Proceedings of the 3rd International Conference on
Wireless and Optical Communication, July, 2002, Banff, Canada, pp.578-584

A CONSTRAINT RESOURCE PLANNING ALGORITHM FOR DEADLINE BASED PACKET FORWARDING FOR OVERLAY NETWORKING

Javed I. Khan & Nouman Bantan
Internetworking and Media Communications Research Laboratories
Department of Math & 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 algorithm for deadline-satisfying packet forwarding and route lookup. This algorithm has been designed after a Constraint Resource Planning (CRP) methodology of approximation algorithm design. The algorithm uses scheduling and forwarding heuristics to produce near-optimal performance. These heuristics execute within a generic shell. In this paper we present the algorithm, the shell architecture, and the simulation results.

KEY WORDS
Intelligent routing, temporal quality of service.

1. INTRODUCTION

There are many applications including control and multimedia systems which require delay bounded communication. Communication with temporal quality is an open problem in network transport research. Any effective time routing algorithm must be backed by a fast time based forwarding algorithm. However, deadline based packet forwarding is itself a difficult problem. The optimum deadline based packet scheduling has been shown to be NP-hard [1]. In this research we present an approximation algorithm based on the heuristics management research in AI and planning. Not too many previous work exists which combines the two fields. This proposed algorithm can produce a near optimum solution. The algorithm has been designed using Constraint Resource Planning (CRP) methodology- a constraint satisfaction heuristics system.

An interesting aspect of the proposed solution is that it is also an optimizing shell and can form a basis for efficient custom overlay routing optimization criteria at run-time. Custom overlay routing has many advantages. For example, as opposed to hop-based routing, a dynamic optimized intelligent routing is more desirable for mobile wireless systems. Mobility, itself, makes it substantially difficult to manage route information. Furthermore, the asymmetric capacity of wireless integrated fiber links makes route optimization unavoidable. Run-time optimization techniques can be used to improve these difficulties. It has increasingly become apparent that various future net-centric nomadic and adaptive systems will require much more advanced and domain knowledge enriched intelligent routing, which is not available in today's network infrastructure. The proposed shell can be potentially used for custom route optimization for overlay networks.

1.1 Related Works

Routing is the key service embedded into the current Internet layer. It has two major components—routing information propagation and forwarding. Most research in routing has been conducted to find stable and scalable route propagation algorithms. The transitions 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 is primarily focused on the design of efficient lookup table data structures [2, 3, 6].

It is highly unlikely any temporal QoS can be provided without intelligent forwarding. Some researchers investigated bounded buffer issue for outgoing packets in forwarding path for time-constrained traffic [5, 1]. There are two principle approaches to provide temporal quality of service – the Integrated Service (int-service) and the Differentiated Service (diff-service) approach [4]. The int-service has been based on per flow based resource reservation and classification of services, admission control and policing [5, 6]. In the *int-service* paradigm, a number of interesting results on queuing behavior under controlled traffic admission have been reported in [7, 8, 9]. One simple approach was introduced in [10] in which multiple copies of time-constrained packets are sent. This approach increased the probability of on-time delivery, but it also increased the overall traffic. Other more sophisticated approaches can be found in [11]. Another technique – choosing a packet out of a buffer of time-constrained packets – can also guarantee a high level of quality of service [12]. Moreover, selecting a packet with the earliest deadline is guaranteed to provide the requisite quality of service.

The other and more recent approach is the *diff-service* [13]. Instead of setting the service on per flow basis, packet flows are simplified into flow classes. Instead of any guaranteed service, differential treatment of flows are

ensured. Diff-services can help simplify the complexity of forwarding operation. Routes have one queue form each class. Packets arrive with class labels and are accordingly diverted in the appropriate queue. The

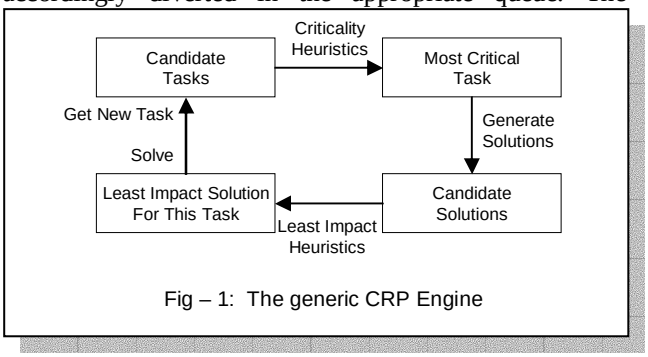


Fig – 1: The generic CRP Engine

packets from time-critical applications will be in a high priority queue that will then be processed first. Both the service architectures require substantial modification of the IP-router architecture, where multiple queues have to be maintained.

1.2 Intelligent Routing

The existing research will show that most of these 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 integrated 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 promising results [14,21], 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 problems are poised to be developed, including network mechanisms for deadline assured delivery mechanism. Network and transport layer intelligence will have a major bearing in any quality of service provisioning effort.

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 current IP layer, where the routing components are responsible for efficient collection and propagation of accurate network connectivity information, and the forwarding component focuses on fast forwarding of packets based on the collected information. In this research we demonstrate that we might be able to infuse deep algorithms in routers for intelligent networking with the emphasis of deadline conformant packet forwarding. In this paper, we present 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, in which we present the overview of the CRP based routing solution, which provides a fast polynomial algorithm for dead-line based forwarding and route lookup. In section three, we will briefly empirically compare our algorithms with the two current packet selection algorithms used for forwarding, round robin and greedy selection. Finally, in section four, we will present the detailed results of experiments using our approach.

2. DEADLINE-BASED ROUTING

2.1 CRP

Optimum deadline scheduling is NP-hard, which is proved in [1]. Instead of attempting to find random approximation techniques, the algorithm has been designed using the Constraint Resource Planning (CRP) methodology of approximation algorithm design. It is critical for network algorithms to be fast. In applied AI and planning research, a large body of knowledge exists form fast approximation algorithms of NP-hard problems [15, 16, 17, 18]. We have selected the CRP methodology, which has the particular appeal of being a generic execution shell [18]. The CRP divides the solution of any complex NP-hard and NP-complete problem into two decision heuristics called respectively as *most-critical-task heuristics* H_{MCT} () and *least-impact solution heuristics* H_{LIS} () [18, 19]. CRP provides a polynomial time bounded execution shell called the CRP engine that uses the heuristics to generate powerful near optimum solutions. While the shell remains intact, the domain specific knowledge required to produce custom solutions for different constraint satisfaction problem can be encoded into these two heuristics. The run-time complexity of the shell is O ($n \log n + m \log m$), where n is the number of tasks at hand, and m is the number of potential immediate assignments to each task. CRP provides a domain-independent, systematic framework to solve constraint satisfactory problem in planning and scheduling. CRP currently provides very competitive near-optimum solutions for more than 50 problems including traveling salesman, job shop scheduling, and map coloring to surface triangulation to optimum packing [20]. It divides any constraint satisfaction planning problems into a four-corner execution model as shown in figure-1. We have designed an intelligent routing shell that is expected to provide various intelligent routing solutions. In this paper, we will share the solution specific to time-bound communication. We demonstrate a novel approximation algorithm where the forwarding bandwidth and alternate routes are first segmented into q paths with a separate sub-queue for each. Then arriving packets are scheduled and assigned to the sub-queues according to the packet deadlines and the expected queuing delay in the sub-queues. According to the CRP methodology, we identify the two critical decision points—the ordering of the most critical task, and assignment of the selected task

based on least-impact strategy. We then formulate a number of candidate heuristics for each of these decision points. We have experimented with more than 24 combinations of probable heuristics. We will share the performance of several combinations that outperformed conventional algorithms in almost every test.

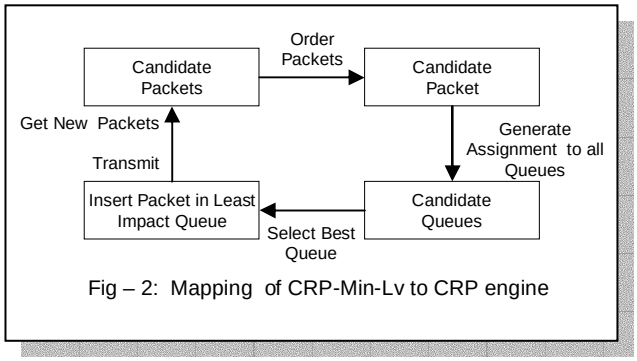


Fig – 2: Mapping of CRP-Min-Lv to CRP engine

2.2 Algorithm Overview

Our proposed algorithm is called *CRP Deadline Minimum Leverage* (CRP-DL-Min-Lv) Algorithm. This iterative algorithm consists of two principal phases – Scheduling and Queuing. Below we describe each. These two phases represents the task generation and solution set generation phases of CRP.

Scheduling: The objective of the scheduling policy is to select a packet from incoming queue using the *most-critical-task heuristics* H_{MCT} () of CRP. The algorithm is executed at two states – the initial state and the continuous state. The initial state represents the cold-start procedures of the algorithm (e.g. deadline-based routing is activated). The continuous state represents the subsequent cyclical execution of the algorithm. At a given time, the router has *n* packets to transmit, located in an internal queue. The scheduling model is as follows:

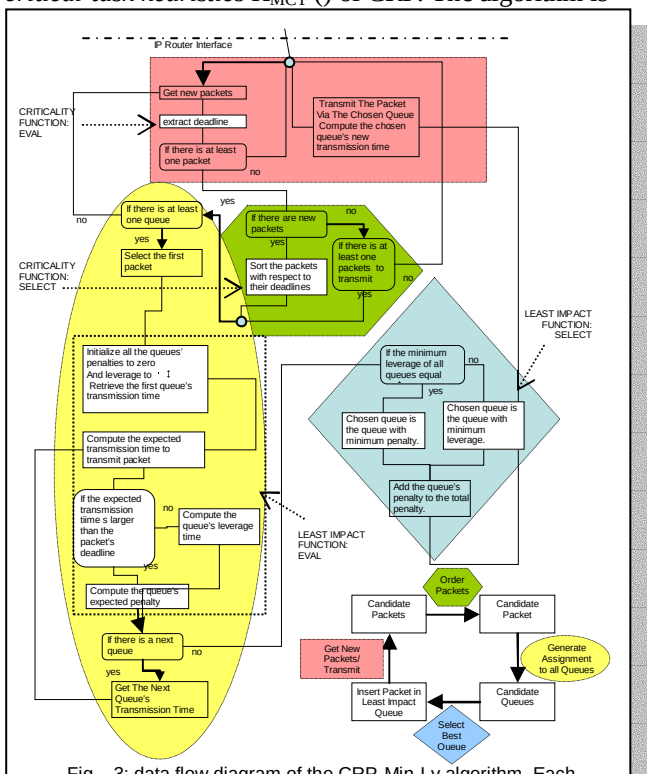


Fig – 3: data flow diagram of the CRP-Min-Lv algorithm. Each group of actions are mapped to an action in the CRP diagram. The white boxes indicate the pluggable heuristics components substitutable for other optimization objectives.

- Initial state
  - If there is at least one packet
    - Retrieve all packets' deadlines
    - Sort all packets in the system, in an ascending order, with respect to their deadlines into the internal queue
    - Select the first packet in the sorted list
- Continuous state
  - If there is at least one packet
    - If there are new packet/s arrived that have since the last run of H_{MCT} ()
      - ❑ Retrieve the new packet/s' deadline
      - ❑ Retrieve the new packet/s' deadline
      - ❑ Sort all packets in the system in an ascending order with respect to their deadlines into the internal queue
    - Select the first packet in the sorted list

Queuing: The objective of the queuing policy is to assign the selected packet to a queue using the *least-impact solution heuristics* H_{LIS} (). At this time, the network has *m* queues or outgoing routes, which will compete to transmit the selected packet. The queuing model is as follows:

- If there is at least one queue
  - Retrieve the first queue's transmission time
  - Initialize
    - Set all queues' leverage times to ∞
    - Set all queues' penalties to 0
  - Compute the expected transmission time
  - If the expected transmission time is larger than the packet's deadline
    - Compute the queue's penalty
  - Else
    - Compute the queue's leverage time
  - For each of remaining queues,
    - Retrieve the queue's transmission time
    - Compute the expected transmission time
    - If the expected transmission time is larger than the packet's deadline
      - ❑ Compute the queue's penalty
    - Else
      - ❑ Compute the queue's leverage time
  - If the minimum leverage of all queues is ∞
    - Chosen queue is the queue with the minimum penalty
  - Else
    - Chosen queue is the queue with the minimum leverage

- Increase the gross penalty by the chosen queue's penalty
- Compute the chosen queue's new transmission time

2.3 Deadline Model

Below we present the solution framework and explain the notations. First we define the packet arrival model, packet properties, and packet queuing model.

```
H_MCT(){
    If n > 0 {
        If w_i > 0 {
            [Retrieve first new packet's deadline]
            w_i = w_i – 1
            while w_i > 0 {
                [Retrieve next new packet's deadline]
                w_i = w_i – 1
            }                          // Evaluation Section
            ------------------------------------------------
            Sort (n)                   // Selection Section
        }
        p_i = RemoveFirstFromTheQueue ()
    }
}
```

Table–1The CRP-Min-Lv scheduling algorithm (init).

- Set of n packets to transmit but only one packet can be queued at any time
- Each packet, p_i where $1 \le i \le n$,
  - takes *transmit time*, t_i
  - must be transmitted before a preset *deadline*, d_i
- Set of m queues
- Each queue, q_j where $1 \le j \le m$,
  - has transmission times, T'i,j and Ti,j, before and after packet pi is transmitted respectively. T'i,j is the summation of transmit times prior packet pi's transmission via qj. Ti,j is the summation of times to transmit after packet pi's transmission via qj. i.e.

$$T'_{i,j} = \sum_{p_b} t_b \ \forall\ p_b \in q_j \text{ where } 1 \le b < i,\ 1 \le i \le n, \quad \text{and } 1 \le j \le m \qquad \ldots(1)$$

$$T'_{i,j} = \sum_{p_b} t_b \ \forall\ p_b \in q_j \text{ where } 1 \le b \le i,\ 1 \le i \le n, \quad \text{and } 1 \le j \le m \qquad \ldots(2)$$

Based on the above, we also define a set of additional concepts. Let

- $e_{i,j}$ be the *expected delivery time* for packet p_i via queue q_j i.e.

$e_{i,j} = t_i + T'_{i,j}$ where $1 \le i \le n$, and $1 \le j \le m$ …(3)

- c_i be the *chosen queue number j*, as in queue q_j, which will transmit packet p_i i.e.

$c_i = j \ni p_i \in q_j$ where $1 \le i \le n$, and $1 \le j \le m$ …(4)

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

$L_{i,j} = d_i - e_{i,j}$ where $e_{i,j} \le d_i$, $1 \le i \le n$, and $1 \le j \le m$ …(5)

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

$L'_i = Min(L_{i,j}) \ \forall j$ where $1 \le i \le n$, and $1 \le j \le m$ …(6)

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

$Y_{i,j} = e_{i,j} - d_i$ where $d_i \le e_{i,j}$, $1 \le i \le n$, and $1 \le j \le m$ …(7)

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

$Y'_i = Min(Y_{i,j}) \ \forall j$ where $1 \le i \le n$, and $1 \le j \le m$ …(8)

- l'_i and l_i be the *current gross penalty* or the penalty of all queues before and after packet p_i is transmitted respectively. The penalty value is the extra time needed to deliver a packet after its deadlines. i.e.

$$l'_i = \sum_{p_b} T_{b,j} - d_b \ \forall j \text{ where } 1 \le b < i,\ 1 \le i \le n, \quad \text{and } 1 \le j \le m \qquad \ldots(10)$$

$$l_i = \sum_{p_b} T_{b,j} - d_b \ \forall j \text{ where } 1 \le b \le i,\ 1 \le i \le n, \quad \text{and } 1 \le j \le m \qquad \ldots(11)$$

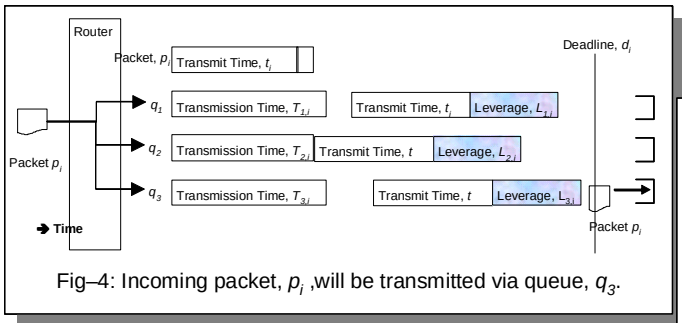
w_i be the number of packet/s that arrived into the network

```
H_LIS(){
    If m > 0 {
        [Retrieve first queue's transmission time]
        For j ← 1 to m {
            L_i,j ← ∞
            Y_i,j ← 0
        }
        e_i,j = t_i + T'_i,j
        If d_i < e_i,j
            Y_i,j ← d_i – e_i,j
    Else L_i,j ← e_i,j – d_i
        For j ← 2 to m {
            [Retrieve the queue's transmission time]
            e_i,j = t_i + T'_i,j
            If d_i < e_i,j
                Y_i,j ← d_i – e_i,j
            Else L_i,j ← e_i,j – d_i
        }                              // Evaluation Section
        ------------------------------------------------
        L'_i = ∞                       // Selection Section
        For j ← 1 to m {
            If L_i,j < L'_i {
                L'_i = L_i,j
                c_i = j
            }
        }
        If L'_i = ∞ {
            Y'_i = 0
            For j ← 1 to m {
                If Y'_i > Y_i,j {
                    Y'_i = Y_i,j
                    c_i = j
                }
            }
        }
        T_i,c_i ← T'_i,c_i + t_i
        l_i ← l'_i + Y_i,j
    }
}
```

Table–2: The CRP-DL-Min-Lv queuing algorithm.

for transmission since the last run of $H_{MCT}()$.

2.4 Algorithm



Fig–4: Incoming packet, p_i ,will be transmitted via queue, q_3.

The detail of the algorithm is given in Tables 1 and 2, and the flow chart is given in Table-2. The four major blocks on the chart identify the major CRP blocks. Further, each heuristics implementation consists of two processes – (i) the evaluation which computes the heuristic cost function for each incoming candidate and (ii) the selection process which orders them. In this case the evaluation process determines all the packets' deadlines and all the queues' information for the selected packet. The selection process identifies both the most critical packet and the least impact queue. Notably, the overall control structure of the algorithm is a generic execution shell. The two heuristics (consisting of the four process modules) confine all the domain specific processing within it. Thus, the same system can be used for other optimizing criteria by changing only these modules.

2.5 Overlay Interface with Base Routing

The intelligent forwarding shell can be placed as a modular overlay component on most existing routing architecture. As shown in Fig-3 (top), the optimizer shell interfaces with the basic routing mechanism via three basic service calls: (a) get packet: accept deadline critical packets (b) get routes: receive alternate routes, (c) put packet: forward a specified packet via a specified route.

2.6 Examples

The following two examples illustrate the execution of the queuing model with three queues. Both examples display the queuing of a packet pi when it enters a router. The first example is shown in figure 4 where q3 has the minimum leverage value out of all the available queues, which is the chosen queue in this example. The second example is shown in figure 5. All the available queues will deliver the packet pi after its deadline. In this case, q2 has the minimum penalty value out of all the available queues, which is the chosen queue in this example.

3. ANALYSIS

In Table-3, we provide the run-time cost analysis of the algorithm. We provide the solutions with the following four assumptions.

1. The Sort function has a cost of O (n lg n).
2. The RemoveFirstFromQueue function, retrieving a packet's deadline, and retrieving a queue's transmission time have a cost of $O(1)$.
3. There is at least one packet in the internal queue at all times.

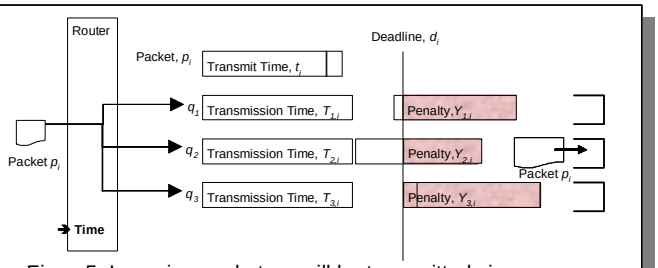


Fig – 5: Incoming packet, p_i ,will be transmitted via queue, q_2.

4. There is at least one available queue at all times.

In the table we provide the number of assignments and number of comparisons separately on the basis of per packet arrival. We also compare it with two conventional forwarding algorithms – Round Robin and Greedy.

	CRP-Min-Lv		**Round Robin**		**Greedy**	
	Best	Worst	Best	Worst	Best	Worst
Assignment	6 + 4*m*	5 + 6*m*	2	3	2	1 + *m*
Comparison	3 + 5*m*	3 + 7*m*	1	1	2*m*	2*m*
Sorting	0	*n* lg *n*	NA	NA	NA	NA
Total Cost	**8 + 9*m***	**11+13*m* +*n* lg *n***	**3**	**4**	**2 + 2*m***	**1 + 3*m***

Table – 3: The run-time cost for the CRP-Min-Lv, Round Robin, and Greedy algorithm.

4. EXPERIMENT RESULT

We have conducted many simulations and have found the performance of several candidate approximation algorithms suitable for deadline based scheduling in the forwarding path of a control network. To evaluate the algorithms, we assigned a penalty function for each packet. If a packet was delivered within the deadline, it was zero. Otherwise, the difference between the deadline and the expected end of transmission time of a queued packet was added to the algorithm's penalty. We did not assign any credit for early delivery. Thus the penalty could only be zero or more. We demonstrated that O ((n lg n)+m) near optimum algorithms were possible, which offered dramatic improvement in deadline assured communication, where n was the expected length of the input queue and m was the number of sub-queue partitions.

To challenge the system we have created the datasets with three different solution complexities. We have started with a perfect solution and then worked backward to generate the challenge dataset. For each packet we assigned a deadline. Then we scrambled the packets and let the algorithm discover back the solution. 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 time it will take just to deliver the data with perfect solution scenario. In each of the three set we put 50 streams; each stream has 20 packets. A data generator program produce theses data sets. The sets differ with their delivery tightness.

We then applied those data sets on three algorithms: Round Robin, Greedy, and CRP-DL-MinLv. We

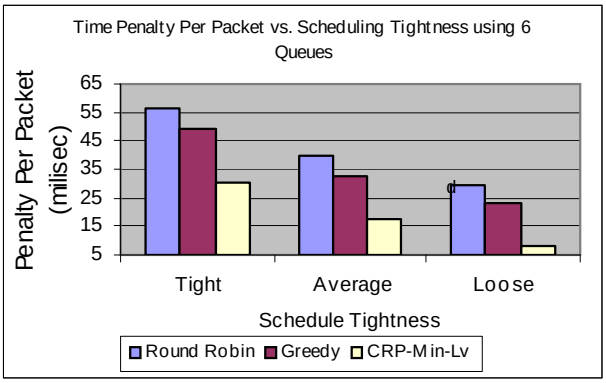


Fig-6(a) Time Penalty of the three algorithms for various levels of optimization difficulties.

assumed that the packets were arriving at uniform interval (typical for audio/video traffic).

Delivery Tightness	Tight	Average	Loose
Range	$1 \geq \frac{t_i}{d_i} \geq 0.5$	$0.5 \geq \frac{t_i}{d_i} \geq 0.25$	$0.25 \geq \frac{t_i}{d_i} \geq 0.1$

Table – 4: The range of delivery tightness of each dataset.

However, their sizes varied. In each epoch one packet from each stream arrived. In Round Robin Algorithm, the arriving packets were routed and queued through the next queue in a cycle. Greedy algorithm ignored the impact on other packets. Arriving packets were routed and queued through the queue that had the best solution (least transmission time or highest bandwidth).

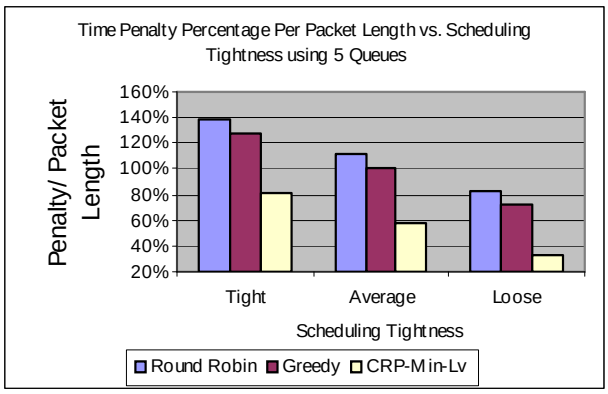


Fig-6(b) Time Penalty of the three algorithms for various levels of optimization difficulties. Normalized with packet lengths

Fig-6(a) displays the average penalty per packet for these three algorithms for three tightness values of the data. The CRP-Min-Lv showed improvement ranging from 39% to 73% against the other methods, and it outperformed the Greedy method. The tightness of the data had some interesting impact. Most improvement was visible when the data was loose, or, in other words, there was some scope for improvement. The simple "greedy" strategy offered about 20% improvement over Round Robin, and failed to take any significant advantage of optimization. On the other hand the CRP algorithm here reduced the penalty by about 300-400% compared to RR or Greedy!

Fig-6(b) displays the average penalty percentage in terms of single packet length. The penalty percentages for the Greedy and the Round Robin were more than 200% of the penalty percentage for the CRP-Min-Lv in the loose dataset. Those methods' penalty percentage never reached below 160% of the CRP-Min-Lv in all the datasets. Fig-6(c) shows the effect of the number of output queues. As the number of output queues increased, so did the performance.

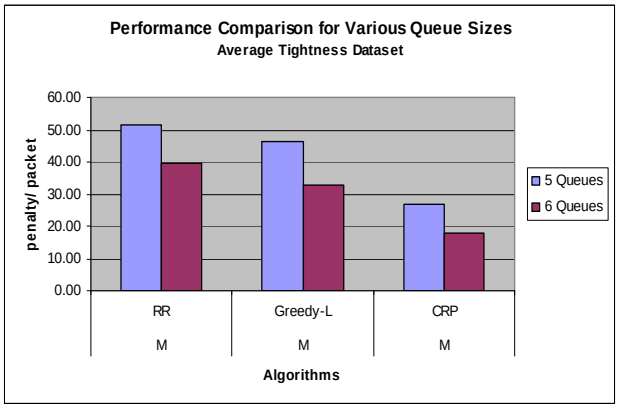


Fig-6(c) Time Penalty of the three algorithms for various Q

5. CONCLUSIONS

This paper presents an intelligent CRP solution for determining near optimum routing for deadline based packet forwarding. The algorithm performs significantly better (200-400%) than simple Round Robin or Greedy algorithm.

In this paper, we did not explain the process route metrics collection and propagation. We also did not discuss the details of the network mechanics. We have designed an IPV6 extension header based mechanism to carry the deadline information.

Also, we are currently exploring several *sparse routing* schemes. In the innovative sparse routing we propose not all packets are time-forwarded at all routers. Only occasionally they will receive 'timeboost' as per some time based forwarding algorithm outlined. Apparently solution quality can be maintained with negligible lookup overhead.

As opposed to hop-based routing, optimized intelligent routing is becoming increasingly important to

internetworking networked 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 (between various land and wireless links) 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. The recent QoS research seems to point out that though more domain specific knowledge have to be applied, but it will be hard to get any consensus about the optimization objective or base metric. 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.

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 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 routing based intelligent networks.

6. REFERENCES

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

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

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

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

[5] 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.

[6] 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.

[7] 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.

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

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

[10] 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.

[11] S. Rampal and D. Reeves. Routing and admission control algorithms for multimedia data: *Computer Communications, 18*, October 1995, 755-768.

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

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

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

[15] A. Fukunaga, G. Rabideau, S. Chien, D. Yan, Toward an application framework for automated planning and scheduling: *Proceedings of the 1997 International Symposium on Artificial Intelligence, Robotics and Automation for Space*, Tokyo, Japan, July 1997.

[16] J. Koehler, Planning under resource constraints: *In Proceedings of the Thirteenth European Conference on Artificial Intelligence (ECAI-98)*, Brighton, UK, 1998, 489-493

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

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

[19] 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.

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

[21] 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.