Uniform Weighted Round Robin Scheduling Algorithms for Input Queued Switches

Idris A. Rai and Murat Alanyali
Department of Electrical and Electronics Engineering
Bilkent University
Bilkent Ankara 06533 Turkey
{rai,alanyali}@ee.bilkent.edu.tr

Abstract— **This paper concentrates on obtaining uniform weighted round robin schedules for input queued packet switches. The desired schedules are uniform in the sense that each connection is serviced at regularly spaced time slots, where the spacing is proportional to the inverse of the guaranteed data rate. Suitable applications include ATM networks as well as satellite switched TDMA systems that provide per packet delay guarantees. Three heuristic algorithms are proposed to obtain such schedules under the constraints imposed by the unit speedup of input queued switches. Numerical experiments indicate that the algorithms have remarkable performance in finding uniform schedules.**

I. INTRODUCTION

A wide range of emerging applications place stringent performance requirements on broadband integrated service networks. Measures of service quality to be delivered by the network are commonly expressed in terms of packet loss, delay, and jitter. Applications involving real-time multimedia are inelastic in their requirements, and their performance measures are seldom expressed in probabilistic terms. Each such application flow, once admitted to the network, has to be virtually isolated from the state of the rest of the network. Appropriate per-hop behavior at network nodes is the essential component in delivering this end-to-end performance, and considerable research effort has been directed to identifying per-hop scheduling policies which enable applications enjoy a perception of privacy while sharing network resources.

Flow isolation on network nodes with output queued switch architectures is well understood. Fair queuing algorithms achieve this end by employing generalized processor sharing to provide a guaranteed service rate for each active flow, and divide any remaining capacity fairly among such flows[1]. When used in conjunction with leaky bucket regulation, these algorithms provide end-to-end delay guarantees[2]. The most elementary implementation of generalized processor sharing is weighted round robin scheduling. Several other versions and implementations of fair queuing with varying complexities have been studied [1], [3], [4].

Output queuing has the disadvantage of requiring a fast switch fabric. For a switch size of $N \times N$, the switching fabric needs to run N times faster than the port rate. Increasing port rates and large switch sizes force designers to consider combined input-output queuing, which entails speeding up the switch fabric by a factor of S, $1 \leq S < N$, with respect to the port rate. This does not incur any performance penalty compared to output queuing if $S \geq 2$; these values of S suffice to emulate packet departures of an output queued switch[5]. The price paid lies in the complexity of the packet scheduling algorithms. These algorithms require packet departure times in advance, thus they are not good candidates in practice. In the extreme case when $S = 1$, the switching fabric runs at the port speed, and the switch employs input queuing only. An input queued switch can provide unit throughput by employing non-FIFO queuing[6], however it cannot emulate an output queued switch in general[5]. Significantly less is known about the quality of service capabilities of input queued switches relatively to those of output queued switches.

In this paper we provide strong evidence that an input queued switch can emulate an output queued switch if the latter employs a fine-granularity weighted round robin scheduler. Namely, we consider nonblocking crossbar switch with N input and N output ports. The demand on the switch is summarized by a matrix $M = [m(i,j)]_{N\times N}$, which is a nonnegative integer matrix whose line sums are identical and equal to P. The goal is to schedule the service provisions through the switch such that for each $i, j \in \{1, 2, \cdots, N\}$ and $k = 1, 2, 3, \cdots$, the switch provides the kth service between input port i and output port j by time slot $\lceil kP/m(i,j)\rceil$. Here $\lceil x\rceil$ denotes the smallest integer that is no smaller than x. This performance can be delivered by an output queued switch in a relatively straightforward manner[4]. However an input queued switch with unit speedup operates under the constraint that each input port and each output port can receive at most one service in each time slot, and it is far from clear whether such a schedule exists and if so how it can be obtained. This paper presents a formulation and an analysis which yields enough insight to develop very effective heuristic scheduling algorithms.

Special cases of the situation considered here have been considered before. Philp and Liu conjectured that it is possible to schedule periodic traffic through an input queued switch so that each packet from a flow leaves the switch before the next packet of that flow arrives, provided that line utilization does not exceed unity[7]. Their setting corresponds to the case when $P/m(i,j)$ is an integer for each (i, j) in the present formulation. Giles provided a constructive proof of this conjecture in the case when each period

$P/m(i,j)$ evenly divides all longer periods[8]. A slightly more general model with time-dependent service rates is considered in [9], and a scheduling algorithm is given for the case $N=2$.

The rest of the paper is organized as follows. Section II involves the formulation of the scheduling problem, and gives a characterization of its solutions. This characterization is used in Section III to obtain a number of heuristic algorithms. These algorithms perform remarkably well, and their numerical study is presented in Section IV.

II. THE SCHEDULING PROBLEM

Consider an $N \times N$ input queued packet switch. The service requirement on the switch is given by a *traffic matrix*, $M = [m(i,j)]_{N\times N}$, which is a nonnegative integer matrix such that for some integer P

$$\sum_{i=1}^{N} m(i,j) = P \qquad \text{for all } j = 1,2,\cdots,N$$
$$\sum_{j=1}^{N} m(i,j) = P \qquad \text{for all } i = 1,2,\cdots,N.$$

The integer P is called the *schedule length.* The traffic matrix dictates that the switch is required to provide service to each connection (i,j) at rate $m(i,j)/P$, and in a regular manner such that the connection should receive its kth service by time slot $\lceil kP/m(i,j)\rceil$. Note that since the switch fabric has unit speedup, this requirement should be satisfied simultaneously for all connections subject to the constraint that at each time slot each port (either input or output) can be served at most once. If the switch can be programmed to deliver this performance, then we say that the traffic matrix admits a *uniform weighted round robin (WRR) schedule.*

We proceed with a characterization of uniform WRR schedules on the input queued switch. Let the *deadline sequence* for a connection (i,j) be the one-sided binary sequence $(d_1(i,j), d_2(i,j),\cdots)$ such that $\sum_{l=1}^{k} d_l(i,j)$ is the smallest number of services that should be received by the connection by the end of time slot k. In particular

$$d_l(i,j) = \begin{cases} 1 & \text{if } l = \lceil kP/m(i,j)\rceil \text{ for some } k \\ 0 & \text{otherwise.} \end{cases}$$

Note that the deadline sequence is periodic, and its period is equal to the schedule length P.

Packet transmission schedule through the switch at any time slot is represented by a *permutation matrix* $\Pi = [\pi(i,j)]_{N\times N}$, which is a binary matrix with exactly one nonzero entry in each row and in each column. Namely, $\pi(i,j) = 1$ indicates that connection (i,j) receives service in the associated time slot. In that case all other entries on the ith row and jth column are zero, thus each input and output port is served at most once at each time slot. If a sequence $(\Pi_1, \Pi_2, \cdots)$ of permutation matrices is adopted so that the switch is scheduled according to Π_l at each time slot l, then $\sum_{l=1}^{k} \Pi_l(i,j)$ denotes the number of services received by connection (i,j) by time slot k. Thus such a sequence constitutes a uniform WRR schedule if and only if for each connection (i,j),

$$\sum_{l=1}^{k} \Pi_l(i,j) \geq \sum_{l=1}^{k} d_l(i,j) \qquad \text{for each } k = 1,2,\cdots \tag{1}$$

Note that if such a sequence exists, then the periodicity of the deadline sequence implies that it can be taken periodic by setting $\Pi_k = \Pi_{(k \text{ modulo } P)+1}$. In the rest of the paper we shall concentrate on determining a single period of a uniform WRR schedule. To further simplify the characterization, a straightforward application of the pigeonhole principle yields that if condition (1) holds then it holds with strict equality for $k = P$. Subtracting both sides of condition (1) from this common value, and defining the matrix $D_l = [d_l(i,j)]_{N\times N}$ yield the following equivalent matrix form of condition (1).

$$\sum_{l=k}^{P} D_l \geq \sum_{l=k}^{P} \Pi_l \qquad \text{for each } k = 1,2,\cdots,P. \tag{2}$$

Here matrix inequalities are understood to hold componentwise. The following example illustrates a traffic matrix and an associated uniform WRR schedule with $N=4$, $P=8$.

Example 1:

$$M = \begin{bmatrix} 3 & 2 & 2 & 1 \\ 2 & 0 & 2 & 4 \\ 0 & 4 & 2 & 2 \\ 3 & 2 & 2 & 1 \end{bmatrix}$$

$$D_8 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}, \quad \Pi_8 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix},$$

$$D_7 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \Pi_7 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix},$$

$$D_6 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}, \quad \Pi_6 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix},$$

$$D_5 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \Pi_5 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix},$$

$$D_4 = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}, \quad \Pi_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix},$$

$$D_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}, \quad \Pi_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$D_2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \Pi_2 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix},$$

$$D_1 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \Pi_1 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}.$$

We now comment on identifying a uniform WRR schedule. Note that the matrices D_l, are determined by the traffic matrix. It is possible to show that, for an arbitrary traffic matrix, a set of $P - k + 1$ permutation matrices can be extracted from $\sum_{l=k}^{P} D_l$ matrix for all k values . However, it is not clear if the extracted set of permutation matrices results in a growing sequence in the increasing values of k, and there is ongoing work in this direction. In the rest of the paper we provide three heuristic algorithms, each of which is inspired by the form of condition (2). These algorithms are based on a greedy method, termed backward extraction, that determines the matrices Π_l in decreasing order of l. The algorithms have outstanding performance as reported in Section IV. These results strongly support the possibility of the existence of uniform WRR schedules for arbitrary traffic matrices.

III. HEURISTIC ALGORITHMS

We start with an observation that motivates the heuristics studied in this paper. In the argument below *extracting* a permutation matrix from a given matrix refers to finding a permutation matrix that is componentwise no larger than the latter matrix. In particular $(\Pi_1, \Pi_2, \cdots, \Pi_P)$ is a uniform WRR schedule if and only if, by condition (2) considered for $k = P$, Π_P is extractable from D_P, and for each $k = P-1, P-2, \cdots, 1$, Π_k is extractable from the matrix $\sum_{l=k}^{P} D_l - \sum_{l=k+1}^{P} \Pi_l$.

The heuristic algorithms proposed in this paper are based on obtaining a schedule in P stages. In the first stage Π_P is chosen as a matrix that is extractable from D_P. At the $k+1$st stage a partial schedule $(\Pi_P, \Pi_{P-1}, \cdots, \Pi_{P-k+1})$ is available, so that Π_{P-k} is chosen as a permutation matrix extractable from $\sum_{l=P-k}^{P} D_l - \sum_{l=P-k+1}^{P} \Pi_l$, in a recursive manner. This procedure is called *backward extraction* since the permutation matrices are determined in the reverse order in which they are used. If a backward extraction procedure succeeds in extracting a permutation matrix at each stage, then the obtained sequence is a uniform WRR schedule.

The three variations of backward extraction presented next differ on their selection criteria among possible permutation matrices.

A. Basic Backward Extraction (BBE):

BBE is obtained simply by extracting an arbitrary permutation matrix at each stage. The full algorithm is given in Fig. 1. At each stage, nonzero entries of the matrix $\sum_{l=k}^{P} D_l - \sum_{l=k+1}^{P} \Pi_l$ are eligible for scheduling, and well known maximum matching techniques are used to extract an arbitrary permutation matrix among eligible entries, without regard to their magnitudes.

In general, multiple distinct permutation matrices can be extracted at a certain stage as can be seen in D_8 in Example 1. The selection of a particular permutation matrix at each stage however affects all later stages, and may determine whether the procedure yields a schedule at all. In particular, for the traffic matrix given in Example 1, if the first three stages result to the permutation matrices $\hat{\Pi}_8$, $\hat{\Pi}_7$, and $\hat{\Pi}_6$ below, then a permutation matrix can not be extracted in the fourth stage and so uniform WRR can not be obtained.

Input: Traffic matrix M.
Output: Either a uniform WRR schedule $(\Pi_1, \Pi_2, \ldots, \Pi_P)$ or failure.
begin
  Compute D_l, $l = 1, 2, \ldots, P$.
  Extract Π_P from D_P
  for $k = P - 1 : 1$ **do**
    Extract Π_k from $\sum_{l=k}^{P} D_l - \sum_{l=k+1}^{P} \Pi_l$
    if no such permutation matrix exists **then**
      The algorithm fails.
    end if
  end for
end

Fig. 1. Basic Backward Extraction.

$$\hat{\Pi}_8 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \hat{\Pi}_7 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix},$$

$$\hat{\Pi}_6 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

B. Balancing Service Ratios (BSR):

BSR attempts mimicking an idealized fluid analogue in which each connection receives service at constant rate. Thus in the idealized fluid model, connection (i, j) receives $m(i, j)/P$ units of service per stage. This quantity, however, is in general not an integer, in turn BSR aims to approximate this analogue.

Service ratio of a connection (i, j) at the kth stage of a backward extraction procedure refers to the ratio of number of services received by the connection by the end of the kth stage to the total number services required by the connection within the schedule length. Denote this quantity by $s_k(i, j)$ and note that $s_k(i, j) = \sum_{l=P-k+1}^{P} \Pi_l(i, j)/m(i, j)$. BSR aims to equalize the service ratios of all connections at each stage. Towards this end, at stage k a connection is considered eligible for scheduling only if

$$s_{k-1}(i, j) \leq \frac{k}{P}.$$

A permutation matrix is now extracted at each stage from the eligible entries, as described for the BBE algorithm above. The description of BSR is essentially the same as that of BBE except for the definition of eligibility.

Input: Traffic matrix M.
Output: Either a uniform WRR schedule $(\Pi_1, \Pi_2, \ldots, \Pi_P)$ or failure.
begin
  Compute D_l, $l = 1, 2, \ldots, P$.
  Set $oldest\_deadline = k = P$.
  Extract Π_P from D_P
  while $k \leq P$ **do**
    while it is possible **do**
      Extract Π_{k-1} from $\sum_{l=oldest\_deadline}^{P} D_l - \sum_{l=k}^{P} \Pi_l$,
      $k \leftarrow k - 1$,
    end while
    $oldest\_deadline \leftarrow oldest\_deadline - 1$
    if $oldest\_deadline < k - 1$ **then**
      The algorithm fails.
    end if
  end while
end

Fig. 2. Oldest Deadline First Algorithm.

C. Oldest Deadline First (ODF):

The ODF variation of backward extraction mimics the earliest deadline first principle in reverse order. Namely, for each k and partial schedule $\Pi_P, \Pi_{P-1}, \ldots, \Pi_k$, the next matrix Π_{k-1} is extracted from $\sum_{l=m}^{P} D_l - \sum_{l=k}^{P} \Pi_l$ where m is the largest integer such that the extraction is possible. Note that $m \geq k-1$ for a uniform WRR schedule. Thus in backward extraction, ODF gives priority to older deadlines. The algorithm is given in Fig. 2.

IV. NUMERICAL RESULTS

In this section, we use simulation to compare the performance of the proposed algorithms in a statistical sense. Numerical experiments are conducted as follows. For given values of switch size N and schedule length P, a number of traffic matrices with these parameters are generated randomly. Each algorithm is then applied to each one of these traffic matrices. The success rate of an algorithm is defined as the ratio of the number of matrices for which the algorithm generated a uniform WRR schedule to the total number of matrices. We measure the performance of the three algorithms in terms of their success rates. Switch sizes $N = 4, 8$, and 16 and for each size schedule lengths $P = 2N, 4N, \ldots, 40N$ are considered. For each (N, P) pair, 10000 traffic matrices are randomly generated for $N = 4, 8$, and 16. Figures 3, 4, and 5 illustrate the success rates of the three algorithms for different values of N. In all cases all algorithms had success rates larger than 95%. Algorithms BSR and ODF had very similar performances, which were notably better than that of BBE. It is intriguing to observe that the performances of all three algorithms seem to increase with increasing values of parameters N and P. This appears to be due to the richer combinatorial structure of the problem for large parameter values. For example the number of permutation matrices of size $N \times N$ increases very fast with N, and when P is large the entries of a typical traffic matrix are more or less uniform, possibly making it easier to identify a uniform WRR schedule. The exact reason seems difficult to determine. This behavior of the algorithms emphasize their merit for increasingly complex scheduling problems.

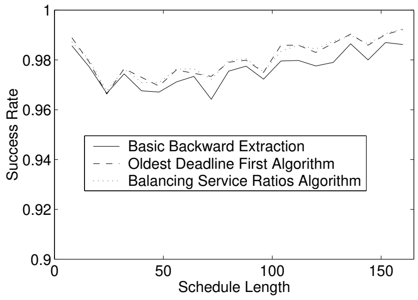

Fig. 3. Success rates of BBE, BSR, and ODF for N = 4 and 10000 traffic matrices per schedule length.

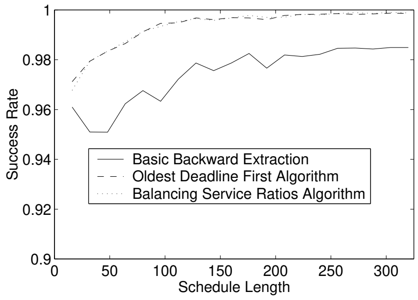

Fig. 4. Success rates of BBE, BSR, and ODF for N = 8 and 10000 traffic matrices per schedule length.

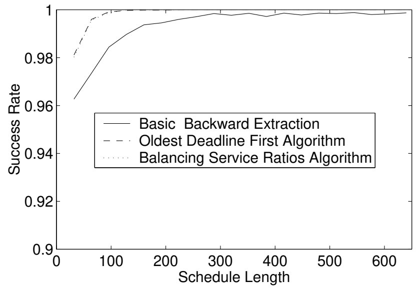

Fig. 5. Success rates of BBE, BSR, and ODF for N = 16 and 10000 traffic matrices per schedule length.

The failure of an algorithm implies that at least one deadline is missed by the obtained schedule. To obtain the number of missed deadlines we extended each algorithm by allowing deadline relaxations in which a deadline is postponed whenever it is absolutely necessary to be able to extract a permutation matrix. The percentage of deadlines which are missed by one time slot, by schedules obtained via these algorithms are given in Figures 6,7, and 8. Note that the total number of deadlines for each considered traffic matrix is PN. These percentages typically lie below 0.1%, and rapidly decrease with increasing schedule length. In neither case a fraction of more than 5×10^{-6} of the deadlines were missed by 2 time slots. No deadlines were missed by more than 2 time slots.

The proposed algorithms in this paper perform significantly better than relevant existing algorithms regardless

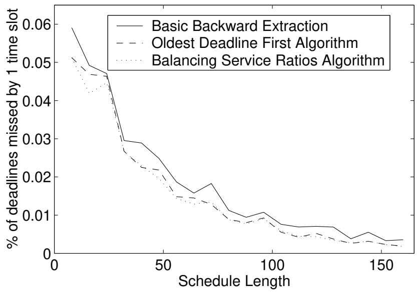

Fig. 6. Percentage of deadlines missed by 1 time slot for BBE, BSR, ODF, for N = 4 and 10000 traffic matrices per schedule length.

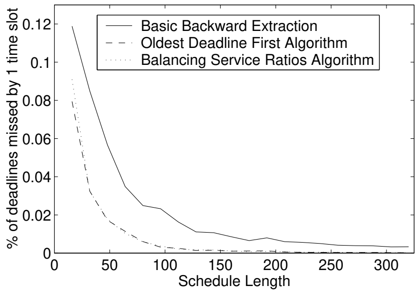

Fig. 7. Percentage of deadlines missed by 1 time slot for BBE, BSR, ODF, for N = 8 and 10000 traffic matrices per schedule length.

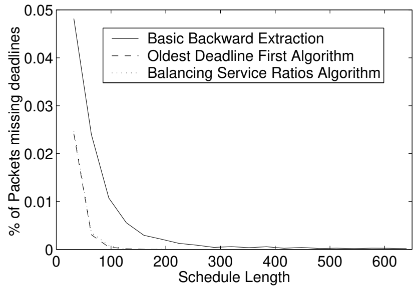

Fig. 8. Percentage of deadlines missed by 1 time slot for BBE, BSR, ODF, for N = 16 and 10000 traffic matrices per schedule length.

the fact that we consider a more general problem and a switch with link utilization of exactly 1.0. All algorithms proposed in [7] have degrading performances as switch size, schedule length, or switch utility increases. When switch size is 8, success rates of these algorithms approach zero. Likewise, success rates of algorithms in [7] for switch of size 4 get below 0.6 and 0.4 as the switch utility and schedule length approach 1.0 and 300 respectively. In [8], a scheduling algorithm is provided to obtain a feasible schedule for periodic traffic whose each period evenly divides all longer periods for a switch with link utility 1.0. For more general period, the algorithm provides a feasible schedule if link utility is less than 1/4. Another greedy algorithm which provides a schedule for arbitrary periods for a switch with link utility less than 1/14 is also provided in [8]. Finally, a heuristic algorithm proposed in [9] provides similar performance with the proposed algorithms in this paper in terms of the percentage of missed deadlines however its performance was obtained for average link utility of 0.92. Additionally, the algorithm proposed in [9] is far more complex due to its critical nodes and links tracking feature.

V. CONCLUSION

In this paper, we present a formulation of uniform weighted round robin service discipline in input queued switches, and propose efficient heuristic scheduling algorithms, namely BBE, BSR and ODF, all based on a backward extraction technique. Simulation results show that the proposed algorithms perform very well in terms of their success rates. Performances of the proposed algorithms are not negatively affected by increase in switch size, schedule length and is independent of the link utilization, as opposed to some of related previous works [7], [8]. BSR and ODF perform better that BBE. It is hard to compare BSR and ODF and their performances become very close as the switch size increases. The offline implementation of the proposed algorithms entails first obtaining the P permutation matrices, and then scheduling packet transmission according to these matrices in a cyclic manner. Online implementation involves solving one maximum matching problem at each time slot, therefore offline implementation may be more suitable for very high speed switches.

REFERENCES

[1] A. K. Parekh and R. G. Gallager, *A generalized processor sharing approach to flow control in integrated services networks: single node case,* IEEE/ACM Trans. on Networking, vol. 1, June 1993, pp. 344-357.

[2] A. K. Parekh and R. G. Gallager, *A generalized processor sharing approach to flow control in integrated services networks: multiple nodes case,* IEEE/ACM Trans. on Networking, vol. 2, April 1994, pp. 137-150.

[3] S. J. Golestani *A self-clocked queuing scheme for broadband applications,* Proc. IEEE INFOCOM'94, pp.636-646, 1994.

[4] N. Matsufuru and R. Aibara, *Efficient fair queuing for ATM networks using uniform round robin,* Proc. IEEE INFOCOM'99, pp. 21-25, March 1999, pp.389–397.

[5] T. Chuang, A. Goel, N. McKeown, and B. Prabhakar, *Matching output queuing with a combined input/output-queued switch,* IEEE Journal on Selected Areas in Communications, vol. 17, No. 6, June 1999. pp. 1030-1039.

[6] N. McKeown, A. Mekkittikul, V. Anantharam, and J. Walrand, *Achieving 100% throughput in an input-queued switch,* IEEE Trans. on Communications, Vol. 47. No. 8, Aug. 1999, pp. 1260-1267.

[7] I. R. Philp, and J. W. S. Liu, *SS/TDMA scheduling of real-time periodic messages,* Proc. International Conference on Telecommunication Systems, pp. 244-251, Mar. 1996.

[8] J. Giles, *Scheduling multirate traffic in a packet switch,* Master's thesis, Department of Electrical and Electronics Engineering, University of Illinois at Urban-Champaign, 1997.

[9] V. Tabatabaee, L. Georgiadis, and L. Tassiulas, *QoS provisioning and tracking fluid policies in input queuing switches,* Proc. IEEE INFOCOM'00, Mar. 2000.