Cooperative Token-Ring Scheduling For Input-Queued Switches

IEEE TRANSACTIONS ON COMPUTERS, MARCH 2009

Authors: Amir Gourgy, and Ted H. Szymanski

Presented by Hsi-Min Lin

4/9, 2009

OUTLINE

- INTRODUCTION
- CLASSICAL ROUND-ROBIN ARBITERS
- HOL and VOQ
- COOPERATIVE TOKEN-RING SCHEDULING
- SIMULATION
- CONCLUSIONs

INTRODUCTION

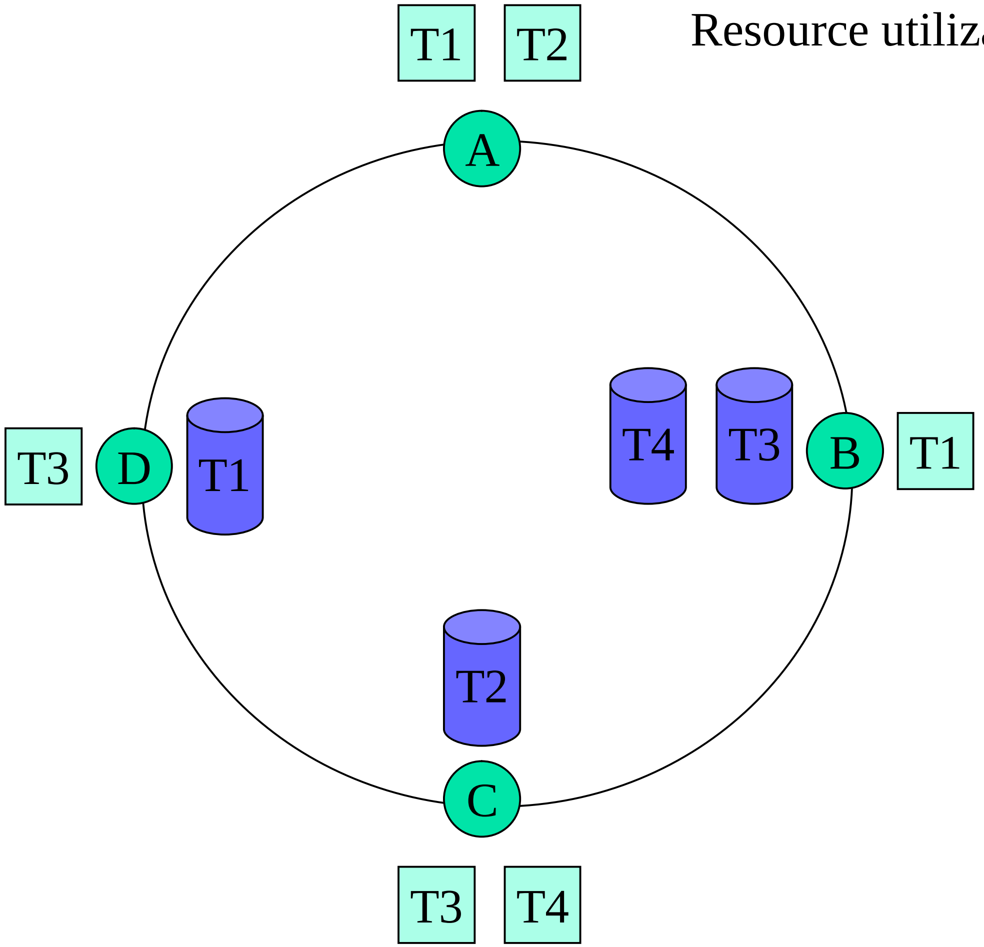
- In an $N \times N$ switch
  - Scheduling can be **optimally** solved using a maximum-weight matching algorithm [5], it requires a runtime complexity of **θ (N^3)**.
  - Most practical algorithms are based on simple heuristics that aim at **maximizing the number of connections between inputs and outputs** and attempt to achieve a maximal match.
    - require N iterations in the worst case (hardware time complexity of **θ (logN)** per iteration).
    - Iterative for maximal matching → Cells transferred → Iterative for maximal matching → …

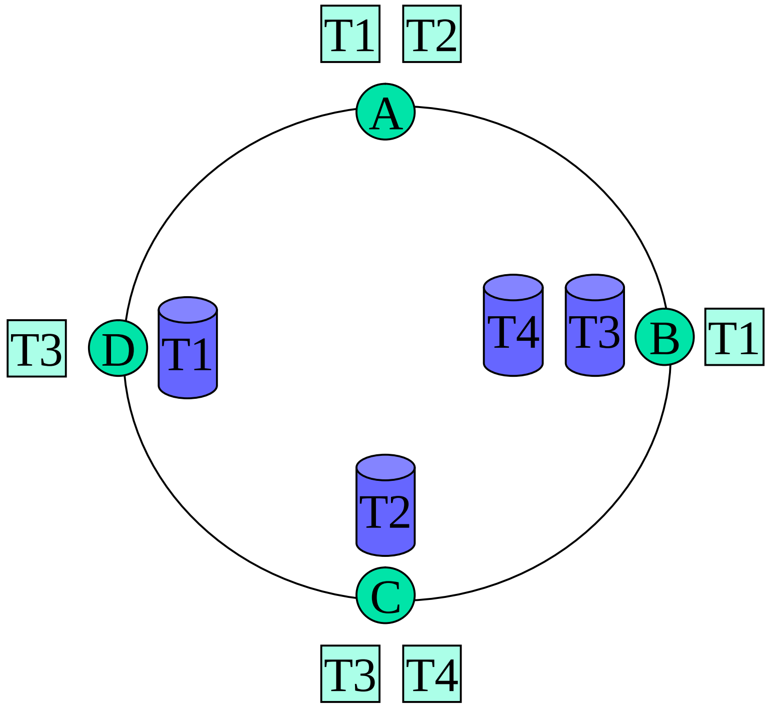
INTRODUCTION

- Most scheduling schemes could be viewed as employing **traditional token rings**
  - **Nodes** in the ring correspond to **input arbiters.**
    - that perform the request and accept phases and where
  - **Tokens** correspond to **output ports**.
  - A token that is **acquired** by a node corresponds to an input port being **matched** to an output port.
- Good throughput for uniform traffic, but the performance degrades for **realistic nonuniform traffic**.

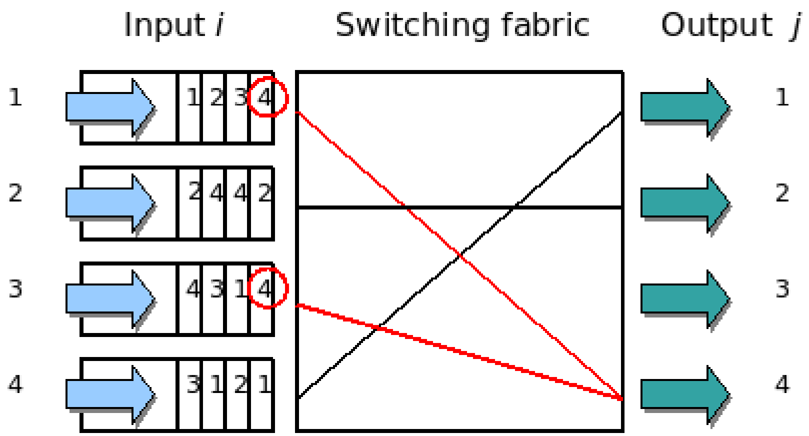
CLASSICAL ROUND-ROBIN ARBITERS

Resource utilization is 50%

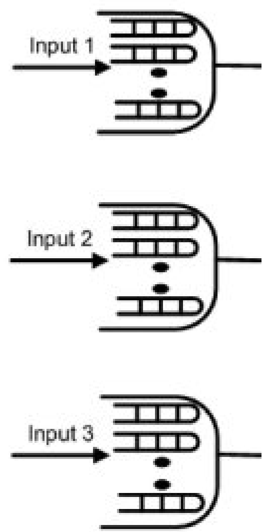




HOL and VOQ



Head-of-line blocking (HOL)



virtual output queuing (VOQ) architecture

COOPERATIVE TOKEN-RING SCHEDULING (CTR)

- In summary, it is a challenge to find a scheduling scheme for IQ switches that meets the following requirements
  - It provides high throughput, **essentially 100 percent**, for both **uniform and nonuniform** traffic.
  - It provides rate guarantees for QoS traffic and proportional bandwidth sharing.
  - It is **readily implemented in hardware**.
    - Most practical schedulers are iterative with a hardware time complexity of θ **(logN)** per iteration, where N is the size of the switch;

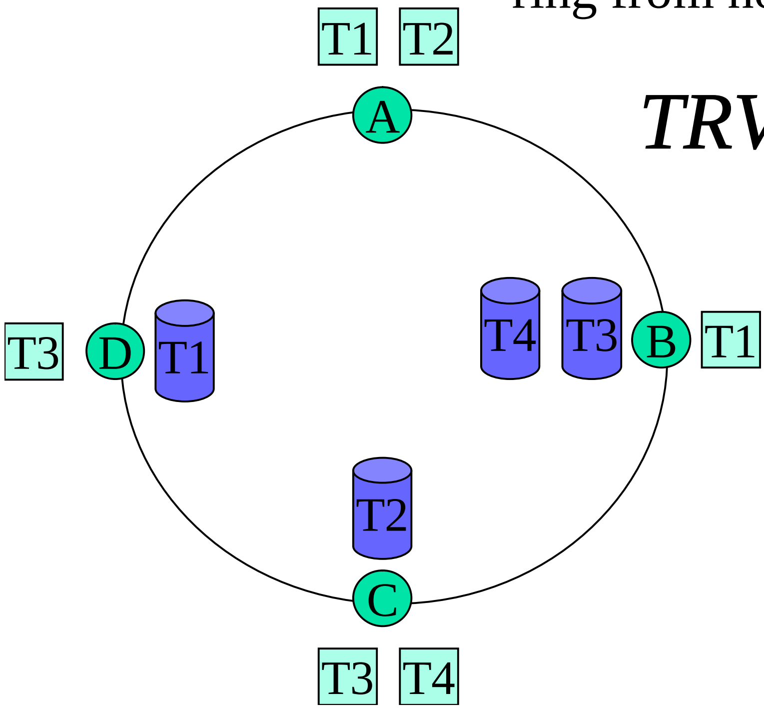
COOPERATIVE TOKEN-RING SCHEDULING (CTR)

- CTR is an **iterative scheme** such that each iteration comprises two phases.
  - Computing the TRV (Token Request Vector)
    - Create TRVs along the ring for each token.
  - Token propagation
    1. Improve a node's resource utilization.
    2. Improve the overall resource utilization of the ring

COOPERATIVE TOKEN-RING SCHEDULING (CTR)

- Computing the TRV (Token Request Vector)
  - $TRV_{i,j}$: the value of the TRV at node i for token j.
    - **True** = Exist an **unmatched downstream node** along the ring from node i that requests token j.

$$TRV_{i,j} = R_{|i+1|,j} + \sum_{t=i+2}^{t=i+N-1} \left(R_{|t|,j} \prod_{k=i+1}^{k=t} \overline{TP_{|K|,j}} \right)$$



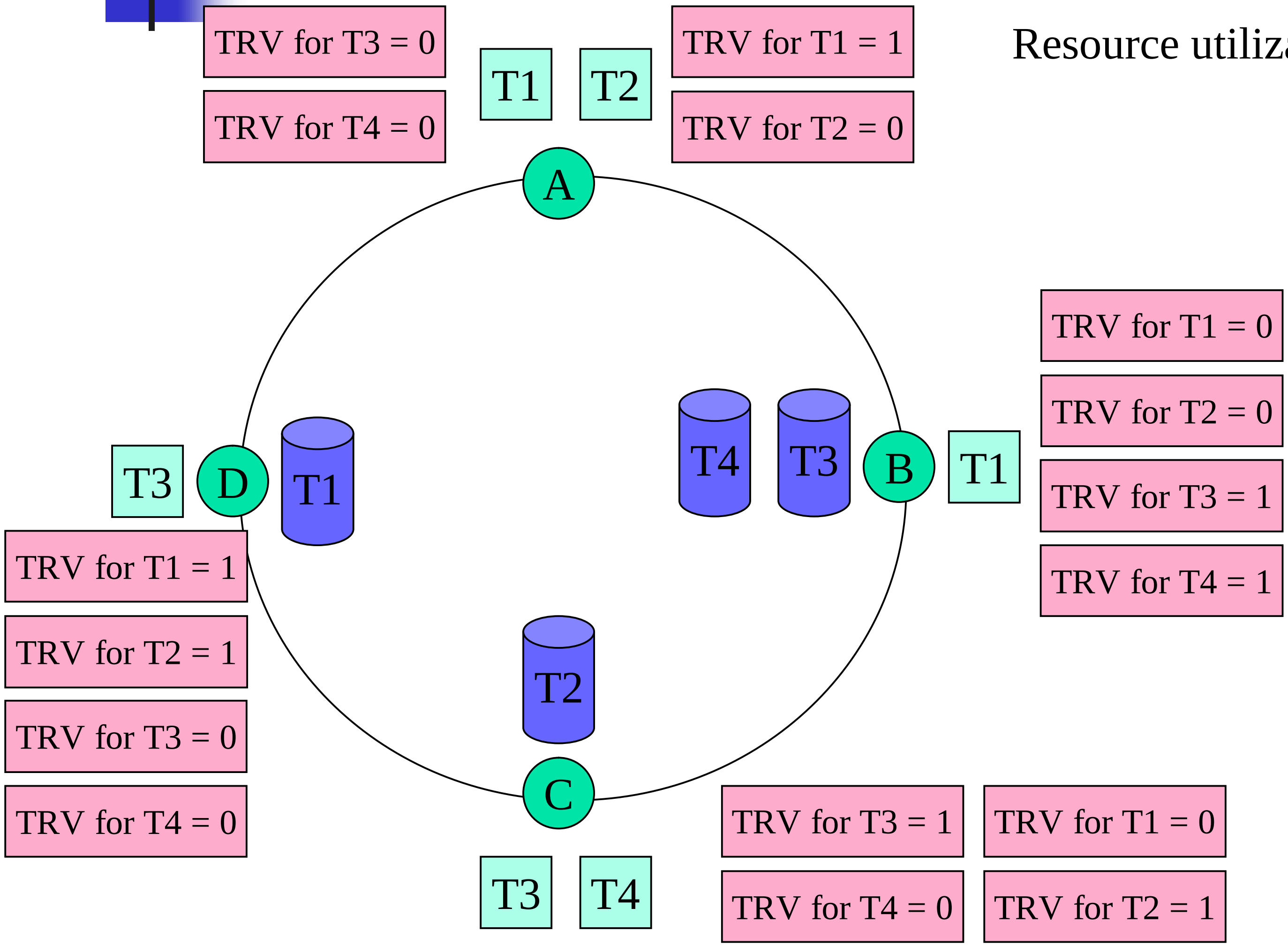
- $TRV_{A,T1} = 1$
- $TRV_{A,T2} = 0$
- $TRV_{A,T3} = 0$
- $TRV_{A,T4} = 0$

COOPERATIVE TOKEN-RING SCHEDULING (CTR)

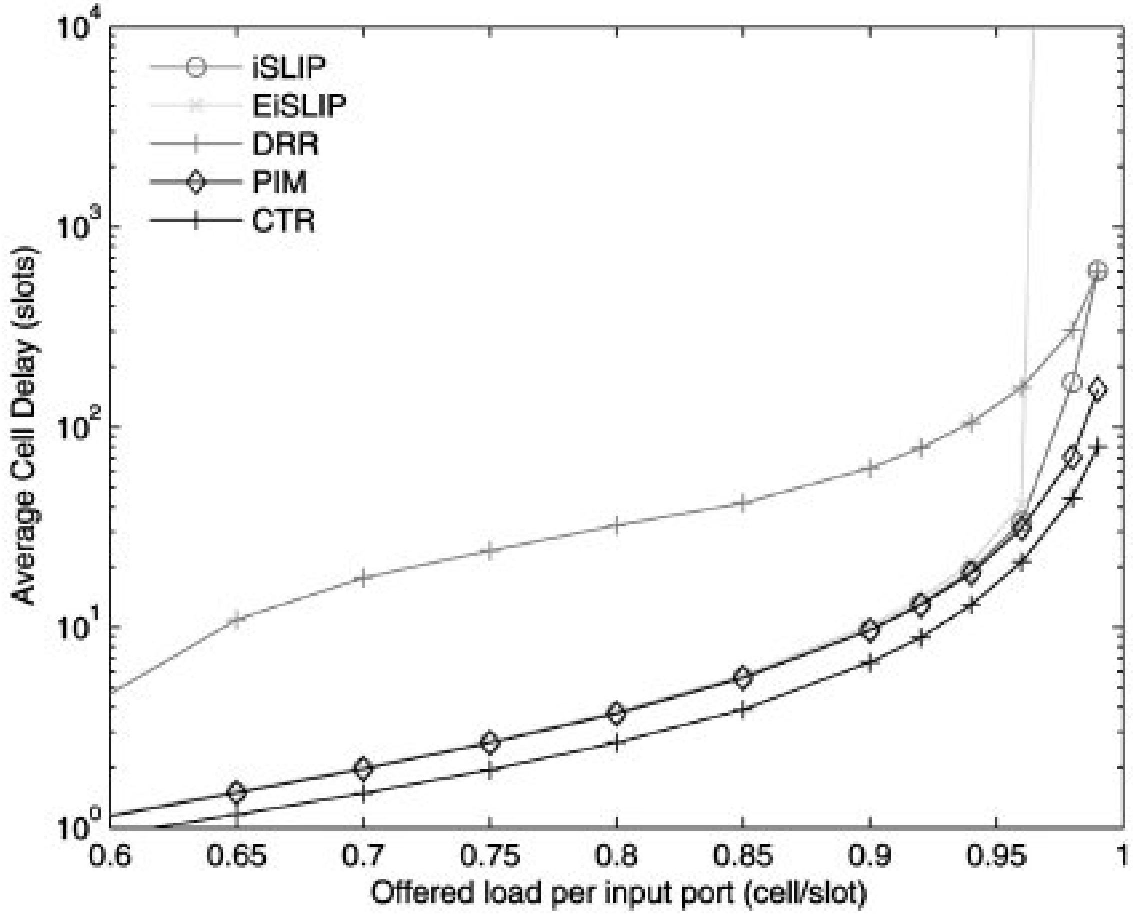
- Token propagation
  1. Improve a node's resource utilization.
     - An input that is **not matched** acquires the first available token regardless of whether this token is critical or not—this ensures that the matching converges.
  2. Improve the overall resource utilization of the ring
     - The acquisition of a critical token takes **precedence** over the acquisition of a noncritical token.
       - An acquired token is swapped for a critical token, when possible.

COOPERATIVE TOKEN-RING SCHEDULING (CTR)

Resource utilization is essentially 100%

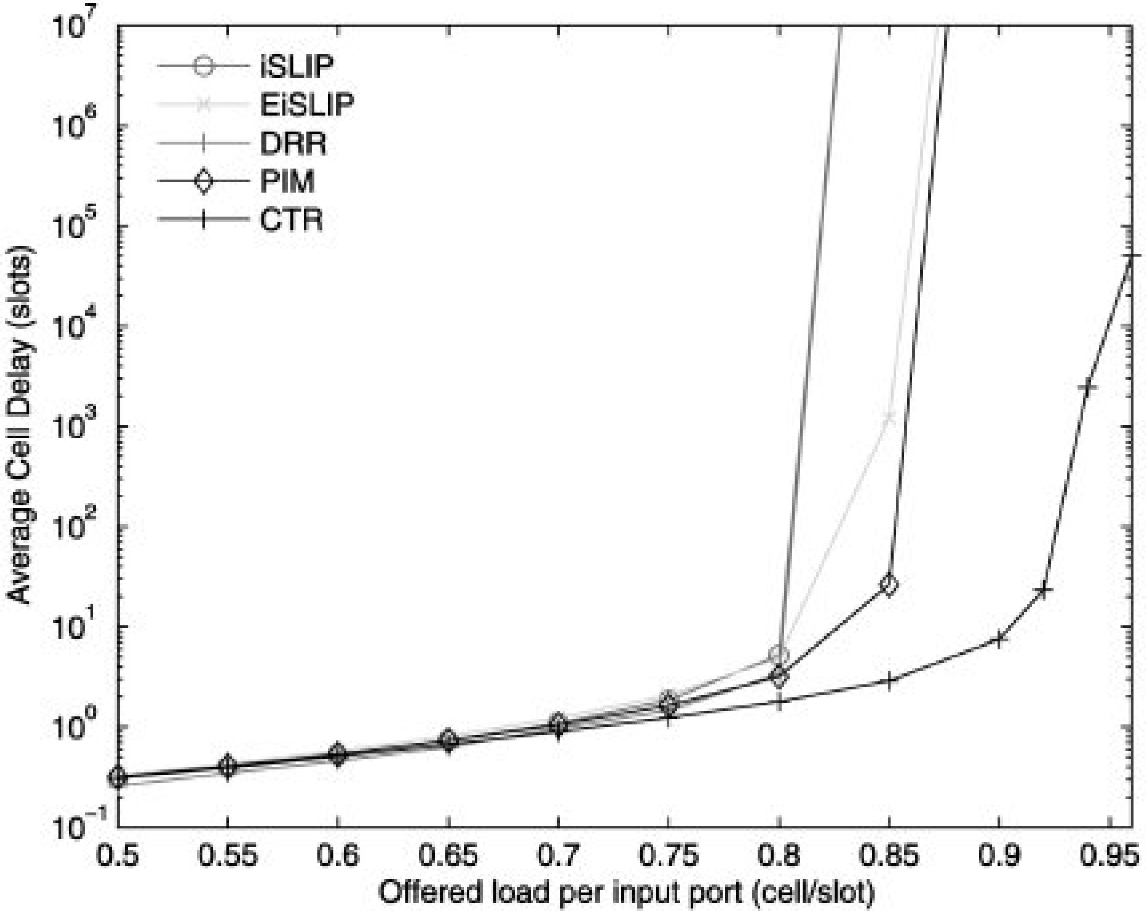


SIMULATION



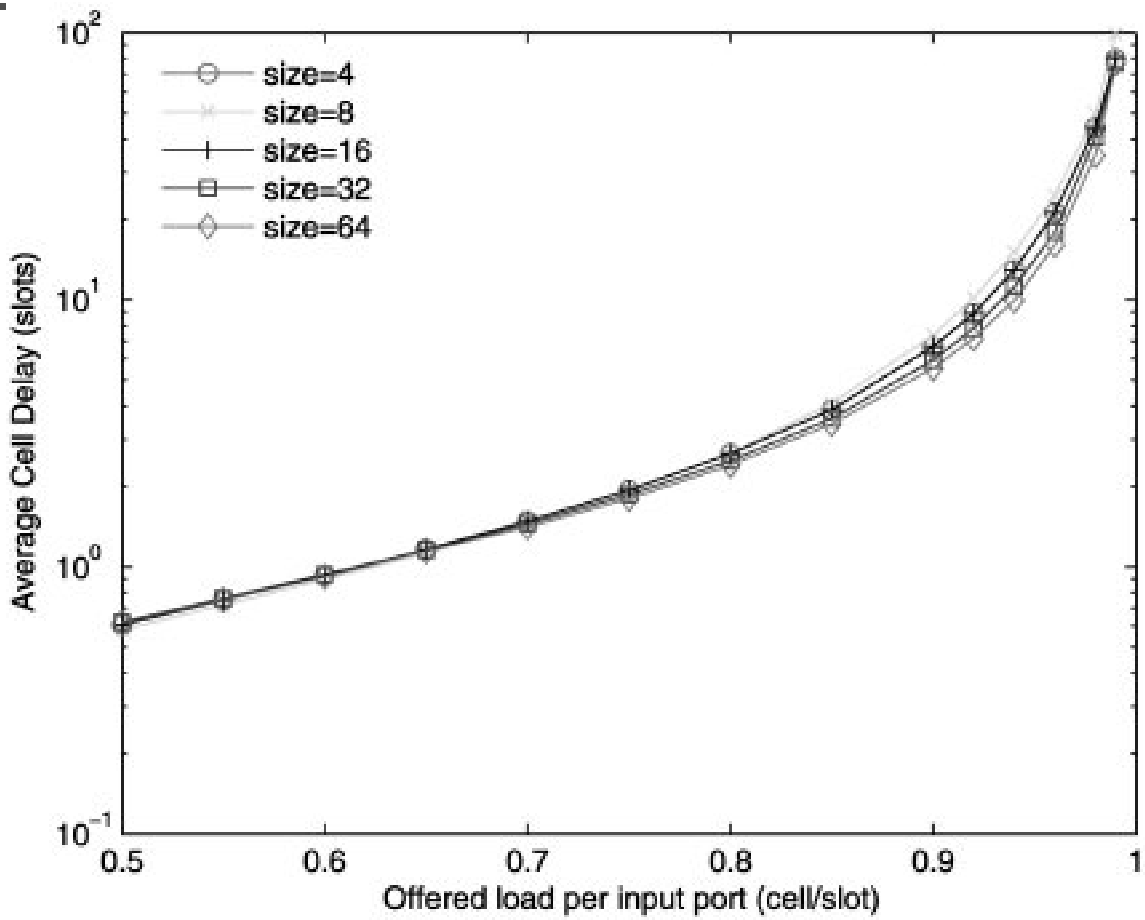
Performance of CTR, iSLIP, DRR, PIM, and EiSLIP for <u>uniform</u> traffic

SIMULATION



Performance of CTR, iSLIP, DRR, PIM, and EiSLIP for diagonal traffic

SIMULATION



The performance of CTR as a function of switch size for uniform i.i.d. Bernoulli arrivals.

CONCLUSIONs

- Achieves **essentially 100 percent** throughput for uniform and nonuniform traffic patterns.
- Has a **complexity comparable** to existing iterative schedulers (time complexity of θ(logN) and a circuit size of θ(NlogN) per port).
- Is **easily implemented** in hardware.
- Generally, the proposed CTR scheme can be applied to solve any resource allocation problem with a set of nodes competing for exclusive access to a set of shared resources.