

Adaptive P2P multimedia communication using hybrid learning*

Juan A. Botía, Pedro Ruiz**, Juan A. Sánchez, Antonio F. Gómez Skarmeta

Departamento de Ingeniería de la Información y las Comunicaciones.
Universidad de Murcia

Abstract. Multimedia communication in mobile and ad hoc networks used by real time applications can be improved by adding intelligent and adaptive capabilities. This new functionality will allow them to adapt to constantly and unpredictably changing network conditions. Derived from this adaptivity, the user will perceive a more or less constant quality instead of the high variable quality perceived in nowadays applications. In this work, we maintain the following thesis: both machine learning and intelligent agents will play an important role in the improvement of the applications we mentioned above. Machine learning, by means of reinforcement learning will provide adaptivity. Intelligent agents will ease P2P computation. This paper focuses on approaches for both topics.

1 Introduction

A mobile ad hoc network (MANET) is a spontaneous association of terminals equipped with wireless interfaces, which form a network. These networks do not require any infrastructure and all the network-layer functions need to be distributed among each of the different nodes. For example, when two distant nodes need to communicate, intermediate nodes act as relays so that multi-hop paths can be created. So, ad hoc nodes perform the functions both of host and routers. These networks are characterized by continuously and unpredictably changing network conditions, mainly due to the movement of the nodes (which provokes topology changes), and other issues at the lower layers like fading, collisions, etc. Traditional real-time multimedia applications are unable to perform well over these networks, and some adaptive functionalities are required at the application layer, to deal with such problems. These new applications called "adaptive applications" are challenged with new components to detect the current network conditions and adapt their internal settings (e.g. audio codecs, video rates, etc.) accordingly.

The main focus of traditional multimedia applications is the reduction of the data rate when the network bandwidth becomes scarce, and the increase of the

* Work supported by the FIT-070000-2003-662, FIT-1603002003-41 and TIC2002-04021-C02-01 projects).

** This work has been partially funded by Spanish Ministry of Science and Technology (MCYT) under the Ramón y Cajal Programme.

data rate whenever more resources become available. Of course, this behaviour improves the QoS perceived by the user. However, the relation between user-perceived QoS and the data-rate required to achieve that QoS is not linear. So, when the network conditions become very bad, a correct change in the internal application settings, could greatly reduce the data rate, while keeping the QoS to an acceptable level. The main problem, is that for these applications to do that, they have to be aware of the user-perception of QoS. This modeling is very complex because it usually has subjective components which cannot be modeled analytically.

In this work we propose an hybrid approach for the design of a mechanism in charge of managing the configuration of a multimedia peer-to-peer application. This configuration mechanism seeks for the best user satisfaction. The hybridization must be understood in terms of the different machine learning techniques [10] we use. We will first obtain an inductive model, using supervised learning, to predict the user perceived quality given concrete network conditions and multimedia application settings. Following, once we are able to score a concrete situation, we will apply reinforcement learning [12] to learn a strategy to decide when and how to change application settings, taking into account the score of the inductive model.

Intelligent agents [14] will be used to ease the control of the P2P multimedia applications. We also propose in this work the use of FIPA (*Foundation for Intelligent Physical Agents*) agents to wrap the elements afore mentioned and to seamlessly integrate them in previously existing multimedia applications.

The rest of the paper is structured as follows: section 2 explains the problem we are faced to. Section 3 introduced the hybrid approach we have used to obtain the adaptive mechanism. Following, section 4 briefly presents the agents architecture we use to integrate the adaptivity in multimedia applications. Finally, section 5 expose lessons learned in this work and pending tasks.

2 Adaptive multimedia applications

Quality of Service (QoS) as defined in ITU-T recommendation E.800, ITU-E.800 [2] is “the collective effect of service performance, which determines the degree of satisfaction of a user of a service”. It is characterized by a combination of service performance factors such as operability, accessibility, retainability and integrity. Thus, it is clear that the user plays an important role in QoS evaluations.

We will start by introducing the application architecture [4]. Main building blocks of it appear at figure 1. The main items in this architecture are the following: (a) multimedia application components like audio, video, slides for a remote presentation, etc., (b) the QoS signaling mechanism and (c) the adaptation logic. The QoS signaling mechanism is the protocol in charge of sending and receiving reports describing the network conditions from the other end. When such a report is received it is passed to the Adaptation Logic as an additional input. Additionally, the Adaptation Logic is in charge of deciding which set of parameters is best suited to the current network conditions.

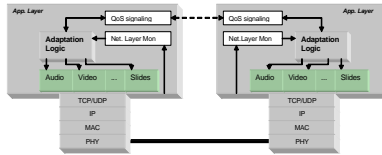


Fig. 1. General adaptive architecture

The most difficult part for the Application Logic is the decision on what components to adapt, and which setting to change, when the application is exceeding the available bandwidth. Many adaptive applications to date, just reduce data rates to use less than the available bandwidth. However, they do not deal with the main difficulty, which is taking the correct decision while taking into consideration the subjective user-perception implications about it.

3 Hybrid learning for P2P control

In this section we will introduce the scheme we have designed to perform hybrid learning. Section 3.1 will introduce the supervised learning part and section 3.2 will present the learning by reinforcement part.

3.1 Supervised learning for user modeling

In order to inductively model the QoS perception of an user, we have to produce a learning data set. And this has to be compound by examples of situations referring to a particular network condition and a particular multimedia application sending to and receiving data from the network. Network conditions have been reproduced by using a reflector. This is a software tool collocated in the middle of a dedicated link between two communicating nodes. It will be in charge of simulating different levels of available bandwidth and estimating packet losses. The multimedia application used, ISABEL-Lite, is a reduced version of ISABEL [1] which allows both manual and automatic change of its settings. This settings must be understood in terms of audio and video codecs. Audio and video codecs are in charge of capturing, coding, sending, decoding and presenting audio and video data respectively. More specifically, for the video we can also specify the size, the number of frames per second sent and a quality factor.

Table 1 summarizes all attributes and the corresponding range of values, which compound the data set used to model the user. Notice that the last row refers to the score given by the user.

The data set consists of 864 instances, each one scored by an user. The data set can be considered to be balanced with the following distribution of examples by score: 241 (27.8%) examples with score 1, 83 (10.4%) for score 2, 181 (20.9%) examples with score 3, 233 (26.9%) with 5 and finally, 125 (14.46%) for the highest score.

Parameter	Values	Explanation
BW	{33, . . . , 384}	Limit of network bandwidth
LOSS	0..100	% loss packets
AUDCOD	PCM, G711-u, G722, GSM	Audio codec
VIDCOD	MJPEG, H.263	Video codec
FSIZE	CIF, QCIF, 160x128	Size of video frames
QFVIDEO	5, 10, 15, 30, 60	Quantify factor of video codec
FPS	{0, . . . , 12}	Frames by second sent
QoS	1, 2, 3, 4, 5	User perceived quality

Table 1. Parameters appearing at the example set used for rule induction by SLIPPER.

Learning experiments have been performed using SLIPPER [6]. This algorithm does not directly use the classic search bias of divide and conquer for rule induction. Instead, it bases its strategy on boosting [7]. It uses a weak learner (i.e. a very simple rule induction algorithm) which boost by modifying learning instances probability each iteration to focus on instances not correctly classified yet. In fact, we also tested IREP, IREP* [8] and RIPPER [5]. Former algorithms which do not use boosting and all of them under-performed SLIPPER. We will consider the possibility of using other kinds of algorithms like, for example, ordinal regression ones. This algorithms predict discrete outputs taking into account a given order between values.

Best classification capacity model we have obtained with SLIPPER appears at figure 2. Ten fold crossvalidation gave a missclassification error of 10%. It is compound by 12 rules and an example is classified as pertaining to the first class (from the upper to the lower class) in which the sum of the confidence values of matching rules in the class is higher than the corresponding negative value. Using that model we can approximate user perceived QoS. However, what we really need is a mechanism to decide, when thing go wrong in the session, what changes have to be applied to the application settings. For that, we will use reinforcement learning.

3.2 Reinforcement learning for adaptivity

In this section we will introduce our approach to obtain the adaptivity scheme. This will be in charge of deciding when and how to change the configuration of our multimedia application to obtain, in the long term, an optimum user satisfaction. The decision model we will obtain will be a multi-layer perceptron [3]. Learning the parameters (i.e. the weights of the arcs in the network) is done by reinforcement learning [13].

In learning by reinforcement, we make use of an entity called agent [12] which is situated in the environment. Its situation comes depicted by the environment particular state at time t , let it be denoted with s_t . The learner agent can perform a set of actions in the environment. Each time the agent executes an action a , it receives a reward from the environment, r . The agent then has to appropriately choose each action it executes in order to maximize the reward obtained at the long term. In the context of this particular application, the agent will learn a

```

if matchConfidence {
  [QFVIDEO >= 60, VIDCOD = MJPEG, FSIZE = QCIF, LOSS <= 10, FPS >= 6] -> 2.8792
  [AUDCOD = GSM, BW >= 80, QFVIDEO >= 30, FSIZE = QCIF, FPS <= 6] -> 1.4357
  [AUDCOD = GSM, BW >= 128, LOSS = 0, QFVIDEO >= 30, FPS >= 3, VIDCOD = MJPEG]
  -> 1.7013
  [] -> -2.4188
} > 0 then 5 else if matchConfidence {
  [BW >= 384, QFVIDEO >= 40, FSIZE <= 2] -> 2.7121
  [QFVIDEO >= 30, VIDCOD = MJPEG, LOSS <= 3, AUDCOD = G722] -> 1.1756
  [FSIZE = CIF, QFVIDEO >= 30, LOSS <= 3, AUDCOD = G722, BW >= 80] -> 1.4437
  [] -> -1.5044
} > 0 then 4 else if matchConfidence {
  [LOSS >= 30] -> 2.1188
  [QFVIDEO <= 5] -> 1.4142
  [LOSS >= 16, FPS <= 3] -> 1.5438
  [] -> -1.0984207275826066
} > 0 then 1 else if matchConfidence {
  [LOSS >= 16] -> 1.9109
  [QFVIDEO <= 10, FSIZE = QCIF] -> 1.5861
  [FSIZE = 160X128, QFVIDEO <= 40,
  VIDCOD = H.263] -> 1.2546
  [] -> -0.3953
} > 0 then 2 else 3

```

Fig. 2. Rule model to estimate user perceived QoS

state-value function, let it be denoted with $V^\pi(s_t)$. This function will be used to predict the long term reward the agent would obtain if, being at time t , it selects the action given by the policy π (i.e. the criteria used to select an action among all the possible ones). This approach is typically used for prediction but we will use it for control. In typical control problems, not only the state is taken into account in the value function but also the actions. This time, the agent has to learn a good approximation of a function, let it be denoted with $Q^\pi(s, a)$ for the current policy π and for all states s and actions a .

Learning is done by iteratively updating the Q function by means of the following expression:

$$Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma Q(s', a') - Q(s, a)],$$

where the pair (s', a') refers to the state s' to which the environment goes to, from s when action a is executed and a' is the action executed at state s' . Constants α and γ are the learning rate and the discount factor, respectively.

In this particular domain, we directly act on the problem. For example, a possible action could be to set the video codec to MJPEG or either to change video size to QCIF or even do nothing at all. Immediate rewards, obtained from executed actions, will be approximated by using the rules model of figure 2.

The estimator will be learnt by using SARSA, without using the eligibilities mechanism (i.e. $\lambda = 0$, see [12], pag. 163). Eligibilities speed up convergence to a good approximation of the Q function, however, in previous simulations we did not perceive any improvement in using that technique.

A world state will be given by concrete network conditions as simulated by the reflector (i.e. packet losses), and settings of the multimedia application. Consequently, an action will be a change in the state (except by packet losses, which

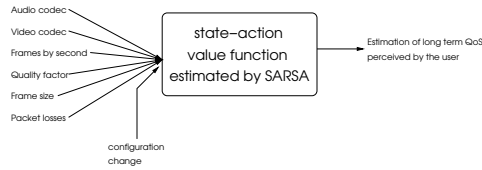


Fig. 3. Functional diagram for the adaptive strategy to be learnt by SARSA

would be given by the reflector). It must be noticed that available bandwidth can not be considered as another dimension of the environment features vector as this parameter can not be obtained from the real application. It can only be simulated with the reflector. A functional scheme, in terms of modules inputs and outputs appears depicted at figure 3.

Again, we have used the ISABEL Light Videophone along with the reflector to reproduce a real application and network conditions the agent will use to learn from. The videophone simulates both communication end points. It reproduces, with no end, a musical clip with a total of 400 video frames which sends to itself through the reflector. This, in turn, is in charge of simulating the network link between the end points. Each one of the learning trials or episodes uses the same bandwidth values changing through time. Bandwidth follows the curve appearing at figure 4. Notice that the density of different bandwidth values grows while approximating to values between 256 and 0KB. That is because this range of values are more sensitive to changes. Depending on the amount of data injected

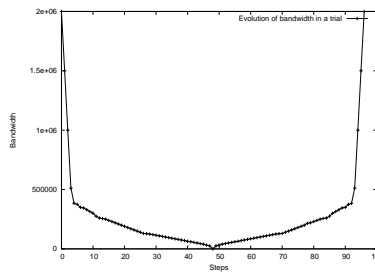


Fig. 4. Bandwidth evolution used in SARSA trials

to the network by the videophone and on the reflector simulated bandwidth, the reflector will generate a concrete packet losses percentage. Both the videophone and the reflector accept commands from outside through a socket. By opening telnet connections to the videophone we can configurate application settings. For example, with the string `set(AUDIO::PCM,VIDEO::MJPEG,QCIF,8,5.0)` we tell the videophone to set the audio codec to PCM, the video codec to MJPEG, the

video size to QCIF, the frame rate to 8 and the quality factor to 5. Bandwidth can be set at the reflector in the same way. We can also read packet losses from the reflector by using a *read* command with the same socket. Communication through sockets is important here because the learning scheme has been developed by using Java and the reflector and videophone are developed in C++. The learning global scheme appears at figure 5.

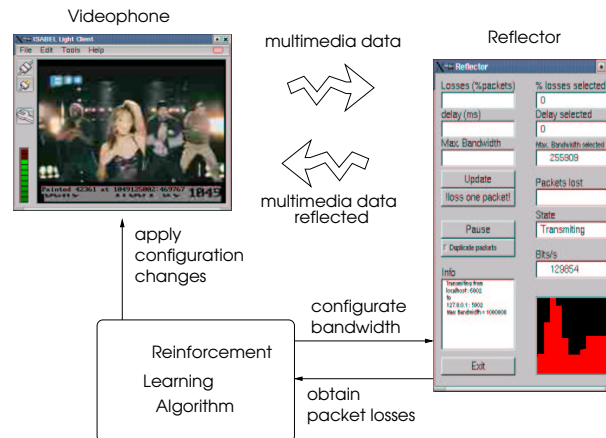


Fig. 5. Arquitectura de aprendizaje global con videófono, reflector y algoritmo SARSA

We have obtained the best results with a multilayer perceptron of 30 hidden nodes, a reinforcement learning rate of 0.05 and a discount factor of 0.9, being the learning rate for the neural network of 0.01. Each one of the episodes was compound by 970 movements (10 actions for each different bandwidth value appearing at figure 4).

Effective learning progress will be demonstrated by using a number of different curves. The first one corresponds to the accumulated r values obtained in each episode, for all s_{t+1} visited states. It is labeled with (a) at figure 6. Notice that, from the very beginning, it grows until it becomes stable. Another interesting point is shown by (b) curve. It refers to packet losses. In the beginning, it shows a minimum level of packet being lost in the network. This is because the learner stands at very conservative states (i.e. states using a low bandwidth and, subsequently, low user scores are obtained). As long as the learning process evolves, packet losses ratio becomes stable between a 3 and a 5% (an acceptable value). RMSE decreases, as expected. Regarding the five curves appearing at graph (d), we can see how that curves which represent low scores (1 and 2 scores) decrease. Also, the number of actions with good quality increase (i.e. 3 scores) and the amount of good and very good quality actions increase.

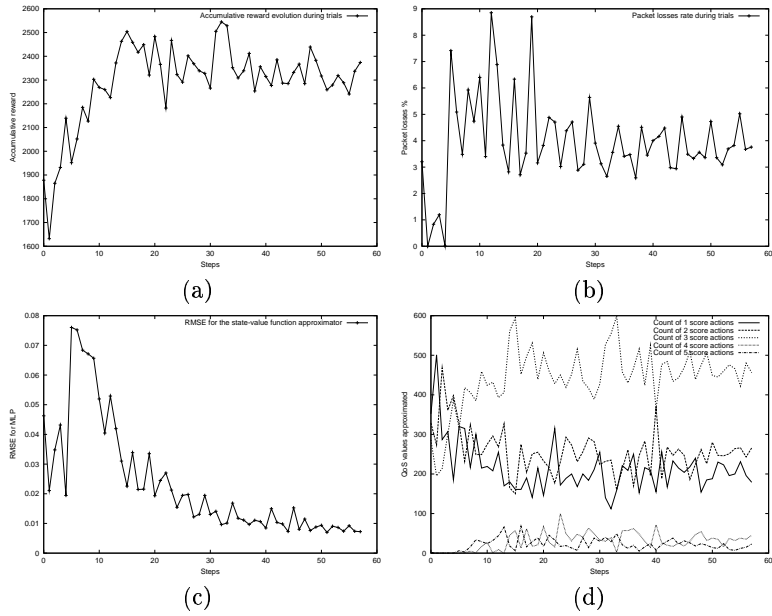


Fig. 6. Curve labeled with (a) shows the evolution of accumulated rewards through episodes. The one labeled with (b) represents the packet losses percentage. The (c) curve is the evolution of the Rooted Mean Squared Error (RMSE) for the multilayer perceptron which approximates the estate-value function. At (d) we have the number of each different scores obtained during learning

4 P2P Control Implementation with FIPA Agents

In this section we will explain an initial implementation we have developed for the adaptive control of the videophone. Our long term goal is to built a complete ambient intelligence [9] system. This concept emphasize the context in which the user is situated. This context depends on the device through which he is connected to the system, the physical network, his personal interests (i.e. his user profile) and available services at the current context. Now, we are working on the first and second mentioned factors.

If we want to provide an implementation of the adaptive level of the architecture (see figure 1) based on intelligent agents, we have to revise the QoS basic signalling mechanism along with the adaption logic (a detailed analysis of the issue can be found at [11]).

The QoS signalling mechanism provides with information of the network state to the other end point at the communication channel. A point to point transport mechanism can be defined in charge of informing the transmitter about the quality the receiver is obtaining. This QoS can be compound by the packet losses ratio and the mean jitter. Each one of the QoS signalling packets can be added a sequence number and the estimated bandwidth. We can express this kind of packets with XML, like in the following example:

```
<?xml version="1.0" encoding="UTF-8"?>
<qosreport>
  <sequence>34</sequence>
  <lostpackets>9.3465</lostpackets>
  <delay>0.093</delay>
  <preferences></preferences>
  <estimatedbw>128000</estimatedbw>
</qosreport>
```

in which we have included an empty preferences part. Agents simply have to exchange messages like that, by using a performative to convey them.

To this moment, implementation is being carried out on laptops, by using videophones coded in C++ and intelligent agents with the JADE platform.

Adaption logic is in charge of deciding when and how to act on application settings. This functionality is given by the multilayer perceptron that, for each action, produces an estimate of how good it will be in the long term. The decision mechanism consists on choosing the action with highest return value. When an `inform` message is received with a `<qosreport>` content, we use the neural network and apply changes.

5 Conclusions

In this work, we have presented an hybrid approach based on both supervised and reinforcement learning. This has been used to obtain and adaptation mechanism to maintain an acceptable QoS in the context of multimedia applications like a videophone. We also outlined initial details of the FIPA agents based architecture to provide a complete ambient intelligence application. Results still can be improved. A possible improvement is that of using ordinal regression models instead of classification ones to approximate the quality perceived by the user.

In this way, error estimations would be more precise as score labels are ordered. However, and with no doubt, this work supposes a very promising start point with respect to the role that artificial intelligence will play in the improvement of ad hoc networks communication. Moreover, another possibility is using simple regression for the same problem.

References

1. *The ISABEL CSCW application.* [On line] <http://www.agora2000.com/productos/isabel.html>.
2. Recommendation E.800 (08/94). *Terms and Definitions Related to Quality of Service for Adaptable Multimedia Communication.* ITU-T, 1994.
3. Christopher M. Bishop. *Neural Networks for Pattern Recognition.* Clarendon Press, Oxford, 1995.
4. Juan A. Botía, Pedro Ruiz, Jose Salort, and Antonio Skarmeta. Improving user-perceived qos in mobile ad hoc networks using decision rules induction. In *Proceedings of the Canadian Conference of Artificial Intelligence*, Ontario, Canada, June 2003.
5. W. Cohen. Fast effective rule induction. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 115–123, Lake Tahoe, CA, 1995.
6. W. Cohen and Y. Singer. A simple, fast, and effective rule learner. In *Proceedings of the Conference of American Association for Artificial Intelligence*, 1999.
7. Yoav Freund and Robert E. Schapire. A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, 14(5):771–780, September 1999.
8. Johannes Fürnkranz. Separate-and-conquer rule learning. *Artificial Intelligence Review*, 13(1):3–54, February 1999.
9. W.A IJsselsteijn G. Riva, F. Davide, editor. *Being There: Concepts, effects and measurement of user presence in synthetic environments.* IOS Press, Amsterdam, The Netherlands, 2003.
10. Tom M. Mitchell. *Machine Learning.* McGraw-Hill, 1997.
11. Pedro M. Ruiz. *Multicast Architecture for MANET Extensions to Fixed IP Networks Supporting Real-Time Adaptive Applications.* PhD thesis, University of Murcia, 2002.
12. R. Sutton and A. Barto. *Reinforcement Learning. An Introduction.* MIT Press, 1998.
13. Richard S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44, 1988.
14. Michael Wooldridge. *An Introduction to MultiAgent Systems.* Willey, 2002.