PROCEEDINGS OF SPIE

SPIEDigitalLibrary.org/conference-proceedings-of-spie

Genetic routing algorithms to optimize availability in broadband wireless networks with load balancing

William S. Hortos

SPIE.

Genetic routing algorithms to optimize availability in broadband wireless networks with load balancing

William S. Hortos
Florida Institute of Technology, Orlando Graduate Center, 3165 McCrory Place, Suite 161, Orlando, FL 32803

ABSTRACT

Packet-switched networks using the Internet Protocol (IP) provide multimedia services through broadband wireless access to mobile and fixed subscribers from an IP core network via bi-directional paths consisting of a hierarchy of high-speed routers, switches, and servers. Packets are aggregated at the nodes that form the ordered links of end-to-end paths between subscriber and gateway. Network resources are allocated at nodes to meet quality of service (QoS) requirements of new and existing calls. If sufficient resources are not available to satisfy a call's QoS, the call is blocked or dropped, reducing network "uptime" or availability. Packet flows are shared among redundant devices, clustered at nodes, to reduce blocking and dropping and speed failure recovery. A two-stage genetic algorithm (GA) is proposed to assign resources to feasible paths to provide calls the best possible resource utilization, availability, and QoS levels, while balancing traffic among devices at nodes. The GA operates on a population of integer-valued vectors of call ID, QoS requirements, and end-to-end paths encoded as node-device pairs. Selection, crossover, and mutation are defined for the GA. At call arrivals and departures, the GA limits the number of candidate paths based on their fitness to provide QoS, path availability, resource utilization, and load balance. Simulation results are discussed for different scenarios.

Keywords: Broadband wireless access; packet-switched networks; Internet routing protocols; multimedia services; genetic algorithms; optimal resource allocation; quality-of-service (QoS) routing; load balancing; network availability

1. INTRODUCTION

Providers of multimedia services are challenged to transition from circuit-switched ATM and frame relay which are already deployed in large-scale networks to build large-scale Internet Protocol (IP)-based networks with the capabilities to provide aspects of quality of service (QoS)/class of service (CoS) and facilitate the use of virtual private networks (VPNs). Added to these challenges are market needs to extend these services to mobile subscribers through broad wireless access technologies based on third- and fourth-generation industry standards. The Internet Engineering Task Force (IETF) is defining new IP concepts in response to numerous interrelated problems to establish large-scale, hybrid-access IP networks. These problems include scaling IP networks to meet the growing demands of Internet traffic, enabling differentiated levels of IP-based services to provisioned, merging traffic types of varying QoS requirements onto a single IP network, and improving operational efficiency in a dynamic environment.[1, 2]

The deployment of new protocols, such as, Multiprotocol Label Switching (MPLS), combining layer 2 (data link layer) switching with layer 3 (network layer) routing, will satisfy the requirements of these service providers.[3] Label-switching technology is a result of the desire to combine the benefits of switching technologies in the core of the network with the benefits of IP routing technologies at the edge of the network. A hybrid network based on both of these technologies as well as broadband wireless access technology and protocols creates a larger problem best described as "how to make IP, ATM and broadband wireless inter-operate." Label switching seeks to combine the best attributes of layer 2 switching, as embodied in ATM and frame relay, with the best attributes of the layer 3 routing embodied in the IP domain. MPLS, as the standards-based approach to label switching, identifies and marks IP packets with labels and forwarding them to modified switch or router, which then uses the labels to switch the packets through the network. The labels are created and assigned to IP packets based upon the information gathered from existing IP routing protocols.

The resulting network architecture for delivering IP traffic places switching at the core of the network, while IP routing continues to dominate the edge. The need to integrate the two different technologies has given rise to the use of overlay networks where the access technology (IP) has been overlaid on the core technology (ATM or frame relay), and,

Applications and Science of Computational Intelligence V, Kevin L. Priddy, Paul E. Keller, Peter J. Angeline, Editors, Proceedings of SPIE Vol. 4739 (2002) © 2002 SPIE · 0277-786X/02/$15.00

in hybrid network architectures, integrated with the mobile access technology of broadband wireless. Such a configuration is shown in Figure 1. Since paths between the routers now go through switches, they require a connection. When an IP network is overlaid on a switched network like ATM, all of the routers appear to be directly connected to each other at the network layer.

A significant feature of current telecommunications networks, particularly the Public Switched Telephone Network (PSTN), is their reliability. The concept of 99.999% uptime, also known as "five nines" availability, is a performance goal of public network providers. Networks and their constituent components meeting this stringent requirement are often referred to as "carrier-class." Public data networks of today need to evolve considerably before they can meet the same level of availability. Network availability translates to the ability to support new and existing calls subject to failures at the hops that form the call's path due to the inherent failure mechanisms in the physical devices, such as routers, switches and servers, forming those hops. Another factor defining availability is the blocking of new calls and dropping of existing calls caused by a lack of resources to meet the call's minimum QoS requirements on the hops of the call's path, when traffic loading overwhelms capacities. To mitigate both threats to carrier-class availability, designers have proposed clustering a number of redundant components at each stage of the network, from the IP gateways to the subscriber terminals within broadband wireless coverage areas. The redundant devices within a cluster, i.e., routers, switches and servers, share the resources processing incoming traffic in order to balance the load among the hops of all calls supported at that node, thereby avoiding call dropping or blocking while maintaining QoS. Moreover, if a device forming one or more hops of the end-to-end path of a call should fail, the hops can be restored by switching to one of the peer devices in the cluster. In both ways, the availability of the hops, and, hence, of the entire path is increased.

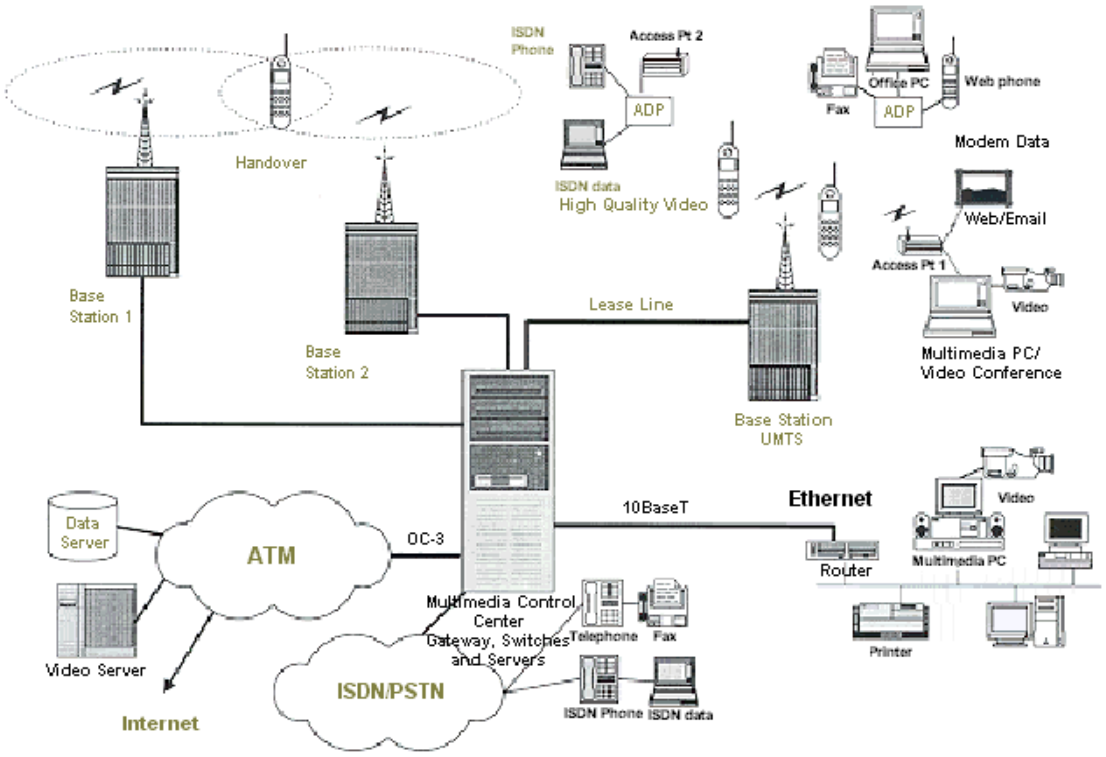

Figure 1. Hybrid wireless/wired multimedia network based on IP overlay model

The application context of this paper is a packet-switched, hybrid IP network that provides mobile and fixed subscribers the multimedia services of voice, video and data through broadband wireless access from a backhaul network to the access gateways of an IP backbone network. The bi-directional paths between subscribers and the IP core begin with radio links between subscriber terminals and wireless base stations located at the edge of the network. Packets are aggregated in queues on upstream links between physical nodes, ordered to form the end-to-end paths between subscriber and gateway. Each node is categorized by its message-processing functions in packet transport: edge router;

core router; core switch; administrative, signaling, application, and firewall servers. Dynamic loading by new and existing calls makes demands on available resources at each node. To avoid packet congestion and capacity overloading, packet flows are shared among physical devices, grouped in clusters based on their common function each node. Multiple devices from a cluster can be selected at a node to balance dynamic traffic loading among the feasible paths through the node. To limit call blocking, the resources of bandwidth, queues, delay, channel capacity, and/or transmit power are allocated at the nodes to meet the minimum QoS requirements of new and existing calls. QoS requirements for each traffic class and application are mapped to corresponding values of the resource parameters. If resources are not available on any path to satisfy a new call's QoS, the call is blocked. Adding redundant devices from the cluster at each node mitigates the impact of dynamic loading on path availability.

An adaptive QoS routing scheme, based on a two-stage genetic algorithm (GA), is constructed to assign resources to the feasible end-to-end paths of each call to provide the best possible path availability and QoS levels to all calls, with load balancing of the resulting IP traffic. Updates occur at each new call arrival or departure. The GA operates in a population of individuals defined as integer-valued vectors consisting of call ID, encoded by its IP home and destination addresses; the call's QoS requirements, encoded as the list of parameter levels; and the end-to-end network path of the call, encoded as a list of its constituent node-device pairs, with each node ID represented by the IP address of the physical device, selected from a common-function cluster. Each device is assigned a time-varying probability distribution of failure, initiated at device activation. Standard genetic operators of selection, crossover, and mutation generate alternate individuals, updated at each call event and, if necessary, in each stage of the algorithm.

The remainder of the paper is organized as follows. Section 2 is a background discussion of IP extensions for QoS, QoS parameters and values, network resources, service-class definitions, and the network environment for adaptive QoS provisioning. Section 3 presents an overview of the proposed two-stage algorithm. The elements and structure of the algorithm is described in terms of basic GA concepts in Section 4. Two subsections detail the two main routines of the proposed algorithm: one focusing on individual selection to optimize resource allocation and sustain the QoS requirements of calls, the other focusing on individual selection to achieve the best possible traffic balance of resource allocations among the paths assigned to calls by the first routine. Simulation results are discussed in Section 5. Section 6 offers conclusions from the application of the GA approach and discusses its extensions to other IP network issues.

2. QOS PROVISIONING IN IP NETWORKS

With the advent of packet switching and the proliferation of many kinds of communications traffic (time-sensitive financial transactions, still images, large data files, voice, video, and so on), there are more than one set of QoS criteria to satisfy. The data rate needed for satisfactory voice service may take an intolerable time to transfer high-resolution images. Conversely, the degree of latency acceptable in some file transfers may be inadequate for real-time voice. Thus, QoS levels are often specified in service level agreements (SLAs) between service providers and their largest customers.

2.1 QoS parameters

As listed in Table 1, the five most important of the QoS parameters in IP, according to the IETF, are:[4]

Availability. Ideally, a network is available 100 percent of the time. A figure of 99.8 percent translates into about an hour and a half of down time per month, which may be unacceptable to a large enterprise. "Carrier-class" networks strive for 99.999-percent availability, which translates into a downtime of 5.2 minutes per year.

Throughput or bandwidth. This is the effective data transfer rate measured in bits per second, and must be distinguished from the maximum capacity, or wire speed, of the network. Sharing network resources among many users lowers the throughput realizable by any user, as does the overhead imposed by the extra bits included in every packet for identification and other purposes. A minimum throughput rate is typically guaranteed by a service provider.

Packet loss. Network devices, such as, switches and routers, sometimes have to hold data packets in buffered queues when a link or hop gets congested. If the link remains congested for too long, the buffered queues will overflow and packets will be lost. The lost packets must be retransmitted, adding to the total transmission time. In a well-managed network, packet loss will typically be less than 1 percent averaged over some time period, e.g., a month or quarter.

Latency or delay. The time taken by data to travel from the source to the destination is known as latency, or delay. For example, the latency of a 5000-km voice call carried by a circuit-switched telephone network is about 25 ms. For the

public Internet, a voice call may easily exceed 150 ms of latency because of delays, such as those caused by signal processing (digitizing and compressing the analog voice input) and congestion (queuing).

Jitter. Jitter is another term for delay variation. Its causes include variations in queue length, variations in the processing time needed to reorder packets that arrive out of order because they have traveled over different paths, and variations in the processing time needed to reassemble packets that were segmented by the source before transmission.

Table 1. QoS parameters and definitions

Category	Parameter	Description/Example
Timeliness	Delay	Time taken for a message to be transmitted
	Response time	Round-trip time from request transmission to reply receipt
	Jitter	Variation in delay or response time
Bandwidth	Systems level data rate	Bandwidth required or available, in bits or bytes per second
	Application level data rate	Bandwidth required or available, in application specific units per second, e.g., video frame rate
	Transaction rate	Number of operations requested or processed per second
Reliability	Mean time to failure (MTTF)	Normal operation time between failures.
	Mean time to repair (MTTR)	Down time from failure to restarting normal operation
	Mean time between failures (MTBF)	MTBF = MTTF + MTTR
	Percentage of time available	MTTF/MTTF + MTTR
	Loss or corruption rate	Proportion of total data that does not arrive as sent, e.g., network error rate

The International Telecommunications Union (ITU) has developed requirements for third-generation (3G) mobile communication networks to provide anywhere, anytime, bandwidth-on-demand multimedia services to users. The leading standard to meet ITU 3G objectives is known alternately as W-CDMA in Japan and as the Universal Mobile Telecommunications System (UMTS) in Europe. A converged version of the two alternatives is called 3GPP, acronym for the 3rd Generation Partnership Project. The 3G architecture is comprised of the circuit-switched, packet-switched and IP access networks of Figure 1 that support emerging as well as legacy services. Since broadband wireless access designs are based on 3rd generation international standards and specifications, it is necessary to extend the discussion from the QoS concepts in wired networks to the framework for QoS delivery according to 3GPP requirements.[5] The 3GPP QoS concept and architecture are based on the following attributes which extend the IETF QoS parameters to service delivery over wireless networks.

Traffic class. This is the type of application for which the service is optimized. By including the traffic class itself as an attribute, 3GPP can make assumptions about the traffic source and optimize the transport for that traffic type.

Maximum bit rate. The maximum bit rate, in kilobits per second, is the upper limit on throughput or bandwidth a subscriber or application can accept or provide. All service attributes may be fulfilled for traffic up to the maximum bit rate subject to network conditions. Maximum bit rate can be used to make code reservations in the downlink of the radio interface. Its purpose is 1) to limit the delivered bit rate to applications or external networks with such limitations and 2) to establish the maximum desired user bit rate for applications that can operate at different rates.

Guaranteed bit rate. The guaranteed bit rate, in kilobits per second, is the bit rate or bandwidth the service will guarantee to the subscriber or application. Guaranteed bit rate may be used to facilitate admission control based on available resources, and for resource allocation within the network. The corresponding QoS attributes, e.g., delay and reliability, are guaranteed for traffic up to the guaranteed bit rate.

Maximum packet size. The maximum packet size in bytes is used for admission control and policing.

Packet format information. The radio subnetwork needs packet size information in bits to operate in transparent Radio Link Control (RLC) protocol mode, which improves spectral efficiency and delay when RLC retransmission is not used.

Packet error or loss rate. This is equivalent to packet loss. The packet error rate is the fraction of packets lost or detected as erroneous and is identified only for traffic that conforms to all other requirements. By reserving resources, packet error rate performance is independent of the loading conditions.

Residual bit error ratio. Residual bit error ratio (RBER) is the undetected BER in delivered packets. If no error detection is requested, RBER indicates the BER in delivered packets. It is used to configure protocols, algorithms and error detection schemes.

Delivery of erroneous packets (y/n/-). The delivery of erroneous packets indicates whether packets detected as erroneous will be delivered or discarded. It is used to decide whether error detection is needed and whether frames or packets with detected errors are to be forwarded. The value 'yes' implies that error detection is used and that erroneous packets are delivered together with an error indication, 'no' implies that error detection is used and that erroneous packets are discarded, and '-' implies that packets are delivered without considering error detection.

Transfer delay. Transfer delay is the maximum delay for the 95th percentile of the distribution of delay for all delivered packets during the lifetime of a service. Delay for a packet is the time from a request to transfer a packet at one service access point (SAP) to its delivery at the other SAP. It is used to specify delay tolerated by an application.

Traffic handling priority. Traffic handling priority specifies the relative importance for handling all packets belonging to the service or subscriber compared to the packets of others. Within some traffic classes, there is a need to differentiate between service qualities. Using the traffic handling priority allows the network to schedule traffic accordingly. Priority is an alternative to absolute guarantees; thus these two parameter types cannot be used together for a single service.

Allocation/retention priority. Allocation/retention priority specifies the relative importance compared to other services for allocation and retention of the service. When resources are scarce, the relevant network elements, e.g., routers and gateways, can use this attribute to prioritize subscribers with a high allocation/retention priority over subscribers with a low priority when performing admission control.

Source statistics descriptor ('speech'/'unknown'). Source statistics descriptor specifies characteristics of the source of submitted packets. Conversational speech possesses a well-known discontinuous transmission (DTX) factor. Informed that the packets are generated by a speech source, networks may calculate a statistical multiplex gain for use in admission control on the applicable interfaces.

2.2 Traffic classes and QoS

Different applications have varying sensitivities to delay, jitter, bandwidth, and packet loss, as shown in Table 2 for common services. Availability, on the other hand, is a network-based characteristic, desirable for all applications. A long file transfer, based on the File Transfer Protocol (FTP), needs high throughput and low packet loss, but is not very sensitive to delay and jitter. Live video conferencing also needs high throughput, but is sensitive to both delay and jitter. Moreover, voice over IP (VoIP) requires very low jitter, and a one-way delay in the order of 100 ms, and *g*uaranteed bandwidth in the range of 8 to 64 kbps, depending on the codec used. These differences must be considered in the SLAs between service providers and their clients. The usual SLA specifies the end-to-end performance to which the client is entitled over a specified time interval, e.g., a month or a quarter.[4]

When defining the 3GPP QoS classes, also referred to as traffic classes, the restrictions and limitations of the air interface have to be taken into account. In the case of Internet applications, the selection of the class and appropriate traffic attribute values is made according to the QoS attributes. Internet applications do not directly use the services of 3G but they use Internet QoS definitions and attributes, which are mapped to 3G QoS attributes at the API.

There are four different QoS classes in 3GPP: conversational; streaming; interactive; and background.[5] As in the IETF specifications for services, the primary distinguishing factor between 3GPP QoS classes is sensitivity to delay. The conversational class is intended for traffic that is very delay sensitive, while the background class is the most delay insensitive traffic class.

The conversational and streaming classes are intended to support real-time applications. The main difference between them is, again, their relative delay sensitivity. Conversational real-time services, like video telephony, are the most delay sensitive applications and those data streams should be carried in conversational class. These classes roughly correspond to the traffic types in Table 2 with "high" jitter sensitivities, with the streaming class somewhat more tolerant of jitter than the conversational class.

Table 2. QoS sensitivities by IP traffic type

QoS Sensitivities by IP Traffic Type				
Traffic Type	Sensitivity Level			
	Bandwidth	Packet Loss	Delay	Jitter
Voice	Very Low	Medium	High	High
E-commerce	Low	High	High	Low
Retail transactions	Low	High	High	Low
E-mail	Low	High	Low	Low
Telnet	Low	High	Medium	Low
Casual web browsing	Low	Medium	Medium	Low
Active web browsing	Medium	High	High	Low
File transfers	High	Medium	Low	Low
Video conferencing	High	Medium	High	High
Multicasting	High	High	High	High

The interactive and background classes are intended for traditional Internet applications like WWW, e-mail, Telnet, FTP and news. With less restrictive delay requirements than the conversational and streaming classes, both provide better error rate by means of channel coding and retransmission. The main difference between the two is that the interactive class is mainly used by interactive applications, e.g., interactive e-mail or Web browsing, while background class is meant for background traffic, e.g., background downloading of e-mails or files. Interactive and background applications are separated to ensure the responsiveness of the former. Interactive traffic has higher priority in scheduling than background traffic, so background applications use transmission resources only when interactive applications do not need them, barring any use of fairness rules, such as weighted fair sharing (WFQ). This prioritization of traffic classes is very important in a wireless environment where bandwidth is low compared to fixed networks.

Table 3 summarizes the QoS attributes, their significance, and range of values that define each 3GPP traffic class. The value ranges reflect the required capability of an access network based on 3rd generation wireless specifications. When a service is defined as a combination of parameters, further limitations may apply; for example, the shortest possible delay may preclude use with the lowest possible packet loss rate.[5]

2.3 Network environment for adaptive QoS provisioning

In the cellular-based wireless segment of the hybrid network, the QoS provisioning is more complex than in the wired segment due to radio channel fading, BER, and mobility. Fading, in addition to BER, increases packet loss and delays due to retransmissions.[6, 7] Bandwidth availability at the wireless access link is less predictable than in the wired backhaul network due to temporal and spatial dependencies in addition to fading effects. During handoff, a call from a mobile subscriber with a certain QoS guarantees would be at risk of diminished service quality or even being dropped. Hence, the network resources of bandwidth and buffer space (in queues) allocated to a call, during the setup phase, could vary significantly over the duration of the call. Therefore, adaptive allocation of resources to support the QoS of a call in the network is required to respond to fluctuations in those resources by changing the call's requested QoS values.

In such an adaptive networking environment, calls can be admitted to the system even if the available resources at all links in the network is not sufficient to meet their QoS requirements. To perform the adaptation, the proposed algorithm seek to lower the QoS levels of existing calls to make some network resources available to additional calls. In terms of the QoS parameters and their values listed in Tables 1 and 3, each call is considered to have a predetermined minimum QoS level (Q_{min}) and a maximum QoS level (Q_{max}) that define the CoS or ToS in a traffic class or SLA. Q_{min} corresponds to the minimum vector of QoS requirements with which the multimedia application can perform, Q_{max} corresponds to the best or ideal QoS levels a call can obtain if there are no resource restrictions. In the context of the IP multimedia network, a call consists of a number of substreams, e.g., voice, video, data, and facsimile, each of which has its own QoS requirements ranging from highest to lowest acceptable levels. For example, a subscriber may permit low

video quality but require high voice quality and high-speed Internet downloads. A subscriber with this profile will be allocated the amount of network resources that supports the Q_{min} level stated in his SLA. Whenever resources become available on the links of the end-to-end path of the call, the network may allot the call more of the available resources at the nodes of the path until either the call's Q_{max} level is reached or the residual resources are exhausted. The concept defines the different grades or classes of service declared in the subscriber's SLA with the service provider.

Table 3. Value ranges for the QoS parameters of 3GPP traffic classes

Traffic class	Conversational	Streaming	Interactive	Background
Maximum bit rate (kbps)	< 2048 (1) (2)	< 2048 (1) (2)	< 2048 - overhead (2) (3)	< 2048 - overhead (2) (3)
Delivery order	Yes/No	Yes/No	Yes/No	Yes/No
Maximum packet size (bytes)	≤ 1500 or 1502 (4)	≤ 1500 or 1502 (4)	≤ 1500 or 1502 (4)	≤ 1500 or 1502 (4)
Packet format information	(5)	(5)		
Delivery of erroneous packets	Yes/No/- (6)	Yes/No/- (6)	Yes/No/- (6)	Yes/No/- (6)
Residual BER	$5*10^{-2}$, 10^{-2}, $5*10^{-3}$, 10^{-3}, 10^{-4}, 10^{-6}	$5*10^{-2}$, 10^{-2}, $5*10^{-3}$, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}	$4*10^{-3}$, 10^{-5}, $6*10^{-8}$ (7)	$4*10^{-3}$, 10^{-5}, $6*10^{-8}$ (7)
Packet loss rate	10^{-2}, $7*10^{-3}$, 10^{-3}, 10^{-4}, 10^{-5}	10^{-1}, 10^{-2}, $7*10^{-3}$, 10^{-3}, 10^{-4}, 10^{-5}	10^{-3}, 10^{-4}, 10^{-6}	10^{-3}, 10^{-4}, 10^{-6}
Transfer delay (ms)	100, max. value	250, max. value		
Guaranteed bit rate (kbps)	< 2048 (1) (2)	< 2048 (1) (2)		
Traffic handling priority			1,2,3 (8)	
Allocation/retention priority	1,2,3 (8)	1,2,3 (8)	1,2,3 (8)	1,2,3 (8)
Source statistic descriptor	Speech/unknown	Speech/unknown		

1) Bit rate of 2,048 kbps requires that the radio access network operates in transparent RLC protocol mode, in this case, the overhead from layer 2 protocols is negligible.
2) Although the 3GPP network has capability to support a large number of different bit rate values, the number of possible values shall be limited to reduce the complexity of terminals, charging, and interworking functions
3) Impact from layer 2 protocols on maximum bit rate in non-transparent RLC protocol mode shall be estimated.
4) In case of PDP type = PPP, maximum packet size is 1,502 bytes. Otherwise, maximum packet size is 1,500 bytes.
5) Requires definition of the possible values of exact packet sizes for which the radio access network can support transparent RLC protocol mode.
6) If *Delivery of erroneous packets* is set to 'Yes,' error indications can only be provided on the mobile equipment or terminal equipment side of the 3GPP call. At the core network gateway, error indications can not be signalled outside of the 3GPP network.
7) Values are derived from cyclically redundant code (CRC) lengths of 8, 16 and 24 bits on layer 1.
8) Number of priority levels will be further analyzed by the S1, N1 and N3 technical groups of the 3GPP.

The traffic parameters of bit rate, packet rate, delay, and jitter, that provide a physical definition of the call (and its substreams), and its SLA parameters are input to controller/scheduler routines resident on the access network's servers. These routines determine the required resources of bandwidth and buffer space in queues from the values of these parameters. The calculated resource values and the SLA of the call are then stored in a directory database on a network server to be accessed by the proposed two-stage GA. One format for the SLA for the call is a table of QoS values

distributed over the application substreams in the call. The first row of the table represents the vector of Q_{max} levels, while the last row is the vector of Q_{min} levels. All intermediate vector levels can be assigned to the call.

In the architecture of the hybrid IP network in this paper, the actual range of QoS values are mapped to four distinct QoS levels (high, midrange, low, and inactive) for each of the *M* individual application substreams. The 4^M rows of the table, each representing a vector of multiple QoS levels can each be assigned a corresponding index. The indices are ordered from highest (1) to lowest (4^M) QoS level, denoting no active service of any type in the call. A subscriber to Grade A service will have a profile with a QoS index ranging between 1 to *r*, $r < 4^M$. It has been shown in [8] that a drop in the bandwidth required occurs when the QoS index increases at certain breakpoints since the video substreams demand more bandwidth and buffer space than the other applications. So the breakpoints in the QoS indices occurs when the video quality drops from high to midrange, from midrange to low, and from low to inactive. This suggests that 4^M indices can be reduced into a smaller set based on the breakpoints. Any decrease in the number of QoS levels significantly lowers the dimensions of the database of subscriber profiles.

For *N* calls in the IP network, one can define an $N \times M$ array $[\boldsymbol{Q}]$, consisting of *N* rows, each of which is the *M*-vector, $\boldsymbol{Q}_n$, of the QoS levels of the multiple service streams of call *n*, or, alternately, define a reduced dimension *N* – vector of QoS indices, $\boldsymbol{Q} = [Q_1, Q_2, \ldots, Q_{N-1}, Q_N]$, where Q_n is the QoS index for call *n* such that $Q_n \in \{1, 2, \ldots, 4^M\}$. To reduce the scope of the paper, the discussion is restricted to consideration of paths in the backhaul network extending from the wireless cells established by the base stations or access points, to the edge routers, core routers, core switches, and access gateways leading back to the IP backbone network or PSTN, from which calls and applications may originate. To further reduce the complexity of the algorithm, it is assumed that the administrative and application servers support the processing at each stage but do enter directly into the links comprising the connection path. In general, at most distinct *L* stages of call processing are assumed in the subnetwork between the backbone and wireless cells; each loop-free, connection path between source and destination in the subnetwork can be decomposed into at most *L*-1 hops. Each hop is enabled by two devices, each selected from the cluster at one of the two nodes that form the hop. Node 0 denotes the access interface to the IP backbone network considered the source of the call, while node *L* denotes wireless cell of the customer premise equipment (CPE) or mobile terminal (MT). It is also assumed that there are no redundant devices at node 0 or *L* for use by the call, while there may be redundant devices at intermediate nodes.

The end-to-end path through the subnetwork for call *n* is represented by the string of index pairs, one index for the processing node, the other for the device selected at that node which supports the connection. Let the network path for call *n* be denoted by $P_n = (<i_{n0}, d_{i0(n)}>, <i_{n1}, d_{i1(n)}>, \ldots, <i_{nj}, d_{ij(n)}>, \ldots, <i_{nk}, d_{ik(n)}>)$ where $1 \le k \le L$ and $i_j \ne i_l$, if $j \ne l$ for $1 \le j, l \le L-1$ (loop-free) and $d_{ij(n)} \in D_j$, the set of the redundant devices at node *j*; i_{n0}, $i_{nk} \in \{0, L\}$ and $d_{n,i0}$, $d_{n,ik}$ is either the singular CPE or MT of the subscriber or the access point of the core network. Let $\wp$ be the set of end-to-end paths for the calls in the network. Let $C_j = (B_j, S_j)$ be the common capacity expressed as the bandwidth, B_j, and buffer or queue capacity, S_j, of each of the redundant devices in group D_j, while $R_n = (b_n, s_n)$ is the resource level (bandwidth and buffer space) to support the QoS requirement of call *n*. The 2*N*-vector $\boldsymbol{R} = [R_1, R_2, \ldots, R_{N-1}, R_N]$ represents the resource requirements of *N* calls in the network. Denote by N_j the number of calls in the network whose path includes node *j* and N_{ij} the number of calls whose path includes node *j* using device d_{ij}, so that $N_j = \sum_{i=1}^{|D_j|} N_{ij}$, where $|D_j|$ is the number of devices in the cluster D_j at node *j*. Let $f_{ij}(h)$ and $F_{ij}(h)$ be the probability density and cumulative probability, respectively, of failure of device d_{ij} in cluster D_j at the instant *h*. Since devices in each cluster are assumed identical, the failure probabilities can be assumed to be the same, i.e., for each D_j, $f_{ij}(h) = f_j(h)$ and $F_{ij}(h) = F_j(h)$.

The QoS-provisioning problem is to determine the highest possible QoS levels for the *N* calls in the network from within the search space of end-to-end paths and the available resources at network nodes. In general, for *M* services, *N* calls, and *L* processing stages in the network, and $|D_j|$ devices clustered at node *j* of the network, the search space will contain $4^{MN} \times \left(2 + \sum_{j=1}^{L-1} |D_j|\right)$ different combinations, where it is assumed that $|D_0| = |D_L| = 1$. The optimization problem

is to determine the set of QoS levels, $\boldsymbol{Q}$, that correspond to the set of resource allocations $\boldsymbol{R}$ and the set of paths $\wp$ for all calls of the acceptable availability, subject to the resource, resource utilization and failure reliability constraints:

$$\sum_{n=1}^{N_j} b_n \le \left|D_j\right| B_j, \sum_{n=1}^{N_j} s_n \le \left|D_j\right| S_j, \text{and} \left(\left|D_j\right| B_j - \sum_{n=1}^{N_j} b_n \right), \left(\left|D_j\right| S_j - \sum_{n=1}^{N_j} s_n \right) \text{ are minima, for } j = 0,1,\ldots,L, \quad (1)$$

the load balance constraints, which require the variances of device utilization at the nodes

$$\frac{1}{\left|D_j\right|^2} \sum_{i=1}^{\left|D_j\right|} \left(B_{ij,av} - \left(B_j - \frac{1}{\left|D_j\right|} \sum_{n=1}^{N_j} b_n \right) \right)^2, \ \frac{1}{\left|D_j\right|^2} \sum_{i=1}^{\left|D_j\right|} \left(S_{ij,av} - \left(S_j - \frac{1}{\left|D_j\right|} \sum_{n=1}^{N_j} s_n \right) \right)^2, \text{for } j = 0,1,\ldots,L \text{ to be minima,} \quad (2)$$

and the path reliability constraints, for each call n, $1 \le n \le N$,

$$F_{P_n}(t) = \prod_{\substack{i=0 \\ <in,d_{ij(n)}> \in P_n}}^{K} F_{ij}(t) \le \rho_n, \rho_n \text{ the path failure thresho ld, for each } P_n \in \wp, \text{ at time } t, \quad (3)$$

where $C_{ij,av} = \left(B_{ij,av}, S_{ij,av}\right)$ is the residual capacity (bandwidth and buffer space) of the ith device d_{ij} at node j. The set of inequalities in (1) indicate that the aggregate resources required by the calls cannot exceed the total resource capacities at the nodes; minimizing the second set of terms achieves optimal utilization of those resources. Minimizing the terms in (2) achieves the optimal balance of traffic at the nodes among the devices active on the paths of the calls. The inequalities in (3) indicate that the network path assigned to call n must meet the reliability requirement, i.e., not exceed the outage probability, declared in the SLA in addition to the acceptable QoS level. Note that the failure threshold can be set to a value independent of the call, thereby making path reliability a network performance metric.

3. OVERVIEW OF THE ALGORITHM

The proposed algorithm is based on genetic algorithms (GAs). Genetic algorithms depend on iterative evaluation of an objective function.[9, 10, 11] This feature gives GAs the advantage as an optimization method of not requiring functional information. Another advantage is that GAs are parallel-search procedures that can be implemented on parallel processing architectures. The structure of the algorithm in this work consists of two main routines. The partition of the algorithm consists of (*a*) the subtasks of resource allocation to achieve optimal QoS and resource utilization and of verification that the assigned paths meet acceptable reliability requirements of the calls; followed by (*b*) the subtask of adjustment of the solutions in (*a*) to achieve optimal balance of the assigned traffic at network. Routine ***A*** assigns the "fair" resource allocations to satisfy the QoS of existing calls, then maximizes the capacity utilization by assigning residual resources at the L processing stages to the existing calls. Routine ***B*** reassigns, as necessary, the final resource allocations at the nodes among the devices clustered at the intermediate nodes to improve load balance and path reliability. In this way, the constraints and conditions of (1) take precedence over the conditions in (2), while the constraints in (3) must be met as an extension of minimum QoS requirements of each call. After repeated application of the two-stage algorithm, the hybrid IP network will not drop an existing call unless there are insufficient resources to satisfy the Q_{min} level of the call along any feasible path in the network.

Routine ***A*** is activated when a new call arrives or an existing call concludes, releasing the resources on its connection path. The inputs to Routine ***A*** include (i) the resource capacities of the system at each of the processing nodes, from the capacity of the wireless cells to the bandwidths and queue sizes of the routers, switches, access points, and gateways in wireline backhaul network; (ii) the number of redundant devices clustered at each node; (iii) the probability distributions of failure for the devices clustered at each node; (iv) the number of calls, N, after the call arrival or departure; (v) the resource requirements of these calls, $R_n = (b_n, s_n)$, and the associated profiles or SLAs, retrieved from the subscriber directory. After the first iteration of the algorithm, the QoS levels from the resource allocation, output from the last iteration of the algorithm, are input to Routine ***A***.

Routine ***A*** searches for the QoS levels that correspond to the resource allocation closest to the "fair" allocation. Computation of the objective function is based on the following "fairness" rules:

$$R_{fair,j} = \left(b_{fair,j}, s_{fair,j}\right) = \frac{\left|D_j\right|C_j}{N} \equiv \left(\frac{\left|D_j\right|B_j}{N}, \frac{\left|D_j\right|S_j}{N}\right), \text{for } j = 0, 1, 2, \ldots, L, \tag{4}$$

where $R_{fair,j}$ is the fair resource allocation and $\left|D_j\right|C_j = \left(\left|D_j\right|B_j, \left|D_j\right|S_j\right)$ is the total capacity of the cluster at node *j*.

In (4) the evaluation of the fair resource allocation can be restricted to calls containing the same type of multimedia substreams. For each of the existing calls with this restriction,

$$R_{a,n} = \left(b_{a,n}, s_{a,n}\right) \in R_n\left(Q_n\right) \tag{5}$$

$$R_{a,n} = \left(b_{a,n}, s_{a,n}\right) = \min\left(\text{abs}\left(R_{fair,j} - R_n\left(Q_n\right)\right)\right), \text{for } j = 0, 1, 2, \ldots, L \tag{6}$$

$$R_{a,n} = \left(b_{a,n}, s_{a,n}\right) \le R_{fair,j},\ j = 0, 1, 2, \ldots, L, \tag{7}$$

where *n* is the call index; $R_{a,n}$ is the resource level corresponding to the QoS level output from Routine ***A*** for call *n*; $R_n(Q_n)$ is the resource requirement corresponding to the QoS level Q_n; and $R_{fair,j}$ is the fair resource allocation at node *j*.

After the fair resource allocations are assigned, the routine then checks to see if the paths created by the node-device pairs determined in Routine ***A*** meet the reliability requirements in (3) of each call. The second subroutine then seeks to satisfy the second condition in (1). The routine redistributes any residual capacity at the nodes of the network among the existing calls. Optimization of the residual capacities in (1) is based on the following rules:

$$Q_{a2,n} \in \boldsymbol{Q}, \text{for each call } n \tag{8}$$

$$R_{a2,n} = R_n\left(Q_{a2,n}\right) \tag{9}$$

$$R_{a2,n} \ge R_{a1,n}, \tag{10}$$

where *n* is the call ID or index; ***Q*** is the set of QoS levels; Q_{a2} is the vector of *N* QoS levels output from the second subroutine of Routine ***A***; R_{a2} is the vector of *N* resource allocations corresponding to Q_{a2}, satisfied at each processing node of the network; and R_{a1} is the vector of *N* resource levels resulting from fair allocation subroutine of Routine ***A***.

The second subroutine ensures that the allocated resources at each node *j* in the end-to-end path are greater than or equal to the "fair" allocations. In this way, each call will be granted at least its equitable share of the resource allocation. An additional advantage of the adjustment in the allocation is that it decreases the search space, as the search will take place only among QoS combinations having indices greater than or equal to those determined by the first subroutine of Routine ***A***. Once the second subroutine of Routine ***A*** minimizes the residual capacities at the nodes, the QoS indices and the node-device pairs that comprise corresponding paths are assigned to the existing calls. These QoS indices and node-device pairs are input to Routine ***B***.

Routine ***B*** seeks to redistribute the residual capacities among the active common-function devices in the cluster at each node *j* on the path of an existing call. In the new distribution, the aggregate resources within the cluster, allocated to all calls through node *j* by Routine ***A***, are shared nearly equally, that is, are "balanced" among the active devices in the cluster. Balancing traffic among redundant devices at the nodes is considered equivalent to minimizing the variances of resource allocation (2) among the devices in the clusters, according to the following rules:

$$Q_{b,n} \in \boldsymbol{Q}, \text{for each call } n \tag{11}$$

$$R_{b,n,j} \equiv R_{b,d_{ij}(n)} = R_n\left(Q_{b,n}\right), \text{for } j \in \{1, 2, \ldots, L-1\}, i \in \{1, 2, \ldots, \left|D_{j,active}\right|\} \tag{12}$$

$$\begin{aligned} R_{b,d_{ij}(n)} &= h \cdot R_{a2,d_{ij}(n)} + \frac{(1-h)}{\left|D_{j,active}\right|} \cdot \sum_{l=1}^{\left|D_{j,active}\right|} \left(\frac{1}{N_{lj}} \cdot \left(\sum_{k=1}^{N_{lj}} R_{a2,d_{lj}(k)}\right)\right) \\ &= h \cdot \left(b_{a2,d_{ij}(n)}, s_{a2,d_{ij}(n)}\right) + \frac{(1-h)}{\left|D_{j,active}\right|} \cdot \left(\sum_{l=1}^{\left|D_{j,active}\right|} \frac{1}{N_{lj}} \cdot \sum_{k=1}^{N_{lj}} \left(b_{a2,d_{lj}(k)}, s_{a2,d_{lj}(k)}\right)\right), 0 \le h \le 1, \end{aligned} \tag{13}$$

such that $Q_{b,n} \ge Q_{a2,n}$, provided the path for *n* connects through node *j*, where *n* is the call ID or index; ***Q*** is the set of QoS levels; Q_b is the vector of *N* QoS levels output from Routine ***B***; R_{a2} (R_b) is the vector of *N* resource allocations

corresponding to Q_{a2} (Q_b), satisfied at each node of the network; R_{a2} is the vector of N resource allocations from the maximum utilization subroutine from Routine ***A***; and $D_{j,active}$ is the subset of D_j consisting of redundant devices which are sufficiently reliable and active on calls whose end-to-end paths connect through node j. The expression in (13) indicates that the final resource allocation at node j to call n using device $d_{ij(n)}$ is the value in the set of convex combinations of the maximum utilitization value at device d_{ij} assigned by Routine ***A***, $R_{a2,d_{ij}(n)}$, and the average of resource allocations assigned by Routine ***A*** to all active devices in the cluster at node j, which is closest to the average of allocations such that the QoS level of call n does not decrease. The main task performed by Routine ***B*** is then an incremental search for the values of resource allocations to the calls, starting with the values output from Routine ***A***, toward the average values of allocated resources in the clusters. At each step of the search, the resulting QoS levels of the calls based on the reallocation are checked to see if the QoS of any call is reduced. If the QoS of a call is reduced, the search is stopped and the values from the previous step are output. To speed execution of this task, the resource and QoS levels are enumerated in discrete values, allowing an efficient search of look-up tables of the resource values and corresponding QoS values. The outputs of Routine ***B*** are fed back to Routine ***A*** to be a subset of the initial population for the next iteration of the algorithm. Thus, in the next iteration Routine ***A*** will have an initial population that is near the "best" solution, reducing the effort to find improved solutions.

4. ELEMENTS OF THE GENETIC ALGORITHM

In its most basic form, called a simple genetic algorithm (SGA), the genetic algorithm consists of five main components: 1) initialization; 2) evaluation of the objective or fitness function; 3) selection; 4) crossover; and 5) mutation.[179] The GA elements can be summarized in the following pseudo-code:

```
Algorithm GA()
{
  randomly generate an initial population of individuals;
  evaluate the fitness of each individual;
  while convergence not achieved and max number of generations are not exceeded
  {
     select individuals probabilistically according to their fitness;
    perform genetic operations of crossover and mutation on the selected individuals;
     evaluate fitness of the resulting individuals;
  }
}
```

1) *Initialization*. In the SGA, a possible solution for the optimization problem is called an individual. A set of individuals represents a group of candidate solutions called a population. A GA starts with an initial population and generates more fit individuals to find the best solution. Each individual is typically represented as a binary vector. It has been shown that more natural representations improve efficiency and produce better solutions.[12] Thus, the individuals in this paper are taken from the set of all paths between the source and destination nodes of a call n, represented by the string of pairs of node-device indices and the associated resource levels allocated at each node to the call's path. This set includes alternate paths made possible by redundant devices available in the clusters at some nodes. Since the node-device pairs can be mapped to a set of integer pairs, and the QoS levels and resources are assumed to take integer values, the individuals here are represented by integer-valued vectors, with intermediate floating-point computations occurring within the algorithm. For N existing calls, the initial population is generated as the set of strings of randomly selected node-device pairs such that the Q_{min} of each call n is satisfied along the path between source and destination without exceeding resource capacities at the nodes. After the first iteration of the two-stage algorithm, the algorithm is initialized by the values fed back from the output of Routine ***B***.

2) *Evaluation*. Each individual is evaluated to obtain its fitness, as defined by a different objective function in each of stage of the algorithm. As previously described, the fitness function in Routine ***A*** is first a fair allocation of resources along the nodes of the path of each call, followed by a reallocation to minimize the residual capacities at the nodes to achieve the highest possible QoS for the existing calls. The failure probability of the path of each call is computed and compared to the acceptable threshold for the call. The achieved multivariate QoS and corresponding resources allocated to a call are observed for each end-to-end path. The QoS values are compared to the minimum QoS requirements for the call and used to calculate a path weight. The fitness function in Routine ***B*** is the optimal balance or sharing of allocated resources among the devices clustered at the nodes of the paths of all calls to maintain or enhance the QoS of each call

found in Routine ***A***. The QoS values are compared to the QoS levels achieved by Routine ***A*** for the call, while the achieved variance of the resource reallocations to the devices at each node is observed to calculate a path weight. Fitness of the path is based on its QoS weight; its resource utilization along its nodes; its balance or load-sharing weight; and its availability, calculated from current failure probabilities of the devices at the nodes of the path.

3) *Selection*. As in [8, 13], normalized geometric selection is employed based on the fitness weights calculated during the evaluation of the individuals. This selection approach uses normalized geometric ranking on the partially ordered set of fitness values.

4) *Crossover*. Two genetic operators, mutation and crossover, are used. Crossover takes two individuals and produces two new individuals, while mutation alters one individual to produce a single new solution. Crossover does not create new genetic material, while mutation can introduce new "genes" into the population. In the two-stage GA, crossover, followed by mutation, are applied if the individual with the best fitness has not yet been determined for each call. Crossover exchanges portions of two selected individuals, considered as parents, so that the offspring preserves part of the first parent and part of the second parent. In this paper, single-point crossover exchanges sub-paths between a pair of end-to-end paths at a crossover node from a set of nodes selected according to a distinct performance criterion at each stage. If this entire set of selected nodes is used, the method can be extended to multiple-point crossovers. In Routine ***A***, following path selection, crossover is performed between the parents using the set of nodes where the devices supporting the path have the largest levels of residual resources after the initial "fair" allocation to the call to meet the $Q_{\min}$ level. In Routine ***B***, crossover occurs at the set of nodes where the devices supporting the path have resource allocations, established by Routine ***A***, nearest to the average resource utilization at those nodes.

5) *Mutation*. The defining characteristic of mutation allows GAs to avoid convergence at local optima. Non-uniform mutation perturbs a path with a given probability (mutation parameter) at one of its nodes based on the failure mechanism and remaining capacity of the device supporting the path at that node. Mutation is used in both stages of the algorithm. In Routine ***A***, mutation is performed at the intermediate node where the device supporting the path has the highest failure probability among all devices supporting that path. The device is removed from the path and the path is switched over to the peer device at the node with the best combination of low failure probability and sufficient residual capacity to support the QoS requirement of the call. In Routine ***B***, mutation occurs at the intermediate node where the device supporting the path has the highest percentage of capacity utilization among all active devices at that node. The mutation parameter is computed as the fraction of unused to total resources allocated to each device at the node.

4.1 GA Routine *A*

Routine ***A*** consists of following tasks: 1) to allocate the fair amount of bandwidth and buffer space at the nodes of the paths of existing calls; 2) to check to see if the resulting allocated paths satisfy the path reliability requirements of the calls; and 3) to allocate any available resources remaining after the two tasks to maximize the link capacity utilization and, consequently, the QoS for the calls. The fitness function for the first task is described in pseudo-code below.

```
Fitness_function_GA_RA1 (call_ID, QoS_index, source_ID, destination_ID, num_hops,
  [(node_ID (1), device(node_ID(1))), …, (node_ID(num_hops+1), device(node_ID(num_hops+1)))],
  [num_devices(1), …, num_devices(L)])
1 for I = 1, …, num_hops+1
2 for device(node_ID(I)) = 1, …, num_devices(node_ID(I))
    if bandwidth(call_ID, QoS_index, device(node_ID(I))) > fair_bandwidth(node_ID(I)) and
    buffer_space(call_ID, QoS_index, device(node_ID(I))) > fair_buffer_space(node_ID(I))
        return (fair_bandwidth(node_ID(I)) – bandwidth(call_ID, QoS_index, device(node_ID(I))),
    fair_buffer_space(node_ID(I)) – buffer_space(call_ID, QoS_index, device(node_ID(I))),
    node_ID,device(node_ID(I)));
        else
        return (bandwidth(call_ID, QoS_index, device(node_ID(I))) – fair_bandwidth(node_ID(I)),
    buffer_space(call_ID, QoS_index, device(node_ID(I))) – fair_buffer_space(node_ID(I)), node_ID(I),
    device(node_ID(I)));
    go to 2
  go to 1
  end
```

The outer loop of the fitness function iterates through the nodes between source and destination of the call, while the

inner loop iterates over the number of redundant devices at each intermediate node, determining if the bandwidth and buffer space of devices at each node assign the fair bandwidth and buffer space allocations to the call. The above routine assumes at each node that bandwidth(call_ID, QoS_index, device(node_ID) and buffer_space(call_ID, QoS_index, device(node_ID)) are the bandwidth and buffer space, respectively, required by the call_ID if granted a QoS level equal to QoS_index. Furthermore, the routine possibly assigns "fair" resource allocations to multiple strings of node-device pairs between source and destination.

Once the fair allocations are assigned to the nodes and devices of the candidate paths for each call, the failure probability of each candidate path is evaluated to determine if it meets the reliability (or availability) requirement for the call. If the candidate does not meet the requirement, it is omitted from the possible individual solutions for the call. The failure events of devices in the same cluster are assumed independent and identically distributed (same failure PDFs), while failures of devices in the clusters at different nodes are assumed independent. Note that, in simulations of the algorithm, the PDFs of failure of the devices on a call's path are progressed in time over the duration of each call. The routine computes the probability of failure for the path incrementally, starting from the device at the source through the device at the last node; at each step the partial result is compared with the failure threshold for the call. If the path meets the reliability requirement, the entire path is returned with the computed path failure probability. If the path fails at an intermediate node, the partial path is returned terminating at the node and device at which the threshold was exceeded, that device ID is assigned a negative value to be omitted from inclusion in a candidate path for this call. The following is a pseudo-code description of the routine.

```
Fitness_function_GA_RA_REL (call_ID, fail_threshold, num_hops, [(node_ID (1), device(node_ID(1)),
  failure_PDF(device(node_ID(l))), …, (node_ID(num_hops+1), device(node_ID (num_hops+1)),
  failure_PDF(device(node_ID(num_hops+1)))], [num_devices(1), …, num_devices(L)])
  path_failure_prob = 1.0;
3 for l = 1, …, num_hops+1
    path_failure_prob = path_failure_prob * failure_PDF(device(node_ID(l));
    if (path_failure_prob < fail_threshold)
       return ([(node_ID (1), device(node_ID(1)),…, (node_ID(l), device(node_ID (l))], path_failure_prob);
    go to 3
       else
       return ([(node_ID (1), device(node_ID(1),…, (node_ID(l), device(node_ID (l)) – (num_devices(l) + 1))],
    path_failure_prob);
    end
```

To efficiently allocate the resources remaining after the first task in order to maximize the capacity utilization at the nodes of each hop, the second task of Routine ***A*** (Routine ***A2***) is based on three components: the generation of the initial population; the fitness function for the GA in this routine; and decomposition of the search space in the event that the number of calls, service classes, or number of nodes in the network impede convergence of the algorithm. Outputs of Routine ***A1*** form just one set of possible solutions to the fair allocation problem. Other candidate solutions for the initial population can be found within a neighborhood of the members of this set of solutions subject to the QoS, reliability, and resource constraints. Each member of the GA population is an array of the QoS indices of size *N* for the *N* calls present in the network at the time Routine ***A2*** is entered, along with their corresponding end-to-end paths. Each path is represented by a string of pairs of node and device indices from source to destination.

One potential procedure to find suitable seeds for the initial population is shown below. Denote the QoS indices, node-device indices from source to destination, for the existing calls, produced by Routine ***A1*** as the "init_seed."

```
Initialize_population(pop_size)
{
   init_pop(1)=fitness_function_GA_A2(init_seed);
   for l = 2, …, (max_QoS_index/2)
            init_pop(l)=fitness_function_GA_A2(init_seed—(l+1));
   for l = (max_QoS_index/2), …, pop_size
            init_pop(l)=fitness_function_GA_A2(random(N) ≺ init_seed);
}
```

The operator "—" denotes conditional vector subtraction. If the result of the vector subtraction is less than 1, the result is set to 1. Moreover, (random(N) ≺ init_seed) denotes the generation of *N* random QoS indices (seeds) that are

less than those of Routine ***A1***. In the model the highest QoS level is 1, while (max_QoS_index) takes the lowest level at 4^M, for M services, representing no active service of any class in the call.

The fitness function for the second task of Routine ***A*** (Routine ***A2***) is described in the following pseudo-code.

```
Fitness_function_GA_RA2 ([call_ID(N), QoS_index(N), num_hops(N),
  [(node_ID(1,1), device(node_ID(1,1))), …, (node_ID(num_hops(1)+1,1),
  device(node_ID(num_hops(1)+1,1)))], …, [(node_ID (1,N), device(node_ID(1,N))),
  …, (node_ID(num_hops(N)+1,N), device(node_ID(num_hops(N)+1,N))],
  [num_devices(1),…, num_devices(L)])
4 for I = 1, …, L
  total_bandwidth(node_ID(I)) = 0;
  total_buffer_space(node_ID(I)) = 0;
5 for J = 1, …, N
  for device(node_ID(I,J)) = 1, …, num_devices(node_ID(I,J))
  total_bandwidth(node_ID(I)) = total_bandwidth(node_ID(I)) + bandwidth(call_ID(J), QoS_index(J),
  device(node_ID(I,J)));
  total_buffer_space(node_ID(I)) = total_buffer_space(node_ID(I)) + buffer_space(call_ID(J),
  QoS_index(J),device(node_ID(I,J)));
  go to 5;
  if total_bandwidth(node_ID(I)) > link_bandwidth_capacity(node_ID(I))
    return ((link_bandwidth_capacity(node_ID(I)) –
  total_bandwidth(node_ID(I)))/link_bandwidth_capacity(node_ID(I)));
  else
    return (total_bandwidth(node_ID(I))/link_bandwidth_capacity(node_ID(I)));
  if total_buffer_space(node_ID(I)) > link_buffer_capacity(node_ID(I))
    return ((link_buffer_capacity(node_ID(I)) –
  total_buffer_space(node_ID(I)))/link_buffer_capacity(node_ID(I)));
  else
    return (total_buffer_space(node_ID(I))/link_buffer_capacity(node_ID(I)));
  go to 4;
  end
```

4.2 GA Routine *B*

Routine ***B*** reassigns, as necessary, the final resource allocations at the nodes among the devices clustered at the intermediate nodes to improve load balance. Routine ***B*** adjusts the allocations, constrained to maintain the QoS provisioning determined by Routine ***A***. The fitness function for Routine ***B*** is described in pseudo-code below.

```
Fitness_function_GA_RB ([call_ID(N), QoS_index_RA2(N), num_hops_RA2(N), inc, threshold,
  [(node_ID_RA2(1,1), device(node_ID_RA2(1,1))), …, (node_ID_RA2(num_hops_RA2(1)+1,1),
  device(node_ID_RA2(num_hops_RA2(1)+1,1)))], …, [(node_ID_RA2(1,N), device(node_ID_RA2(1,N))),
  …, (node_ID_RA2(num_hops_RA2(N)+1,N), device(node_ID(num_hops_RA2(N)+1),N))],
  [num_devices(1),…, num_devices(L)])
6 for I = 1, …, L
  avg_bandwidth(node_ID_RA2(I)) = 0;
  avg_buffer_space(node_ID_RA2(I)) = 0;
  total_active_devices(node_ID_RA2(I)) = 0;
7 for J = 1, …, N
  QoS_index_RB(J) = QoS_index_RA2(J);
8 for device(node_ID_RA2(I,J)) = 1, …, num_devices(node_ID_RA2(I,J))
  if failure_probability(device(node_ID_RA2(I,J)) > threshold
    go to 8;
    else
    total_active_devices(node_ID_RA2(I)) = total_active_devices(node_ID_RA2(I)) + 1;
    avg_bandwidth(node_ID_RA2(I)) = avg_bandwidth(node_ID_RA2(I)) +
        bandwidth(call_ID(J), QoS_index_RA2(J), device(node_ID_RA2(I,J));
    avg_buffer_space(node_ID_RA2(I)) = avg_buffer_space(node_ID_RA2(I)) +
```

```
Fitness_function_GA_RB ( ) (continued)
   buffer_space(call_ID(J), QoS_index_RA2(J), device(node_ID_RA2(I,J));
   go to 7;
   avg_bandwidth(node_ID_RA2(I)) =
         avg_bandwidth(node_ID_RA2(I))/total_active_devices(node_ID_RA2(I));
   avg_buffer_space(node_ID_RA2(I)) =
         avg_buffer_space(node_ID_RA2(I))/total_active_devices(node_ID_RA2(I));
   return(avg_bandwidth(node_ID_RA2(I)), avg_buffer_space(node_ID_RA2(I)));
   go to 6;
 9 for J = 1, …, N
10 for I = 1, …, num_hops_RA2(J)+1
   for device(node_ID_RA2(I,J)) = 1, …, num_devices(node_ID_RA2(I,J))
   if (bandwidth(call_ID(J), QoS_index_RA2(J), device(node_ID_RA2(I,J)) >
         avg_bandwidth(node_ID_RA2(I)))
   bandwidth_delta(node_ID_RA2(I,J)) = bandwidth(call_ID(J), QoS_index_RA2(J),
         device(node_ID_RA2(I,J))) – avg_bandwidth(node_ID_RA2(I));
     else
     if (bandwidth(call_ID(J), QoS_index_RA2(J), device(node_ID_RA2(I,J))) <
         avg_bandwidth(node_ID_RA2(I)))
     bandwidth_delta(node_ID_RA2(I,J)) = avg_bandwidth(node_ID_RA2(I)) –
         bandwidth(call_ID(J), QoS_index_RA2(J), device(node_ID_RA2(I,J)));
   else
   if (buffer_space(call_ID(J), QoS_index_RA2(J), device(node_ID_RA2(I,J)) >
         avg_buffer_space(node_ID_RA2(I)))
   buffer_space_delta(node_ID_RA2(I,J)) = buffer_space(call_ID(J), QoS_index_RA2(J),
         device(node_ID_RA2(I,J))) – avg_buffer_space(node_ID_RA2(I));
     else
     if (buffer_space(call_ID(J), QoS_index_RA2(J), device(node_ID_RA2(I,J)) <
         avg_buffer_space(node_ID_RA2(I)))
   buffer_space_delta(node_ID_RA2(I,J)) = avg_bandwidth(node_ID_RA2(I)) –
        buffer_space(call_ID(J), QoS_index_RA2(J), device(node_ID_RA2(I,J)))
   else
11  for K = 1, …, inc
   bandwidth_RB(call_ID(J), QoS_index_RA2(J), device(node_ID_RA2(I,J))) =
      bandwidth(call_ID(J), QoS_index_RA2(J), device(node_ID_RA2(I,J))) +
      (K * bandwidth_delta(node_ID_RA2(I,J))/inc);
   buffer_space_RB(call_ID(J), QoS_index_RA2(J), device(node_ID_RA2(I,J))) =
      buffer_space(call_ID(J), QoS_index_RA2(J), device(node_ID_RA2(I,J))) +
      (K * buffer_space_delta(node_ID_RA2(I,J))/inc);
   if ((QoS_index(bandwidth_RB(call_ID(J), QoS_index_RA2(J), device(node_ID_RA2(I,J))),
      buffer_space_RB(call_ID(J), QoS_index_RA2(J), device(node_ID_RA2(I,J)))) < QoS_index_RA2(J)) or
      (bandwidth_RB(call_ID(J), QoS_index_RA2(J), device(node_ID_RA2(I,J))) >
      bandwidth_capacity(device(node_ID_RA2(I,J))) or
      (buffer_space_RB(call_ID(J), QoS_index_RA2(J), device(node_ID_RA2(I,J)) >
      buffer_capacity(device_ID_RA2(I,J)))
      return(QoS_index_RB(J), bandwidth_RB(call_ID(J), QoS_index_RA2(J), device(node_ID_RA2(I,J))) –
   bandwidth_delta (node_ID_RA2(I,J)), buffer_space_RB(call_ID(J), QoS_index_RA2(J),
   device(node_ID_RA2(I,J))) – buffer_space_delta(node_ID_RA2(I,J)), node_ID_RA2(I),
   device(node_ID_RA2(I,J)));
      else
      QoS_index_RB(J) = QoS_index(bandwidth_RB(call_ID(J), QoS_index_RA2(J),
   device(node_ID_RA2(I,J))), buffer_space_RB(call_ID(J), QoS_index_RA2(J), device(node_ID_RA2(I,J))));
   go to 11;
 go to 10;
 go to 9;
 end
```

The fitness function routine first computes the average resource utilization of Routine ***A*** among the active devices at the nodes of the path used for all calls. It then incrementally adds a uniform fraction of the difference between the

average allocation at the node and the allocation from Routine ***A2*** to the device on the path of a call to that device's allocation. Lastly, the routine compares the new QoS index of each call_ID that results from the adjusted allocations with the QoS index from Routine ***A2*** to ensure that the bandwidth and buffer capacities of the device are not exceeded and the new QoS is not lower. If either constraint is violated, the routine returns the last computed QoS index and corresponding last adjusted resource allocation, along with the node ID and device ID on the path of the call. Otherwise, the routine computes the new QoS index based on the current adjusted resource allocation, calculates the next resource adjustment, and computes the resulting QoS until some constraint is exceeded. Routine ***B*** thus achieves improved balance of resources among the devices clustered at the nodes without degrading the QoS levels attained with maximal resource utilization. The QoS levels in Routine ***A2*** take precedence over the extent of load sharing in Routine ***B***.

After a call arrives or departs the network, the steps of the algorithm are executed in the following order: (1) assign resource allocations on the paths of all calls to meet their maximum QoS and stop, unless the resource capacities are exceeded; in which case, set allocations as close as possible to their "fair" values to meet their minimum QoS levels based on evaluation of the fitness in Routine ***A1***; (2) calculate failure probabilities of the paths assigned in (1) to each call to see if they satisfy the reliability (availability) requirement of the call; if requirement is not met, discard the path and device where the requirement fails, test another candidate path; if no path remains, return to (1) with discarded devices omitted from the clusters at the nodes; otherwise, go to (3) Routine ***A2*** to search for the global optimum resource allocation. In (3), generate or input the initial population, evaluate the fitness of each individual; if the most fit individual is found for the call, save the best QoS and corresponding path indices and resource allocations in a results array and stop; otherwise, create intermediate population members while the population is below limit by first performing the crossover defined for Routine ***A2*** on random pairs of members to get new individuals with greater fitness, then performing the mutation defined for Routine ***A2*** on random individuals, insert new members into population until the population limit is met, then return to evaluate the fitness of the new members; go to (4) Routine ***B*** to readjust the optimum QoS indices and resource allocations to the paths found in (3) to achieve the best possible load balance among devices at the nodes. In (4), input the results array from (3) as the initial population, compute the average resource utilization at each of the nodes, determine the best increment from the allocations in (3) to the computed averages that maintains the optimum QoS index of each call in the population and stop; otherwise, create intermediate population members while the population is below limit by first performing crossover defined for Routine ***B*** on random pairs of individuals, then performing the mutation defined for Routine ***B*** on random individuals, insert new members into the population until the population limit is met, then reevaluate the average utilization at the nodes and determine best adjustments in the allocations to achieve best possible load balance. The algorithm stops unless the population is too large to ensure convergence.

4.3 Techniques for improved convergence

The network architecture can yield a search space of large dimensions due to the number of multi-hop paths possible for each call, compared to one-hop networks.[8, 9] If the search space is too large, it may be difficult for the GA to avoid convergence to local optima. It has been found that, for a single-resource, single-hop wireless network, when the number of calls is greater than $N = 10$ calls, then the results of the GA approach do not converge satisfactorily.[8] This motivates the use of partitioning schemes to limit the search process. For the QoS-provisioning problem with multiple processing nodes and multiple resources at each node, the first stage of the partition naturally occurs at each node of the paths of existing calls, starting from source, to the kth processing node, the destination. This partition results in k one-hop path searches, $k \leq L$. The two-stage algorithm is repeated on each hop in a forward recursion from source to destination. The second step of the partition occurs among the calls based on required QoS level. For N calls in the network, those calls requiring QoS levels in the range from 1 to $4^M/H$, H a factor of 4^M, can be grouped into the first set. Those with QoS levels in the range $4^M/H + 1$ to 4^M (2/H) can be grouped into a second set, and so on, until all calls are grouped into one of H sets. The scheme then seeks to find the best solution within each set of the second partition on each hop of the first partition of the paths of the calls. Next the scheme determines the best solution among those that were selected from each partition. In the next step, the best solution will be the one that maximizes the utilization and local balance at each node, and simultaneously corresponds to the maximum achievable QoS level for each call. Lastly, the one-hop solutions are used to increment a forward recursion on the next hop in the path of each call. Initial simulation results indicate that a partition of calls based on QoS into $H = N/4$ sets and a population of individuals into sets of $4^M/L$ assures convergence of the two-stage GA. For example, with $N > 64$ calls, $M = 4$ services, $L = 4$ processing nodes, and $D = 4$ redundant devices in the cluster at each node, the number of call partitions is $H = 16$ and the population

of individuals is partitioned into search sets of 256 quality level-device pairs for each hop.

5. SIMULATION RESULTS

In the simulations performed to date, representative service substreams from the traffic classes in Tables 1 and 3 have been used. The values for the QoS attributes for each class are varied within ranges recommended by the IETF and 3GPP for the QoS architecture.[4, 5] The video component of the multimedia calls is an example of high activity of video traffic. The voice and data substreams, i.e., the conversational, interactive and background classes, are generated according to exponential distributions and truncated Pareto distributions, respectively; the latter approximate the heavy-tail distributions for the self-similar behavior exhibited by Internet downloads during web browsing. The parameter values of the distributions are changed from one simulation run to another. The duration of calls is allowed to vary from 60 to 300 seconds. The throughput requirement for the video substream used in the simulation is 24 packets/s, with 1,500 bytes per packet, yielding a required bit rate of 144 kbps. To generate high, medium and low QoS levels for video, the I, P, and B frames in the substream are removed.[14] The B frames are dropped to obtain the medium QoS level, then both the B and P frames are dropped to represent low QoS. The high, medium, and low QoS levels for audio are generated by selecting the average bit rate of the source as 32, 16, and 8 kbps, respectively. The delay requirement for speech is fixed at 100 ms and speech traffic is assigned a priority of 1 in the descending, eight-level priority scheme in the ToS specification. The high, medium, and low QoS levels for interactive and background services, such as file transfer and web browsing, are generated by varying the average bit rate as 128, 64, and 8 kbps, respectively. Bursts from 100 to 5000 packets occur at the rate of 2,048, 512, and 256 kbps for high, medium and low service, respectively.

The baseline IP network configuration for the simulations consists of $L = 6$ processing nodes: a single high-speed Ethernet access gateway; 4 redundant high-speed, 1000BaseSX Ethernet core switches; 4 redundant high-speed, 1000-Mbps core routers; 8 redundant high-speed, Ethernet edge routers; 4 redundant wireless base stations that cover up to 3 overlapping cells. Each cell has a maximum bandwidth of 8.192 Mbps, or, approximately, 682 packets/s. Queue capacity of each device at the intermediate nodes is assumed to be 4 Gigabytes (GB); otherwise, capacities are determined by the device specifications at each node. The failure mechanism of the redundant devices at each processing node follows a common exponential distribution. Exponential rate parameters differ between the nodes and are based on estimates of the mean-time between failure (MTBF) for each device type. The path reliability (failure probability) is set to a uniform level of 10^{-5}, equivalent to "five nines," for all calls. The average arrival rate of calls is varied from 1 to 5 per minute. The number of calls is varied between 1 and 250.

Simulation runs of the algorithm required from 3 to 14 hours for output. Results indicate that, as long as the available resources along network paths meet the required bandwidth and buffer requirements of the calls, the resource allocations to existing calls based on the two-stage GA algorithm coincide with the allocations to these calls if they are assigned the highest QoS levels. During runs with higher arrival rates of calls of longer duration and runs with long data bursts at the highest bit rates, overload of network resources occurs. In these cases, the two-stage GA shows significantly increased call blocking and dropping rates, with minimal levels of unused capacities at the nodes.

Initial simulation results show that the algorithm is able to admit as many as 90 calls to the network simultaneously, albeit with some calls at the lowest QoS levels and in the absence of data bursts. Under the same conditions, the network without the algorithm was able to process no more than 29 calls. Hence, these results represent an improvement of greater than 310% in call admissions. The average improvement in the number of admitted calls over all runs, as model parameters were varied, is a more modest 55.4%, while the maximum gain occurs in scenarios of slow call arrival rates with high levels of data bursts and is nearly 367%. However, gains in number of calls simultaneously processed come at the expense of the QoS of the calls when resource usage is high. For a three-service network, with 64 levels of quality, then minimum average QoS index delivered to existing calls was 24.7, with the worst reduction in QoS of 50-70% during overload conditions. However, these averages do not account for the grade-of-service advantages that the same quality in one service may have over the quality of another. Simulations of the same network scenarios show that call blocking occurs at significantly higher rate when the algorithm is not used. The blocking rate is as high as 91% in cases when high call arrival rates and data burst rates cause capacity overload. In the same cases, use of the two-stage GA reduces the blocking rate to less than 10%, by decreasing QoS levels of existing calls to make additional resources available. In longer simulation runs, a gradual reduction in available resources at nodes is observed due to the discarding of devices that contribute to the failure of the "hard" requirement of path reliability.

6. CONCLUSIONS

An adaptive allocation of the resources of bandwidth and buffer space has been proposed for a hybrid network of broadband wireless access points and wired nodes that provides multimedia services over IP to subscribers. Starting from the CPE located in wireless cells, the architecture consists of several packet-processing stages implemented at intermediate nodes by wireless base stations, edge routers, core routers, core switches, and access gateways to an IP backbone network. The multi-hop network model includes the concept of path reliability in the set of QoS attributes and the use of clusters of redundant devices at the intermediate nodes to enhance path reliability. Clusters allow capacity expansion and automatic switchover to peer devices in the event a device fails or its capacity is depleted on the end-to-end path of a call. A population of individual strings representing node-device pairs along candidate paths from source to destination of all calls is constructed to meet QoS requirements of the calls. A two-stage GA is formulated to maximize resource utilization on both wireless and wired hops, while maintaining high availability through the adaptive assignment of the redundant devices at the nodes to reduce downtime and call blocking. Subscribers to multimedia services are assumed to have SLAs that specify a range of acceptable QoS levels instead of a fixed QoS for each service. The flexibility of QoS requirements allows the GA to reallocate resources adaptively to calls in order to jointly maximize the number of calls in the network and achieve the best possible path availability. The former objective is equivalent to minimizing call blocking, while the latter is equivalent to simultaneously minimizing call dropping and failure recovery time of the devices on each path. Such adaptation lowers the QoS levels of existing calls to levels at or above the minimum levels found in the SLAs. Each stage of the GA is designed to meet distinct performance objectives. The first stage assigns "fair" resource allocations to each call along candidate paths and then reallocates any residual capacities at the nodes that remain after the fair allocations to improve QoS. The reliability of each candidate path is also checked. The second stage adjusts the allocations of the first among the active devices at each node to achieve improved traffic load balance at the nodes. Initial results from simulations of a few network scenarios indicate significant improvements in minimizing call blocking and attaining load balance among network elements with a limited number of calls, while providing acceptable QoS levels to subscribers. Moreover, the adaptive response of the GA allows increases in the QoS levels of existing calls in the event that one or more calls leave the network or failed devices are restored to clusters at the nodes. In this way, the GA approach adapts to changing traffic loading and device availability.

The approach has been limited to the resource provisioning of multimedia service requests. As such, it indirectly addresses call admission and congestion management. However, administrative and control signaling as well as security messages have not been addressed. As suggested by the priority scheme of the WFQ hierarchy of MPLS, the auxiliary messages can be aggregated in separate queues and assigned higher handling and retention priorities than those for real-time conversational services. The GAs can be extended to allocate resources to a hierarchy of queues that share the common resources at each node according to their assigned priorities. Moreover, representation of individuals in the population as strings of node-device indices suggests application of graph-theoretic concepts, including the notion of a cut set of nodes, to the GA operators of crossover and mutation as well as in the search procedures. Mapping the range of acceptable QoS levels for each subscriber into a corresponding set of resource values suggests the use of fuzzy logic methods to create normative rules for resource assignment. These rules could improve selection in large populations.

ACKNOWLEDGMENTS

The author wishes to thank FIT for its support and to cite the leadership of the Internet Engineering Task Force (IETF) in establishing the discourse on QoS, high availability and load sharing in wideband IP networks for multimedia services.

REFERENCES

1. C. Giammarco, K. Malick, and P. Morreale, "Wireless quality of service assurance for network survivability," *IEEE Military Commun. Conf. Proc., MILCOM 1999*, **2**, pp. 893-896, 1999.
2. S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated services", *RFC 2475*, IETF, 1998.
3. Y. Bernet, R Yavatkar, P. Ford, F. Baker, L. Zhang, M. Speer, R. Braden, B. Davie, J. Wroclawski, and E. Felstaine, "A framework for integrated services operation over DiffServ networks", *RFC 2998*, IETF, 2000.
4. A. Dutta-Roy, "The cost of quality in Internet-style networks," *IEEE Spectrum*, **37**, pp.57-62, 2000.

5. "Technical specification group services and system aspects; QoS concept and architecture (Release 5)", *3GPP TS 23.107 V5.3.0 (2002-01)*, 3rd Generation Partnership Project, Valbonne, France, 2002. http://www.3gpp.org
6. M. Naghshineh and M. Willebeek-LeMair, "End-to-end QoS provisioning in multimedia wireless/mobile networks using an adaptive framework," *IEEE Commun. Mag.*, **35**, pp. 72-81, 1997.
7. A. Acampora, "Wireless ATM: a perspective on issues and prospects," *IEEE Trans. Pers. Commun.*, **3**, pp. 8-17, 1996.
8. M.R. Sherif, I.W. Habib, M. Nagshineh, and P. Kermani, "Adaptive allocation of resources and call admission control for wireless genetic algorithms," *IEEE J. Sel. Areas Commun.*, **28**, pp. 268-282, 2000.
9. D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison Wesley, Reading, MA, 1989.
10. J. Chang, C. Sun, and E. Mizutani, *Neuro-Fuzzy and Soft Computing*, Prentice-Hall, Englewood Cliff, NJ, 1997.
11. S. R. Ladd, *Genetic Algorithms in C++*, M&T Books, New York, NY, 1996.
12. Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer-Verlag, New York, NY, 1994.
13. C. Houck, J. Joines, and M. Kay, "A genetic algorithm for function optimization," *Tech. Rep. 95-09*, North Carolina State Univ., Dept. of Indust. Eng., 1995.
14. N. Yeadon, F. Garcia, D. Hutchison, and D. Shepherd, "Filters: QoS support mechanisms for multipeer communications," *IEEE Select. Areas Commun.* (*Special Issue Distr. Multimedia Sys. and Technol.*), **4**, pp. 1245-1262, 1996.