
KEYWORDS: Smart home, artificial intelligence, pattern matching, reinforcement learning, Markov Model

INTRODUCTION

Smart home system is an intelligent system that needs to adapt to the inhabitant's lifestyle, predict their future actions and minimize the user device interaction. Such requirements will never be achieved without proper analysis of inhabitant's device interaction history for each particular state of the home environment. The basic idea behind prediction is given a sequence of user device interaction events, how can the next user action be predicted [1].

In the previous year's many researchers developed techniques for predicting user actions. Some of these methods were used for intelligent Human Computer Interaction (HMI) [2]; others were used for smart-home environment [3, 4, 5]. Although the domain were different prediction techniques, the idea behind the techniques was the same and it is possible to apply a method of a particular domain to another by modifying the representation of the environment and action sequence.

Some of the techniques that were previously used has employed Markov models to optimally predict the next user action in any stochastic sequence, one such example of these algorithms is Smart Home Inhabitant Prediction (SHIP) [6,7,8]. SHIP can predict the user future actions but only achieve 60% prediction accuracy which is undesirable for sensitive application such as Smart home. On the other hand, some techniques have simply used sequence matching to predict the next user action like Incremental Probabilistic Action Modeling (IPAM) and On-line implicit State Identification (ONSI) [2]. While IPAM performs better than SHIP, it chooses pairs of actions occurring in sequence as a pattern and summarizes them by increasing the probability of those that occur and decreasing the probabilities of all others. IPAM make an implicit Markov assumption, namely the last action together with the current summary provided by the probability estimation, which contain enough information to predict the next state. Looking at real user interaction traces, it has been found that often users enter modes that can be easily identified by examining the behavioral patterns they engage in, but these patterns span more than two actions in a row. While the techniques that is currently being used passively predicts the user actions where they do not take into consideration the user most preferred action for a particular environment state. ONSI solves IPAM problem by using varying pattern matching length however it tend to ignore the adaptation to the user preferred actions and the relation between the actions and the environment state.

Inspired by behaviorist psychology, reinforcement learning is an area of machine learning in computer science, concerned with how an agent ought to take actions in an environment so as to maximize some notion of cumulative reward. The reinforcement learning algorithms do not need the knowledge of the Markov decision process (MDP). Reinforcement learning differs from standard supervised learning in that correct input/output pairs are never presented, nor sub-optimal actions explicitly corrected. A reinforcement learning agent interacts with its environment in discrete time steps. At each time t, the agent receives an observation o_t, which typically includes the reward r_t. It then chooses an action a_t from the set of actions available, which is subsequently sent to the environment. The environment moves to a new state s_{t+1} and the reward r_{t+1} associated with the transition (s_t, a_t, s_{t+1})

is determined. The goal of a reinforcement learning agent is to collect as much reward as possible. The agent can choose any action as a function of the history and it can even randomize its action selection [9-13].

In computer science, pattern matching is the act of checking some sequence of tokens for the presence of the constituents of some pattern. In contrast to pattern recognition, the match usually has to be exact. The patterns generally have the form of either sequences or tree structures. Uses of pattern matching include outputting the locations (if any) of a pattern within a token sequence, to output some component of the matched pattern, and to substitute the matching pattern with some other token sequence (I.e., search and replace). Graph rewriting languages rely on pattern matching for the fundamental way a program evaluates into a result [14-15].

This research proposed a novel technique by combining the pattern matching and reinforcement learning techniques to predict the user next action, which was never been thought of using earlier. By applying reinforcement learning the intelligent home system can receive positive reward for each action that is correctly predicted, on the other hand negative reward is given for each wrong action. By using this method the system can adapt the user ideal actions. On the other hand pattern matching will be used to match the most recent user event sequence with the history of the recorded sequences, by combining the results obtained from both methods the algorithm should have sufficient information to calculate the probability of the next user action thus the performance improvement should be significant.

DESIGN METHODOLOGY

The goal of the proposed algorithm is to predict the user future actions and to base the prediction decision on the user's preferences. The first thing that has to be taken into consideration in designing any decision algorithm is how the state of the environment will be represented. The algorithm was designed with the aim to employ the algorithm for a smart-home system, therefore the environment representation is based on smart-home environment.

The environment of the home can include massive amount of information such as room temperature, time (time of the day and the date of the month), devices states, user location and many other parameters that can be sensed by the inhabitant. For the purpose of simplification the explanation will take into consideration the time and the devices state as the state of the environment, and the action performed on these devices will be simply a device turning switching On or OFF actions.

Graph theory will be used to visualize the representation of the environment states and the actions [16]. According to graph theory, a directed graph G is an ordered pair of nodes N and vertices or arrows V as expressed by $G := (N, V)$.

According to this representation each state will be represented as a node in the graph and the action that is performed in that particular state will yield an edge that will produce another state in the environment as shown in Figure 1.

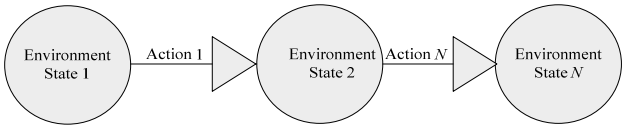

Figure 1. Environment state and action representation using graph theory

It is worth mentioning that although the representation of the environment state is simplified it can be easily extended to include more information. But it must be taken into consideration that by including more environment variables will require more storage and processing time however prediction accuracy will be improved since the inhabitant interaction history will be significantly more comprehensive.

ALGORITHM STRUCTURE

After illustrating how the environment state and the actions will be represented to clarify the internal construction of the algorithm. The algorithm has two main components and they are:

- *The reinforcement learning component.*
- *The pattern matching component.*

The Reinforcement Learning Component

The reinforcement learning component is modeled using Q-learning algorithm. Q-learning algorithm is a recent form of reinforcement learning technique that can be implemented to support online decision making and does not need a model of the environment, which makes it a great choice for Smart home system because Smart home system is a real-time application.

Q-Learning is a reinforcement learning technique that works by learning an action-value function that gives the expected utility of taking a given action in a given state. Given the environment devices states as [S], and the device actions that can be taken on a given environment state as [A], then Q value array of reinforcement learning can be formed as shown in Equation 1.

$$Q = S \times A \qquad (1)$$

According to Q-Learning algorithm the Q value array is used to store the rewards the agent has received by performing a particular action at a given environment state. Each time the agent makes a correct decision, the agent is given a positive reward or a negative reward. The reward is calculated based on the user feedback to the agents performed action, which can be sensed by the system through monitoring the devices state constantly. The Q value function will be calculated as shown in Equation 2.

$$Q^*(x,a) = (1-\alpha)Q^*(x,a) + \alpha(r + \gamma V^*(y)) \qquad (2)$$

Where Q is the Q-learning value function, x is the environment states, a is the action that can be taken,* α *is the learning rate,* γ *is the value of future reinforcement, r is the immediate reward, y is the next state resulting from taking action in state x and V* is the future Q-learning value function.*

The uniqueness is that Q-learning does not specify which action to be taken, it also allow us to perform experimental trials while preserving the current optimal state. Furthermore, since this function is

updated according to the ostensibly optimal choice of action at the following state, it does not matter what action is actually followed at that state. For this reason, the estimated returns in Q-learning are not contaminated by "experimental" actions so Q-learning is not experimentation-sensitive [17].

The Pattern Matching Component

The pattern matching of the component of the algorithm is inspired by method used in ONSI algorithm [2]. In ONSI algorithm, a pattern matching is extracted from the user interaction history and summary of those pattern is formed, then those pattern summaries is ranked accordingly and the prediction is made based on the ranking. The algorithm presented in this paper follows the same approach used by ONSI for pattern matching. The algorithm maintains two main directed graphs data structure, multiple graphs of device usage history sequence and a graph of the most recent device usage sequence. The history sequence is an ordered set of device usage history that has occurred at a particular environment state. The history sequence is used to train the algorithm. During the training state of the algorithm, the Q value array of the reinforcement learning algorithm is initialized based on the user selected action in the history. This will make the algorithm learn the optimal user actions for a given state.

The other graph of the algorithm is the recent device usage patterns that are supposed to be sensed by the smart-home system. Each graph will represent the device usage for one particular day. Therefore, the recent device usage graph will store the usage patterns for the current day that the smart-home system is running at.

The pattern matching operation starts from the most recent state recorded in the recent graph and going backward to match with each history graph. This operation will continue until the history graphs are completely searched for pattern matching. After the last iteration of this process is reached the longest pattern matching is calculated. This operation can be seen in Figure 2. As shown in Figure 2, the yellow nodes in the graph represent the pattern matched in the history graphs based on the recent graph starting at node with state S4. The pattern match will be taken into account only if the successor state, the predecessor state and the action between them is similar to the most recent graph states actions pairs.

The pseudo code of the algorithm is shown in the following steps:

i. Let α is the threshold value $0 \le \alpha \le 1$, $f(s,a)$ is the action under consideration under the latest current state in the recent graph of event sequence.
ii. Calculate the longest pattern match starting from the current state in the recent graph of event sequence and store it in variable L.
iii. Calculate the Q-value function for the current action under consideration based on the previous history of events using and store it in variable R.
iv. $h(s,a)$ is the total number of occurrences of action a under the current state s and $\sum_i h(s,a_i)$ is the calculated total number of occurrences of other actions a_i at the same state s
v. Calculate the probability for the current action a under consideration using the formula

$$f(s,a) = (1-\alpha)L + \alpha\left\{\frac{h(s,a)}{\sum_i h(s,a_i)} + R\right\} \qquad (3)$$

vi. Repeat steps 2 to 5 for the subsequent actions for the same state s and select the action with the highest ranking.
vii. After all the actions have been taken under consideration receive feedback from the home inhabitant. If the action was satisfying give the action a positive reward otherwise give the action a negative reward while taken the user chosen action under consideration. Calculate the Q-value for the selected action by the algorithm using Q-learning equation (2).

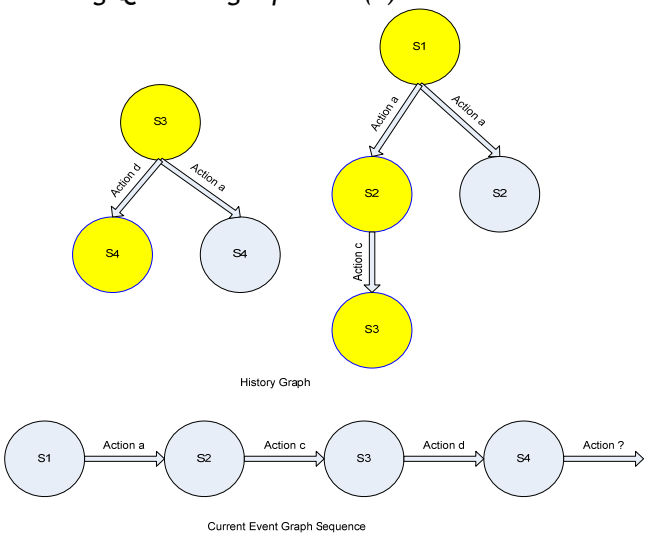

Figure 2. Illustrating pattern matching operation

TESTING RESULTS AND COMPARISON

To test the algorithm, the JAVA programming language is used to implement the testing module for the algorithm. The data used for testing the algorithm has been generated synthetically to simulate device usage patterns of home inhabitant. The data was generated with 1 months usage of 5 devices. The total data volume was 4800 with random actions. The data then has been formulated and stored in a database file then fed into the testing module to train the algorithm and the Q-learning module. After the training phase pattern, testing is performed with carefully analyzing the result and the synthetic generated patterns. In the testing it is assumed that the home consist of five devices and the actions that can be taken on these devices is simply device (On, Off) actions. Sample data format is shown in Table 1.

Table 1. Sample synthetic data used to train the multi-agent system

Date and Time	Action	Device	Location
2011-03-03 / 09:21	On	Lamp1	Living Room
2011-03-03 / 10:26	Off	Fan1	Bedroom
2011-03-03 / 10:29	On	Tv1	Living Room
2011-03-03 / 18:21	Off	Lamp2	Living Room
2011-03-03 / 20:22	Off	Tv2	Bedroom

Random trials have been performed to set the most suitable threshold value a. The optimal result was found while the threshold value was set to 6.0. Figure3 shows the results together with the comparison of results from other researchers [2, 18].

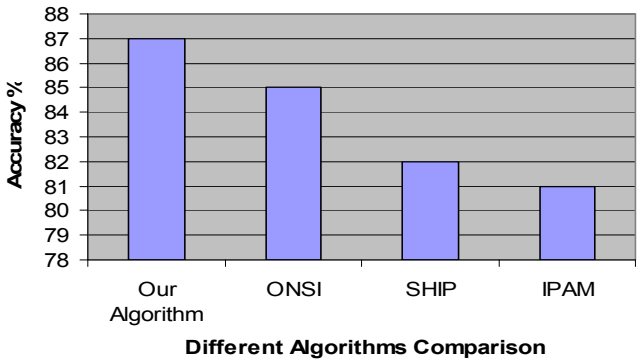

Figure 3. Result comparison with different prediction algorithms

The formula used to rank the accuracy of the algorithms is shown below:

$$f(h) = \frac{i}{q} \qquad (4)$$

where i is the number of accurate predictions and q is the number of input data that is fed to the algorithm.

Using Equation 4, the result shows that the accuracy of the proposed algorithm is 87%, where the accuracy of ONSI algorithm is 85%, accuracy of SHIP algorithm is 82% and the accuracy of IPAM algorithm is 81%, which is graphically represented in Figure 3. Therefore, by combining the pattern matching and reinforcement learning techniques in predicting the user next action of smart home devices clearly shows that it gives better accuracy than others.

CONCLUSIONS

This paper has presented a novel technique of user action prediction for smart home system. The algorithm introduces a new technique of adapting the user proffered actions using a combination of pattern matching and reinforcement learning. The result shows that the algorithm performs better than ONSI, SHIP and IPAM, which validate the supremacy of the proposed technique compare to others.

REFERENCES

[1] *ALAM, M.R., REAZ, M.B.I., ALI, M.A.M., SAMAD, S.A., HASHIM, F.H., HAMZAH, M.K. Human activity classification for smart home: A multiagent approach. In IEEE Symposium on Industrial Electronics & Applications (ISIEA), Penang (Malaysia), 2010, p. 511 - 514.*

[2] *GORNIAK, P., POOLE, D. Predicting future user actions by observing unmodified applications. In Proceedings of the Seventeenth National Conference on Artificial Intelligence, Cambridge, MA, 2000, p. 217-222.*

[3] *EDWIN, O.H., COOK, D.J. Improving home automation by discovering regularly occurring device usage patterns. In Third IEEE International Conference on Data Mining. Washington DC (USA), 2003, p. 537- 540.*

[4] *RASHIDI, P., COOK, D.J., HOLDER, L.B., SCHMITTER-EDGECOMBE, M. Discovering activities to recognize and track in a smart environment. IEEE Transactions on Knowledge and Data Engineering, 2011, vol. 23, no. 4, p. 527 - 539.*

[5] *WU, C. L., FU, L.C. Design and realization of a framework for human-system interaction in smart homes. IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, 2011, vol. PP, no. 99, p. 1-17.*

[6] *RAO, S.P., COOK D.J. Identifying tasks and predicting action in smart homes using unlabeled data. International Journal on Artificial Intelligence Tools, 2004, vol. 13, no. 1, p. 81-100.*

[7] *VINTAN, L., GELLERT, A., PETZOLD, J., UNGERER, T. Person movement prediction using neural networks. In Proceedings of the KI2004 International Workshop on Modeling and Retrieval of Context (MRC 2004). Ulm (Germany), 2004, vol. 114, p. 618-623.*

[8] *KANG, W., SHINE, D., SHIN, D. Prediction of state of user's behavior using Hidden Markov Model in ubiquitous home network. In International Conference on Industrial Engineering and Engineering Management (IEEM), Singapore, 2010, P. 1752 - 1756.*

[9] *KAELBLING L.P., MICHAEL L.L., ANDREW W.M. Reinforcement Learning: A Survey. Journal of Artificial Intelligence Research, 1996, vol. 4, p. 237-285.*

[10] *SUTTON R.S. Learning to predict by the method of temporal differences. Machine Learning, 1988, vol. 3, p. 9-44.*

[11] *BUSONIU L., ROBERT B., BART D.S., DAMIEN E. Reinforcement Learning and Dynamic Programming using Function Approximators. Taylor & Francis, CRC Press, 2010.*

[12] *SUTTON R.S., BARTO A.G. Reinforcement Learning: An Introduction. Cambridge: MIT Press, 1998.*

[13] *ZHANG Z.C., HU K.S., HUANG H.Y., LI S., ZHAO S.Y. (2010). A multi-step reinforcement learning algorithm. Applied Mechanics and Materials, 2010, p. 3611-3615.*

[14] *TROCHIM W. M. K. Pattern Matching, Validity, and Conceptualization in Program Evaluation. Evaluation Review, 1985, vol. 9, p. 575-604.*

[15] *LIN J. W. (2011). Neural network model and geographic grouping for risk assessment of debris flow. International Journal of the Physical Sciences, 2011, vol. 6, No. 6, p. 1374-1378.*

[16] *DIESTEL, R. Graph Theory. Springer: 2005.*

[17] *SUTTON, R.S., BARTO. A.G. Reinforcement Learning: An Introduction. Cambridge: MIT Press, 1998.*

[18] *DAS, S.K., COOK, D.J., BATTACHARYA, A., HEIERMAN, E.O., TZE-YUN, L. The role of prediction algorithms in the Mavhome smarthome project. IEEE Wireless Communications, 2002, vol. 9, no. 6, p. 77- 84, December 2002.*

ACTA TECHNICA CORVINIENSIS - Bulletin of Engineering

ISSN: 2067-3809 [CD-Rom, online]