Available online at www.sciencedirect.com

ScienceDirect

Performance Evaluation 64 (2007) 1053–1061

PERFORMANCE EVALUATION
An International Journal

www.elsevier.com/locate/peva

A distributed switch scheduling algorithm[☆]

Petar Momčilović

Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48105, United States

Available online 29 June 2007

Abstract

The maximum weight matching algorithm is a high-performance scheduling algorithm for cross-bar switches. It is known that it performs optimally under heavy loads. However, its centralized nature and high computational complexity limit the algorithm's applicability. This paper presents a randomized algorithm for distributed switch scheduling that is capable of delivering high throughput.

Keywords: Cross-bar switch; Distributed scheduling; Randomized algorithm

1. Introduction

The virtual output queued (VOQ) switch architecture has emerged as the architecture of choice for high speed switches. This is primarily due to the high speed memory requirement for output queued switches and the fact that pure input queued switches suffer from performance degradation due to head-of-line (HOL) blocking [1]. On the other hand, the VOQ architecture is based on a switching fabric operating at line speeds and does not suffer from HOL blocking. The reduced hardware requirements and improved performance come at the cost of increased control complexity. Namely, the key performance metrics (throughput, delay, etc.) crucially depend on the employed scheduling algorithm. It has been shown that a greedy algorithm performs poorly under certain traffic scenarios [2].

It is known that the Maximum Weight Matching (MWM) algorithm can provide high throughput and low delay [3]. The algorithm operates as follows. The inputs and outputs are represented by vertices in a bipartite graph. Each input–output pair (i, j) (edge) is assigned a weight that is a measure of congestion, e.g., a number of packets awaiting transmission from input i to output j. The algorithm selects an independent set of input–output pairs with the highest sum weight. Packet transmissions are scheduled between the pairs in this set, i.e., a maximum weight matching. In [3,4] it was shown that the MWM algorithm is stable with Bernoulli arrival processes when no input or output is overloaded; the result was extended for more general arrival processes in [5]. Furthermore, the MWM algorithm with appropriately chosen weights performs optimally under the heavy load scenario [6,7]; the cross-bar switch considered here is a special case of the generalized switch[1] examined in [7]. MWM-like algorithms were also argued to perform well when applied in the network of constrained queues [8] and switches [9].

[☆] This work was supported by the National Science Foundation under grant CNS-0643213.
E-mail address: petar@eecs.umich.edu.

[1] In the generalized switch data rates between inputs and outputs might not be uniform and fixed as in the cross-bar switch.

0166-5316/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.peva.2007.06.019

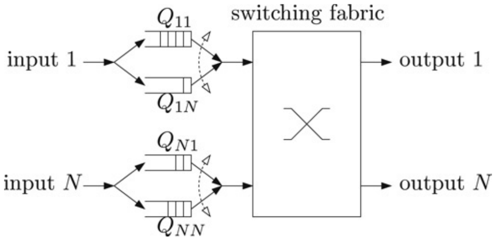

Fig. 1. The structure of a virtual output queued switch. Packets are queued at inputs based on their destinations. Each input and output can be connected to at most one output and input, respectively, at the same time.

However, computing maximum weight matchings at line speeds presents significant challenges. The best known algorithms [10–12] either are of significant complexity or require a large number of arithmetic operations when the number of inputs/outputs is large. Hence, a number of practical algorithms were developed, including iSLIP [13], MUCS [14] and RPA [15]. These algorithms do not attempt to approximate the MWM algorithm explicitly and are inferior to the MWM algorithm when the input traffic is not uniform [16]. On the other hand, algorithms based on MWM approximations were also considered, see e.g. [17–19]. They are based on an observation that queue sizes exhibit correlations in time, and, hence, matching weights do not change significantly over small time intervals. Recently, an algorithm based on an auction algorithm was presented in [20].

The problem of switch scheduling can be viewed as an instance of a general problem of scheduling in interference graphs. In the context of ad hoc wireless networks low-complexity distributed algorithms are of interest. Namely, local back-pressure policy [8], the Longest Queue First (LQF) policy [21] and maximal scheduling were considered in the literature. The last scheme is particularly desirable due to its simplicity — packets are scheduled in an arbitrary way as long as it is feasible. However, this approach suffers from a significant performance degradation [22,23] in comparison with the MWM algorithm. For further studies of distributed scheduling in wireless networks see [24–26] and references therein.

2. Model

The structure of the VOQ switch is shown in Fig. 1. Packets arriving to the input i with the destination j are enqueued in a virtual output buffer (i, j) of infinite size. The number of packets in this buffer at time t is denoted by $Q_{ij}(t)$. All packets are of fixed size, time is slotted and it takes a time slot to transmit (switch) a packet from an input to an output. Let $A_{ij}(t)$ be the number of packet arrivals to the queue (i, j) at discrete time t. The time evolution of queue lengths is governed by the following recursion

$$Q_{ij}(t+1) = Q_{ij}(t) - S_{ij}(t) + A_{ij}(t+1), \tag{1}$$

where $S_{ij}(t) = 1$ if the queue (i, j) is served in time slot t and $Q_{ij}(t) > 0$, i.e., if a packet is switched from input i to output j; otherwise $S_{ij}(t) = 0$. Event $\{S_{ij}(t) = 1\}$ indicates that the switching fabric is configured in such a way that in time slot t one packet is transferred from input i to output j. The switching fabric operates under the constraint that at most one packet can be switched from each input and that at most one packet can be switched to each output, i.e.,

$$\sum_{i=1}^{N} S_{ij}(t) \leq 1, \qquad \sum_{j=1}^{N} S_{ij}(t) \leq 1. \tag{2}$$

A scheduling algorithm determines the connectivity pattern provided by the switching fabric. Equivalently, the algorithm sets the values of $\{S_{ij}(t)\}$ subject to (2). Hence, in view of (1), the impact of the scheduling scheme on the queue sizes is through $\{S_{ij}(t)\}$.

It is assumed that sequences $\{A_{ij}(t)\}_{t\geq 0}$ are i.i.d. and independent of each other with $\max A_{ij}(t) = A < \infty$. The expected number of arrivals to queue (i, j) within one time slot is given by $\lambda_{ij} = \mathbb{E}A_{ij}(t)$. The input traffic is called admissible if the traffic intensities $\{\lambda_{ij}\}$ satisfy

$$\lambda_{\cdot j} := \sum_{i=1}^{N} \lambda_{ij} < 1, \qquad \lambda_{i\cdot} := \sum_{j=1}^{N} \lambda_{ij} < 1, \tag{3}$$