2007 IEEE International Symposium
on Signal Processing and Information Technology

Selectively Weighted Multicast Scheduling Designs For Input-Queued Switches

Mohammed Shoaib
Department of Electrical Engineering
Indian Institute of Technology, Madras
Chennai - 600036, India.
Email: shoaib.m@iitm.ac.in

Abstract— **The objective of this work is to propose hardware-efficient schemes for multicast scheduling in input-queued switches based on the Weight Based Arbiter (WBA), motivated by the practical implementation of a scheduler for a 64-port optical crossbar switch. We demonstrate that alternating fanout- and age-based weight calculations in subsequent time slots lead to higher clock speeds and better FPGA area utilization, with performance characteristics close to the conventional WBA. Our FPGA sizing experiments and clock speed evaluations show improvements of upto 35.25% and 47.06%, respectively, over the WBA. In addition, latency–throughput results for the proposed variations highlight the trade-offs between fairness, throughput, hardware complexity and speed.**

Index Terms— **Multicast Scheduling, Weight Based Algorithm (WBA), Input Queued Switches, High Performance Computing, FIFO queues, Routing, Switching, Policy based networking.**

I. Introduction

Broadcast networks and multi-user applications have caused an increase in multicast traffic density over the internet, facilitated by MBone [5], [6]. Dense multicast traffic is also a natural result of growing demand for network services like audio and video distributions. To cope with the increasing multicast traffic volume, high performance routers must be able to handle multicast traffic with minimum hardware complexity and high efficiency. The kind of growth entails new hardware challenges and objectives in contemporary switch design.

A number of implementations and architectures have been proposed for multicast switches [7], [8]. Queueing for multicast traffic has been widely studied and various queueing schemes have been proposed. Virtual Output Queues (VOQs) for multicast traffic [11] avoid Head-of-Line (HOL) blocking to a large extent but are impractical, requiring $2^N - 1$ queues for a NxN switch. The multiple-queue architecture [14], [3] uses k queues for a NxN switch, where $1 < k \ll (2^N - 1)$, but is unable to achieve high performance or run at high speeds. FIFO queues introduce HOL blocking but are more practical and hardware efficient [15], [13]. We restrict our attention to packet-switched, crossbar-based, input-queued (FIFO) switches. Input-queued (IQ) switches usually operate on fixed-size data units called *cells*. A practical and efficient way to integrate multicast and unicast scheduling was proposed in [4].

A. Fifo queue

Shown in Fig.1 is the structure of a FIFO queue. R1, R2, and R3 are incoming multicast requests which are queued up in a FIFO fashion. The structure of request R2, shown enhanced in Fig. 1, is essentially a bitmap of N bits, N being the number of switch ports, where the bit in position i indicates whether the corresponding multicast cell requests output port i. There are N such FIFO queues corresponding to the N switch inputs.

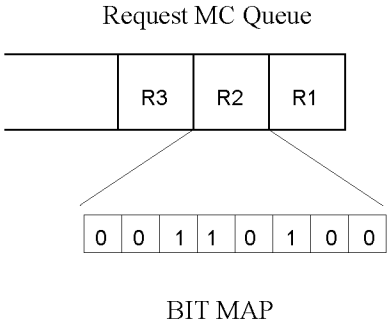

Fig. 1. Multicast FIFO Input Queue, $N = 8$.

B. Fanout

The total number of active requests in a particular bitmap constitutes the fanout of that particular cell. As the input queues are organized in a FIFO fashion, only the bitmaps at the heads of the FIFOs are considered by the multicast scheduler. Fig. 2 shows a 64×64 crossbar scheduler with FIFO queues. In the figure, queue Q_{00} has an input cell destined for the outputs 1,2,3,4 and hence the fanout of queue Q_{00} is 4 (Number of requested outputs is four. Hence the fanout is 4). Similarly, the fanout of Q_{63} is 3 - Here fanout is used as a general term to stand for both the constitution and the cardinality of the input vector.

† This work was performed while the author was at the IBM Zurich Research Laboratory, Switzerland.

† OSMOSIS is a contract for IBM-Corning to develop optically-switched interconnects for supercomputers, funded by the Department of Energy (DoE) - National Weapons Lab and The National Nuclear Security Administration (NNSA), USA

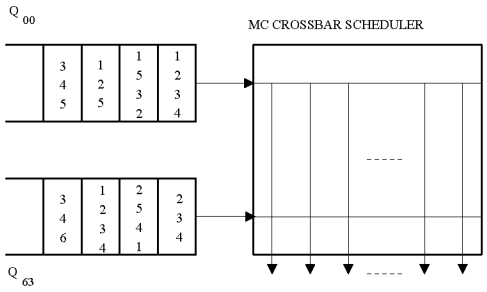

Fig. 2. 64×64 multicast crossbar switch with a single FIFO queue at every input.

C. Scheduler

A critical component of the switch is the scheduler, which in every time slot computes a matching between the inputs and the outputs and configures the crossbar. Our time slot target is 51.2 ns(256-byte cells at a line rate of 40 Gb/s [10]). Known optimum multicast matching algorithms, e.g. the Concentrate and TATRA algorithms proposed in [1], are typically complex in terms of hardware implementation. Hence, practically feasible, approximate algorithms are used. These algorithms employ independent selectors (also referred to as arbiters [12]). A popular approximate algorithm is the Weight Based Arbiter (WBA), proposed in [1]. It was shown to be hardware implementable and close to Concentrate in performance. Therefore, we focus on WBA in the rest of our discussion.

The main contribution of this work is an optimization of the WBA algorithm. Our proposed schematic variations yield sizeable improvements in hardware area and clock speed while achieving a performance close to the WBA.

Section II discusses the WBA scheduling policy and its hardware topology. Section III shows FPGA synthesis results (area occupancy and clock speeds) and simulation results of the WBA latency–throughput performance. Section IV presents the proposed alternatives to the WBA scheduling policy and Section V shows the hardware implementation results as well as performance simulations demonstrating the sizeable gains of the proposed schemes in comparison with the WBA. Finally, we conclude in Section VI.

II. The Weight Based Arbiter

An algorithm that maximizes residue concentration at the expense of fairness can starve some inputs even though it may achieve high throughput. If an algorithm aims to be fair, it may not achieve the best possible residue concentration and thereby sacrifice some throughput. The demarcation between the choice for fairness and throughput is decided based on their relative importance in a particular field in comparison with hardware complexity and performance speed. There is no room for such flexible decision making in the conventional WBA design. It is very strict in granting the input ports and follows a regular, fixed and strict methodology for weight computation. If in a design, hardware efficiency is important and not fairness or throughput, the WBA does not provide insights into the possible trade-offs with a possible bargain.

A. Motivation and objectives

The motivation for the WBA is the search for a multicast scheduling algorithm that can be implemented in an FPGA—specifically, in a Xilinx Virtex II Pro XC2VP100-6FF1704—for a 64-port switch. This requires a greater collective effort for the organization of the queued packets and provides a more broader perspective of parallelism by distribution. In the WBA design, the definition of fairness is very rigid and uniformly same for all the inputs. This kind of rule-setting does not help especially when hardware efficiency is the design constraint and not latency or throughput performance.

When the WBA was proposed in by McKeown et.al in [1], the main design objectives were the search for an alternative algorithm to the WBA, which is :

1) Simple to implement in hardware.
2) Fair in the scheduling of traffic and achieves a high throughput.

Still the primary objective was to propose a practical alternative to the concentrate algorithm, with reduced hardware complexity.

B. Operation of WBA

The operation of WBA is based on assigning weights to the input cells depending on their age and fanout at the beginning of every time slot. Once the weights are assigned, each output chooses the heaviest cell among the inputs subscribing to it. In case of multiple requests with the same weight, the scheduler breaks ties randomly. Dictated by fairness objectives, a positive weight is given to age while fanout is weighted negatively to maximize throughput. Thus, the older the cell, the heavier it is and larger the fanout, the lighter it is. Basing cell grants on age and fanout results in a compromise between extremes of pure residue concentration and strict fairness. The weight w_i of input i is computed as

$$w_i = f * \text{fanout}(\text{HOL}(i)) + a * \text{age}(\text{HOL}(i)), \qquad (1)$$

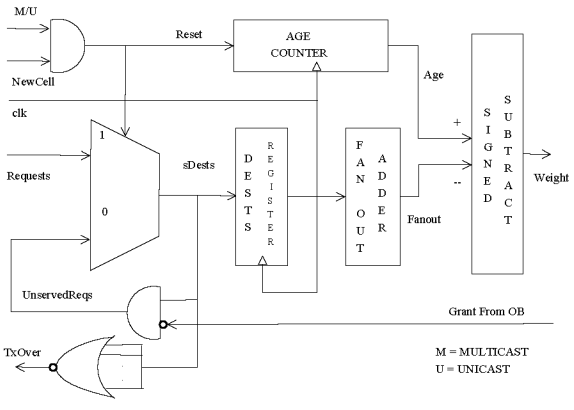

Fig. 3. 64 × 64 WBA Input Block (IB) Design Schematic.

where f and a are the weights assigned to fanout and age, respectively. It can be shown that for an $M \times N$ switch, no cell waits at the HOL for more than $M + f * N/a - 1$ cell times [1]. By means of the weights f and a, the weight calculation can prioritize either age or fanout. In particular, if we assign equal weights to age and fanout, no cell waits at the HOL for more than $M + N - 1$ cell times.

WBA employs *fanout splitting*, meaning that a cell can be served over the course of multiple time slots, where in every slot a subset of the remaining fanout is served. This is the opposite of *one-shot* scheduling, in which every cell must be served in a single slot. As fanout splitting achieves a significant performance benefit at a small cost, it is usually preferred [2], [16].

As the weight computation for an input cell only depends on local information, it can be done at each input separately and in parallel. Also, the weight comparisons at the outputs can be done in parallel. This leads to a separation of WBA into two separate sections:

- The input blocks (IB) which perform the weight computation.
- The output blocks (OB) which perform the weight comparison and grant selection.

The implementation complexity of the WBA is of order O(1). The hardware implementation is therefore simple.[1]

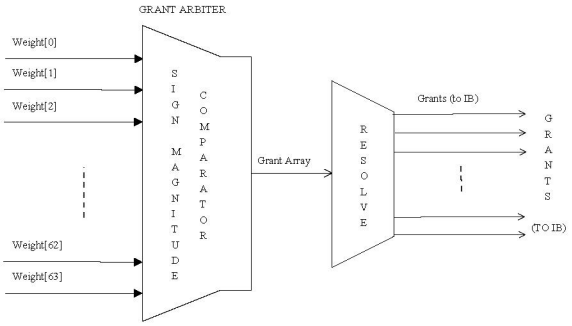

Fig. 4. 64×64 WBA Output Block (OB) Design Schematic.

1) The input block (IB): Fig. 3 shows the architecture of an IB, which has to calculate the weight for each of the input cells based on their age and fanout. The age counter is reset whenever a new cell advances to the HOL, and is incremented in every time slot until the cell has been served completely. The fanout adder determines the fanout of the HOL cell. The grants coming from the OBs are fed back to the IBs to update their age and fanout values for the next time slot.

2) The output block (OB): For a MxN switch, the output block has N comparators (corresponding to the N output ports), each with M-inputs which takes in an array of M weights, each of length $(log_2 N + 2)$. In every time slot, each comparator identifies the highest weight forwarded by the input blocks and grants the input with the highest weight denying all other requests. See Fig.5 and Fig.4.The grant vectors coming from each of the N comparators (corresponding to the N output ports) are then rearranged into grants for the input blocks and then fed-back to them to recompute the weights for a new operation cycle.

C. Topology

To reduce implementation complexity, an input cell must wait in line until all the cells ahead of it have gained access to all of the outputs that they have requested. As discussed, the WBA has an input block and an output block. There are numerous simplifying features included in the WBA. The connectivity of the scheduler is shown in Fig.5. The topology of the broadcast algorithm is loop-back type and hence needs resolution of grants sent from the OBs to the IBs.

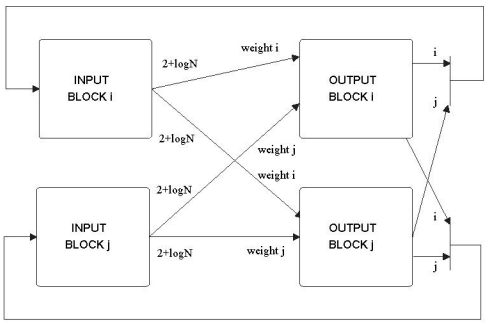

Fig. 5. $N \times N$ WBA scheduler connection details – Connecting N IBs and N OBs forming an $N \times N$ WBA Scheduler.

III. Implementation Results

A. FPGA synthesis

The motivation for this synthesis exploration is the implementation of a crossbar scheduler for a 64-port optical switch demonstrator (called OSMOSIS) with 40-Gb/s ports and a predefined scheduling cycle (time slot) of 51.2 ns for high performance computing applications [9], [10]. A major challenge in this project is to implement a 64-port scheduler with this time slot duration in FPGAs, which were used for minimizing cost and improving the flexibility of the design.

The FPGA synthesis results, shown in Table I, show that the 64-port WBA scheduler fits in the target FPGA device, nearly saturating it (84% full). The device used for implementation was "xc2vp100-6ff1704", a Xilinx Virtex-II Pro series FPGA with 8 M system gates (100 K logic cells†) and 1040 User IOs.

B. Performance simulations

The latency–throughput simulation results are shown in Fig.6. The mRRM (multicast Round Robin Matching) scheme is a simple multicast round robin arbiter granting the requesting cells in a round robin fashion. This is the simplest in implementation complexity but has poor latency–throughput performance. The Concentrate algorithm [1] gives the best performance score and the mRRM the worst. WBA has much lower implementation complexity than Concentrate, yet has comparable performance. On the other hand, WBA clearly outperforms mRRM.

†Virtex logic cell = One, 4-Input LUT + One, Flip Flop + Carry Logic. One Virtex Slice = Two Virtex logic cells.

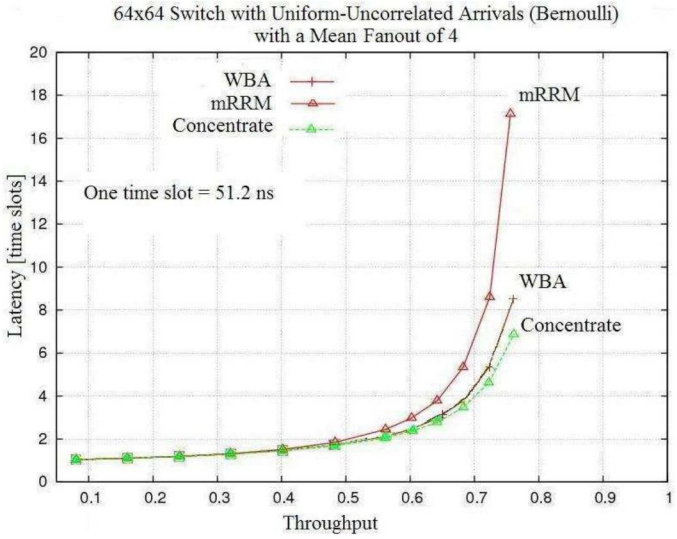

Fig. 6. Latency–throughput performance of the WBA scheduler.

C. Issues with WBA

Table I shows the FPGA implementation results for the structural design of the WBA. The structural design has a clock speed of 27.927 ns for a 64×64 switch and occupies 83.96% (Table.I) of the chip. When the same design was implemented behaviorally, the WBA fills up the chip rapidly, allowing a maximum implementable switch size of only 40-ports. This occupancy (even of the structural design) leaves little room for additional logic for data serializing and deserializing (*SerDes*) and signal resolution circuitry to be implemented along with the design. In Sec. IV we explore further optimizations of WBA to achieve a clock period reduction by about 15-20% and about 10% area reduction, while maintaining a level of performance comparable to the basic WBA as shown in Fig. 6.

IV. Proposed WBA Variations

Design enhancement explorations of the WBA scheme suggest alternate scheduling strategies to be potential performance enhancers, where the weight estimation methodology is altered to choose between simpler subsets of weight components. The calculation of weights in the WBA led to its IB having the hardware structure described in Fig. 3. The signed subtractor in the IB subtracts the fanout from the age of a particular input port to determine the weight of that port. This subtractor could be potentially substituted for selective weight estimation between age and fanout rather than using both of them in every cycle. There are 64 such subtractors at the head of the 64 IBs. Hence, subtractor optimization (replacement) could be a substantial performance enhancer.

TABLE I
NxN WBA scheduler – FPGA synthesis results in Xilinx Virtex-II-Pro[speed grade-6].

N	2	4	8	16	32	64
# slices	22	124	538	1995	8399	37024
% slices	0.05	0.28	1.22	4.52	19.05	83.96
Min. clock period (ns)	2.023	5.173	10.377	12.891	18.96	27.927
Max. clock freq. (MHz)	494.31	193.31	96.37	77.57	52.74	35.81

A. The OCF Scheme

The Oldest Cell First (OCF) scheme uses weights based only on the HOL cell age, i.e., w_i = age(HOL(i)). As a result, the subtractor at the head of the IBs is no longer needed. This scheme clearly simplifies the architecture of the WBA design, but has other repercussions on the scheduling, which are discussed in the following sections.

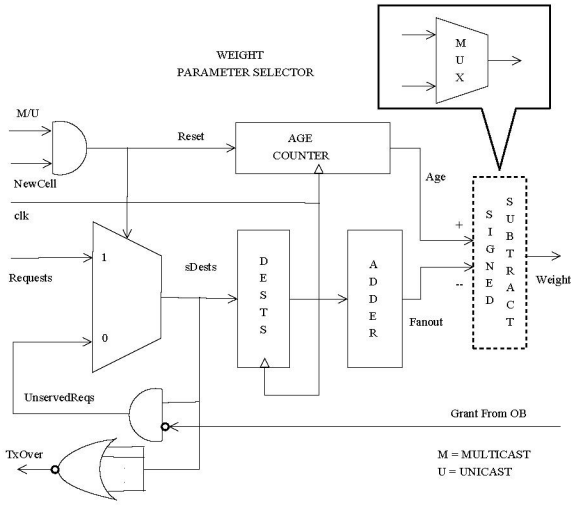

Fig. 7. OCF, LFF and AF schemes – Replace the signed subtractor with the simple weight parameter selector operating during every cell cycle in the IB.

B. The LFF Scheme

The Largest Fanout First (LFF) scheme uses weights based only on the HOL cell fanout, i.e., w_i = fanout(HOL(i)). Hence, the age counter and the subtractor in the IB are removed. Another advantage of OCF and LFF is that the weights are one bit less, so the OB has a comparator instead of a more complex sign-magnitude comparator (Now we have weights of $(log_2 N + 1)$ bits instead of $(log_2 N + 2)$ bits earlier, where the MSB was for the sign of the weight which could have been negetive because of the subtraction of age and fanout) The LFF design is as simple as the OCF but has other implications.

C. Fairness and performance issues

The main concern in the OCF and the LFF schemes is performance. In Sec. V we will see that there are substantial gains in terms of lower logic complexity and higher clock speeds, but the schemes are either not fair or less performing in thoughput. LFF tends to be unfair in not serving all the queues

with equal consideration (the FIFO with the least fanout may dominate and starve the other ports at the HOL) and the OCF reduces the throughput performance of the scheduler. The issue is the trade-off between performance characteristics and hardware simplicity - The WBA was also a result of trade-offs and is not better performing than the concentrate algorithm but simplifies complexities to a large extent. The above schemes are a step forward in this simplification process.

D. Mixed AF Design

1) 1A1F: To balance performance and complexity, we propose to mix the OCF and LFF schemes. Instead of using both parameters during every time slot as in WBA, our scheme uses age and fanout separately in subsequent time slots in an alternating fashion. We refer to this scheme as 1A1F, i.e., in one slot, age is used and in the next one, fanout.

The investment, compared to OCF or LFF, is that we have to have additional hardware to multiplex the age or fanout in specific cycles to serve as the weights to be forwarded to the OB, see Fig. 7. An acceptable level of fairness and throughput is achieved with less complexities and higher speed than WBA. The signed subtractor is replaced by a simpler multiplexer and the sign-magnitude comparator of $(log_2 N+2)$ bits in the WBA is replaced by an unsigned, ordinary comparator of $(log_2 N+1)$ bits. The reduction in the number of bits is explained by the fact that in the bitmap coming out of the IB corresponding to the weight (age and fanout included), the most significant bit is the sign bit obtained after subtracting the fanout from the age of that cell, which may lead to negative weights. Hence, the comparator in the OB is required to include the most significant bit during the comparison.

2) 1A3F: Fig. 10 demonstrates that the latency–throughput performance of OCF is significantly worse than LFF. Therefore, we can expect improved performance by using fanout more often than age; for instance, in a three to one ratio. This exploits the good latency–throughput performance of LFF, while ensuring no port is starved indefinitely. This scheme is referred to as 1A3F. The latency–throughput performance and hardware utilization are very close to WBA.

V. Hardware Implementation Results

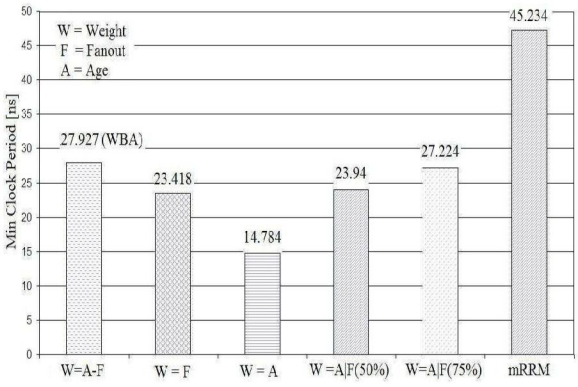

Fig. 8. Comparative Minimum Clock periods for the designs.

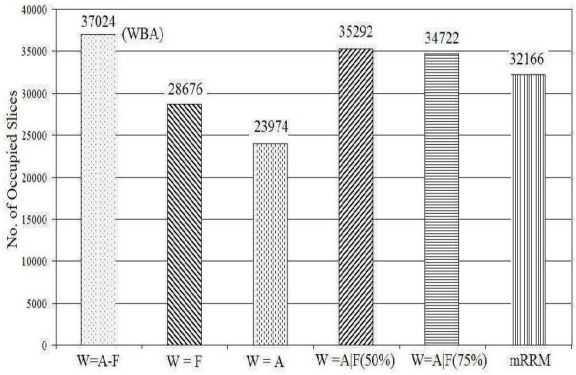

Fig. 9. FPGA area occupancy.

A. FPGA synthesis

The implementation results of the proposed schemes are shown in Table II, Fig. 8 and Fig. 9. OCF gives the lowest FPGA area occupancy (54.37%) and best clock speeds (14.784 ns), an improvement of 35.25% in area and 47.06% in clock speed over WBA. LFF yields improvements of 22.55% and 16.14% in area and speed respectively. The implementation of 1A1F results in gains of 6.22% in area and 14.28% in speed over WBA, see Fig. 9. In the synthesis behaviour there are additional routing overheads and other signal resolution issues that disallow direct interpolation of the results from one scheme to another.

TABLE II
64 × 64 MULTICAST SCHEDULER – FPGA SYNTHESIS RESULTS IN XILINX VIRTEX-II-PRO[SPEED GRADE-6]. A = AGE, F = FANOUT

64 × 64	WBA	LFF	OCF	1A1F	1A3F	mRRM
# Slices	37024	28676	23974	35292	34722	32166
% Slices	83.96%	65.03%	54.37%	80.03%	78.74%	72.94%
Min. clock period (ns)	27.927	23.418	14.784	23.94	27.224	47.234
Max. clock freq. (MHz)	35.81	42.70	67.64	41.77	36.73	21.17

B. Performance simulations

Fig. 10 shows that OCF is very poor in performance and that the latency–throughput characteristics of LFF are practically identical to WBA. Hence, a delicate balance between them is the way to better performance. The 1A1F and 1A3F schemes achieve this, having latency–throughput curves in between the OCF and LFF curves. The 3A1F, 1A1F, and 3A1F curves are all in between OCF and LFF. Moreover, the larger the relative frequency of using fanout as opposed to age, the closer the curve shifts to LFF. In doing this, we include OCF occasionally, so the scheduler does not starve any of the contending input ports. The WBA has strict fairness and throughput constraints even for applications which may not require such strictness. The xAyF-design is more flexible,

allowing application-specific customization of weight computation.

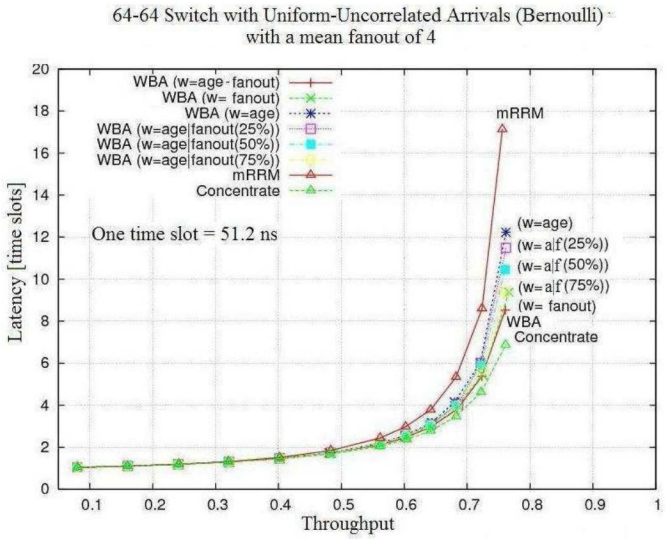

Fig. 10. Latency vs Throughput performance of the Age-Fanout WBA Schemes.

VI. Conclusion

The WBA scheme proposed in [1] for scheduling of multicast cells was synthesized on the Xilinx Virtex II Pro FPGA. The scheduler had an area occupancy of 83.96% and a minimum clock period of 27.927 ns for the 64×64 switch. Improvement of clock speeds by 15-20% and reduction in FPGA area by at least about 10% led us to investigate compromises between strict fairness and throughput constraints incorporated in the WBA design. The main contribution of this paper is that a hybrid weight calculation methodology with relaxed constraints, which achieves substantial area reductions (up to 35.25%) and improved clock speeds (up to 47.06%) with respect to WBA. The schemes proposed use the age and fanout of the HOL cells separately in a configurable number of subsequent cycles, instead of using both of them in every cycle. This reduces the hardware complexity of WBA, making it more practical to implement. There is an acceptable trade-off between performance and complexity making it a practical scheduler choice in multicast scheduling.

Acknowledgment

The author would like to extend special thanks to and acknowledge the help of François Abel and Cyriel Minkenberg, Research Staff at the IBM Research GmbH, Zurich Research Laboratory, CH-8808, Rüschlikon,Switzerland for their assistance in simulations and helpful suggestions in the manuscript.

References

[1] B. Prabhakar, N. McKeown, R. Ahuja, "Multicast scheduling for input queued switches," *IEEE J. Sel. Areas Commun.*, vol. 15, no. 5, pp. 855–866, June 1997.

[2] J.F. Hayes, R. Breault, and M. Mehmet-Ali, "Performance analysis of a multicast switch," *IEEE Trans. Commun.*, vol. 39, no. 4, pp. 581–587, Apr. 1991.

[3] S. Gupta and A. Aziz, "Multicast scheduling for switches with multiple input queues," in *Proc. Hot Interconnects*, pp. 28-33, 2002.

[4] E. Schiattarella and C. Minkenberg, "Fair integreated scheduling of unicast and multicast traffic in an input queued switch," in *Proc. IEEE ICC*, Istanbul, Turkey, 2006.

[5] H. Eriksson, "MBone: The Multicast Backbone," *Communications of the ACM*, vol 37 (no.8) pp.54–60, August 1994.

[6] V. Paxson, "Growth trends in wide-area TCP connections," *IEEE Network*, vol. 8, no. 4,pp. 8–17, July-Aug 1994

[7] T.T. Lee, "Non-Blocking copy networks for multicast packet switching," *IEEE J. Sel. Areas Commun.*, vol. 6,no. 5,pp. 1455–1467, Dec. 1988

[8] J.S. Turner, "Design of a broadcast switching network," *Proc. IEEE INFOCOM*, pp.667–675, 1986.

[9] R.P. Luijten, C. Minkenberg, B.R. Hemenway, M. Sauer, and R. Grzybowski, "Viable opto-electronic HPC interconnect fabrics," *Proc. CM/IEEE SC2005 Conference on High-performance Networking and Computing*, Nov. 12–18 2005, Seattle, WA, USA, p. 18.

[10] B.R. Hemenway, R.R. Grzybowski, C. Minkenberg, R.P. Luijten, "An optical packet-switched interconnect for supercomputer applications," *OSA J. Opt Netw*, vol. 3, no. 12, pp. 900–913, Oct. 2004.

[11] M. Karol, M. Hluchyj, and S. Morgan, "Input versus output queueing on a space division switch," *IEEE Trans. Commun.*, vol. 35, no. 12, pp. 1347–1356.

[12] N. McKeown, "iSLIP scheduling algorithm for input-queued switches," *IEEE Trans. Netw.*, vol. 7, no. 2, pp 188–201, Apr. 1999.

[13] M. Andrews, S. Khanna, and K. Kumaran, "Integrated scheduling of unicast and multicast traffic in an input-queued switch," in *Proc. IEEE INFOCOM*, pp. 1144–1151, 1999.

[14] A. Bianco, E. Leonardi, F. Neri, C. Piglione, and P. Giaccone, "On the number of input queues to efficiently support multicast traffic in input queued switches," in *Proc. IEEE HPSR*, pp. 111–116, Jun. 2003.

[15] N. McKeown, "A fast switched backplane for a gigabit switched router," *Business Commun. Rev.*, vol. 27, no. 12, 1997.

[16] J.Y. Hui and T. Renner, "Queueing analysis for multicast packet switching," *IEEE Trans. Commun.*, vol. 42, no. 2/3/4, pp. 723–731, Feb. 1994.