# Active Resource Management for The Differentiated Services Environment[*]

Ananthanarayanan Ramanathan, Manish Parashar
The Applied Software Systems Laboratory
Department of Electrical and Computer Engineering
Rutgers, The State University of New Jersey
94 Brett Road, Piscataway, NJ 08854
{ananthr, parashar}@caip.rutgers.edu

## Abstract

*This paper presents a mechanism for active resource management (ARM) in a differentiated services environment. While the differentiated services architecture and the bandwidth broker agent provide a mechanism for QoS management through resource reservation, this mechanism is based on a static provisioning of resources. As bandwidth requirements are typically dynamic, such a static reservation approach can either lead to wasted bandwidth or leave applications resource-starved. The active resource management approach presented in this paper addresses this problem by dynamically reallocating resources based on current network state and applications requirements. An implementation and evaluation of ARM using the NS-2 simulation toolkit is also presented.*

## 1. Introduction

A large percentage of the traffic on the Internet today is multimedia related real time data that is critical to an application. Such time-critical data require some level of Quality of Service (QoS) guarantees. The Internet Protocol (IP), however, is based on best effort and lacks the ability to provide such QoS guarantees [1]. Various solutions have been proposed to address this problem and guarantee applications their required resources. These include integrated services (e.g. RSVP), differentiated services and MPLS.

The Differentiated Services (DS) network architecture attempts to provide QoS guarantees in the most scalable and least complex manner. A DS domain defines two levels of service provisioning: the standard best effort service, which is similar to IP, and the premium service where a client's request for service guarantees are satisfied. In the DS architecture, the Bandwidth Broker (BB) manages a domain's resources using service policies defined based on the clients requirements. The BB reserves the bandwidth requested by clients, for a price. This reservation however, is made without any understanding of the nature of the information that will be transmitted. Although such a reservation provides a uniform resource allocation, the resource provisioning remains static, and can lead to wastage of bandwidth.

This paper presents the active resource management (ARM) approach that actively manages the resources in a DS domain by dynamically reallocating resources at the BB based on the current requirements of applications and the state of the network. The ARM approach is motivated by the observation that the actual traffic generated by a client rarely approaches its peak rate that has been reserved for the client. Consequently, when the clients traffic drops below the reserved rate, a portion of the unused bandwidth can be returned to the pool of available bandwidth. ARM is implemented and evaluated using the NS-2 [4] simulator toolkit. Our experiments demonstrate that by actively over provisioning and dynamically reallocating available resources, ARM can effectively increase the overall utilization of the available bandwidth and support increased numbers of clients while honoring QoS guarantees.

The rest of this paper is organized as follows. Section 2 presents background material and outlines related work. Section 3 describes the ARM approach. Section 4 outlines the implementation and evaluation using the NS-2 network simulator, and present experimental results. Section 5 presents some conclusions.

## 2. Background

### 2.1. Differentiated Services

Differentiated Services is a set of technologies that are used to provide quality of service in a world of best ef-

[*] The research presented in this paper is based upon work supported by the National Science Foundation under Grant Numbers IIS 9872995 (KDI) and ACI 9984357 (CAREERS) awarded to Manish Parashar.

fort service provisions [2]. In DS, all the complexities are pushed out to the edge routers and the core routers are maintained as simple as possible. The differentiated services architecture is based on a simple model where the traffic entering a network is classified and possibly conditioned at the boundaries of the network, and then assigned to different behavior aggregates. In the approach taken by DS individual micro flows are classified at the edge routers in the network, into one of the many classes. It then applies a per-class service in the core of the network. The classification is done at the ingress router, based on one or more bits in the packet. Then the packet is marked, using code points, as belonging to one of the many classes and injected into the network. The core routers that forward the packet examine this marking and and use it to decide how the packet should be treated. Most of the work in this scheme is done at the edge routers. These routers are responsible for classifying, using a multifield classifier and a traffic meter, and decide the next action to be taken on the packet. The traffic meter is used to ensure that the packet conforms to the traffic profile previously agreed upon by the network provider and the customer. The packets are then marked with Diffserv Codepoint (DSCP). DS uses six bits of the IPV4 or IPV6 header to convey the DSCP, which selects a per hop behavior (PHB).

All packets with the same code point are grouped together and are known as a behavior aggregate (BA). There are two defined PHBs: expedited forwarding (EF), and assured forwarding (AF) [7]. EF PHB supports low loss, low delay, and low jitter. AF PHB defines four relative classes of service with each service supporting three levels of drop precedence.

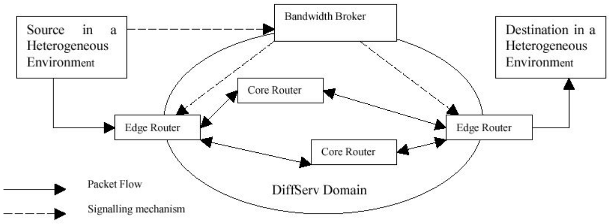

**Figure 1. A generic model of a bandwidth broker in a DS domain.**

Figure 1 shows a generic model of the DS, and how the BB interacts within a DS domain. Differentiated Services is being regarded as a reasonable solution to provide Quality of Service on the Internet. Research is being carried on in various universities, and IETF has a DS working group. Companies like Cisco and IBM provide DS functionality in their routers, and Nortel Networks has evaluated DS using NS-2 toolkit [4].

## 2.2. Bandwidth Broker

Bandwidth Broker is an agent that provides a centralized mechanism to control the resources within a DS domain. All agreements between the customer and the service provider that pertain to the type of service required are known as service level agreements (SLA). The BB manages a domain's resources using service policies defined based on the clients requirements. These SLAs are used to define the relation between policies and the PHBs, while a service provisioning policy (SPP) indicates how traffic conditioners are configured at the edge of the domain and how the traffic streams are mapped to the DS behavior aggregates. The bandwidth broker requires both the SLAs and the SPPs to achieve a range of services, which are provided to the user. Based on the SLAs the broker decides whether it can provide the allocation, and configures the edge router to mark and classify the packets as decided in the SLA [6].

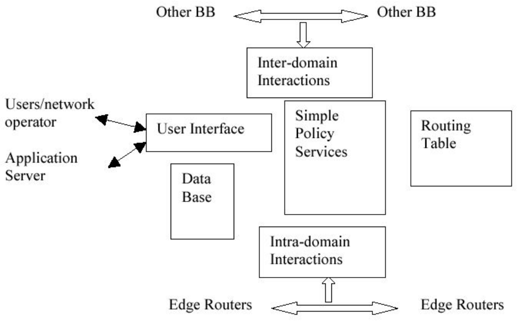

**Figure 2. Functional decomposition of the bandwidth broker as defined by the Qbone team.**

The bandwidth broker consists of a few basic components shown in Figure 2 [13]:

1. **User Interface:** The user/application interface provides a means for the user to make resource requests directly, or to the network operator who enters the users' requests. The interface also receives messages from setup protocols (for example RSVP messages).

2. **Inter-domain Interactions:** The interactions provide a method of allowing peer BBs to make requests for resources and take admission control decisions to enable flow of traffic.

3. **Intra-domain Interactions:** The interactions provide a method for the BB to configure the edge routers within the domain so as to provide quality of service.

4. **Routing Table:** A routing table is maintained at the BB to access inter-domain routing information so that

the BB can determine the edge routers and the downstream routers before allocating their resources. Further, additional routing paths may be maintained in the routing table for different flows within the domain.

5. **Database:** A database is used to store information about all the BB parameters. The information that is stored within the repository includes SLAs, current reservations, router configurations, DSCP mapping, and policy information.

The Bandwidth Broker has been designed to add intelligence to the DiffServ, to help optimize the existing resources. An advisory committee has been initiated to define the protocols implemented by the broker.

## 3. Active Resource Management (ARM)

### 3.1. Architectural Framework:

As described above, in the DS model, predefined policies are used to allocate resources to a particular client. These policies are based on the clients peak traffic rate, the time for which the service is required, and the acceptable delay and jitter. For many applications, for e.g. where the transmitted information is in the form of streaming media, the traffic rate is bursty and is rarely at the peak transfer rate. In such a situation, a portion of the allocated bandwidth remains unused. However, as this bandwidth is provisioned to that particular client, no other client can use it. Furthermore, bandwidth is reserved for a particular client and this reservation is applicable to all applications belonging to the client. As a result, a client will make reservations based on its maximum requirements, and every application belonging to the client, whether it is a streaming media application or a simple mail application, will get this allocated service.

Bandwidth broker agents are used in the DS architecture to enable a more intelligent allocation of resources. BB agents maintain a database of parameters pertaining to the various flows. These parameters include service level agreements (SLAs), current reservations/allocations, edge router configurations, service mappings/DSCP mappings, policy information, and management information. Based on these parameters the broker agent makes a reservation for the client and assigns a DSCP for that service. This leads to a better allocation of resources as it takes the relevant sets of parameters into consideration for each reservation. Each client gets to define its requirements and these get translated into SLAs, which affect the resource allocation. The resulting allocation, however, is still static and can lead to wasted bandwidth and/or starved clients.

There is therefore a need for actively managing resource allocations so as to reclaim the bandwidth wasted on each reservation and if possible re-allocate the reclaimed bandwidth to another client. ARM reallocates the excess bandwidth without loss of promised service, by effectively monitoring the rate at which the client generates traffic and the amount of the alloted bandwidth that is used at any given time.

In DS, each client is provided with a service level specification (SLS) specifying the amount of bandwidth, the duration of the connection and a few other parameters. These parameters together map to particular DSCP that defines the clients assigned level of service. Incoming client packets are then marked using this DSCP to inform the routers to forward the packets with appropriate priority. In order to measure the traffic rate of every client, the bandwidth broker agent can use a meter provided by the DS. For example, the TSW Tagger [9] is a meter that measures the average traffic rate, using a specified window size for the TSW2CM and TSW3CM [10] policers. With the knowledge of incoming traffic, different DSCPs are defined for various traffic rates.

In the ARM approach, when the broker agent notices a traffic rate that is less than the rate agreed upon, it steps down to a lower DSCP that suits the current rate. The remaining unused bandwidth is now sent to a pool of bandwidth that is maintained for best effort services. This enables a larger number of clients that can be supported and translates to more revenue for the same amount of bandwidth. In the worst-case scenario, if all the clients generate in traffic at their peak rates, the required bandwidth is retrieved by dipping into the pool of bandwidth belonging to the best effort services. For the ARM algorithm to work, we follow set of predefined conditions. They are 1) the number of reservations in the EF class must be limited as this class is rigid takes up a large amount of the bandwidth, and 2) some of the available bandwidth is reserved for best effort so that it can be used in case of over allocation without affecting paying customers.

### 3.2. An Illustrative Example

The functioning of the ARM algorithm is explained with the help of a test network. The test network includes two DS domains, and an external client (Source1) that may not belong to a DS domain. The two DS domains demonstrate the inter-domain interaction between the BB agents to provide end-to-end resource allocation for a source-destination pair, while the external client provides means of verifying the allocation rules for out of domain traffic sources.

When Source1, which is outside the DS domain, requests service, it contacts BB1 on DS1 enroute to Destination1. If Destination1 is within this domain, as shown in Figure 3, the BB1 looks into its database, and decides upon the best available bandwidth, jitter and delay parameters, defines the SLA, and assigns a DSCP for the traffic flow be-

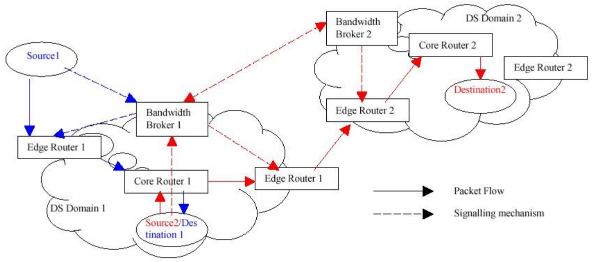

**Figure 3. Test network showing different scenarios.**

tween this source-destination pair. It then configures the edge router to mark the packets from this client with the correct DSCP so that the core routers forward the packet according to the priority assigned to the flow. BB1 also assigns a set of lower DSCPs, which define slightly lesser bandwidth requirements. During the entire packet exchange between Source1 and Destination1, BB1 periodically uses the meter to check the traffic rate from the source. If the measured rate is below the allocated bandwidth, it steps down the service to a lower DSCP to provide only the required amount of bandwidth, and returns the remaining unused bandwidth to a pool of unused bandwidth. The unused bandwidth can now be allotted to another client. Now, if the source is in DS1 domain and the destination is DS2 domain, as shown by Figure 3, the source contacts the BB1 in its domain. BB1 then looks at the database and the routing table to figure out the downstream edge router and peering BB2, and sends a resource allocation request (RAR) based on the SLA it has with the source. When BB2 confirms the reservation requested by BB1, BB1 allocates a set of DSCPs corresponding to different levels of reservation, and configures the edge router in its domain to mark the packets from the source accordingly. The peering BB2 also configures its routers to perform the same kind of marking for that source-destination pair. Both BBs define their own set of DSCPs for this flow. When there is need for conserving bandwidth in either domain, the corresponding BB decides to step down on the DSCP marking for the flow through that particular domain, thus saving bandwidth for reuse.

# 4. Implementation and Evaluation of the ARM Algorithm using NS-2

## 4.1. Implementation Overview

We have implemented the ARM algorithm using the NS-2 network simulation toolkit. The NS-2 (Network Simulator-2) toolkit has substantial functionality for simulating different network topologies and traffic models. NS also has an open architecture that allows users to add new functionalities.

Our implementation builds on the NS DiffServ patch provided by Nortel Networks, to generate DS domains and create suitable test networks [5]. This DS implementation has three modules - two of these implement the edge router and core routers, while the third module is the policy and resource manager. The policy class handles the creation, manipulation and enforcement of edge router policies. A policy defines the treatment of packets at an edge router. Policies are set using Tcl [8] scripts. The policy class uses a policy table to store the parameter values. The table is in the form of an array that maintains various fields such as SLA, current reservation, router configuration, policies, and DSCP mappings. Packets arriving at the edge router are checked to determine which traffic aggregate they belong to. A specified meter is used to check the average traffic rate of a client to make sure it corresponds to the current allocated rate. If the rate of traffic is below the requested rate, then the packets get downgraded to a lower DSCP.

The bandwidth broker is used to configure the policy module of the DiffServ. We define two modules for the broker agent. The user interface module, through which the user/network operator can allocate resources, and a DiffServ manager that does all the resource allocations.

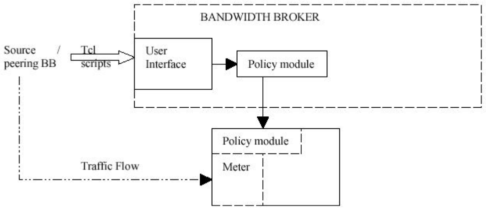

**Figure 4. Modular breakup of the BB and its interactions in a traffic flow.**

The agent performs resource provisioning, based on the SLAs as agreed upon with the client/user (through the user interface module), using Tcl scripts. Provisioning is done in accordance with other parameters in the database module such as the current reservations and the router configurations. all configuration changes are made to the policy module, and these changes are reflected in the policy module of the DS edge router. The provisioning so far is static.

The Active Resource Management (ARM) implementation keeps a track of every clients average traffic rate using a meter (such as the TSW Tagger), which is a part of the edge router. Within the policy module we associate every source-destination flow with a policy type, meter type, current rate of traffic (the rate agreed upon with the client) and

| Name | Type | Parameters |
|---|---|---|
| B1 | Link Between E1 and Core | 5Mbps, 5ms delay, with 3 drop precedence, Priority scheduling mode. |
| B2 | Link Between Core and E2 and E2 | 3Mbps, 5ms delay, 3 queues with 3 drop precedence, Priority scheduling mode. |
| S1 | Source 1 and Destination 4 | Bursty Source with Pareto distribution. Pkt size 500, burst time 500ms,idle time 300ms, rate 1Mbps. |
| S2 | Source 2 | CBR type Source on UDP. Rate 2Mbps. |
| S3 | Source 3 | Bursty Source with Exponential Pkt size 500, burst time 500ms, idle time 300ms, rate 1.5Mbps. |
| D3 | Destination 3 | FTP application on TCP. It produces packets at regular intervals. |
| D12 | Destination (1 & 2) and Source 4 | Generic node |
| BB | Bandwidth Broker agent | Configures the Edge routers. |

**Table 1. Legend for the test network.**

other policer specific parameters. We also associate a set of DSCPs with this flow. Each DSCP corresponds to a different traffic rate and the ARM algorithm switches the current DSCP marking of the packet flow according to the traffic rate indicated by the meter.

### 4.2. Experimental Evaluation

We have evaluated the ARM algorithm using three sets of experiments. The first experiment allocates the entire available bandwidth, the second experiment pushes the allocation over the limit, and finally the third experiment tests the system for an increased duration of simulation time. Each experiment consists of three evaluations - the experiment is first performed on a DS domain that does the resource provisioning in its own capabilities (DS), then on a DS environment that uses a BB provisions the resources intelligently (DS+BB), and finally on the DS environment that uses BBs implementing the ARM algorithm (DS+BB+ARM).

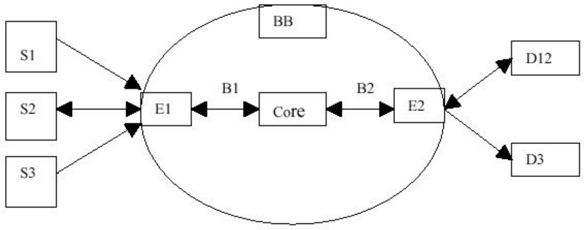

**Figure 5. Test Network**

Our test network is shown in Figure 5 and consists of 5 nodes and 3 DS enabled routers. S1, S2, S3, D12 and D3, are the nodes, E1 and E2 are the edge routers, and Core is a core router. The DS Domain includes the three routers and has one bandwidth broker agent that configures the edge routers. Table 1 lists the various parameters used in the test network and their values.

Table 2 below shows the initial request for resources made by the client and Table 3 shows the alternate policy requests in case the initial request cannot be fulfilled.

| Source | CIR | PIR | POL |
|---|---|---|---|
| S1 | 1Mbps | 2Mbps | EF |
| S2 | 1Mbps | 2Mbps | TSW3CM |
| S3 | 1.5Mbps | 3Mbps | EF |
| D12 | 1.5Mbps | 3Mbps | TSW3CM |

**Table 2. Initial policy request.**

| Source | ALTCIR | ALTPIR | ALTPOL |
|---|---|---|---|
| S1 | 750Kbps | 1.5Mbps | EF |
| S2 | 750Kbps | 1.5Mbps | TSW3CM |
| S3 | 750Kbps | 1.5Mbps | TSW3CM |
| D12 | 1Mbps | 2Mbps | TSW3CM |

**Table 3. Alternate policy request.**

The total bandwidth available for allocation is set to 5Mbps. The table entry for each method indicates the amount of bandwidth used. The notations used in the table are: Committed Information Rate (CIR), Alternate Committed Information Rate (ALTCIR), Peak Information Rate (PIR), Alternate Peak Information Rate (ALTPIR), Policer for first set of parameters (POL) and Policer for alternate set of parameters (ALTPOL).

**Experiment 1: Exact allocation of resources**

In this experiment we test the DS's capability to provide service when all the bandwidth is used up. Both the BB experiment and the ARM algorithm experiment allocate less than maximum bandwidth available and this results in a better utilization of bandwidth. Shown below (Table 4) are policy tables of the three evaluations - DS, DS+BB, and DS+BB+ARM. The bandwidth used in each case is calculated by adding the CIR. Evaluation 3, using the ARM algorithm, shows two sets of policy tables. The first table corresponds to the initial allocation made, and the second table shows the resultant run time allocations made by the ARM algorithm. The ARM algorithm shows improved bandwidth utilization and we realize a conservation of nearly 70% bandwidth.

**1.A: DS simulation results:**

| Flow | Policer Type | Initial Codepoint | CIR | PIR |
|---|---|---|---|---|
| S1 to D12 | EF | 10 | 1Mbps | 2Mbps |
| S2 to D12 | TSW3CM | 11 | 1Mbps | 2Mbps |
| S3 to D3 | EF | 12 | 1.5Mbps | 3Mbps |
| D12 to S1 | TSW3CM | 13 | 1.5Mbps | 3Mbps |

**1.B: Broker simulation results:**

| Flow | Policer Type | Initial Codepoint | CIR | PIR |
|---|---|---|---|---|
| S1 to D12 | EF | 10 | 1Mbps | 2Mbps |
| S2 to D12 | TSW3CM | 11 | 1Mbps | 2Mbps |
| S3 to D3 | EF | 12 | 750Kbps | 1.5Mbps |
| D12 to S1 | TSW3CM | 13 | 1Mbps | 2Mbps |

**1.C: ARM Algorithm simulation results:**
**Before dynamic allocation**

| Flow | Policer Type | Initial Codepoint | CIR | PIR |
|---|---|---|---|---|
| S1 to D12 | EF | 10 | 1Mbps | 2Mbps |
| S2 to D12 | TSW3CM | 11 | 1Mbps | 2Mbps |
| S3 to D3 | EF | 12 | 750Kbps | 1.5Mbps |
| D12 to S1 | TSW3CM | 13 | 1Mbps | 2Mbps |

**After dynamic allocation.**

| Flow | Policer Type | Initial Codepoint | CIR | PIR |
|---|---|---|---|---|
| S1 to D12 | EF | 10 | 289Kbps | 2Mbps |
| S2 to D12 | TSW3CM | 11 | 981Kbps | 2Mbps |
| S3 to D3 | EF | 12 | 750Kbps | 1.5Mbps |
| D12 to S1 | TSW3CM | 13 | 1Mbps | 2Mbps |

**Table 4. Policy request for experiment 1.**

**1.A: DS simulation results:**

| CP | TotPkts | TxPkts | ldrops | edrops |
|---|---|---|---|---|
| All | 24286 | 24173 | 112 | 1 |
| 10 | 6151 | 6151 | 0 | 0 |
| 11 | 8169 | 8056 | 112 | 1 |
| 12 | 9966 | 9966 | 0 | 0 |

**1.B: Broker simulation results:**

| CP | TotPkts | TxPkts | ldrops | edrops |
|---|---|---|---|---|
| All | 22527 | 22381 | 26 | 120 |
| 10 | 5406 | 5406 | 0 | 0 |
| 11 | 1698 | 1698 | 0 | 0 |
| 12 | 173 | 173 | 0 | 0 |
| 18 | 6768 | 6709 | 10 | 49 |
| 20 | 8482 | 8395 | 16 | 71 |

**1.C: ARM Algorithm simulation results:**

| CP | TotPkts | TxPkts | ldrops | edrops |
|---|---|---|---|---|
| All | 22527 | 22381 | 26 | 120 |
| 10 | 3157 | 3157 | 0 | 0 |
| 11 | 812 | 812 | 0 | 0 |
| 12 | 173 | 173 | 0 | 0 |
| 16 | 2249 | 2249 | 0 | 0 |
| 18 | 7654 | 7595 | 10 | 49 |
| 20 | 8482 | 8395 | 16 | 71 |

**Table 5. Packet statistics for experiment 1.**

The Legend for Table 5: cp - Codepoint; TotPkts-Total packets; TxPkts-Transmitted packets; ldrops-Late drops; and edrops-Early drops. The packet statistics for the same three experiments are shown in (Table 5). These statistics were calculated after 40.0 time steps of the simulation. From these statistics it can be seen that service guarantees are met for all the cases, as there is no difference in the output results. Thus, we managed to conserve bandwidth by using ARM algorithm while providing the same level of service.

**Experiment 2: Over allocation of resources.**

In the second experiment, we stress the allocation limits by over-allocating the resources. This is achieved by increasing Source 2 requirements to 2Mbps. The BB manages to keep the allocation under control in evaluation 2, and the ARM algorithm improves upon the BB's allocation in evaluation 3. Once again, the total CIR for each experiment shows that we conserve bandwidth utilization by about 50% when using the ARM algorithm. The policy table for this experiment are shown in Table 6.

**2.A: DS simulation results:**

| Flow | Policer Type | Initial Codepoint | CIR | PIR |
|---|---|---|---|---|
| S1 to D12 | EF | 10 | 1Mbps | 2Mbps |
| S2 to D12 | TSW3CM | 11 | 2Mbps | 4Mbps |
| S3 to D3 | EF | 12 | 1.5Mbps | 3Mbps |
| D12 to S1 | TSW3CM | 13 | 1.5Mbps | 3Mbps |

**2.B: Broker simulation results:**

| Flow | Policer Type | Initial Codepoint | CIR | PIR |
|---|---|---|---|---|
| S1 to D12 | EF | 10 | 1Mbps | 2Mbps |
| S2 to D12 | TSW3CM | 11 | 2Mbps | 4Mbps |
| S3 to D3 | EF | 12 | 750Kbps | 1.5Mbps |
| D12 to S1 | TSW3CM | 13 | 1Mbps | 2Mbps |

**2.C: ARM Algorithm simulation results:**
**Before dynamic allocation.**

| Flow | Policer Type | Initial Codepoint | CIR | PIR |
|---|---|---|---|---|
| S1 to D12 | EF | 10 | 1Mbps | 2Mbps |
| S2 to D12 | TSW3CM | 11 | 2Mbps | 4Mbps |
| S3 to D3 | EF | 12 | 750Kbps | 1.5Mbps |
| D12 to S1 | TSW3CM | 13 | 1Mbps | 2Mbps |

**After dynamic allocation.**

| Flow | Policer Type | Initial Codepoint | CIR | PIR |
|---|---|---|---|---|
| S1 to D12 | EF | 10 | 578Kbps | 2Mbps |
| S2 to D12 | TSW3CM | 11 | 1.95Mbps | 2Mbps |
| S3 to D3 | EF | 12 | 750Kbps | 1.5Mbps |
| D12 to S1 | TSW3CM | 13 | 1Mbps | 2Mbps |

**Table 6. Policy request for experiment 2.**

The packet statistics for the 3 evaluations in experiment 2 are listed in Table 7. These statistics were also calculated after 40.0 time steps of the simulation. From these statistics it can be seen that once again service guarantees are met for all the cases, and that using ARM algorithm conserves bandwidth.

**2.A: DS simulation results:**

| CP | TotPkts | TxPkts | ldrops | edrops |
|---|---|---|---|---|
| All | 24459 | 21935 | 1798 | 726 |
| 10 | 6017 | 6017 | 0 | 0 |
| 11 | 5 | 5 | 0 | 0 |
| 12 | 8568 | 8568 | 0 | 0 |
| 18 | 9869 | 7345 | 1798 | 726 |

**2.B: Broker simulation results:**

| CP | TotPkts | TxPkts | ldrops | edrops |
|---|---|---|---|---|
| All | 18924 | 18397 | 316 | 211 |
| 10 | 4299 | 4299 | 0 | 0 |
| 11 | 3337 | 3334 | 3 | 0 |
| 12 | 224 | 224 | 0 | 0 |
| 18 | 8895 | 8607 | 139 | 149 |
| 20 | 2169 | 1933 | 174 | 62 |

**2.C: ARM Algorithm simulation results:**

| CP | TotPkts | TxPkts | ldrops | edrops |
|---|---|---|---|---|
| All | 18928 | 18402 | 305 | 221 |
| 10 | 3539 | 3539 | 0 | 0 |
| 11 | 2264 | 2264 | 0 | 0 |
| 12 | 224 | 224 | 0 | 0 |
| 16 | 761 | 761 | 0 | 0 |
| 18 | 9964 | 9667 | 139 | 158 |
| 20 | 2176 | 1947 | 166 | 63 |

**Table 7. Packet statistics for experiment 2.**

As seen above, the packet loss due to DS is more than that due to ARM algorithm. This is because the dynamic allocation done by the ARM algorithm, evenly distributes the packets to different queues depending upon the rate of traffic at a given point of time. In the case of DS, the packet distribution to the various queues are decided by the reservations made at the beginning, thus causing the allocations to be static.

**Experiment 3: Increased duration of traffic.**

In the third experiment, we extended the second experiment to stress the system by doubling the period of simulation to 80.0. It can be seen that, as the traffic increases, the results for the BB the ARM algorithm evaluation match the DS result in packet statistics. There is however an improvement in allocation. The policy tables for this experiment are listed in Table 8 and the packet statistics are listed in Table 9. The packet statistics are now calculated for 80.0 time steps of the simulation. As the traffic increases after a period of time, as expected, the queue starts dropping packets evenly irrespective of the method used, and the results even out.

**3.A: DS simulation results:**

| Flow | Policer Type | Initial Codepoint | CIR | PIR |
|---|---|---|---|---|
| S1 to D12 | EF | 10 | 1Mbps | 2Mbps |
| S2 to D12 | TSW3CM | 11 | 2Mbps | 4Mbps |
| S3 to D3 | EF | 12 | 1.5Mbps | 3Mbps |
| D12 to S1 | TSW3CM | 13 | 1.5Mbps | 3Mbps |

**3.B: Broker simulation results:**

| Flow | Policer Type | Initial Codepoint | CIR | PIR |
|---|---|---|---|---|
| S1 to D12 | EF | 10 | 1Mbps | 2Mbps |
| S2 to D12 | TSW3CM | 11 | 2Mbps | 4Mbps |
| S3 to D3 | EF | 12 | 750Kbps | 1.5Mbps |
| D12 to S1 | TSW3CM | 13 | 1Mbps | 2Mbps |

**3.C: ARM Algorithm simulation results:**
**Before dynamic allocation.**

| Flow | Policer Type | Initial Codepoint | CIR | PIR |
|---|---|---|---|---|
| S1 to D12 | EF | 10 | 1Mbps | 2Mbps |
| S2 to D12 | TSW3CM | 11 | 2Mbps | 4Mbps |
| S3 to D3 | EF | 12 | 750Kbps | 1.5Mbps |
| D12 to S1 | TSW3CM | 13 | 1Mbps | 2Mbps |

**After dynamic allocation.**

| Flow | Policer Type | Initial Codepoint | CIR | PIR |
|---|---|---|---|---|
| S1 to D12 | EF | 10 | 578Kbps | 2Mbps |
| S2 to D12 | TSW3CM | 11 | 1.95Mbps | 2Mbps |
| S3 to D3 | EF | 12 | 750Kbps | 1.5Mbps |
| D12 to S1 | TSW3CM | 13 | 1Mbps | 2Mbps |

**Table 8. Policy request for experiment 3.**

From the above experiments it can be seen that even though outputs are similar in terms of amount of packets dropped, the amount of bandwidth utilized is lesser for the ARM algorithm as compared to the other two methods. Furthermore, it can be noted that SLA agreed with the client is maintained, thus providing the guaranteed level of service.

**3.A: DS simulation results:**

| CP | TotPkts | TxPkts | ldrops | edrops |
|---|---|---|---|---|
| All | 45624 | 41411 | 3149 | 1064 |
| 10 | 12254 | 12254 | 0 | 0 |
| 11 | 5 | 5 | 0 | 0 |
| 12 | 15145 | 15145 | 0 | 0 |
| 18 | 18220 | 14007 | 3149 | 1064 |

**3.B: Broker simulation results:**

| CP | TotPkts | TxPkts | ldrops | edrops |
|---|---|---|---|---|
| All | 42122 | 37611 | 3367 | 1144 |
| 10 | 10998 | 10998 | 0 | 0 |
| 11 | 7005 | 6941 | 64 | 0 |
| 12 | 416 | 401 | 15 | 0 |
| 18 | 13605 | 12393 | 705 | 507 |
| 20 | 10098 | 6878 | 2583 | 637 |

**3.C: ARM Algorithm simulation results:**

| CP | TotPkts | TxPkts | ldrops | edrops |
|---|---|---|---|---|
| All | 43790 | 38652 | 3217 | 1921 |
| 10 | 5644 | 5644 | 0 | 0 |
| 11 | 2787 | 2765 | 22 | 0 |
| 12 | 299 | 298 | 1 | 0 |
| 16 | 4702 | 4702 | 0 | 0 |
| 18 | 17812 | 15871 | 1083 | 858 |
| 20 | 12546 | 9372 | 2111 | 1063 |

**Table 9. Packet statistics for experiment 3**

Loss of packets is observed only in the case of the clients that are using constant bit rate traffic (such as FTP and CBR), while clients sending variable traffic (such as multimedia traffic) do not lose packets and their bandwidth usage is only what is required. Thus, using the active reallocation algorithm, the number of clients can be increased without loss of service.

## 5. Conclusion

Real time media data and mission critical traffic require guaranteed services, which cannot be provided by standard IP methods. The Differentiated Services framework provides a scalable and relatively simple means for providing these guarantees. Furthermore, with the help of the bandwidth broker agent, intelligent resource provisioning can be achieved. However, these techniques provide only static provisioning and can either lead to wasted bandwidth or leave applications resource-starved. In this paper we presented ARM - an Active Resource Management algorithm that attempts to optimize bandwidth usage, by dynamically reallocating a client's unused, but allocated bandwidth to other clients, based on the current state of the network and the application. We have implemented and evaluated ARM using NS-2 simulation toolkit. Experimental results presented in this paper show that ARM conserves between 50% to 75% bandwidth while providing clients with the required QoS.

## References

[1] C. Metz. IP QoS: traveling in the first class on the Internet. *IEEE Internet Computing*, Vol. 3, No. 2, March/April 1999.

[2] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An Architecture for Differentiated Services.*RFC 2475*. December 1998.

[3] K. Fall, and K. Varadhan, editors. ns Notes and Documentation. *The VINT Project, UC Berkeley, LBL, USC/ISI, and Xerox PARC*, January 1999. Available from http://www.mash.cs.berkeley.edu/ns/.

[4] The Network Simulator, http://www.isi.edu/nsnam/ns/

[5] P. Pieda, J. Ethridge, M. Baines and F. Shallwani. A Network Simulator Differentiated Services Implementation, *Open IP, Nortel Networks*.

[6] K. Nichols, V. Jacobson and L. Zhang. A Two Bit Differentiated Services Architecture For The Internet, *RFC 2638*, July 1999.

[7] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski. Assured Forwarding PHB Group. *RFC 2597*. June 1999.

[8] B. Welsh, *Practical programming in Tcl and Tk*, Third Edition, Prentice Hall, 1995.

[9] D. Clark, and W. Fang. Explicit Allocation of Best Effort Packet Delivery Service. *IEEE/ACM Transactions on Networking*, Aug. 1998.

[10] W. Fang, N. Seddigh, and B. Nandy. A Time Sliding Window Three Colour Marker, *Internet draft*, March 2000.

[11] J. Heinanen, T. Finland, and R. Guerin. A Single Rate Three Colour Markerl, *Internet draft*, May 1999.

[12] J. Heinanen, T. Finland, and R. Guerin. A Two Rate Three Colour Marker, *Internet draft*, May 1999.

[13] Qbone Forum, http://qbone.internet2.edu/bb/.