Users and Services in Intelligent Networks

Erol Gelenbe
Dennis Gabor Chair
Intelligent Systems and Networks
Electrical & Electronic Engineering Dept.
Imperial College, London SW7 2BT
e.gelenbe@imperial.ac.uk

***Abstract*— We present a vision of an Intelligent Network in which users dynamically indicate their requests for services, and formulate needs in terms of Quality of Service (QoS) and price. Users can also monitor on-line the extent to which their requests are being satisfied. In turn the services will dynamically try to satisfy the user as best as they can, and inform the user of the level at which the requests are being satisfied, and at what cost. The network will provide guidelines and constraints to users and services, to avoid that they impede each others' progress. This intelligent and sensible dialogue between users, services and the network can proceed constantly based on mutual observation, network and user self-observation, and on-line adaptive and locally distributed feedback control which proceeds at the same speed as the traffic flows and events being controlled. We illustrate these concepts via an experimental test-bed at Imperial College, based on the Cognitive Packet Network (CPN), that embodies some of these functionalities thanks to "smart packets" and reinforcement learning. At its edges, CPN is fully compatible with the IP protocol, while internally it offers routing that is dynamically modified using on on-line sensing and monitoring, based on users' QoS needs and overall network objectives.**

***Index Terms*— Network Intelligence. Autonomic Networks. Users and Services. User Goals and Quality of Service. Cognitive Packet Networks.**

I. Introduction

Sheer *technological capabilities and intelligence*, on their own, are of limited value if they do not lead to enhanced and cost-effective capabilities that are of value to human – or even beyond humans – to living users.

In the field of telecommunications, fixed and then mobile telephony and the Internet have been enablers for major new developments that improve human existence. However advances in telecommunications have also had some undesirable and unexpected outcomes during the past century. A case in point is television broadcasting. It was initially thought that television broadcasting would become a wonderful medium for education. Unfortunately in many instances it has lowered public standards for entertainment by forcing a limited number of programs upon the public; it has often displaced reading, sophisticated cinema, theatrical and musical forms by the introduction of facile talk shows and soap operas. This is a great example of a tremendous success in technology which has not been applied in the most broadly intelligent manner. The "one-to-very-many" broadcast nature of television does not give users, or communities of users, the possibility to significantly influence the system that they use. Other models of communications, such as the peer-to-peer concept which was born in the Internet, can offer a greater degree of user choice. Thus we suggest that through intelligent organisation of networks, which should include a just compensation for services and intellectual property ownership, one can achieve improved communications for the sake of an enhanced cultural and humanistic environment.

We envision Intelligent Networks (INs) to which users can ubiquitously and harmoniously connect to offer or receive services. We imagine an unlimited peer-to-peer world in which services, including current television broadcasts, voice or video telephony, messaging, libraries and documentation, live theater and entertainment, and services which are based on content, data and information, are available at an affordable cost. In these networks the technical principles that support both the "users" and the "services" will be very similar if they are framed within an autonomic self-managing and self-regulating system. In fact This network will be accessible via open but secure interfaces that are compatible with a wide set of communication standards, including the IP protocol.

We imagine an IN in which users and services play a symmetric role: users of some services can be services of other users, and services can be users of some other services. Users and services can express their requests dynamically to the network in terms of the services that they seek, together with Quality-of-Service (QoS) criteria that they need, their estimate of the quantity or duration of the requested service and the price that they are willing to pay. The users could also have the capability to monitor on-line to what extent their requests are being satisfied. In turn the services and the network would dynamically try to satisfy the user as best as they could, and inform the user of the level at which their requests are being satisfied, and at what cost. The network would also provide guidelines to users to avoid that the latter impede each others' progress. Similarly, network entities and services would also conduct a dialogue, so that they can collectively and autonomously provide a stable, evolving and cost effective network infrastructure. We will sometimes find it useful to distinguish between users and services, merely to indicate the relationship that exists between a specific user requesting a specific service. But we wish to stress that at a certain level of abstraction, these two entities are indeed equivalent.

This research was supported by the UK Engineering and Science Research Council under Grant No. GR/S52360/01 and by the EU Marie Curie Programme under project SAPAD No. MIRG-CT-2004-506602.

The IN will offer the facilities for an intelligent and sensible dialogue between all users, including services, and it will adapt to users' needs based on mutual observation, network and user self-observation, and on-line distributed feedback control which acts in response to the events that are being controlled.

II. An Architecture for the Intelligent Network

A sketch of the IN architecture is shown in Figure 1. The IN is based a standard communication interface inspired by the Internet Protocol (IP). Users U (shown with small purple rectangles as U1, U2, etc.) are generally mobile and can be recognised via their ID and password. Users have a credit with the network and with certain network services, as represented by a credit allocation or via a "pay as you go" scheme (e.g. with a credit card), or they can access certain free services or services that may be paid for by the service provider (e.g. advertisements). Users can have a user terminal which may be as simple as a Personal Digital Assistant or mobile phone, or as complex as intelligent network routers (INRs) shown as blue octagons in Fugure 1. Users are connected to the IN via INRs or directly to a network cloud (shown as clouds of different colours). Services S (shown as S1, S2, ..) are very similar

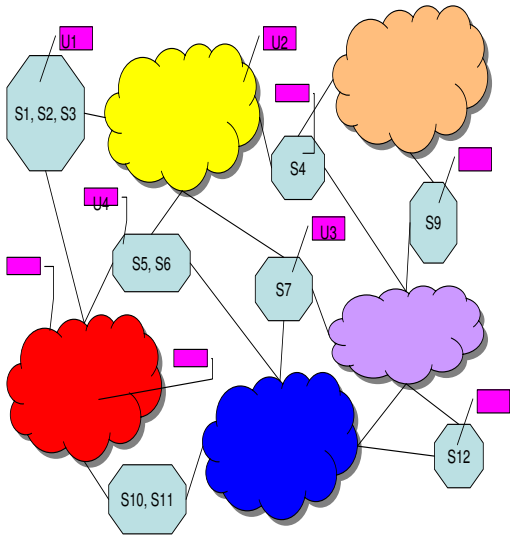

Fig. 1. Architecture of the Intelligent Network

to users in that they have an ID and they may have a credit allocation; they can also receive credit when their services are used by users, just as users may be reimbursed by services or by other users. Services can also be mobile. However:

- Users will in general be light-weight (a mobile phone, a PDA, or just a user ID and password),
- While services will be much more complex and may often be resident on one or more INRs, or they may own one or more INRs for their needs.

When some other user or service asks something of a user, the chances are that there will be an automatic answer saying "sorry no; I am just a simple user". On the other hand, services will often be equipped with authentication schemes to recognise the party who is making a request, billing schemes that allow for payment to be collected, schemes allowing a service to be used simultaneously by many users, and so on, depending on the complexity of the service being considered.

INRs are machines or clusters which can be identified by the community of users and services. Network clouds on the other hand are collections of routers internally interconnected by wire or wireless and which are only identified as far as the users and services are concerned via the ports of INRs which are linked to a cloud; in other words, users and services do not actually know who and what is inside a network cloud. However INRs, and hence users and services, can observe the QoS related to traversing a network cloud; this may include billing of the transport service by the cloud. Also, clouds may refuse traffic, or control and shape the traffic that wishes to access them, depending on the clouds own perception of the traffic.

The IN architecture we have described can be viewed as an overlay network composed of INRs with advanced search, QoS (including pricing and billing), that links different communities of users and services. The networked environment of the future will include numerous INs, and there may be specific INs whose role is to find the best IN for a given user. Some of the se INs may be quite small (e.g. a network for a single extended family), while others would be very large (e.g. a network that provides sources of multimedia entertainment, or educational content). In the three following sub-sections we will discuss three important enabling capabilities of the system: finding services and users, routing through the network, and self-observation and network monitoring to obtain the best QoS and performance.

A. Finding services and users

We expect that the IN will have different free or paying directory services that will be used to locate users and services. When appropriate, these directories may provide a "street address and telephone number" for a service that is being sought out; however, since in many cases the services will have a major virtual component, they will especially provide a way to access them virtually, either via an IP address, or more probably via one or more INR addresses or one or more network paths.

The directory services will offer "how to get there" information similar to a street map service, providing a network path in terms of a series of INRs or of network clouds, from the point where the request is made, to the INR where the service can be found. Directory services may have a billing option which is activated by services to reward the directory for being up-to-date, or services or users can subscribe to them, or they may be paid for via advertisement information, and so on. These directories will be updated pro-actively by the services or by the directories themselves, or on demand when the need occurs. Updates would also occur when INR or network cloud landmarks change.

Directories can be "smart" in the sense that they offer information about faster or less congested paths to services that are requested, or paths to less expensive services, or paths that

are better in some broader sense. An approach for achieving this based on the Cognitive Packet Network (CPN) [4], [5], [7], [10] protocol is described in Section III. If the user does not know how to find a service, it can broadcast its request which will be relayed by INRs and directories, again using a smart routing algorithm similar to CPN.

III. Smart search and routing

Let us go through the steps of the establishment of a connection between some user U and a service S:

- U first searches for a directory; assuming he finds one, U formulates his request in the form of (SX, QY, PZ) meaning that he wants a service SX at QoS value QY for a price of PZ. The directory either is unable to answer the request, or it provides one or more paths $\pi(U, SX, QY, PZ)$ which best approximate this request for several possible locations of the service.
- Assuming that the directory does provide the information, U sends out (typically via the INR) a sequence of smart packets SPs which have the desired QoS information, with several following each of the possible designated paths. The first SP for each of the paths will follow it to destination, with the purpose of verifying that the information provided by the directory is correct. Subsequent SPs on each route will be used to search for paths: they will invoke an optimisation algorithm at all or some of the INRs they traverse so as to seek out the best path with respect to the user's QoS and pricing requirements.
- INRs collect measurements and store them in mail boxes (MB). These can concern both short term measurements which proceed at a fast pace comparable to the traffic rates, and long term historical data. INRs will measure packet loss rates on outgoing links and on complete paths, delays to various destinations, possibly security levels along paths (when security is part of a QoS requirement), available power levels at certain mobile nodes, etc.. This constant monitoring can be carried out using the SPs and other user related traffic, or using specific sensing packets generated by the INRs.
- The network monitoring function can also be structured as a special set of users and services whose role is to monitor the network and provide advice to the users and to the directories.
- Each SP also collects measurements from the INRs it visits which are relevant to its users QoS and cost needs, about the path from the INRs which it visits.
- When a SP reaches a service SX, an acknowledgement ACK packet is sent back along the reverse path back to U; the ACK carries the relevant QoS information, as well as path information which was measured by the SP and by the ACK, back to the INRs and to the user U. The ACK may thus be carrying back a new path which was unknown to the directory.
- For a variety of reasons, both SPs and ACKs may get lost. SPs or ACKs which travel through the network over a number of hops (ERs or total number including routers within the clouds) exceeding a predetrmined fixed number, will be destroyed by the routers to avoid congesting the IN with "lost" packets.
- Note that the SPs and ACKs may be emitted by the directory itself, rather than by U. This would be an additional service offered by certain directories. One could also imagine that both users and directories have this capability so as to verify that the request is being satisfied.

A. Individual versus collective QoS goals

The usual question that any normally constituted telecommunications engineer will ask with respect to the vision that we have sketched is what will happen when individual goals of users and services conflict with the collective goals of the system. We are allowing for users to set up the best paths they can find, from a selfish perspective, with services, and for services to actually do the same, in parallel with the behaviour of users. This has the potential for:

- Overloading the infrastructure, because services have an interest in maximising their positive response to user's needs, and they may even overdo it in terms of solliciting users; because of the possibility of billing, portions of the infrastructure itself may have an interest in getting overloaded.
- Creating traffic congestion and oscillations between hot spots, as users and services switch constantly to a seemingly better way to channel their traffic.
- Opening the door to malicious traffic whose sole purpose may be to deny service to legitimate users through the focused creation of overload in the services or the infrastructure (e.g. denial of service attacks).

The first of these points, which does not relate to malicious behaviour, can be handled through overall self regulation of the INRs, the users and services:

- When a new part of the infrastructure joins the IN, for instance a INR, it will be allocated an identity within the IN. We could have a virtual regulating agency (VRA) which sets up a dialogue with the INR to provide it with its identity, and which ascertains its type and nature from its technical characteristics. The VRA then enables the INRs operating systm with a set of parameters which in effect limit the number of resident processes and the amount of packet traffic that this particular INR can accept.
- Services and users which join the IN, also need to be identified by the VRA. Just as a shop rents a certain space in a building and on a particular street, the VRA can provide the service with a "footprint", depending on the rent it is willing to pay, and on the VRA's knowledge of currently available resources. This footprint can then determine the fraction and amount of processing power and bandwidth that it is allowed inside the IN and at any given INR.
- Note that the overall quality and seriousness of the VRA will make a particular IN more or less desirable to users and services.

The second point is related to dynamic behaviour. Each INR, in its role as a service support centre enabled by the VRA, will run the dynamic flow and workload control algorithms for each service and user that it hosts. However it will also run a monitoring algorithm which has IN-wide implications.

- For some user U assume that $RU(S)$ is the rank ordered set of best instantaneous choices for some decision (e.g. what is the best way to go to service S with minimum delay).
- At the same time, let $RN(U,S)$ be the rank ordered set of best instantaneous choices for the network (e.g. what is the best way to go to where service S is "sitting" so that overall traffic in the IN is balanced).
- The decision taken by the INR will be some weighted combination of these two rank orders. The weights can depend on the priority of the user, of the price it is willing to pay, and so on.
- Choices which are impossible or unacceptable to either of the two criteria (user or network) will simply be excluded. If there are no mutually possible choices, then the request will be rejected. When there are ties between choices, any one of the tied choices can be selected at random.

As an example, suppose that the ranking indicating the user's preference, in desecending order, among six possible choices is $\{1,2,3,4,5,6\}$, while the network's preference ranking could be $\{5,4,2,3,1,6\}$. If we use rank order as the decision criterion and weigh the INR and the user equally, then the decision will be to choose 2 whose total rank order is 5. If the network's role is viewed as being twice as important, we can divide the network's rank for some choice by 2 and add the resulting number to the rank that the user has assigned to that choice, which results in a tie between the three top choices $\{1,2,5\}$. If the network's role is three times more important, then we get a tie for the top choice between $\{1,5\}$, and so on.

In the aproach that we have suggested for finding services, the user U formulates some request (SX, QY, PZ) for a service SX at quality level QY and for the price PZ. Both the quality of service value and the price constitute "goals" in the sense that the term is used in the CPN algorithm [7]. They may be treated as separate goals to be minimised, and combined in some manner as outlined above, or combined into some single common metric.

For instance, if QY is some non-negative number such as "loss" or "delay", we could combine the two considerations in a single metric such as $G = QY/PZ$ (quality for a given price), or as

$$G = PZ\ 1[QY < Qmax] + \frac{QY}{PZ} 1[QY \geq Qmax] \qquad (1)$$

where $1[x]$ is the function which takes the value one if the predicate $x = true$ and takes the value zero if $x = false$. Thus (1) means, for instance, that as long as the delay is less than some maximum acceptable value $Qmax$, we are happy to minimise the price; however if the delay is larger than this maximum value, we want simply to minimise the delay per price unit that we pay for the service.

B. The eternal problem of scalability

It is often said that the main impediment to the broad use of QoS mechanisms in the Internet is the issue of scalability. Indeed, if each Internet router were enabled to deal with the QoS needs of each connection, it would have to identify and track the packets of each individual connection that is transiting through it. The routing mechanism we propose for all requests through the IN is based on dynamic source routing[1]. In other words, the burden of determining the path to be used rests with the INR that hosts the service or user. In our proposed scheme, routers have two roles:

- The INR generates SPs for its own use that monitor the IN as a whole, and the user or service process resident at a INR generates the SPs and ACKs which are related to its connections to monitor their individual traffic.
- As a result of the information that it receives from SPs and ACKs, of the information similarly received by users and services that are resident at the INR, and of the compromise between global (IN) and local (user and service) considerations, the INR generates source routes for its resident users and services.
- Each INR also provides QoS information to SPs and ACKs that are not locally generated but which are transiting through it, such as "what is the loss rate on this line", or "what time is it here now", or "what is the local level of security".

Thus we propose to avoid the scalability issue by making each INR responsible only for local users and services, much as a local telephone exchange handles its local users. Source routing removes the burden of routing decisions from all but the local INR, reducing overhead, and removing the need of "per flow" information handling except at INRs where the flows are resident. However, it comes at the price of being less rapidly responsive to changes that may occur in the network. This last point can be compensated by constant monitoring of the flow that is undertaken with the help of SPs and ACKs. Our scheme also requires that INRs be aware of the overall IN topology in terms of other INRs (but there is no need to know what is inside the "clouds"), although this can be mitigated if one accepts the possibility of staged source routing, i.e. with the source taking decisions up to a given intermediate INR, which then takes decisions as far as some other INR, and so on.

IV. An Algorithm for Smart Search

We will now illustrate the search process by extending some ideas from the Cognitive Packet Network (CPN) algorithm [10], and its implementations [6], [7], [8] in test-beds at the University of Central Florida and at Imperial College. In CPN, the purpose of the search is to find a network destination (rather than a service), and SPs and ACKs in CPN play a role that is identical the one we described earlier, except that we are looking for a path to some destination which optimises a QoS requirement. Thus we can imagine that the CPN algorithm

[1]Note that MPLS is a form of distributed virtual source routing where label switching at each node maps virtual addresses into physical link addresses.

which runs at the packet transport level and finds destination nodes, can be abstracted to a higher level where it searches for services.

In order to provide a practical grounding for the preceding discussion, we will discuss how the CPN protocol currently runs. In CPN dumb packets (DPs) carry the payload traffic, while CPN routers are similar to INRs, and are interconnected either via portions of the Internet which plays the same role as the network clouds that we have described in Figure 1, or via point-to-point Ethernet or other (e.g. ATM) connections. SPs find routes and collect measurements, but do not carry payload.

DPs are source routed, using paths which best match the users' QoS requirements. On the other hand, SPs are routed using a Reinforcement Learning (RL) algorithm that uses the observed outcome of previous decisions to "reward" or "punish" the mechanism that lead to the previous choice, so that its future decisions are more likely to meet the QoS goal.

When a SP arrives to its destination, an acknowledgement (ACK) packet is generated; the ACK stores the "reverse route" and the measurement data collected by the SP. It will travel along the "reverse route" which is computed by taking the corresponding SP's route, examining it from right (destination) to left (source), and removing any sequences of nodes which begin and end in the same node. For instance, the path $< a, b, c, d, a, f, g, h, c, l, m >$ will result in the reverse route $< m, l, c, b, a >$. Note that the reverse route is not necessarily the shortest reverse path, nor the one resulting in the best QoS. The route brought back by an ACK is used as the source route by subsequent DPs of the same QoS class having the same destination, until a newer and/or better route is brought back by another ACK. A *Mailbox (MB)* in each node is used to store QoS information. Each MB is organized as a Least-Recently-Used (LRU) stack, with entries listed by QoS class and destination, which are updated when an ACK is received.

A. The Random Neural Network

We use recurrent random neural networks (RNN) [1] whose weights is modified using RL in order to implement the SP routing algorithm.

The RNN is an analytically tractable spiked random neural network model whose mathematical structure is akin to that of queuing networks. It has "product form" just like many useful queuing network models, although it is based on nonlinear mathematics. Consider a RNN with n interconnected neurons. The state q_i of ith neuron is the probability that it is excited and it satisfies the following system of nonlinear equations:

$$q_i = \frac{\sum_j^n q_j w_{ji}^+ + \Lambda_i}{r(i) + \sum_j^n q_j w_{ji}^- + \lambda_i} \tag{2}$$

where Λ_i and λ_i are external parameters that are set for the RNN as a whole. w_{ji}^+ is the rate at which neuron j sends "excitation spikes" to neuron i when j is excited, w_{ji}^- is the rate at which neuron j sends "inhibition spikes" to neuron i when j is excited, and $r(i)$ is the total firing rate from the neuron i or $r(i) = \sum_j^n [w_{ij}^+ + w_{ij}^-]$. For an n neuron network, the network parameters are these n by n "weight matrices" $W^+ = \{w^+(i,j)\}$ and $W^- = \{w^-(i,j)\}$ which need to be adapted from measurement data about QoS.

Turning now back to the manner in which the RNN is used in CPN, in each CPN node we have an RNN per QoS class and per destination. Each output link of the node is associated with a neuron of the RNN. The arrival of a SP triggers the calculation of the q_i using (2); then the output link corresponding to the neuron whose q_i value is largest, is chosen as the output link for the SP. On the other hand, each time the MB is updated, the weights of the RNN are updated so that decisions are reinforced or weakened depending on how they contribute to the success of the QoS goal. This point will be detailed below.

B. Reinforcement Learning: Updating the Network Weights

As an example, if the QoS goal G is hop count H, the reward function $R = 1/G$ of the RL algorithm will be:

$$R = \frac{1}{\beta \cdot H}. \tag{3}$$

H can be measured by SPs by incrementing a counter that increases each time an SP visits one more node. H is brought back to the nodes on a path by the ACKs as they return to the source, and is then stored in the node's MB.

Successive values of R, denoted by $R_l, l = 1, 2, \cdots$, are used to compute a decision threshold which represents the (recent) historical value of the hop count from some intermediate node to the destination:

$$T_l = \alpha T_{l-1} + (1 - \alpha) R_l \tag{4}$$

where α is some constant($0 < \alpha < 1$) that is used to tune the responsiveness of the algorithm. For instance $\alpha = 0.2$ means that on the average five past values of R are being taken into account. Suppose that the RL algorithm's lth decision was to select output link (neuron) j and that the lth reward calculated for the QoS information received from the network is R_l. We first determine whether R_l is larger than, or equal to, the threshold T_{l-1}. If this is the case, then we increase significantly the excitatory weights going into neuron j and make a small increase of the inhibitory weights leading to other neurons. If R_l is less than T_{l-1}, then we simply increase moderately the excitatory weights leading to all neurons other than j and increase significantly the inhibitory weight leading to neuron j in order to punish it for not being successful this time:

- If $T_{l-1} \leq R_l$

$$\begin{aligned} w^+(i,j) &\leftarrow w^+(i,j) + R_l \\ w^-(i,k) &\leftarrow w^-(i,k) + R_l/(n-2), \text{ for } k \neq j. \end{aligned}$$

- Else

$$\begin{aligned} w^+(i,k) &\leftarrow w^+(i,k) + R_l/(n-2), \text{ for } k \neq j \\ w^-(i,j) &\leftarrow w^-(i,j) + R_l \end{aligned} \tag{5}$$

The probabilities q_i are computed using equations (2), and the next SP will be forwarded to the output link which corresponds to the neuron which has the largest excitation probability.

We can also construct a QoS goal that combines the number of hops H and the forward delay D:

$$G = H + \gamma D \tag{6}$$

where γ is a constant that is used to relate the measurement units of D (milliseconds) to the integer value of H.

C. Some Experimental Results

We report the results of some experiments that were run on a CPN test-bed consisting of 17 nodes shown in Figure 2. Each pair of INRs is connected by point-to-point 10Mbps

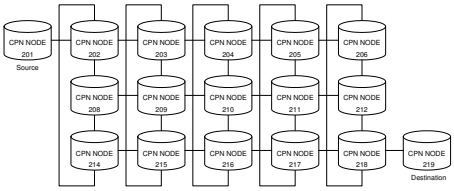

Fig. 2. The test-bed topology used in the experiments

Ethernet links. All tests were performed using a flow of UDP packets entering the network at constant bit rate (CBR) with $1024B$ packets. Each measurement point is based on $10,000$ packets that were sent from the source to the destination, and we inserted random background traffic into each link in the network with the possibility of varying its rate. The CPN routing algorithm is used throughout the experiments using three different QoS goals: (a) delay [*Algorithm-D*], (b) hop count [*Algorithm-H*] and (c) the combination of hop count and forward delay [*Algorithm-HD*]. Measurements concern average hop count, the forward delay and packet loss rate under different background traffic conditions.

From Figure 2, we see that the shortest path length from the source node $(\#201)$ to the destination node $(\#219)$ is 7, and there are only five distinct shortest paths. For example, one of them is route $\langle 201 \rightarrow 202 \rightarrow 214 \rightarrow 215 \rightarrow 216 \rightarrow 217 \rightarrow 218 \rightarrow 219\rangle$.

Figure 3 reports the average number of hops traversed from source to destination when different algorithms are used. When hop count is used as the QoS goal, we see that the average number of hops under different background traffic conditions is close to the minimum of 7. In order to get the detailed view of the routes used, we also collect the routes used by each packets at the source node without introducing any background traffic. Since the routing algorithm is running in kernel mode, we only collect the routes used by the first 2000 DPs due to the memory concern.

Figure 4 shows the routes used when the packet transmit rate is 100 packets/sec. Twenty-five different routes are used in total and one shortest path is discovered. we notice that 1805 of 2000 packets use the route $\langle 201 \rightarrow 202 \rightarrow 203 \rightarrow 204 \rightarrow 205 \rightarrow 206 \rightarrow 218 \rightarrow 219\rangle$, which is one of the shortest routes. At 500 packets/sec (Figure 5), 20 routes are used and 4 of them are shortest paths. 1711 of the 2000 packets use one of the shortest paths, route #3. When the connection's traffic rate is 1000 packets/sec (Figure 6), 12 routes are used and 4 of them are a shortest path. In this case, only 1021 packets

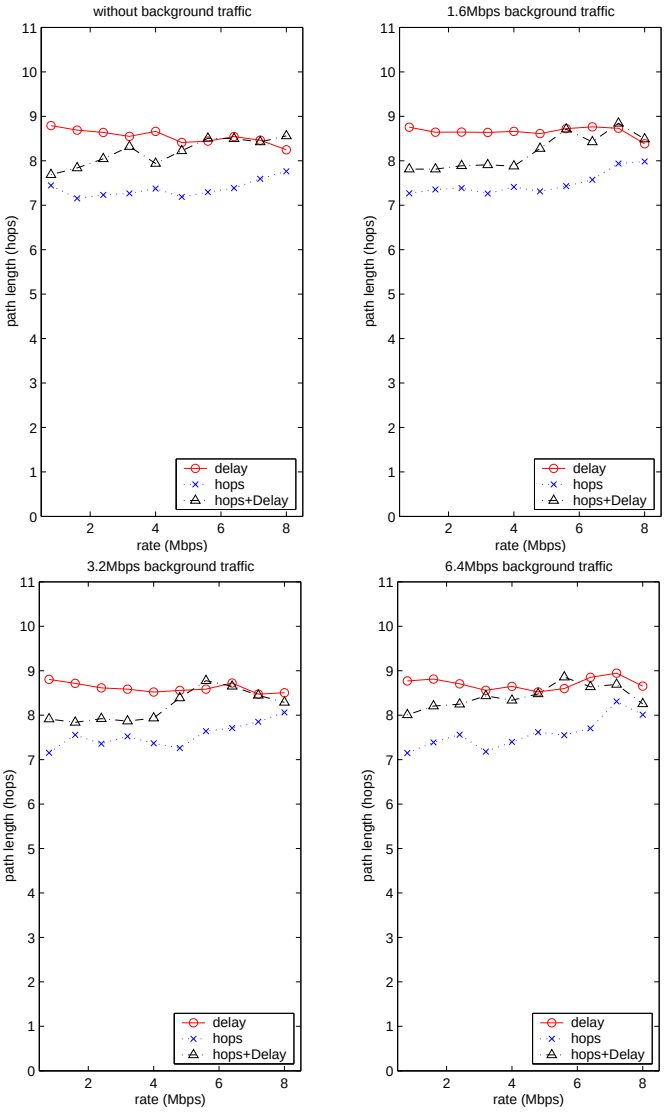

Fig. 3. Path length comparison

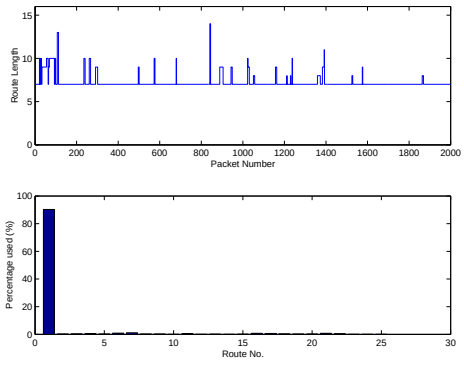

Fig. 4. Route usage with low transmit rate

are using the shortest paths. We can conclude that when *Algorithm-H* is used, the shortest paths are indeed discovered by the the SPs and are used by most of the DPs. We note that most of the DPs keep on using the same shortest path even if more than one of them has been discovered; since the topology of out test-bed does not change, once a shortest path

is discovered and used, the positive feedback brought back by the ACKs will reward the previous choice so that the RNNs keep recommending that the same path be chosen. When the connection's traffic rate is very high at 1000 packets/sec, we observe that only 12 distinct routes were discovered. Although we are not certain about the cause of this observation, it may be due to a higher loss rate of SPs, and hence a smaller number of ACK packets that update the RL algorithms running at the nodes. When forward delay is used as the QoS goal, the

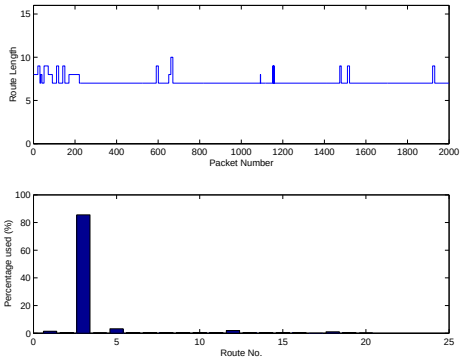

Fig. 5. Route usage with medium transmit rate

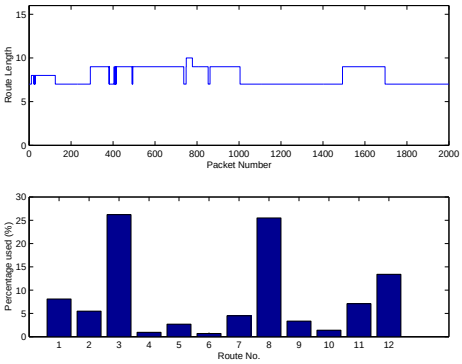

Fig. 6. Route usage with high transmit rate

average number of hops is not at the minimum (here it is close to 9) as seen in Figure 3), and Figures 7,8 and 9 show which routes are being used for different levels of connection traffic rate. Compared to *Algorithm-H*, more routes are being discovered and used. The numbers of routes used is 40, 35 and 15 when the connection's traffic rate is 100 packets/sec, 500 packets/sec and 1000 packets/sec respectively. The average delay decreases when traffic is spread out over more paths, when the ACKs bring back a larger number of alternate paths which have been discovered by the SPs.

Figure 10,11 and 12 show how routes are discovered with *Algorithm-HD*. The numbers of routes discovered by the smart packets are 43, 45 and 12 when the connection's traffic rate is 100 packets/sec, 500 packets/sec and 1000 packets/sec respectively. From Figure 3, we can see that the average path length is close to 8 when the connection's traffic rate is low or medium; when it is high, the average path length is close to 9. With respect to path length, the experiments confirm our expectations: *Algorithm-H* is the best, and *Algorithm-HD* is better than *Algorithm-D*.

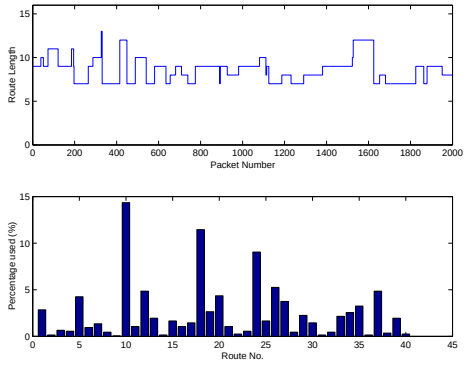

Fig. 7. Route usage with low transmit rate

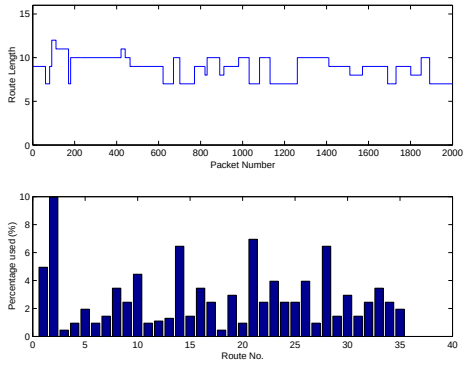

Fig. 8. Route usage with medium transmit rate

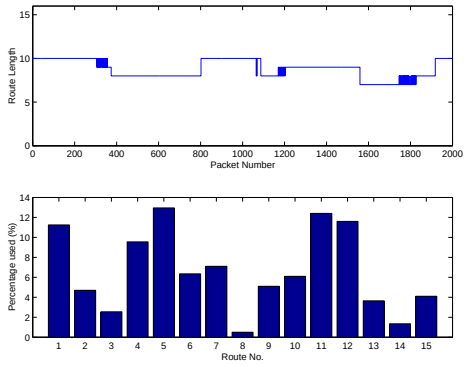

Fig. 9. Route usage with high transmit rate

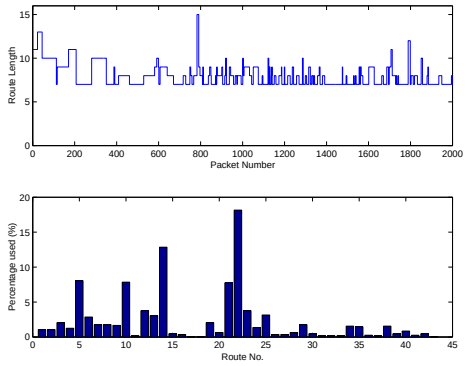

Fig. 10. Route usage with low transmit rate

We also report the forward delay and the packet loss rate.

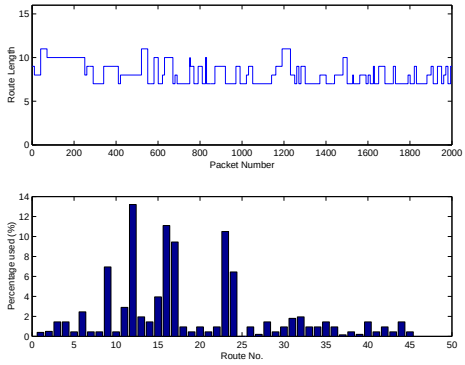

Fig. 11. Route usage with medium transmit rate

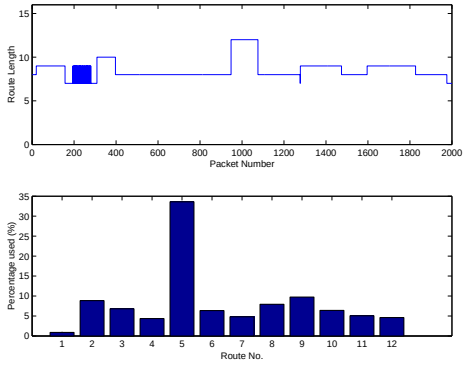

Fig. 12. Route usage with high transmit rate

The forward delay is approximated as one half of the round trip delay. To measure the loss rate, we keep track of the number of packets s sent out from the source node and the number of packets r which are received at the destination. The packets loss rate is obtained then $L = 1 - r/s$. From

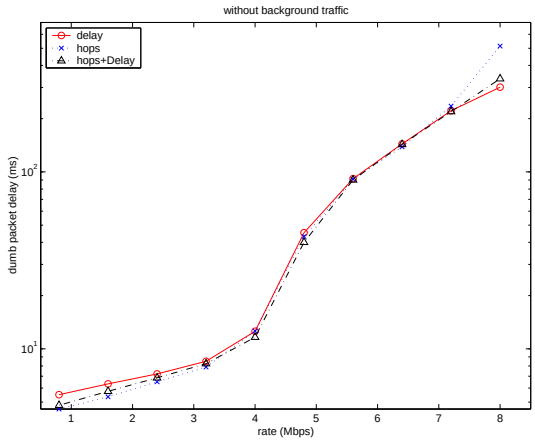

Fig. 13. Delay without background traffic

Figure 13, we are surprised to observe that if the connection's traffic rate is less than 3.2Mbps, then *Algorithm-H* achieves the smallest delay, while *Algorithm-D* is the worst. However, when the connection's traffic rate is between 3.2Mbps and 5.6Mbps, the performance of *Algorithm-HD* is better but *Algorithm-H* and *Algorithm-D* are almost the same. All algorithms are equivalent with respect to measured delay when the connection's traffic rate is between 5.6Mpbs and 7Mbps. When the connection's traffic rate is extremely high(>7Mbps) *Algorithm-D* gives the smallest delay, the and *Algorithm-H* is the worst.

V. Conclusions

We present an architecture for Intelligent Networks (INs) which offers a communication environment for users and services. The IN is composed of Intelligent Network Routers capable of supporting the user and service needs, and able to sense and adapt network paths and user to service connections dynamically as a function of network state and user and service quality of service needs. It uses smart packets for the search for services, for on-line dynamic sensing. We also suggest that the IN can use reinforcement learning (RL) and neural networks for local control. We illustrate these ideas with experiments on the CPN test-bed that carries out QoS driven network routing, based on similar concepts with decentralised control using reinforcement learning, and which we have implemented and tested. Athough the CPN test-bed is focused on routing only, the experimental results obtained show that these ideas can actually work in practice. A future test-bed will focus on connecting users and services in a smart adaptive framework. Our current work aims at extending these concepts to the connection between users and services, as outlined in this paper. Future work will also show how similar concepts can be used to protect users and services from malicious attacks.

References

[1] E. Gelenbe. Learning in the recurrent random neural network. *Neural Computation*, 5 (1), pp. 154–164, 1993.

[2] D. Williams and G. Apostolopoulos. QoS Routing Mechanisms and OSPF Extensions. RFC 2676, Aug. 1999.

[3] E. Gelenbe, E. Seref and Z. Xu. Simulation with learning agents. *Proceedings of the IEEE*, 89 (2), pp. 148-157, 2001.

[4] E. Gelenbe, R. Lent and Z. Xu. Measurement and performance of a cognitive packet network. *Computer Networks*, 37, pp. 691-791, 2001.

[5] E. Gelenbe, R. Lent and Z. Xu. Design and performance of cognitive packet networks. *Performance Evaluation*, 46, pp. 155-176, 2001.

[6] E. Gelenbe, R. Lent, and Z. Xu. Cognitive Packet Networks: QoS and Performance. *Proc. IEEE MASCOTS Conference*, ISBN 0-7695-0728-X, pp. 3-12, Fort Worth, TX, Oct. 2002.

[7] E. Gelenbe, M. Gellman, R. Lent, P. Liu, Pu Su. Autonomous smart routing for network QoS. *Proc. First International Conference on Autonomic Computing*, (IEEE Computer Society), ISBN 0-7695-2114-2, pp. 232-239, May 17-18, 2004, New York.

[8] E. Gelenbe, R. Lent, A. Nunez. Self-aware networks and QoS. *Proceedings of the IEEE*, 92 (9), pp. 1478-1489, 2004.

[9] E. Gelenbe and R. Lent. Adhoc power aware Cognitive Packet Networks. *Ad Hoc Networks Journal*, Vol. 2 (3), pp. 205–216, 2004 (ISN: 1570-8705).

[10] E. Gelenbe. Cognitive Packet Network. *U.S. Patent No. 6,804,201 B1*, Oct. 12, 2004.

[11] E. Gelenbe and P. Liu. QoS and Routing in the Cognitive Packet Network. Submitted for publication, 2005.