Proceedings of the 13th IEEE International Conference on Computer Communications and Network, ICCN04, October 11-13, 2004, Chicago, pp93-100

INTELLIGENT EXTENSIBLE ROUTING FOR OVERLAY NETWORKS WITH EMBEDDED CONSTRAINT RESOURCE PLANNING SHELL: A CASE STUDY WITH DEADLINE BASED PACKET FORWARDING

Javed I. Khan & Nouman Bantan
Internetworking and Media Communications Research Laboratories
Department of Computer Science, Kent State University
233 MSB, Kent, OH 44242 U.S.A.
javed@kent.edu & nbantan@cs.kent.edu

Abstract

This paper presents a fast near linear run-time system for customizable criteria based packet forwarding and route lookup. It presents the case for deadline line based packet forwarding. This system has been designed after the Constraint Resource Planning (CRP) methodology for algorithm designs. The resulting near-optimum algorithm operates within an extensible CRP shell and uses two pluggable scheduling and forwarding heuristics to produce near-optimal performance. In this paper we present the shell architecture, the algorithm, and the simulation results of this system for dead-line based packet forwarding.

Key Words

Shell interface, intelligent routing, temporal quality of service, forwarding, admission control, leverage, deadline

1 Introduction

The need for time-constraint application (e.g. multimedia application) has created a wide area of research. An efficient time routing algorithm will make major contribution to this field of study. Since the optimum deadline base packet scheduling has been shown to be NP-hard [1], any routing algorithm must be fast in order to serve the purpose of the original issue that is delivering the data on time. In this research, we define a routing interface based on the heuristics management research in AI and planning. Very few work exists which uses AI in communication research area. Upon this infrastructure, we present various algorithms which have been designed using the Constraint Resource Planning (CRP) methodology – a constraint satisfaction heuristics system [5,10]. Since this methodology proposes an optimizing shell rather than a final solution, future research might even produce better results than the outcome of this paper.

The proposed architecture serves as an optimizing shell, which can form the basis for efficient custom, overlay routing optimization criteria at run-time. With the assumption that total knowledge of the network traffic is available in distributed fashion, the proposed shell can potentially be used for custom route optimization for overlay networks.

1.1 Related Work

Routing is the key service embedded into the current Internet layer. It has two major component- routing information propagation and forwarding. Most research in routing has been conducted to find stable and scalable route propagation algorithms. The transition to OSPF from RIP and the advent of BGP has dramatically improved the scalability of route information management. However, the research on forwarding is relatively little and primarily focused on the design of efficient lookup table data structure [13,17,19]. It is highly unlikely any temporal QoS can be provided without intelligent forwarding.

There are now two principle approaches to provide temporal quality of service- the Integrated Service (int-service) approach and the Differentiated Services (diff-service) approach [8]. The int-service has been based on per flow based resource reservation and classification of services, admission control and policing [2,19]. In the int-service paradigm, a number of interesting results on queuing behavior under controlled traffic admission have been reported in [3,6,9]. One simple approach was introduced in [14] where multiple copies of the time-constraint packet are admitted into the network. This approach increases the probability for on-time delivery, but it also increases the overall flow of traffic. The use of fixed input queue has triggered a lot of research as well [1,2]. Moreover the fixed input queue is being implemented in some current router architecture, e.g. Cisco routers.

The other and more recent approach is the diff-service [4]. Instead of setting the service on per flow basis, packet flows are simplified into flow classes. Packets arrive with class labels and are accordingly diverted in the appropriate queue (we will use the term queue instead of route). The packets from time-critical application will be in a high priority queue that will then be processed first [15]. This heuristic provides high level of QoS. Both services architectures require substantial modification of the IP router architecture, where multiple queues have to be maintained.

Prior work that provides a high level of QoS with the support the CRP methodology had produced excellent results when used in communication research. We presented a CRP methodology in a single hop overlay network in [11] where the outcome yielded exceptionally good result over Greedy routing. This paper differs a great deal from [11] since other elements like multi-hops, cost function, different network models, and many more are implemented here.

1.2 Intelligent Routing

The existing research will show that most of them have been proposed with an implicit assumption that no intelligent action at the junction points other than IP standard processing is permissible. The flow-based into a service model essentially performs resource reservation and entry policing at the network endpoints.

The diff-service has been a notable advancement where it has looked into standard based extension. Intelligent routing inside the overlay network was indeed a serious limitation- until now. However, recent advances in programmable networking such as OPENSIG and Active Network have shown some satisfying results [12,16], where network layer can be customized. In the light of such advancements it seems that direct and more efficient solutions of many of the currently hard to solve problem are poised to be developed, including network mechanisms where deadline assured delivery is guaranteed.

We have focused on two central mechanisms: (1) a distributed network temporal tracking system and (2) a deadline-based forwarding algorithm. These two techniques are analogous to the routing and forwarding in the current IP layer, where the routing components are responsible for efficient collection and propagation of accurate network connectivity information and forwarding components focuses on fast forwarding of packet based on the collected information. In this research we demonstrate that we might be able to infuse deep algorithm in routers for intelligent networking with the emphasis of deadline conformant packet forwarding. This paper presents an efficient deadline based forwarding queue-scheduling algorithm.

Now that we have focused on the general precepts of our approach, we will now move to section two, where we present the overview of the CRP shell base architecture for router extension that enables the researcher to plug-in an algorithm of interest. In section three, we present the CRP routing model which illustrates the forwarding and route lookup prototype. In section four, we present a CRP based routing solution, which provides a fast polynomial algorithm for deadline based routing. Finally, The details of our experiments implementations and comparison results and analysis for various heuristics are presented in section five.

2 CRP Model

The Constraint Resource Planning (CRP) is a methodology that enables the design of near optimum approximation algorithms- with linear execution shell loop. In applied AI and planning research, a large body of knowledge exists for fast approximation algorithm of NP-hard problems [8,10]. We have selected the CRP methodology because of its ability to become a generic execution shell (interface) that serves the purpose of this paper. The CRP divides the solution of a complex NP-hard or NP-complete problem into two decision heuristics: i) most-critical task heuristic $H_{MCT}()$ and ii) the least-impact solution heuristic $H_{LIS}()$[5,10]. CRP provides a polynomial time bound execution shell called the CRP engine that uses the heuristic to generate powerful near optimum solutions based on the proposed solution within the CRP shell.

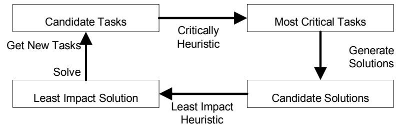

Fig – 1: The generic CRP engine

CRP provides a domain-independence systematic framework to solve constraint-satisfaction problem in planning and scheduling. CRP provides very competitive near optimum solutions for more than fifty problems including traveling salesman, job shop scheduling, and map coloring to enhance triangulation to optimum packing [18]. It divides any constraint satisfaction-planning problem into a four-corner execution model as shown in figure 1. We have designed an intelligent routing shell that is expected to host or facilitate various intelligent routing solutions. We shall present the solution specific for delay-bound communication.

The advantage of CRP is that within this generic four-corner shell the optimization criterion can be radically changed just by changing these two heuristics. This feature enables one to use it as a basis for extensible routing. Interestingly at the same time, it helps one to experiment with wild number of probable heuristics pairs in short time. We have formulated a number of candidate heuristics for each of those heuristics and have experimented with over fifty combination probable heuristics.

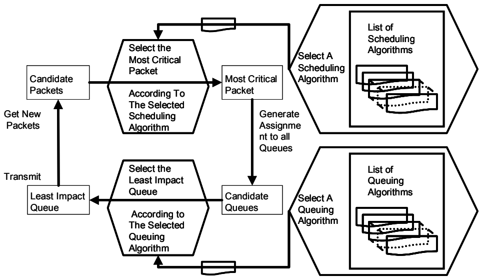

Fig – 2: data flow diagram of the CRP-Interface for the deadline-base routing. Each heuristic decision box facilitate a pluggable action.

2.1 CRP Interface Blocks

The interface consists of four data blocks, two heuristic decision points, and three mechanical procedures. Those procedures remain unchanged and are not affected by various algorithms which are plugged in the heuristic decision points. The setup of our research CRP interface is shown in figure 2. At the entry stage of the interface, packets which are ready for transmission are inserted into the Candidate Packets data block which is the first mechanical procedure. The first heuristic decision point decides which packet is the most critical based on the plugged scheduling algorithm. Now the most critical packet is inserted into the Candidate Packet data block. The second mechanical procedure is the execution of a cost function which will estimate the arrival time of a packet based on network propagation information on a specific queue. The arrival times of all the available queues for the most critical packet are the candidate assignment solutions. Then these solutions are inserted them in the Candidate Queue data block. The second heuristic decision point selects the queue that has provided the "best results". The definition of the "best results" is decided upon by the plugged queuing algorithm. Once a queue is found to have produced the best results, it is inserted into the Least Impact data block. At this point, the third mechanical procedure proceeds to transmit the most critical packets via the least impact queue. The next cycle of the CRP shell starts after this transmission.

3 Deadline Base Routing

The CRP shell act as a host to various scheduling and queuing heuristics. We have defined many scheduling algorithms which may be plugged in the admission control heuristic and, also, have defined many queuing algorithms which can be plugged in the forwarding heuristics.

3.1 Scheduling

The object of the scheduling policy is to select a packet from the input queue using the most critical-task heuristic $H_{MCT}()$ of CRP. The algorithm is executed in a cyclical execution mode while the router is up and running. At most *n* packets are inserted in the input queue. Then the packets in the input queue are evaluated according to the scheduling heuristic. The router will enter into a phase where all the packets in the input queue are forwarded before allowing any other packets into the input queue. Once a packet is selected, it is removed from the input queue for transmission. The transmission is the responsibility of the queuing model, where a queue is selected based on the queuing model heuristic. The scheduling model is in table 1.

- While the router is running
  - ☐ If there is no outgoing packet in the router
    - Wait until at least one packet arrive in the router
  - ☐ For a fixed time interval or *n* or more packets are in the router
    - Insert at most *n* packets in the input queue (packet insertion is in accordance to their arrival time)
  - ☐ (Within the input queue)
    - Retrieve all the packets' deadlines
  - ☐ While the input queue is not empty
    - Select the most critical packet according to the scheduling heuristic
    - Remove the selected packet from the input queue
  - ☐ Run the queuing algorithm to transmit the selected packet

Table–1: The CRP scheduling model

- Packet's end of transmission time – is the time used to deliver the packet.
- Packet's real penalty – is the positive difference between the end of transmission time and the deadline if the end of transmission time is larger than the deadline.
- Gross penalty – is the accumulated real penalties after all the packets are transmitted.
- Queue's expected end of transmission time – is the time that the queuing model produces which is the assumed time of delivery
- Queue's hop number – is the iterative order of hops in the queue.
- Queue's hop count – is the number of hops in the queue.
- Chosen queue – is the queue that is assigned to deliver the packet.
- Hop flight time – is the time used to transmit a packet fully over this hop.
- Hop activity – is a special value which is retrieved from the network. Each time a packet is transmitted through this hop, the hop flight time is recorded. The hop activity is the accumulation of all the flight times through this hop.
- Hop number cost time – is produced by the first part of cost function and is a parameter of final part of the cost function that produces the expected end of transmission time.
- Hop activity cost time – is produced by the first part of cost function and is a parameter of final part of the cost function that produces the expected end of transmission time.

Table–2 The terms of the queuing model

3.2 Queuing

The objective of the queuing policy is to assign the selected packet to a queue using the least impact solution heuristic $H_{LIS}()$. The queuing policy uses a network cost function, explained later, to determine which queue produces the least impact solution.

3.2.1 Model Terms

Prior defining the queuing model, we must explain some of the terms used inside the model, see table 2.

3.2.2 Queuing Model

The queuing model is implemented for one packet and many queues which will compete to deliver this packet, see table 3.

- If there is at least one queue to transmit the selected packet to its destination router
    - ❑ For each of all the available queues to the packet's destination node
        - Retrieve the hop count and the available bandwidth of each hop
        - Compute the expected end of transmission time
    - ❑ Select the least impact queue according to the selected queuing heuristic
    - ❑ Transmit the packet
        - Send the packet via the chosen queue
        - Retrieve the end of transmission time
    - ❑ Increase the gross penalty by the chosen packet's real penalty (if any)
- Else
    - Schedule this packet for later delivery (not reachable in our research)

Table–3: The CRP queuing model

3.2.3 Cost Function

The cost function will produce a single value that represents the expected end of transmission time for a single packet. The cost function's main variables are the queue's hop count and the queue's hop activity. The cost value grows exponentially as the hop count increases whereas the bandwidth has a moderate influence on the cost value.

- Initialize the expected transmission time to zero
- For each hop number
    - ❑ Hop number cost time is the area of the circle with radius 2 raised the power of the queue's hop number.
    - ❑ Hop activity cost time is the hop activity multiplied by the area with radius that is equal is the hop number.
    - ❑ Add both cost times to the expected end of transmission time.

Table–4: The cost function

3.3 Solution Framework

Below we present the total solution framework which explains the scheduling model, queuing model, and the cost function in details.

3.3.1 Scheduling Framework

- w be the number of packet/s in the router but not in the input queue at any time
- Set of n packets to transmit
- Each packet, p_i where $1 \le i \le n$,
    - ❑ has size, s_i
    - ❑ requires transmit time, t_i
    - ❑ must be transmitted before a preset *deadline*, d_i

3.3.2 Queuing Framework

- Set of m queues, at most, for each packet
- Each queue, q_j where $1 \le j \le m$,
    - ❑ has an $e_{i,j}$ be the *expected end of transmission time* for packet p_i via queue q_j
- c_i be the *chosen queue* number j, as in queue q_j, which will transmit packet p_i i.e.

$$c_i = j \ni p_i \in q_j \tag{1}$$

- $T_{i,j}$ be the *packet's end of transmission time* for packet p_i via queue q_j
- $l_i^{'}$ and l_i be the *gross penalty* of all queues before and after packet p_i's arrival respectively i.e.

$$l_i^{'} = \sum_{b=1}^{i-1} (T_{b,j} - d_b) \text{ if } (T_{b,j} - d_b) > 0 \ \forall j \tag{2}$$

$$l_i = \sum_{b=1}^{i} (T_{b,j} - d_b) \text{ if } (T_{b,j} - d_b) > 0 \ \forall j \tag{3}$$

3.3.3 Cost Function Framework

- h_j be the *queue's hop count* for queue q_j
- $rn_{j.k}$ be the radius used for the *hop number cost time* at k^{th} hop in queue q_j. i.e.,

$$rn_{j.k} = 2^k \text{ where } 1 \le k \le h_j \tag{4}$$

- $ra_{j.k}$ be the radius used for the *hop activity cost time* at k^{th} hop in queue q_j. i.e.,

$$ra_{j.k} = k \text{ where } 1 \le k \le h_j \tag{5}$$

- $A_{j.k}$ be the *hop activity* of the k^{th} hop in queue q_i
- $CN_{i.j.k}$ is the *hop number cost time* for packet p_i via queue q_j at k^{th} hop. i.e.,

$$CN_{i.j.k} = \pi * rn^2_{j.k} \text{ where } 1 \le k \le h_j \tag{6}$$

- $CA_{i.j.k}$ is the *hop activity cost time* for packet p_i via queue q_j at k^{th} hop.

$$CA_{i.j.k} = A_{j.k} * \pi * ra^2_{j.k} \text{ where } 1 \le k \le h_j \tag{7}$$

$$e_{i,j} = \sum_{k=1}^{h_j} (CA_{i.j.k} + CN_{i.j.k}) \tag{8}$$

4 DML Algorithms

4.1 DML Overview

Our proposed algorithm is called Deadline Minimum Leverage (DML) algorithm. This iterative algorithm consists of two principal phases – Scheduling and Queuing. These two phases represents the task generation and solution set generation phases of CRP which is presented in figure 2.

```
H_MCT(){
  While router is running
    If w > 0 {
    - While the input queue size < n and w > 0{
        Insert earliest packet residing outside the input queue in the input queue
        Decrement w by 1
    }
    - Retrieve the deadline value for all the packets residing in the input queue
    - While the input queue is not empty {
        Select the Most Critical Packet according to the selected scheduling heuristic
        Remove first packet from the input queue
    }
  }
}
```

Table–5: The CRP scheduling algorithm pseudo code.

4.1.1 DML SCHEDULING

Once the packets are inserted in the input queue, the DML scheduling algorithm will sort all packets within the input queue in a descending order according to their deadline. The selection process is to pick the packet sitting on top of the input queue which is the packet with highest deadline. Now, this packet is the most critical packet.

```
H_LIS(){
  If $m > 0$ {
    For $j \leftarrow 1$ to $m$ {
      $e_{i,j} = 0$
      For $k \leftarrow 1$ to $h_j$ {
        $rn_{j,k} = 2^k$
        $ra_{j,k} = k$
        $CN_{i,j,k} = \pi * rn^2_{j,k}$
        $CA_{i,j,k} = A_{j,k} * \pi * ra2_{j,k}$
        $e_{i,j} = CN_{i,j,k} + CA_{i,j,k} + e_{i,j}$
      }
    }
    Select the least impact queue according to the selected queuing heuristic
    If $T_{i,c_i} > d_i$
      $l_i \leftarrow l'_i + (T_{i,c_i} - d_i)$
  }
}
```

Table–6: The CRP queuing algorithm pseudo code

4.1.2 DML Queuing

The DML queuing algorithm requires a few extra parameters, see table 7.

- Queue's leverage time – is the positive difference between the expected end of transmission time and the deadline if the deadline is larger than the expected end of transmission time.
- Queue's fake penalty – is the positive difference between the expected end of transmission time and the deadline if the deadline is smaller than the expected end of transmission time.

Table–7: The extra parameters for DML queuing algorithm

The mathematical representations of the DML extra parameters are presented below.

- $L_{i,j}$ be the *queue's leverage time* for packet p_i via queue q_j

$$L_{i,j} = d_i - e_{i,j} \text{ where } e_{i,j} \le d_i \tag{9}$$

- L'_i be the minimum *queue's leverage time* for packet p_i

$$L'_i = Min(L_{i,j}) \quad \forall j \tag{10}$$

- $Y_{i,j}$ be the *queue's fake penalty* for packet p_i via queue q_j

$$Y_{i,j} = e_{i,j} - d_i \text{ where } d_i \le e_{i,j} \tag{11}$$

- Y'_i be the minimum *queue's fake penalty* for packet p_i

$$Y'_i = Min(Y_{i,j}) \quad \forall j \tag{12}$$

The DML queuing model is shown in table 8.

- Set all queues' leverage time to ∞
- Set all queues' fake penalties to 0
- For each of all the available queues to the packet's destination
  - If the expected end of transmission time is less than the packet's deadline
    - Compute the leverage time
  - Else
    - Compute the fake penalty time
  - If the minimum leverage time of all queues is ∞ (not a single queue was thought to deliver the packet by its deadline)
    - The chosen queue is the queue with the minimum fake penalty
  - Else (at least one queue is thought to deliver the packet by its deadline)
    - The chosen queue is the queue with the minimum leverage time

Table–8: The DML queuing algorithm model

The example shown in figure 3 illustrates the execution of the DML queuing algorithm with three queues. The example displays the queuing of packet p_i when it enters a router where q_1 has the minimum leverage value out of all the available queues. So it is the chosen queue to deliver packet p_i.

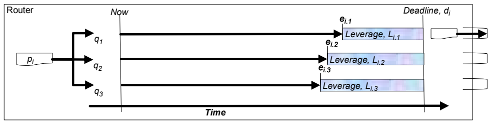

Fig 3: The most critical packet p_i is tested on all available queues. According to the DML queuing algorithm, q_1 has the least leverage time thus q_1 is the chosen queue.

5 Experiment

We have conducted thousands of simulations and found the performance of several candidate scheduling and queuing algorithms fit for deadline based scheduling/queuing problem. To evaluate the algorithms, we assigned a penalty for each packet. If a packet is delivered within the deadline, the penalty is zero. Otherwise, the difference between the deadline and the end of transmission time of a queued packet is added to the algorithm's penalty. We do not assign any credit for early delivery. Thus the penalty can only be zero or more. We demonstrated that $O((\lg n)+m)$ near optimum algorithms is possible, which offered dramatic improvement over other conventional methods in deadline assured communication, where n is the expected length of the input queue and m is the number of output queues. The cost for sorting is only found once for processing n packet which, in Amortized analysis, averages out to $O(\lg n)$.

5.1 Experiment Environment

We have defined a set of criterions on the simulation to resemble real working environments. Defining the environment of any research is a major task since the success of any research depends greatly on how similar it is to a real environment. The criterions are topology, deadline scheme, input queue size, and the method of creating routing tables.

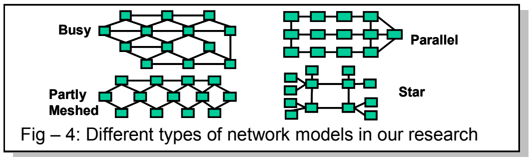

Fig – 4: Different types of network models in our research

5.1.1 Network Models

The network design plays a critical role in real life environment. Each business establishment must consider the layout structure of its network to produce the desired goal. Thus we have developed four types of network models. Each one of these models expands with respect to its criteria defined in table 9.

- Parallel – Routers are connected serially on each level. Backbone connections exist at both ends of each level.
- Star – A number of star networks are connected together
- Partly meshed – Routers are connected like a mesh. The mesh grows only toward the center of the grid. A backbone connection resides at one end.
- Busy – Each router has at least three links and at most six links.

Table–9: The four network models used in our research

5.1.2 Input Queue Size

Each input queue has a size that indicates the maximum number of packets that may be placed in the queue. This input queue size is a variable that can be set to any desirable value. We have chosen three arbitrary sizes, 11, 22, and 32.

Level \ ST	1	2	3
N1	$0.146\geq t_i/d_i>0.04$	$0.081\geq t_i/d_i>0.05$	$0.051\geq t_i/d_i>0.047$
N2	$0.253\geq t_i/d_i>0.146$	$0.112\geq t_i/d_i>0.081$	$0.055\geq t_i/d_i>0.051$
N3	$0.36\geq t_i/d_i>0.253$	$0.143\geq t_i/d_i>0.112$	$0.059\geq t_i/d_i>0.055$
N4	$0.466\geq t_i/d_i>0.36$	$0.174\geq t_i/d_i>0.143$	$0.063\geq t_i/d_i>0.059$
N5	$0.573\geq t_i/d_i>0.466$	$0.205\geq t_i/d_i>0.174$	$0.067\geq t_i/d_i>0.063$
N6	$0.68\geq t_i/d_i>0.573$	$0.236\geq t_i/d_i>0.205$	$0.071\geq t_i/d_i>0.067$
N7	$0.786\geq t_i/d_i>0.68$	0.267≥ti/di>0.236	$0.075\geq t_i/d_i>0.071$
N8	0.893≥ti/di>0.786	0.298≥ti/di>0.267	0.079≥ti/di>0.075
N9	1≥ti/di>0.893	0.33≥ti/di>0.298	0.083≥ti/di>0.079
W1	0.36≥ti/di>0.04	0.143≥ti/di>0.05	0.059≥ti/di>0.047
W2	0.68≥ti/di>0.36	0.236≥ti/di>0.143	0.071≥ti/di>0.059
W3	1≥ti/di>0.68	0.33≥ti/di>0.236	0.083≥ti/di>0.071

Table 10– The narrow and wide ranges for the three solution tightness (ST)

5.1.3 Deadline Ranges

To create various levels of challenges we assigned the deadlines using the criterion of solution tightness. This is the ratio of the assigned deadline and the transmit time of a packet. To challenge the system we have created the datasets with sixteen different solution tightness complexities. These complexities vary from loose to tight tightness. We will present only three complexities since these are the ones that have shown significant differences in the results. Each one of the four complexities is configured in ranges. Each range represents a ratio of the assigned deadline and the transmit time of a packet. Within a set, the ranges shift from a loose end to a tight end, table 3.

We present two types of ranges, a narrow range and a wide range. The wide ranges confine three narrow ranges. For each solution tightness, there are nine deadline narrow ranges and three wide ranges. In each of the range's sets we put 50 streams; each stream has 11, 22, or 32 packets (depending on the input queue size). A data generator program produce theses data sets.

5.1.4 Routing Table

We used three different methods in creating the routing tables. For each *x* and *y* routers where *x* is the source router and *y* is the destination router, we have defined an *xy* master table of all the queues between those two routers. The *xy* master table shown in table 6 contains one row for each hop count and all queues are stored in their appropriate row accordingly where the order of the queues in each row is not required.

We have defined three routing table schemes. Scheme one will randomly select a maximum of two queues for each hop count. Each subsequent scheme increases their maximum queue selection by one.

Hop count	Queue 1	Queue 2	Queue 3	Queue 4	Queue 5
:					
14	Queue	Queue			
15	Queue	Queue	Queue	Queue	
:					

Table – 11: An example of the xy master table.

Hop count	Queue 1	Queue 2	Queue 3
:			
14	Queue	Queue	
15	Queue	Queue	Queue
:			

Table – 12: An example of the second routing table scheme for the xy connections.

- 7 to 20 routers are used
- Each simulation consists of 50 runs.
- Each router sends 2800 to 8000 packets for each run, depending on the number of routers
- The size of the packet varies from 1Kb to 1Mb.
- Only one routers can transmit at an instance
- The order of transmitting between routers are done in a random order
- Each router must transmit to all the other routers

Table–13: The criteria and assumption of simulation

5.2 Simulation Criteria

We have defined certain criteria that resemble an actual network, such as different router counts, variable packet sizes, and various numbers of packets. (see table 13)

5.3 Greedy Vs DML

We have tested over fifty combinations of different scheduling and queuing algorithms. DML outperformed almost all of the other algorithms (the other algorithms were omitted from this paper due to space limitations). Only Greedy queuing algorithm had shown substantial better results in certain conditions. A simple definition of the Greedy algorithm is:

Greedy – Arriving packets were routed and queued through the queue that had the best solution.

The results of our work are enormous, over 200,000 simulations. It is extremely difficult to show all the results that we have obtained through this paper. As a result, four figures are produced. All the graphs show the average improvement of *penalties per average packet size* of DML algorithm over Greedy algorithm.

- P_{DML} be the average *gross penalty* of all queues at the end of the simulation using the *DML* algorithm.
- P_G be the average *gross penalty* of all queues at the end of the simulation using the *Greedy* algorithm.
- S be the average packet size for all the packets in the data set.
- $I_{DML.G}$ be the average improvement of penalties per packet size from the DML algorithm over Greedy algorithm. i.e.,

$$I_{DML.G} = (P_{DML} - P_G)/ S \qquad (13)$$

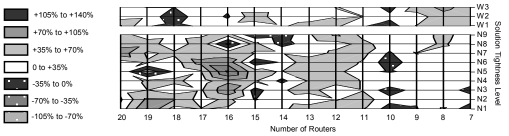

Fig – 5: The $I_{DML.G}$ for a Busy network, 2nd Solution Tightness, 2nd Routing Table, and an Input Queue of 32.

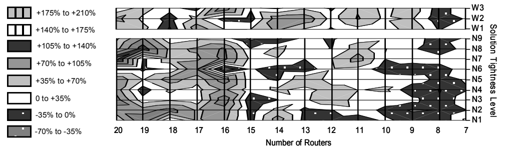

Fig – 6: $I_{DML.G}$ for a Meshed network, 1st Solution Tightness, 3rd Routing Table scheme, and an Input Queue size of 32.

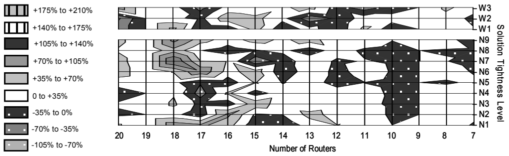

Fig – 7: $I_{DML.G}$ for a Parallel network, 2nd Solution Tightness, 2nd Routing Table scheme, and an Input Queue size of 11.

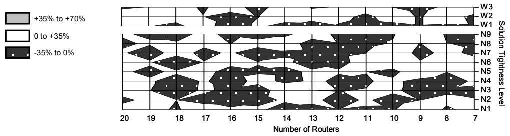

Fig – 8: $I_{DML.G}$ for a Star network, 3rd Solution Tightness, 2nd Routing Table scheme, and an Input Queue size of 22.

In some details, all figures have 2 graphs. The top graph shows the results for wide range deadlines namely, W1 through W3. The bottom graph show the results for narrow range deadlines namely, N1 through N9.

	Solution Tightness	Routing Table	Input Queue
Star	Both are equal	Both are equal	Both are equal
Busy	DML is better for all deadlines but it is much better if it is less tight.	DML is slightly better if routing table increases	DML is slightly better if input queue increase
Parallel	Both are equal	DML is slightly better if routing table increases	Both are equal
Meshed	DML is better for all deadlines but it is much better if it is less tight.	Both are equal	Both are equal

Table–14: Comparison table for all the network topologies and the environment conditions.

6 Conclusions

This paper presents an intelligent CRP solution for determining near optimum routing for deadline based packet forwarding. The suggested algorithm performs significantly better (up to 200%) than Greedy algorithm in a most cases.

In this paper, we did not discuss the protocols such as the network mechanics that might be designed to distribute and install heuristics in routers, or to collect network bandwidth information. Also, how IP intercepts the concept of intelligent routing has not been covered here. However, the research in extensible routers, programmable networks, and active networks, has looked into these issues in depth. Rather, in the paper we focused on designing high quality extensible routing solutions.

As opposed to hop-based routing, optimized intelligent routing is becoming increasingly important to internetworking applications. Routing is particularly important in mobile wireless systems. The mobility itself adds a fundamental challenge in routing. On top of that, asymmetry in network bandwidth and in devices capacity make route optimization particularly critical. Overlay networks proposed in many current systems such as grid computing, content distribution and content services network, all require custom routing. Various future net-centric nomadic and adaptive systems will require much advanced and domain knowledge enriched intelligent routing, which are not available in today's network infrastructure. Indeed such variability is not an external problem but is ingrained deep into the nature of the applications. For example, an entertainment video application generally tries to minimize the variance in inter-frame arrival times (also known as 'jitter".) A tele-surgery application will be much more interested to minimize the deviation of arrival time from its arrival time. These are different quantities and have different data dependency. Thus, QoS problem intrinsically is not solvable at non extensible network layer.

An interesting aspect of the proposed approach is that the key optimization shell is versatile. Even though we have discussed the solution against one particular type of optimization criteria, the CRP shell that guided the packet selection/queue evaluation process has already been proven to be generic in constraint satisfaction planning research in AI research. In CRP, this can be achieved by substituting the two heuristics without any fundamental change to the control structure. Consequently, shell-systems such as this, if embedded into networks, have the potential to be the corner stone for a new generation of smart extensible routing architecture for intelligent networks.

7 References

[1] M. Adler, A.L. Rosenberg, R.K. Sitaraman and W. Unger, Scheduling time-constrained communication in linear networks: *Proceedings 10th Annual ACM Symposium on Parallel Algorithms and Architectures*, Puerto Vallarta Mexico, 1998, 269-278.

[2] I. Ben-Aroya, D.D. Chinn, and A. Schuster, A CLT-type lower bound for nearly minimal adaptive and hot potato algorithms, Technical Report LPCR #9405, CS dept. Technion, May 1994.

[3] J. C.R. Bennett and H. Zhang, Hierarchical packet fair queueing algorithms: *IEEE/ACM Transactions on Networking* 5(5), Oct 1997, 675-689.

[4] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W, Weiss, An architecture for Differentiated Service, *RFC-2475*, December 1998.

[5] Y. Ge, D. Y. Y. Yun, A general planning and scheduling methodology for solving task scheduling with multiple objectives: *Proceedings of the IASTED International Conference*, Honolulu, USA, August 1996.

[6] Pawan Goyal, Harrick M. Vin, and Haichen Cheng, Start-time fair queuing: A scheduling algorithm for integrated services packet switching networks, *Proceedings of ACM SIGCOMM*, Palo Alto, CA, USA, August 1996, 157–168.

[7] P. Haslum, and H. Geffner, Heuristic planning with time and resources: *Workshop Notes of the IJCAI-01 Workshop on Planning with Resources*, Seattle, WA, USA, August 2001.

[8] Christian Huitema, *Routing in the Internet* (Englewood Cliffs, NJ: Prentice Hall PTR, 1999).

[9] S. Jamin, P. Danzig, S. Shenker, and L. Zhang. A measurement-based admission control algorithm for integrated service packet networks. *IEEE/ACM Transactions on Networking* 5(1), February 1997, 56–70.

[10] N. P. Keng and D. Y. Y. Yun, A planning/scheduling methodology for the constrained resource problem: *Proceedings of the 11th IJCAI*, August 1989, 20-25.

[11] Javed I. Khan & Nouman Bantan, A Constraint Resource Planning Algorithms for deadline based Packet Forwarding for Overlay Networking, *Proceedings of the 3rd International Conference on Wireless and Optical Communication*, July, 2002, Banff, Canada, pp.578-584.

[12] Javed I. Khan and Seung. S. Yang, A framework for building complex netcentric systems on active network: *Proceedings of the DARPA Active Networks Conference and Exposition*, San Jose, CA, USA, May 2002.

[13] T. V. Lakshman and D. Stiliadis, High speed policy-based packet forwarding using efficient multi-dimentional range matching: *SIGCOMM'98*, Vancouver, Canada, September 1998.

[14] Parameswaran Ramanathan and Kang G. Shin, Delivery of time-critical messages using a multiple copy approach: *ACM Transactions on Computer Systems* 10(2), May 1992, 144-166.

[15] H. Sariowan, R.L. Cruz, and G.C. Polyzos, Scheduling for quality of service guarantees via service curves, *Proceedings ICCCN'95*, Las Vegas, USA, September 1995.

[16] Jonathan M. Smith, Programmable networks: Selected challenges in computer networking: *IEEE Computer* 32(1), January 1999, 40-42.

[17] M. Waldvogel, G. Varghese, J. Turner, and B. Plattner: *Proceedings SIGCOMM'97*, Cannes, France, September 1997, 25-36.

[18] D. Y. Y. Yun, CRP - An intelligent engine for resource and production management: exhibition and brochure, *COMDEX'92*, November 1992.

[19] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zapala, RSVP: A new resource reservation protocol: *IEEE Network* 7(5), September 1993, 8–18.