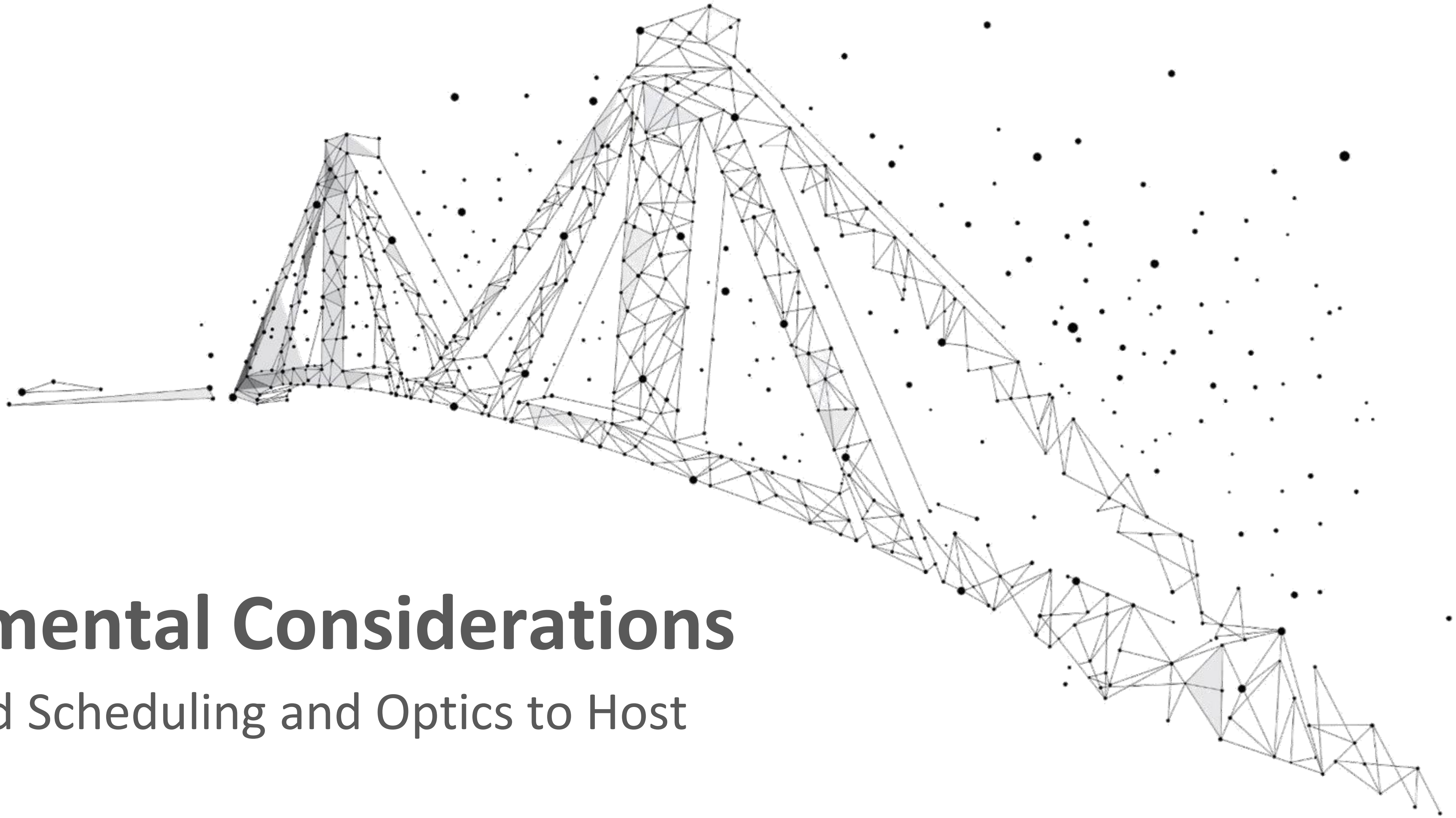


ODCNs
Fundamental Considerations

Distributed Scheduling and Optics to Host

ONDM 2019

Why ODCNs?

- Higher Aggregate Bandwidth Needed
  - **Host** bandwidth demands are exponential (see Jupiter Rising [1])
  - Hence, keeping the DCN scale require exponential ToR switch aggregate bandwidth AGG_{BW} [2]

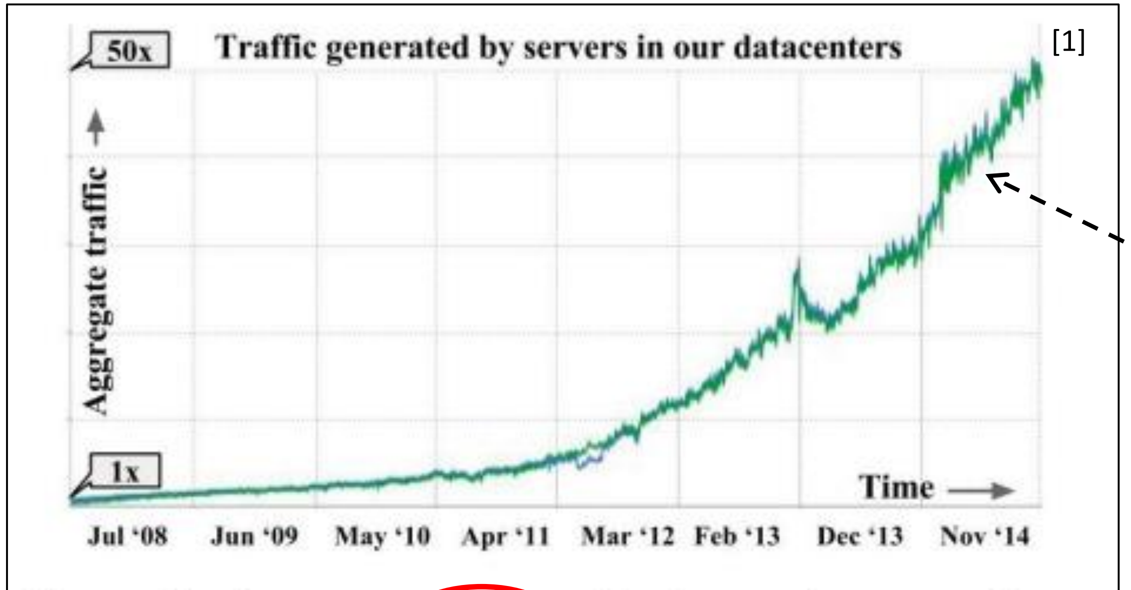
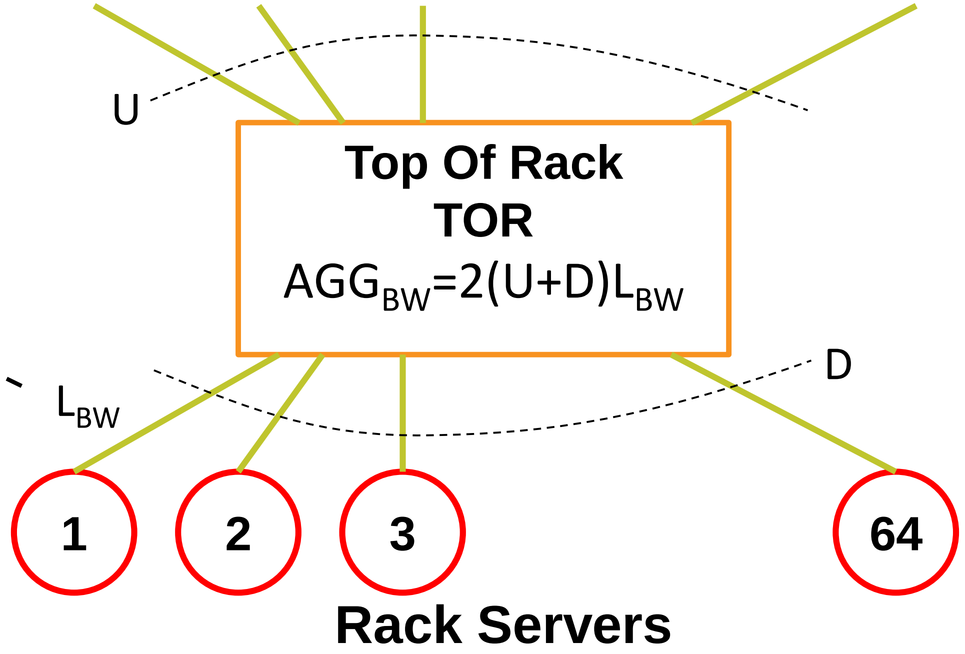


Figure 1: Aggregate server traffic in our datacenter fleet.



[1] A. Singh et al., "Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network," in SIGCOMM, 2015, pp. 183–197.
[2] W. M. Mellette, A. C. Snoeren, and G. Porter, "P-FatTree: A multi-channel datacenter network topology," in *Proceedings of the 15th ACM Workshop on Hot Topics in Networks*, 2016, pp. 78–84.

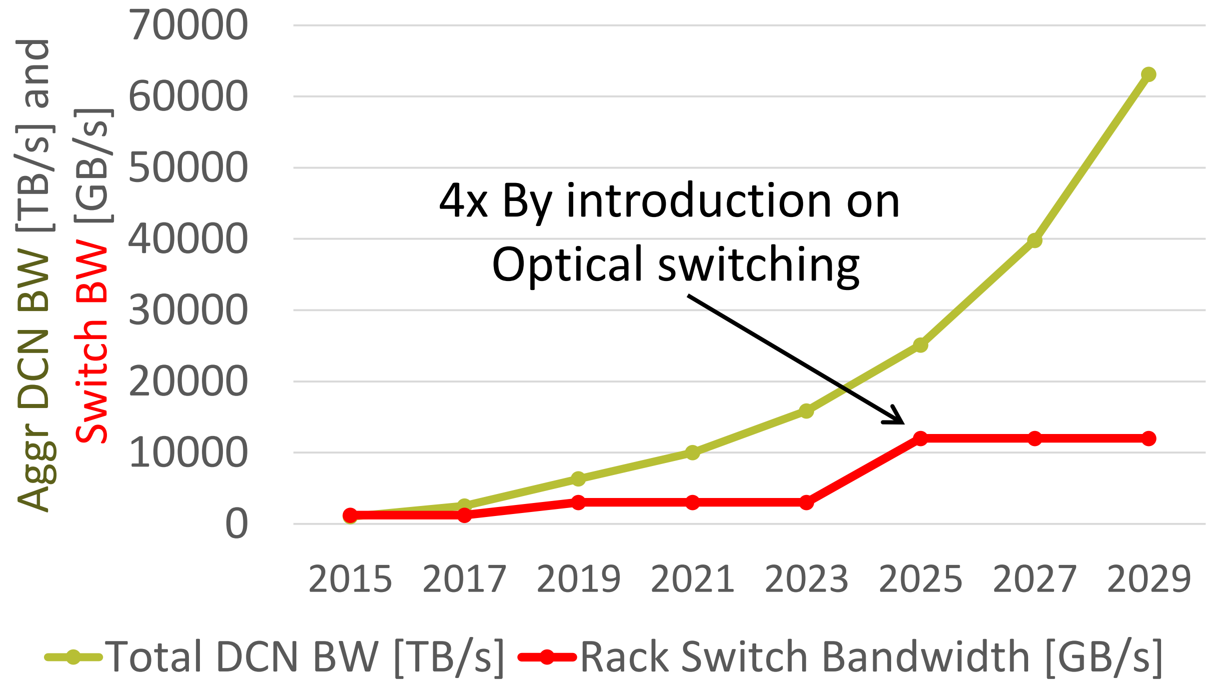
And the Problem is?

- **Silicon manufacturing technology** started to saturate ("The end of Moore law")
  - VLSI clock **frequency stay flat** since the end of the 90's
  - While transistor **area** scaling is maintained, wire density **start saturating**
    - Idea area scaling is ~0.54 transistor area reduction
    - **Effective** wire density scaling is ~0.7
  - Power density per mm^2 scales ~0.7

- Can switches aggregate bandwidth grow exponentially?
  - For fixed clock frequency 2x BW => 2x data path width (wires)
  - With **ideal** area scaling 0.54 switches scale too
    - => 2x cells * 0.54 area => ~constant chip size, logic power x0.7
  - Today true area scaling is saturating ~0.7
    - => 2x cells * 0.7 area => **1.4 chip size**
    - Logic power has to grow to drive long distances
    - => **power of the chip grows**
  - **What if wire density scaling is only 0.8 ?**

ITRS Predicted Required DCN BW vs. Switch Aggregated BW

[3]



[3] ITRS2.0 2015 System Integration Vol 1

Fundamentals

PetaCloud collaboration with Prof' Yithak Birk (Technion)

From Electrical Packet Switching to Optical Circuits

- **Ethernet networks are "packet switching":**
  - Small message segments are sent over the network
  - Packets from different messages can mix on the same wire
  - When the wire is busy with a packet, others wait at the buffer

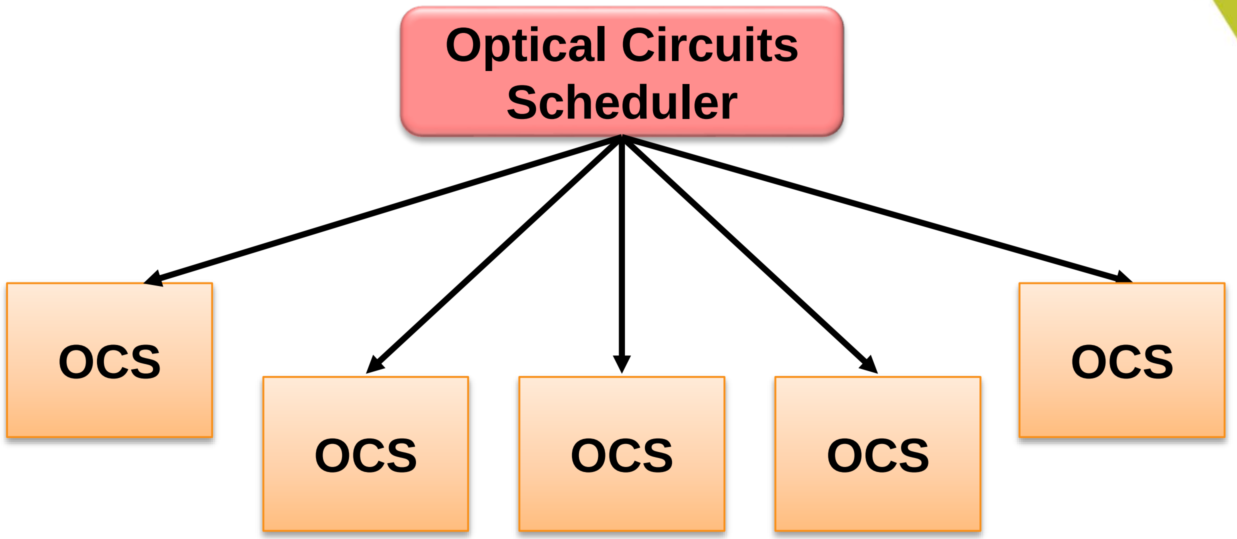
- **Optical network have no *Buffers***
  - Once data enters the fabric it cannot wait for scheduling
  - Packets are destroyed if they "collide"

Light must use the Green Wave

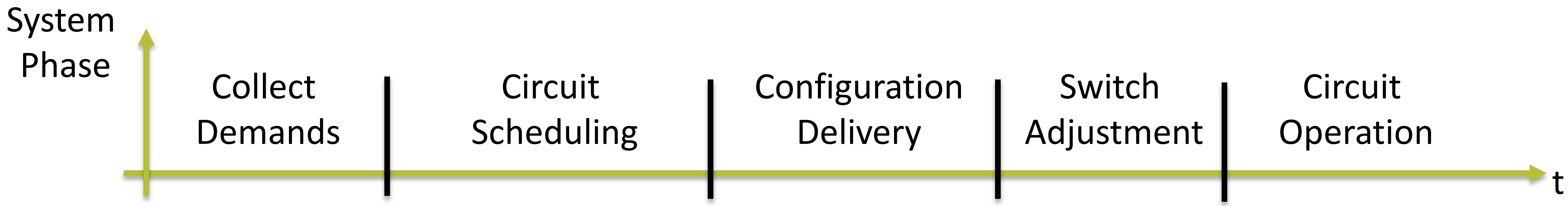
ODCNs use Circuit Switching instead of Packet Switching

Centrally Controlled ODCNs

- A Central Controller should
  - Know the **required traffic matrix**
  - **Compute** light circuits allocation
    - Online: A single permutation, or **Offline: a TDMA schedule**
      - To avoid **starvation** schedule offline the entire matrix
  - Send the **configuration** over to the network elements



- The following system phases are required



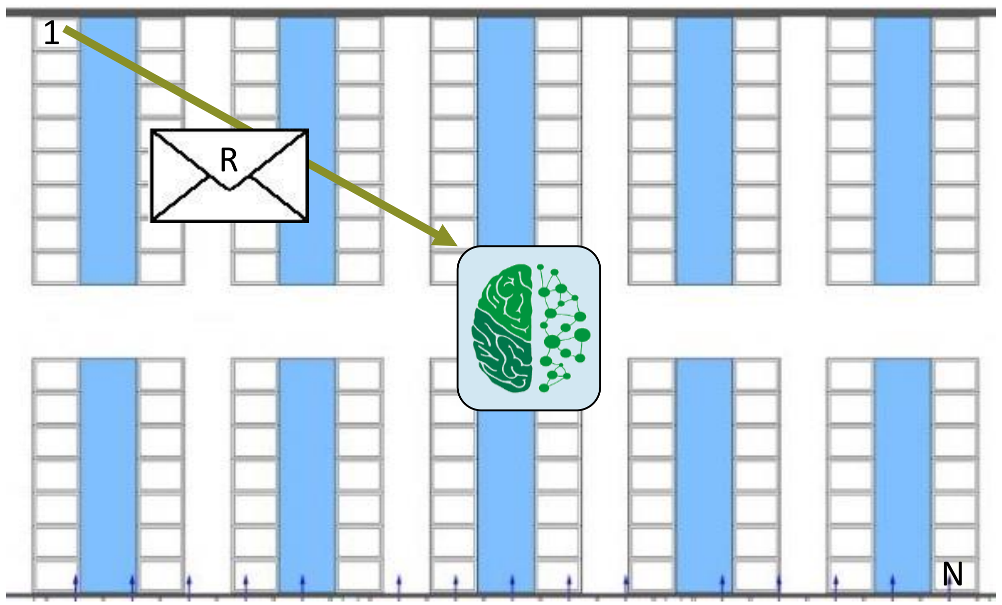
- Pipelining can help but slowest phase dictate throughput == slot time

Central Scheduling Fundamental Limitations: Demand Collection

- We calculate the size of the traffic demand matrix = [D] = N x N
- The time it takes to collect the Traffic Matrix = T_D

- Assuming TOR as an aggregation point the matrix size is N x N
- Assuming resolution of B bytes per entry and no overhead
- Control network bandwidth of C_{BW}
- $T_D = B*N^2/C_{BW}$

- Example:
  - **N=1000**
  - |D| = 1000*1000 = 1e6
  - Entry is 2 bytes
  - C_{BW} = 100Gbps = 12.5GB/s
  - T_D = |D| * 2 / 100Gbps = 2e6/12.5e9 = **160usec**

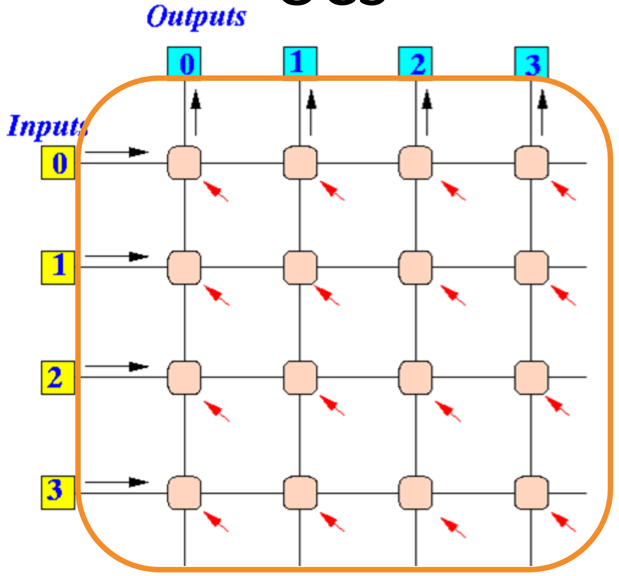


Traffic Demands Collection is a Slot Time Limiter

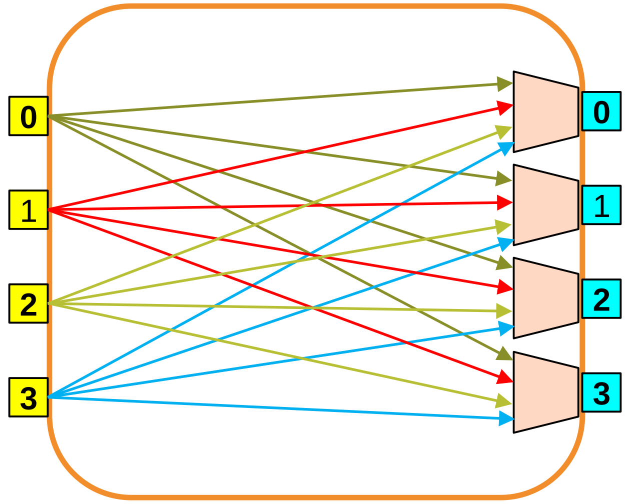
Central Scheduling Fundamental Limitations: Configuration Time

- Configuration Data Size – D_C, and sending time T_C
  - The amount of data the **central** resource allocator/scheduler has to deliver
- Most ODCNs built using Crossbar Optical Switches (OCS) or Broadcast and Select Switches (BSS)
  - Since optical circuits cannot intersect (on same color/mode/angular momentum)
- How much data is required to configure OCS/BSS that carry F new flows?
  - Common representation is the permutation
    - Assuming K ports switch $\log_2(K)$ bits for representing ports
    - Permutation is $K*\log_2(K)$ bits
- How much time does it take to configure all switches?
  - Example: 100 L2 switches of K=1000 (like RotorNet)
  - $K=1000$, $\log_2(K) = 10$
  - $D_C = 1000*10*100 = 1e6$ [bit]
  - $T_C = D_C / C_{BW} = 1e6 / 100Gbps = 1e6/100e9 =$ **10usec**

OCS



BSS



Configuration Delivery is NOT negligible

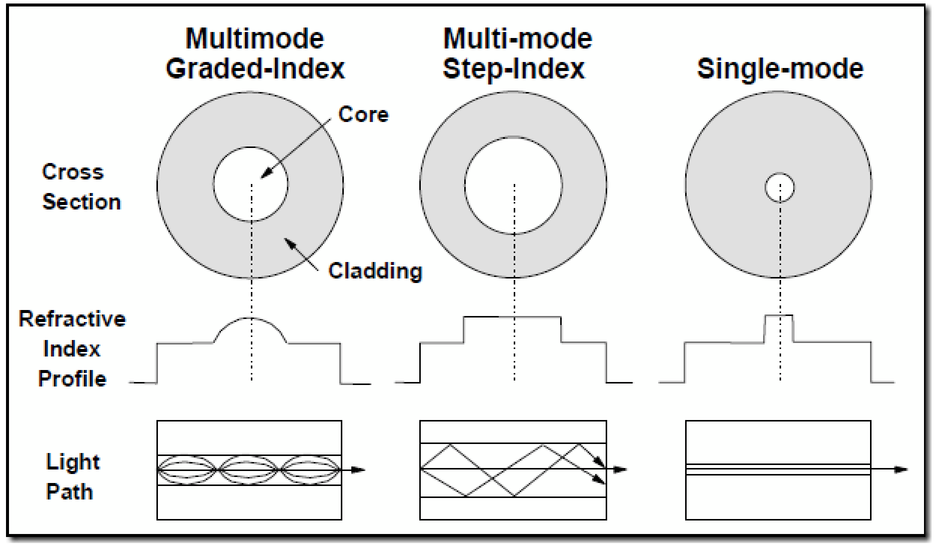
Central Scheduling Fundamental Limitations: Circuit Operation

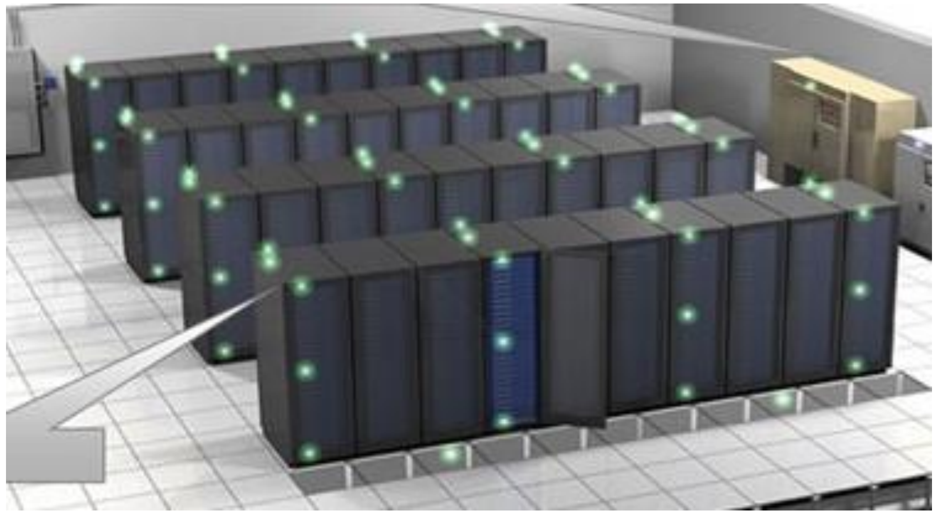
- How long does take the Light to cross the data center?
  - We denote it T_L
  - The speed of light in the refractive fiber is ~5nsec/meter

- How far apart are hosts from each other?
  - The most compact distance geometric shape: Circle
  - A realistic approximation: Square

  - Most packed Floor Plan calculation for T ToRs
    - Rack Width 60cm, Depth 100cm, Isle 100cm (on the depth side)
    - Nw*Nd=T, Nw*0.6=Nd*2.0 => $\widehat{N}_d = \sqrt{3T/_{10}}$

  - Example: T=1000
    - =>$\widehat{N}_d = \sqrt{3*1000/_{10}} = 17$ => Nd = 17, Nw = 59
    - Max Manhattan distance between racks = 2.0*17+0.6*59=69m
    - Max latency between racks T_L =~ **0.3usec**





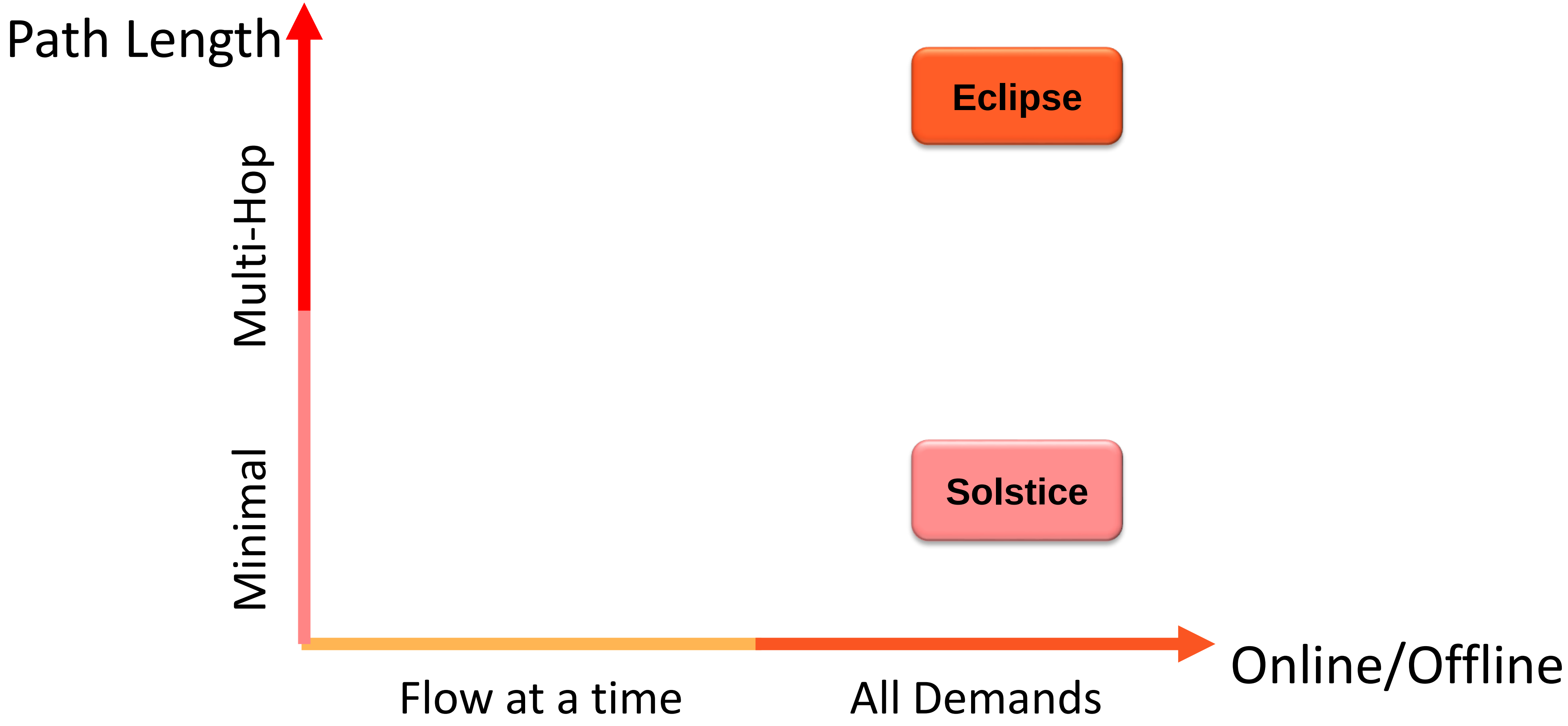
Intrinsic Propagation Latency is < 0.5usec

Taxonomy Of Circuit Scheduling Options

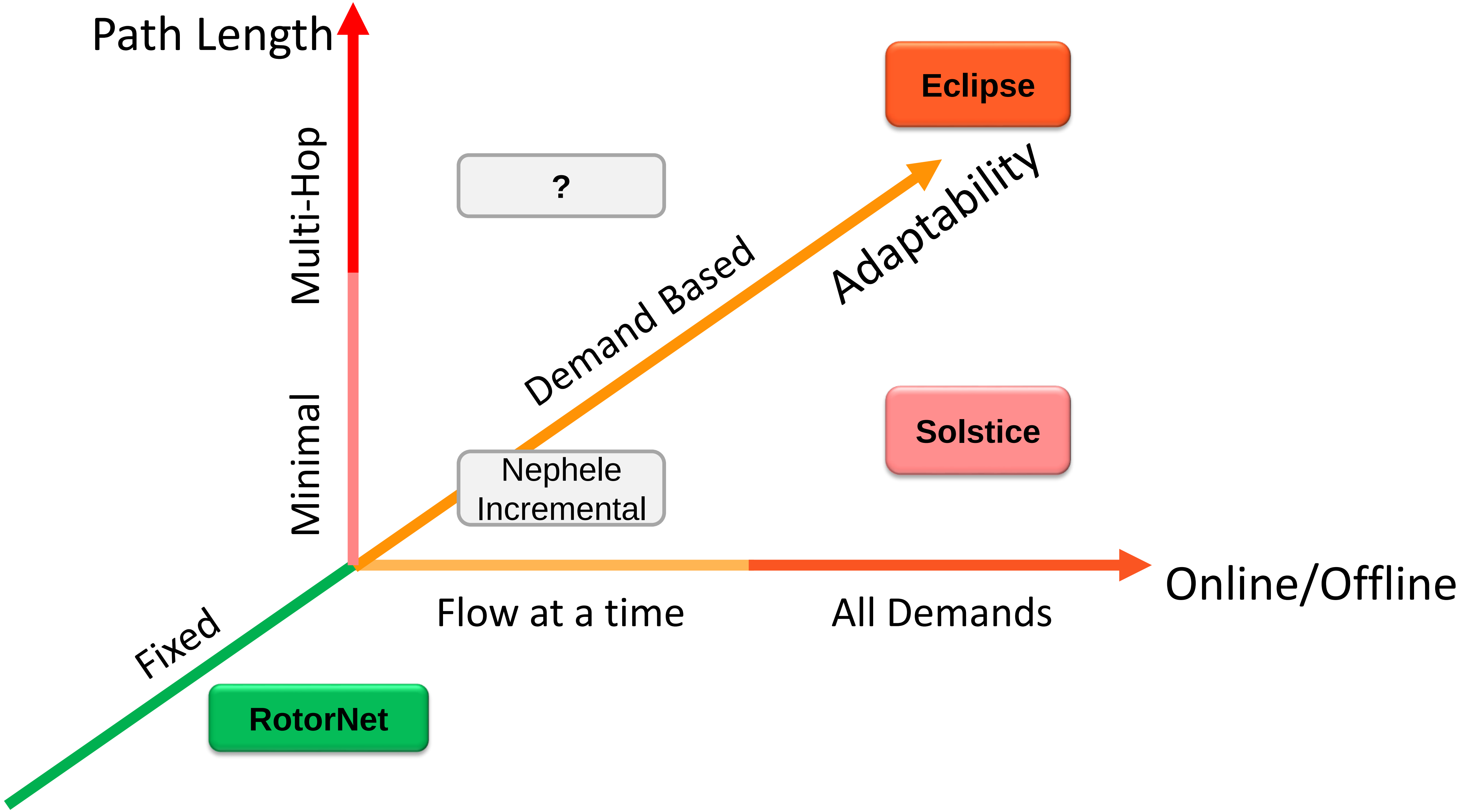
Solstice

Flow at a time — All Demands → Online/Offline

Taxonomy Of Circuit Scheduling Options



Taxonomy Of Circuit Scheduling Algorithms



Central Scheduling Fundamental Limitations: Computation Complexity

- Scheduling problem: how to allocate light paths to meet the traffic demand

- To avoid potential starvation allocate a complete "Schedule" of multiple "slots"

- Single Maximum Matching (non weighted) Hopcroft Karp
  - complexity $O\left(E\sqrt{V}\right) = O(N^{3/2})$
    - Assuming Clos where V=N/k and E = N (permutation at minimum - each host send to just one other)

- Solstice: a leading single hop algorithm
  - Complexity $O(N^2 log^2(N))$

- Eclipse: utilizing available multi hop paths (optical, electrical, optical…)
  - Complexity is even higher

Dynamic Scheduling Time is not Scalable

Central Scheduling Faith

Central Scheduling is a Dead End

- What can be done?
  - Fixed Schedule RotorNet
    - Support All-to-all demand, make any demand all-to-all
    - Pay in latency
  - Distributed Scheduling
    - Tradeoff the “infinite” bandwidth of Optical Fibers with less accurate scheduling
    - Lose some bandwidth, win much time
    - Avoid both requirements collection, offline scheduling and configuration fundamental limits

New Architectures Enable ODCN

The Hybrid ToR Paradox

Motivation for ODCNs?

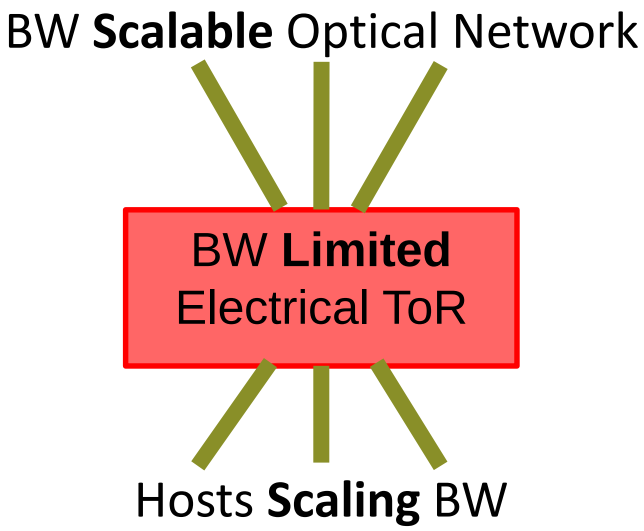
Saturation of
Electrical Packet Switches
Aggregate Bandwidth

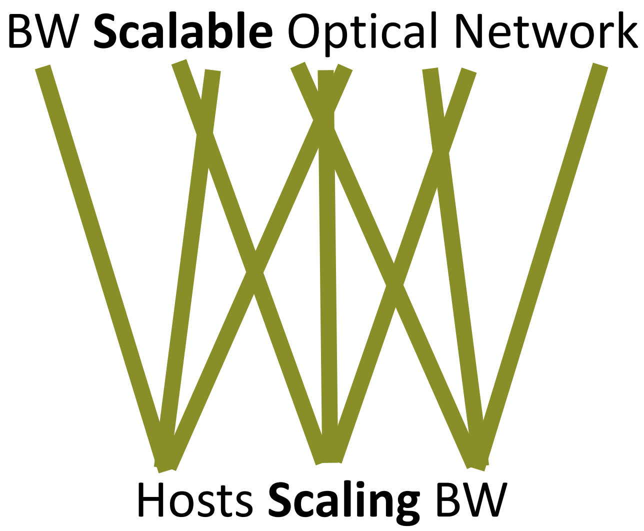
Avoiding the Electrical Switch BW Bottleneck

- Our motivation for ODCN is Saturation of EPS Aggregate Bandwidth
- Hence we must avoid using Electrical ToR
- Otherwise they become our Bisectional Bandwidth Scaling **bottleneck**

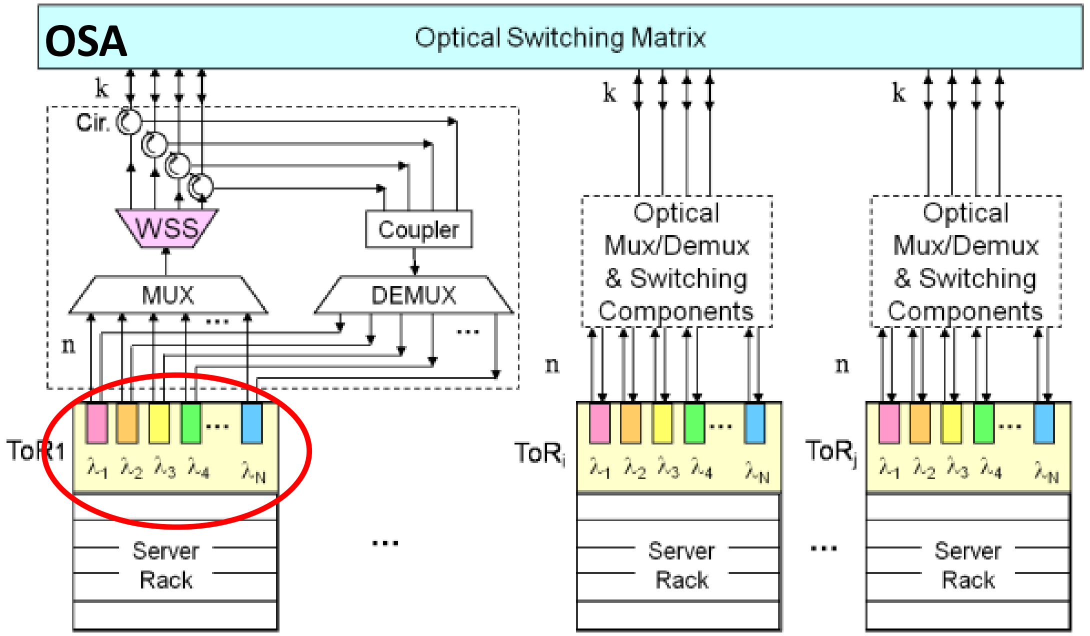
We assume Electrical ToR have saturated BW

=> use Optical to the Host

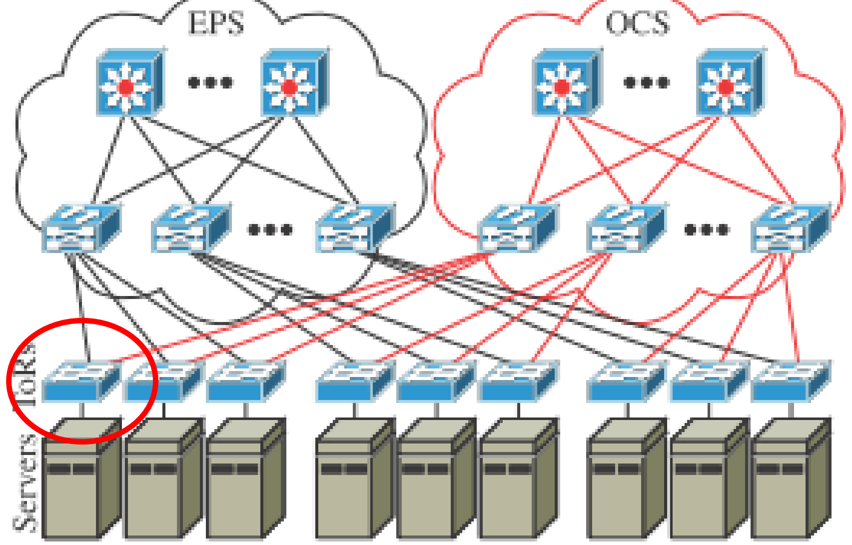




But Most ODCNs Utilize Electrical ToR !!!

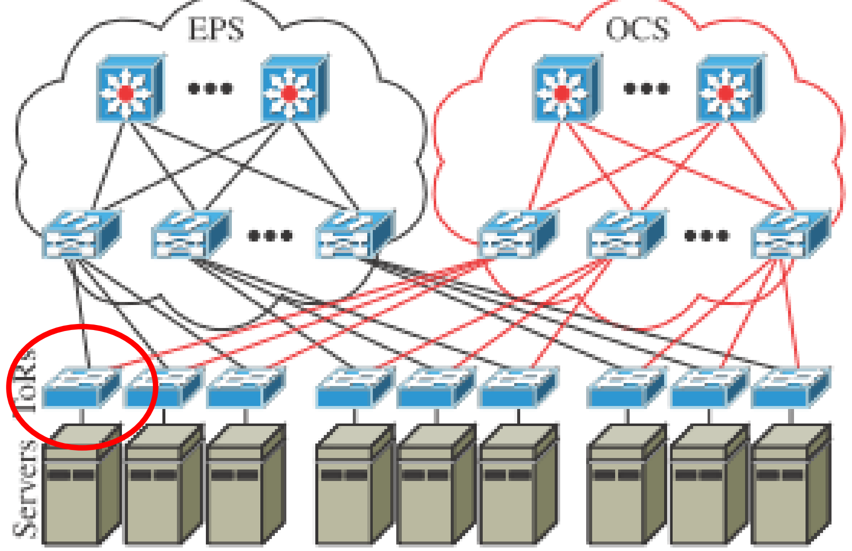


Optical DCN with Elastic OCS

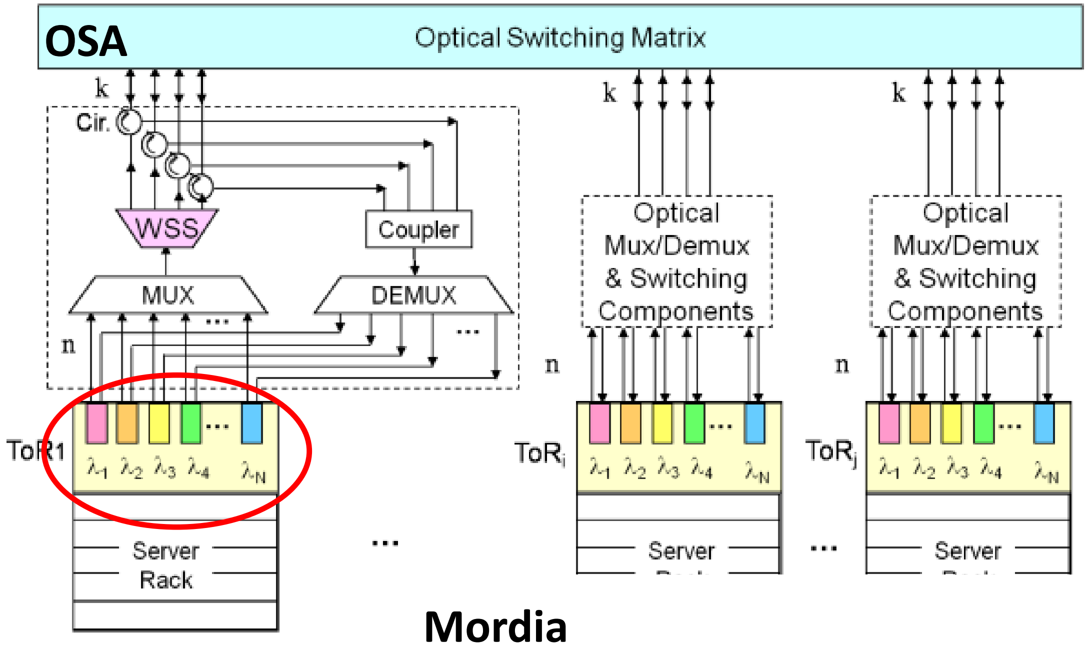


But Most ODCNs Utilize Electrical ToR !!!

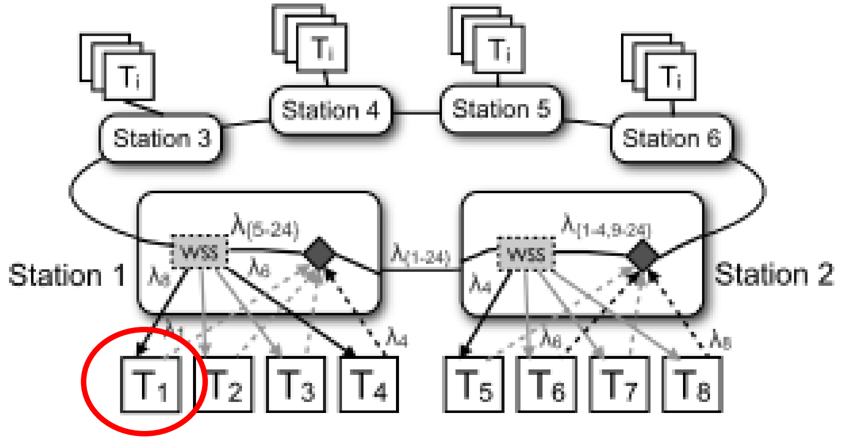
Optical DCN with Elastic OCS



OSA

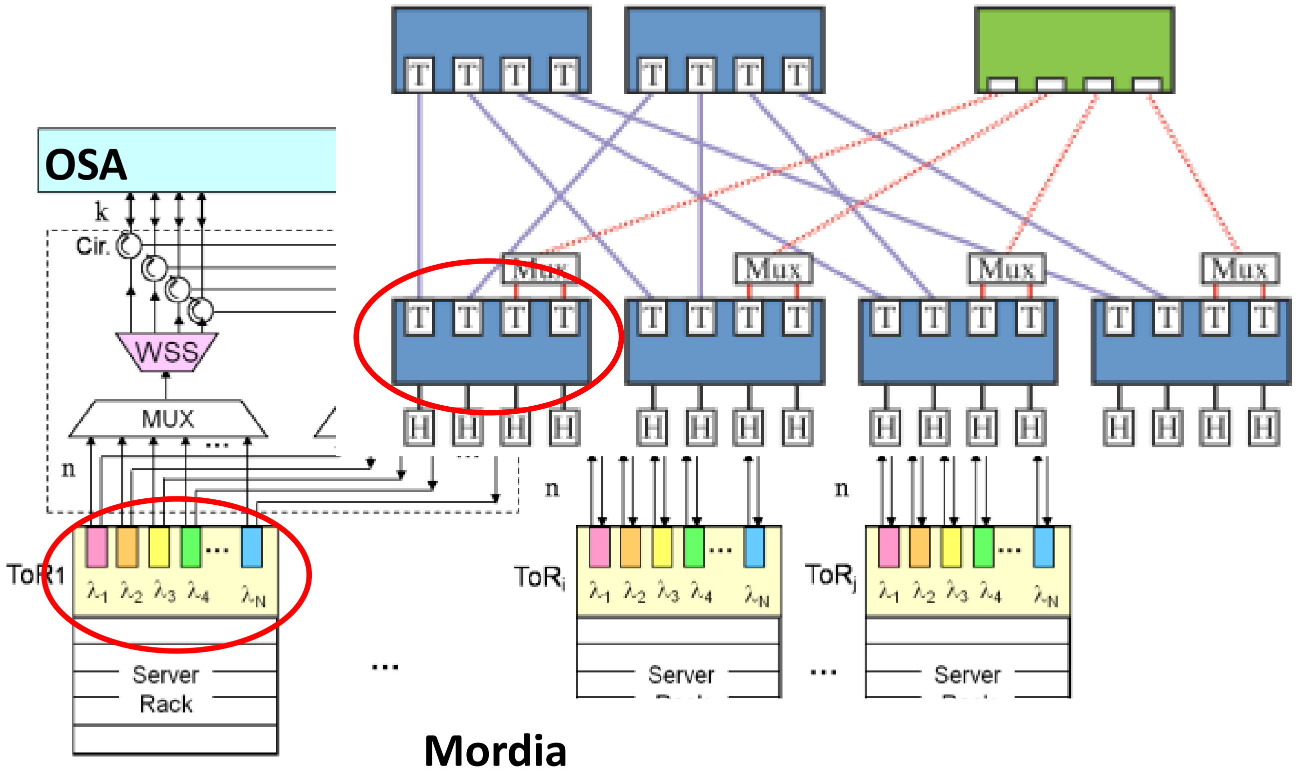


Mordia

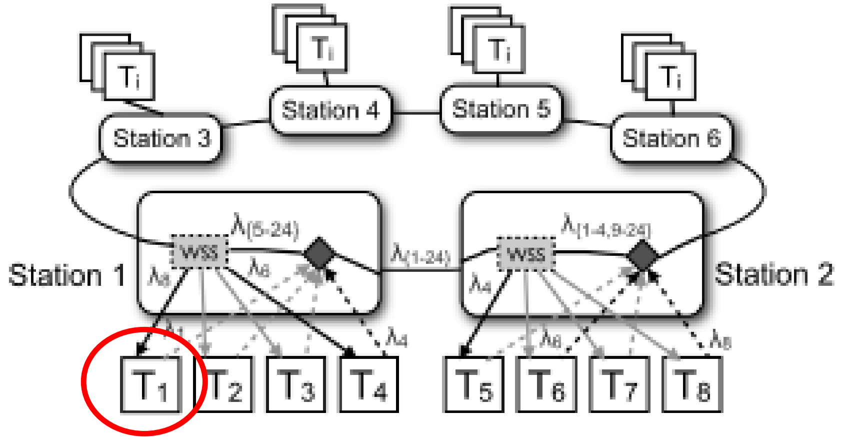


But Most ODCNs Utilize Electrical ToR !!!

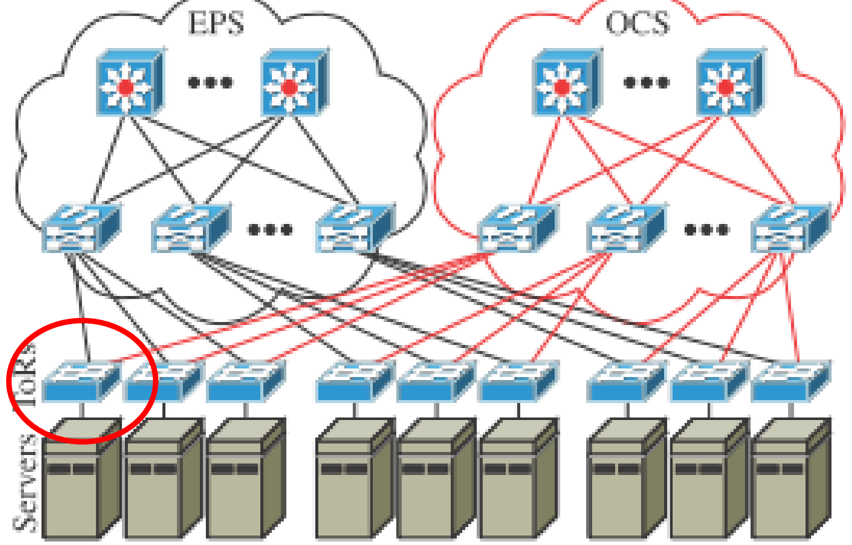
HELIOS



Mordia

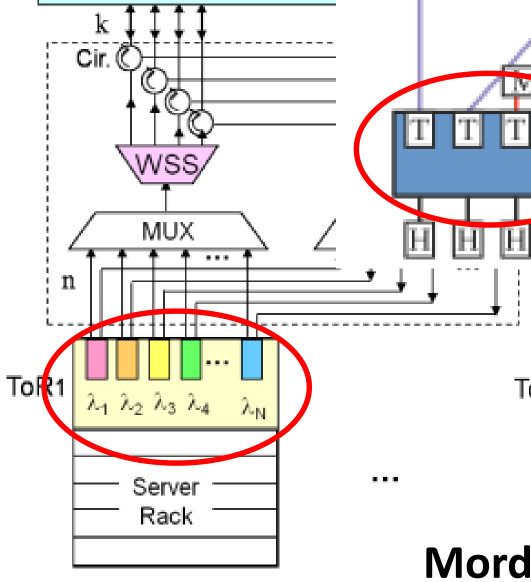


Optical DCN with Elastic OCS

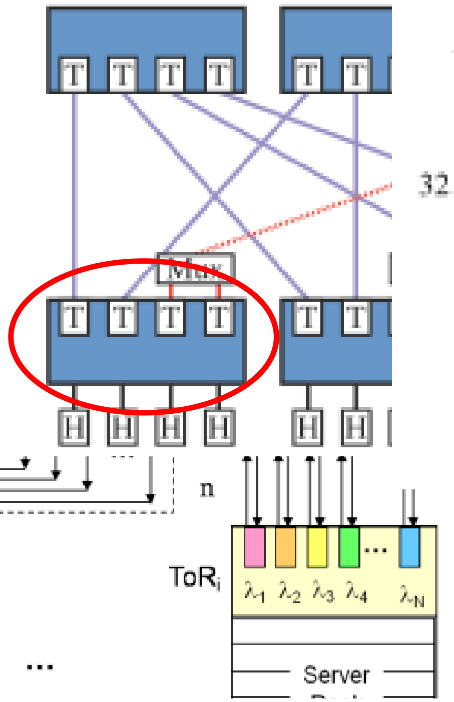


But Most ODCNs Utilize Electrical ToR !!!

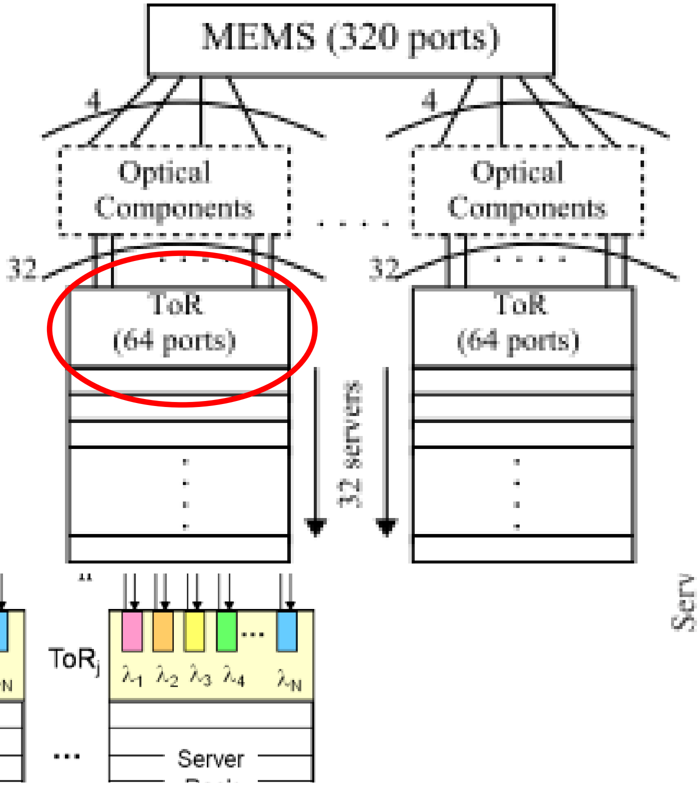
OSA



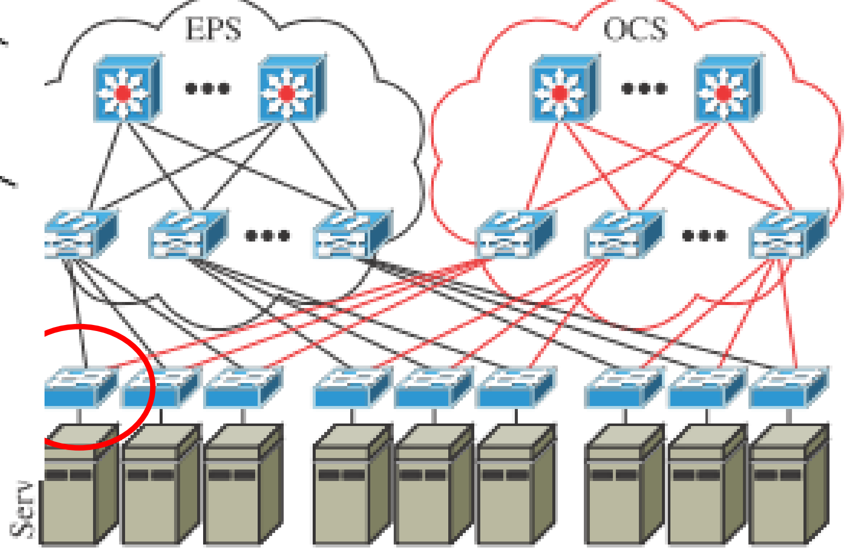
HELIOS



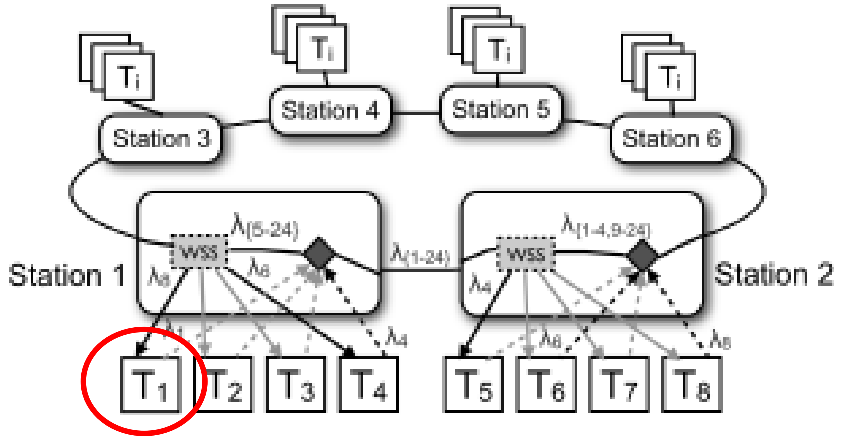
Proteus



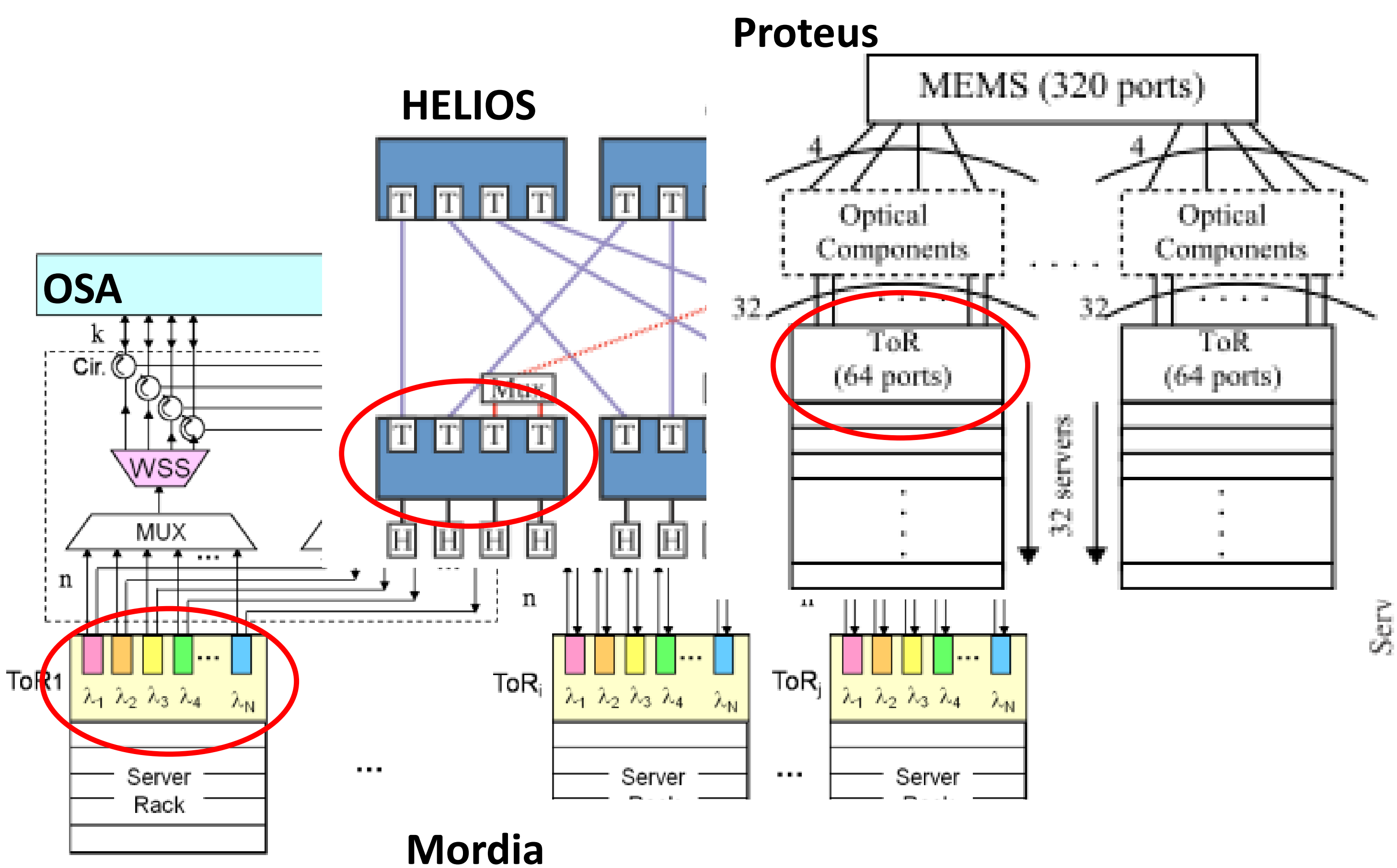
Optical DCN with Elastic OCS

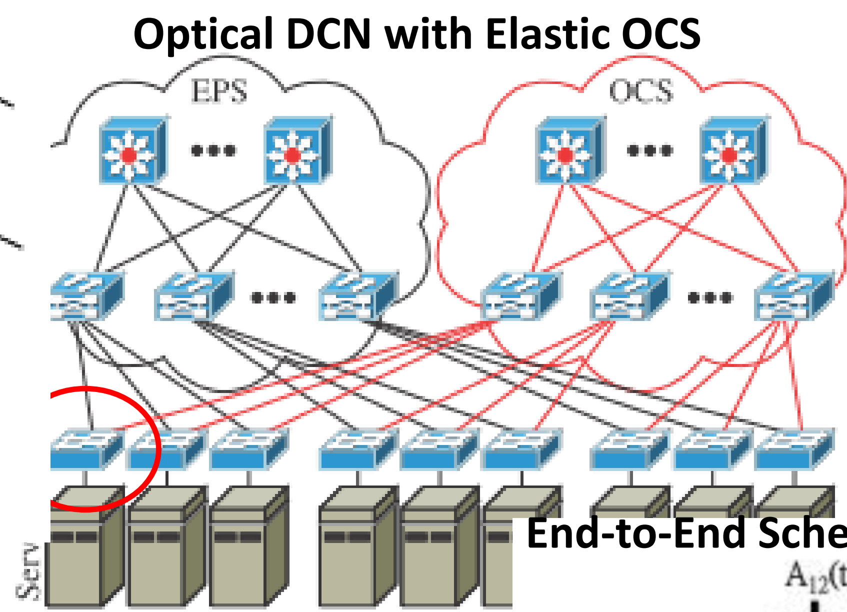


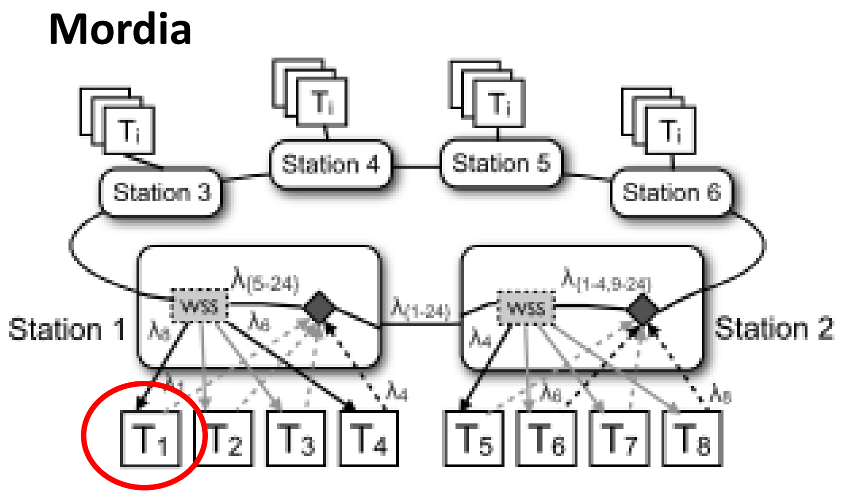
Mordia

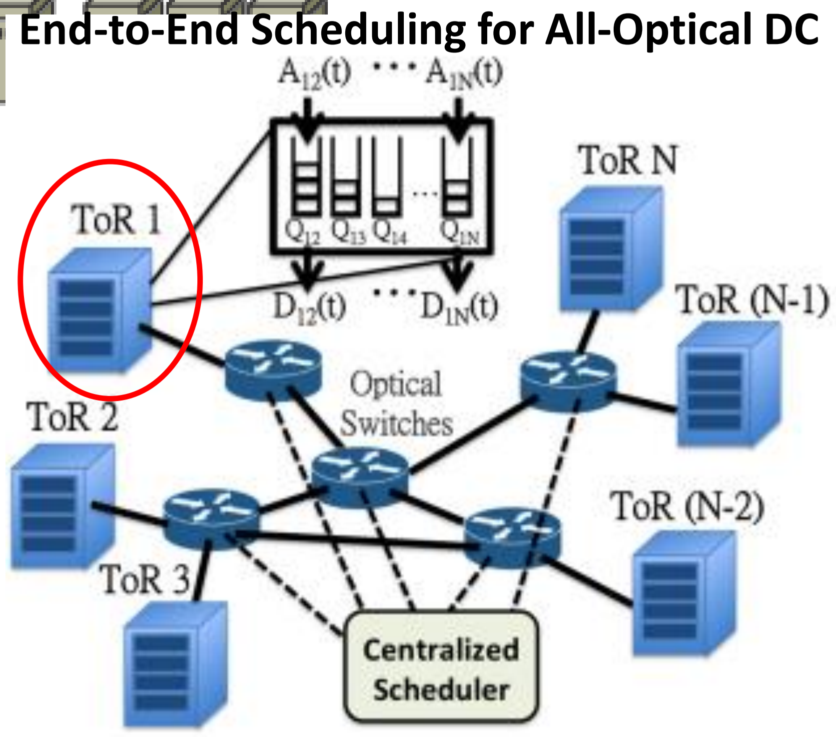


But Most ODCNs Utilize Electrical ToR !!!

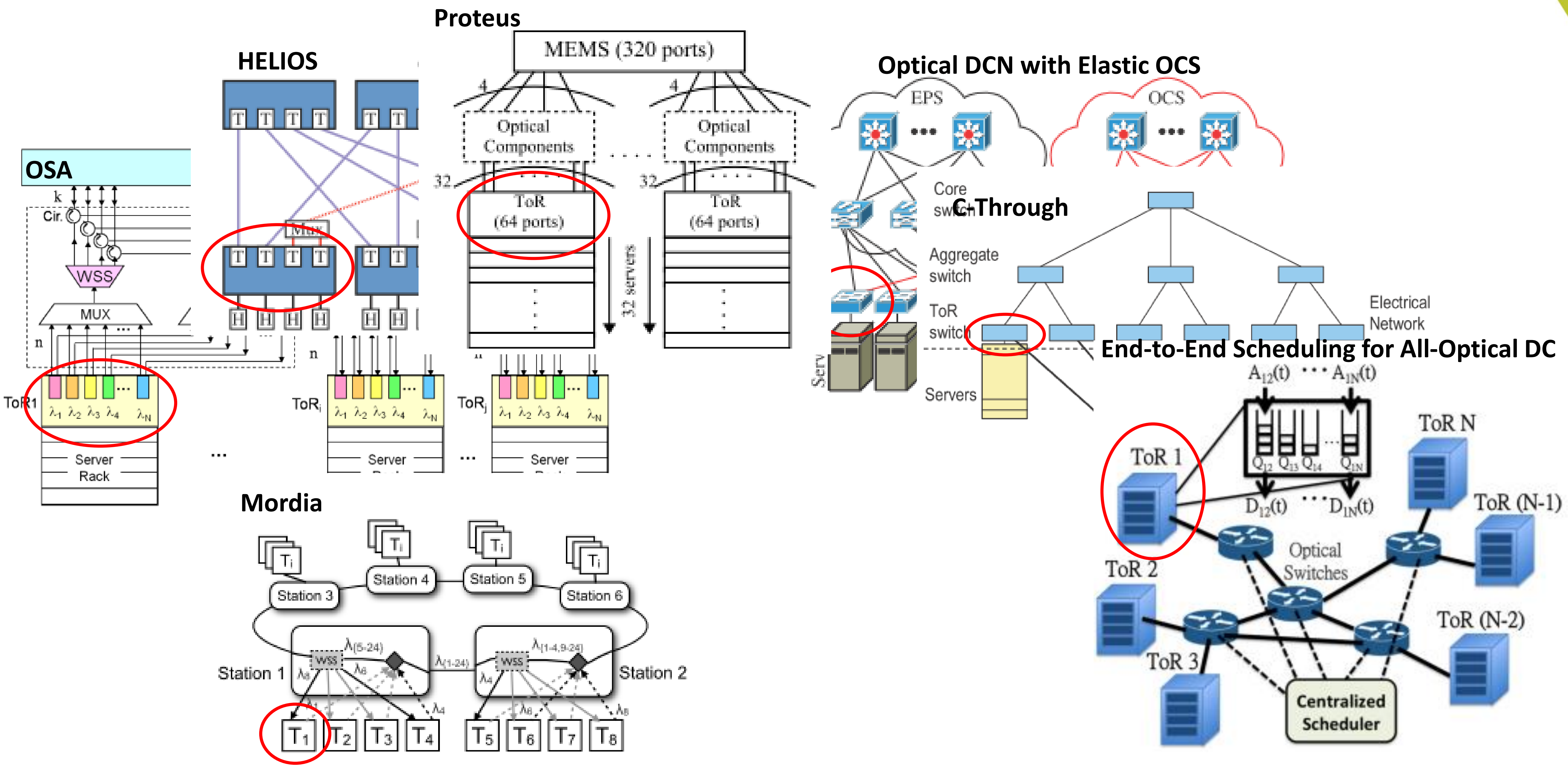




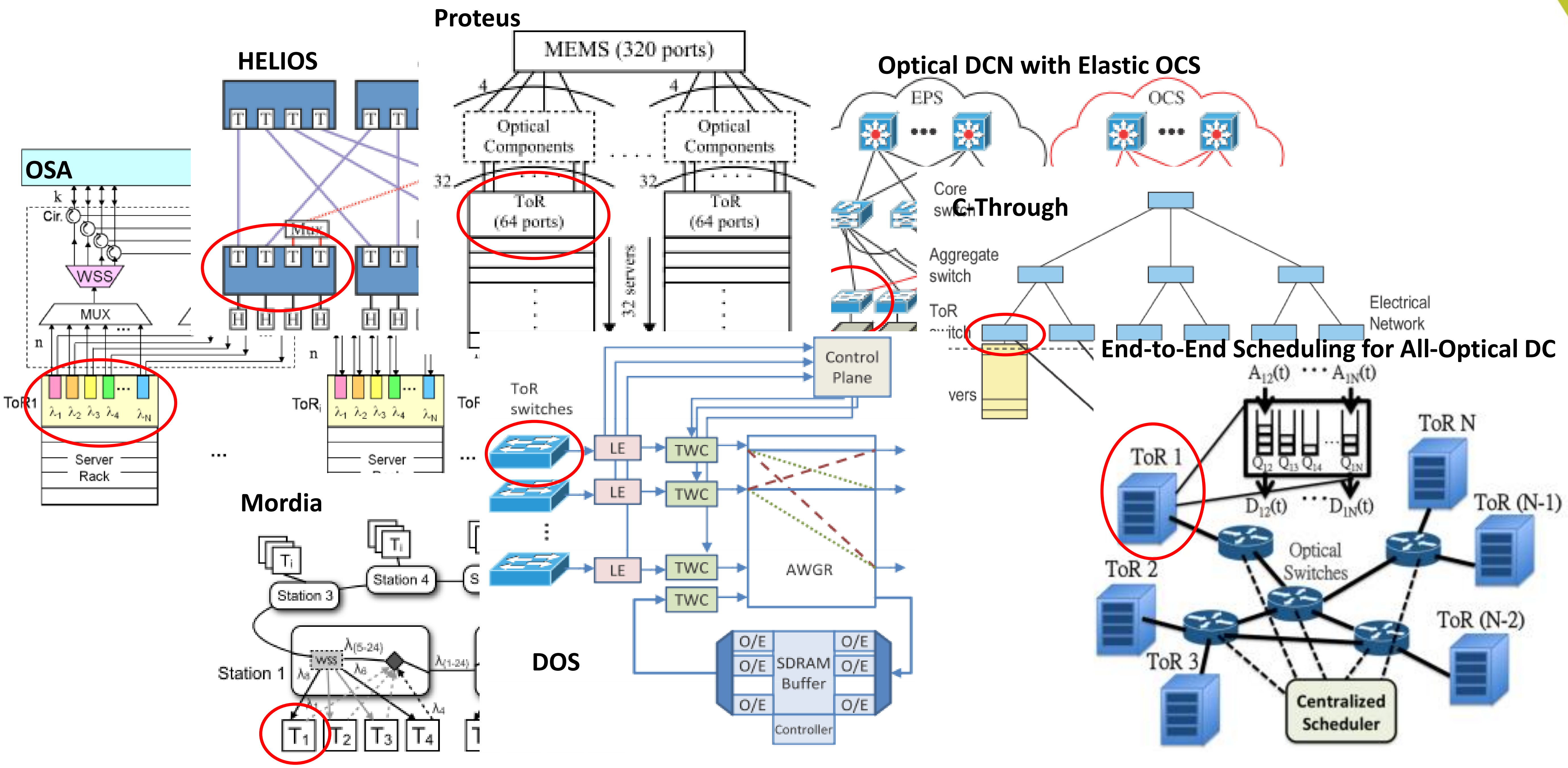




But Most ODCNs Utilize Electrical ToR !!!

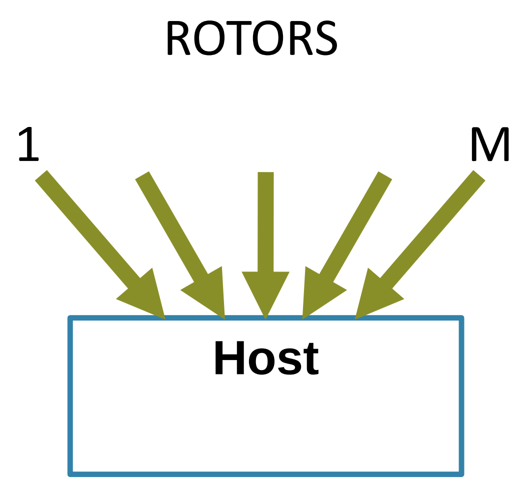


But Most ODCNs Utilize Electrical ToR !!!



Why RotorNet cannot do Optical to the Host?

- RotorNet assumes each ToR connects to all M rotors
  - With clear tradeoff between network latency and that number
- Connecting the hosts to all rotors is costly
  - Most of today hosts utilize 1 or 2 ports of 4 lanes each
- Moreover, required host peak input bandwidth is M x lane bandwidth
  - Since there is no coordination between senders to same host



Lack of host input bandwidth scheduling

Prevents **Optical to the Host**

=> Must Schedule Host Inputs

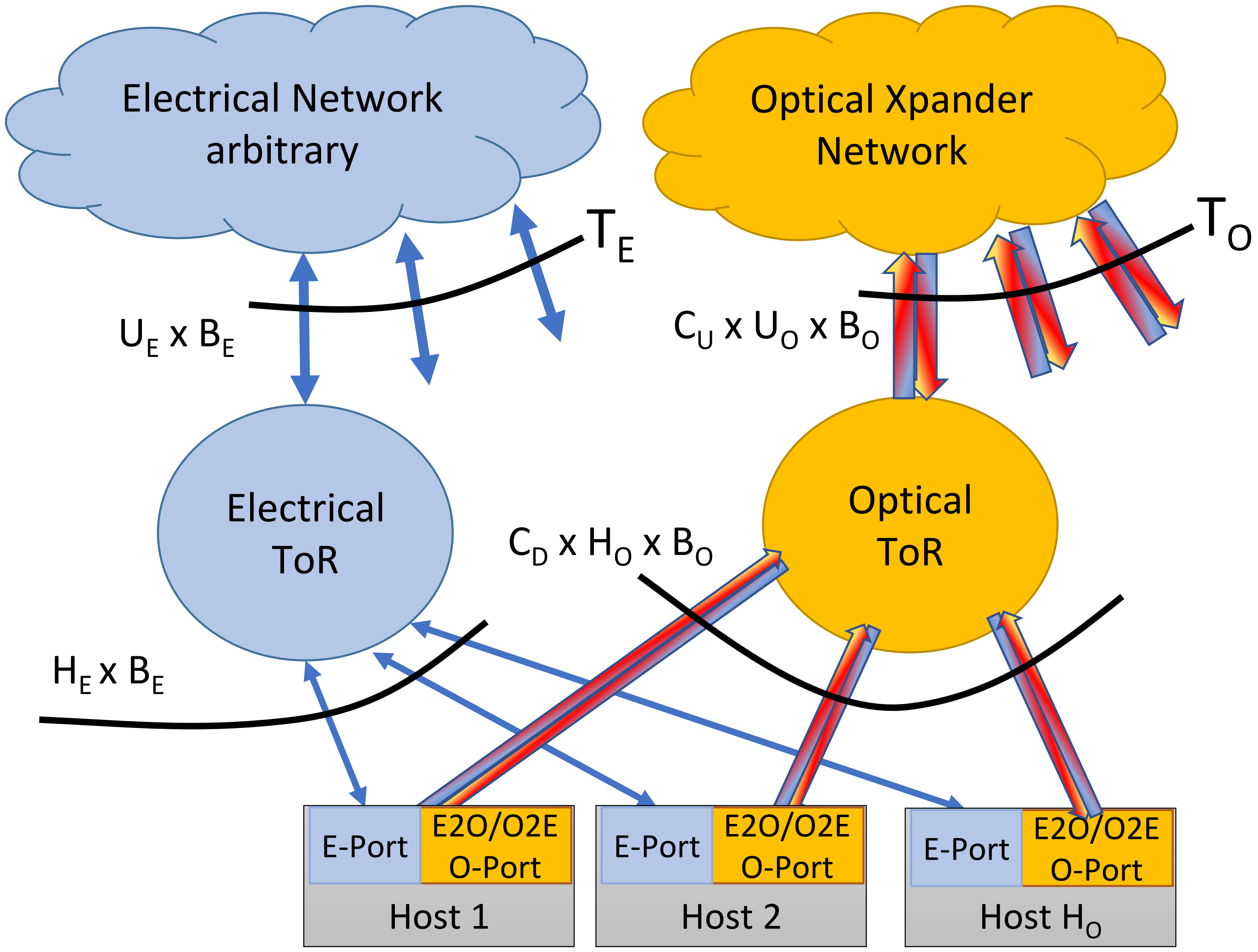
SOX

Server attached Optical eXpander

Prof' Ankit Singla (Eth),
Prof' Michael Schapira (HUJI),
Eitan Zahavi, Paraskevas Bakopoulos (Mellanox)

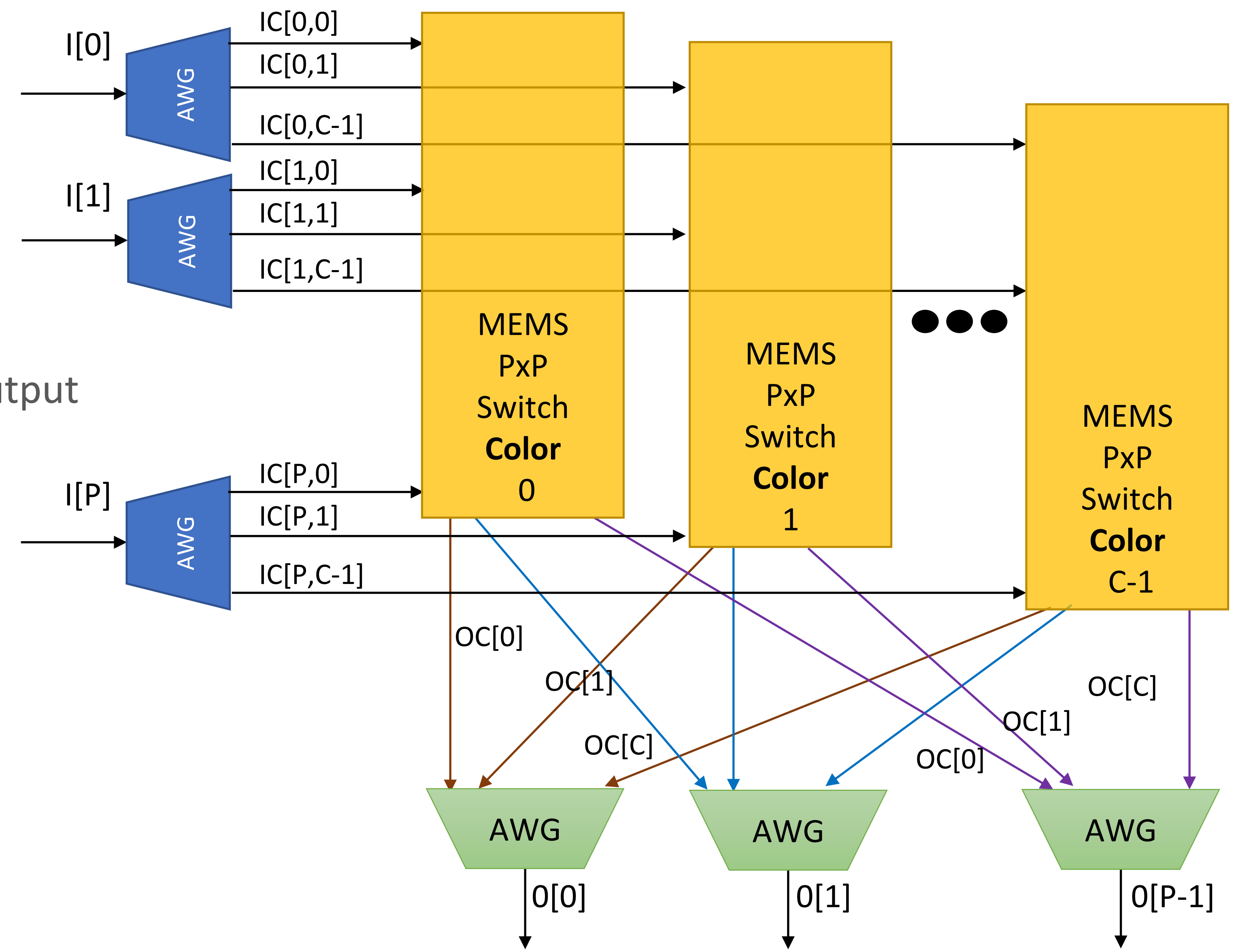
Server attached Optical eXpander (SoX)

- Optical Expander provides
  - Low diameter
  - Low cost (no multi-layer)
  - High bandwidth via DWM
- Electrical
  - Low bandwidth network

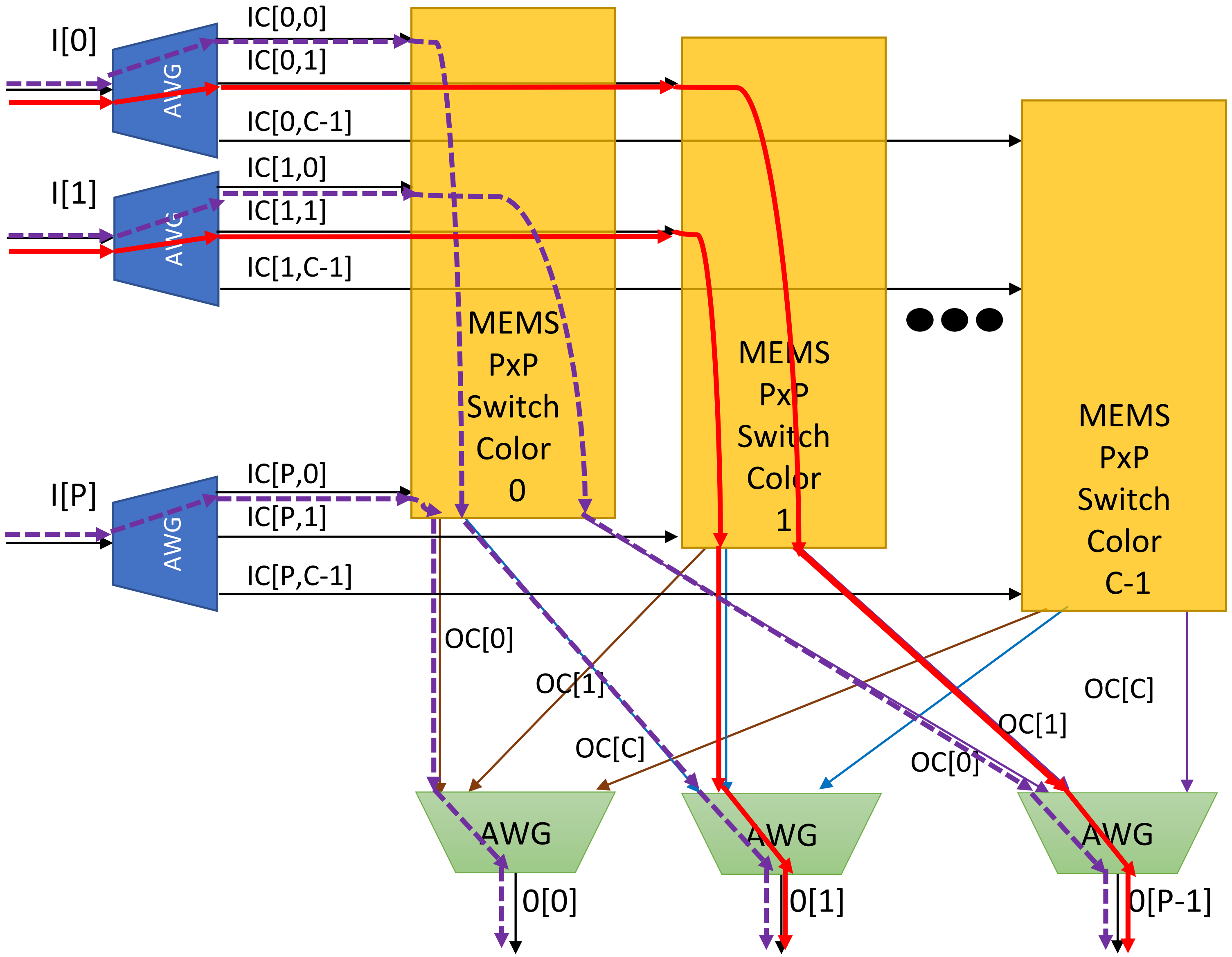


SoX Switch

- Wavelength Selective Switches
- Input AWGs
  - Split the input to different colors
- MEMS
  - Per color
  - A crossbar from every input to every output
- Output AWGs
  - Act as DWM combiners



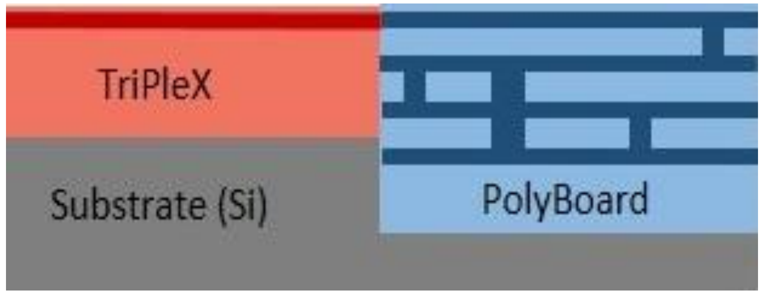
SoX Switch



I[0]
I[1]
I[P]
AWG
IC[0,0]
IC[0,1]
IC[0,C-1]
IC[1,0]
IC[1,1]
IC[1,C-1]
IC[P,0]
IC[P,1]
IC[P,C-1]
MEMS PxP Switch Color 0
MEMS PxP Switch Color 1
MEMS PxP Switch Color C-1
OC[0]
OC[1]
OC[C]
OC[1]
OC[0]
OC[C]
O[0]
O[1]
O[P-1]

SoX Switch Integration - The 3PEAT Technology

Combine PolyBoard with TriPlex platform to develop large photonic switches

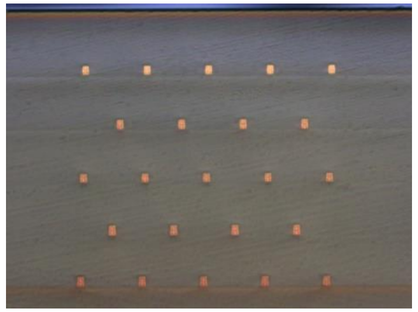


Target prototypes

- ✓ 36×36 active switch
- ✓ 72×72 AWGR

HHI 3D PolyBoard

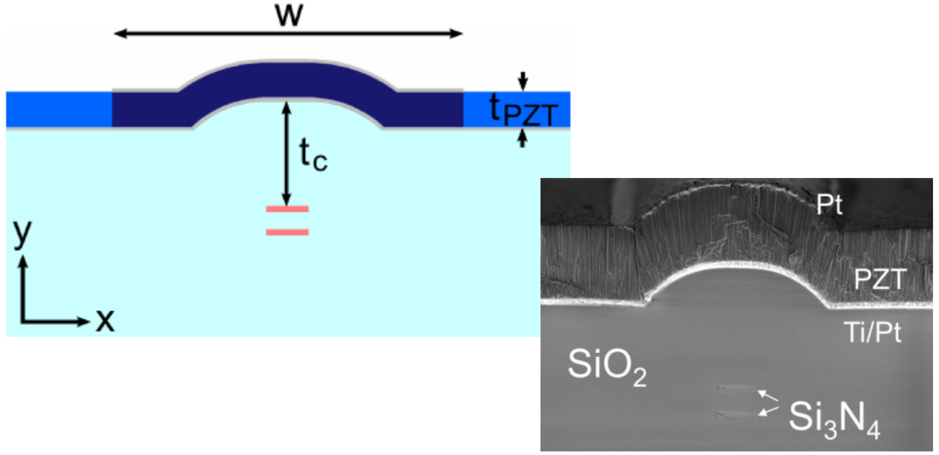
- > 7 layers
- 8x8 and 9x9 AWGRs
- >250 MMI couplers on a single chip



Multi-layer polymer waveguide (HHI)

LioniX TriPleX

- PZT films on the top of TriPleX chips
- 20ns target reconfiguration time

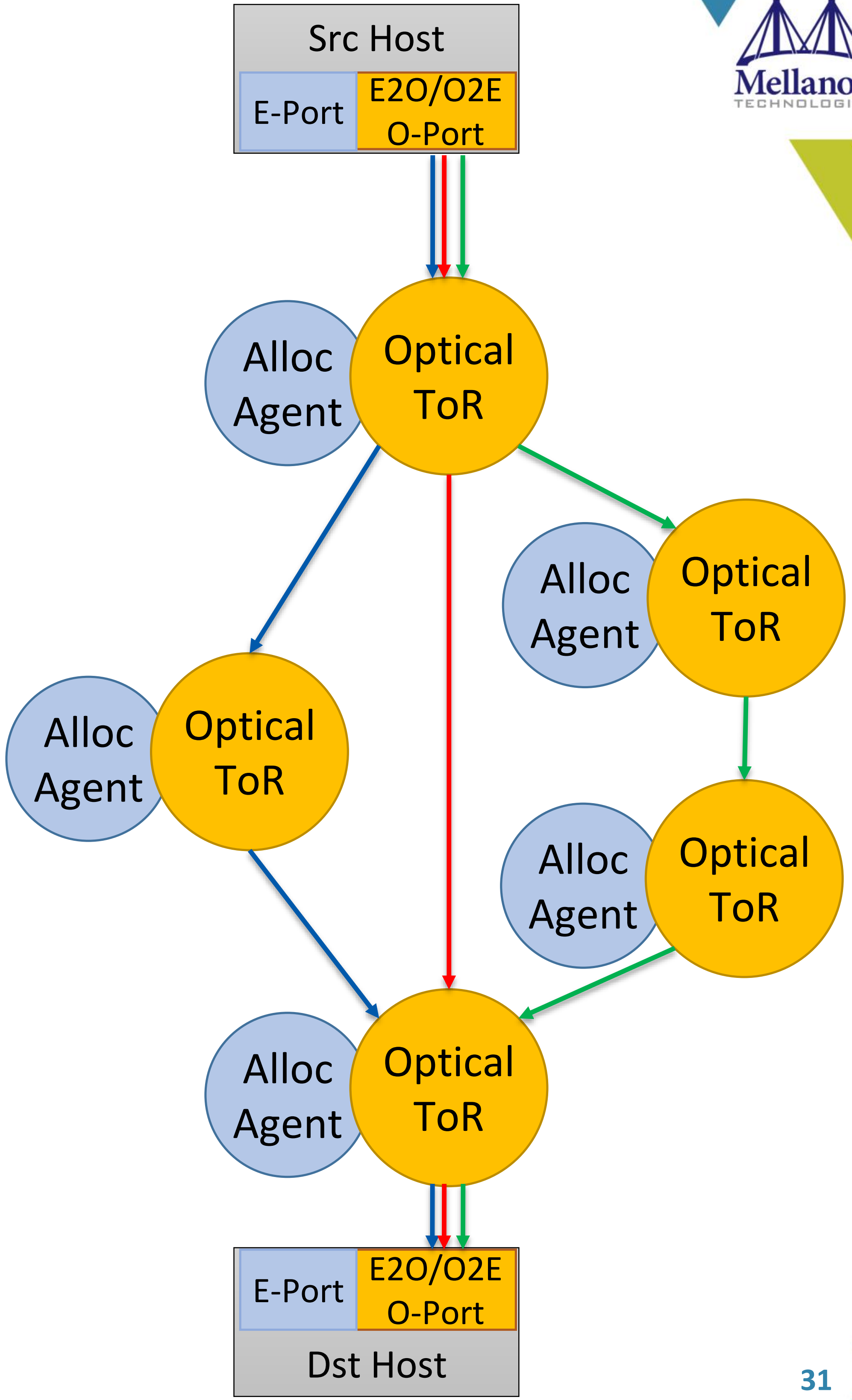


PZT film on TriPlex switch (LioniX)

SoX Scheduling

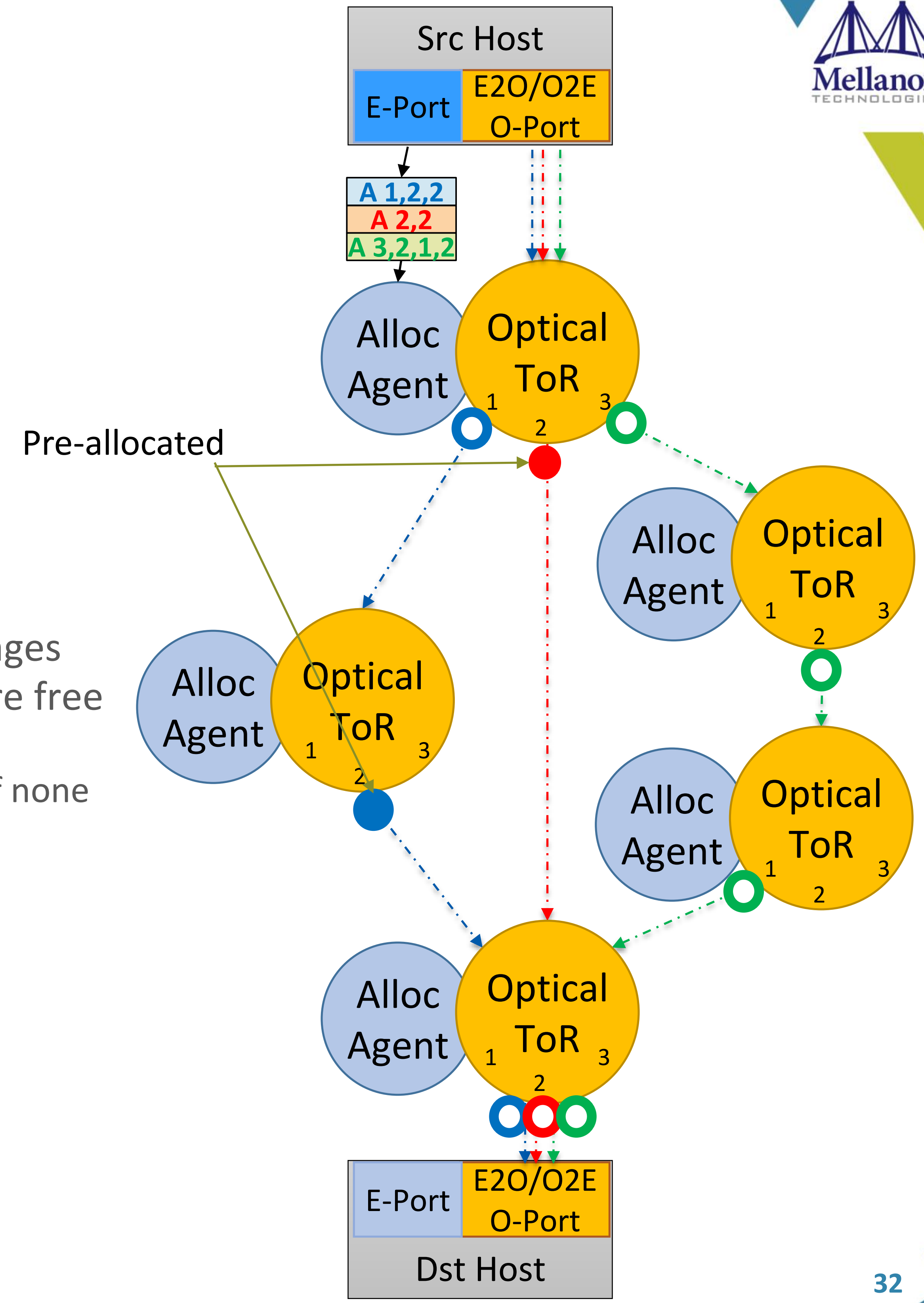
- Distributed and Scalable
  - K Optical switches are managed by a single Allocation Agent (AA)
  - The more Optical Switches the more agents

- High Level Algorithm
  - Try N pre-calculated paths from source to destination
  - Paths colors are preassigned
  - AAs tracks the used colors on each output of the Optical Switches it manages
  - When a request arrives to an AA it looks to see if the Output/Color pair are free
    - If they are free
      - Reserve the color and send the request to next Agent or back to the origin if none
      - Configure the optical switch accordingly
    - If not free send Backward Cancellation to previous AA
  - When AA receive Backward/Forward Cancellation request
    - It send it to previous/next AA (if not last)
  - Requestor uses the first reserved path



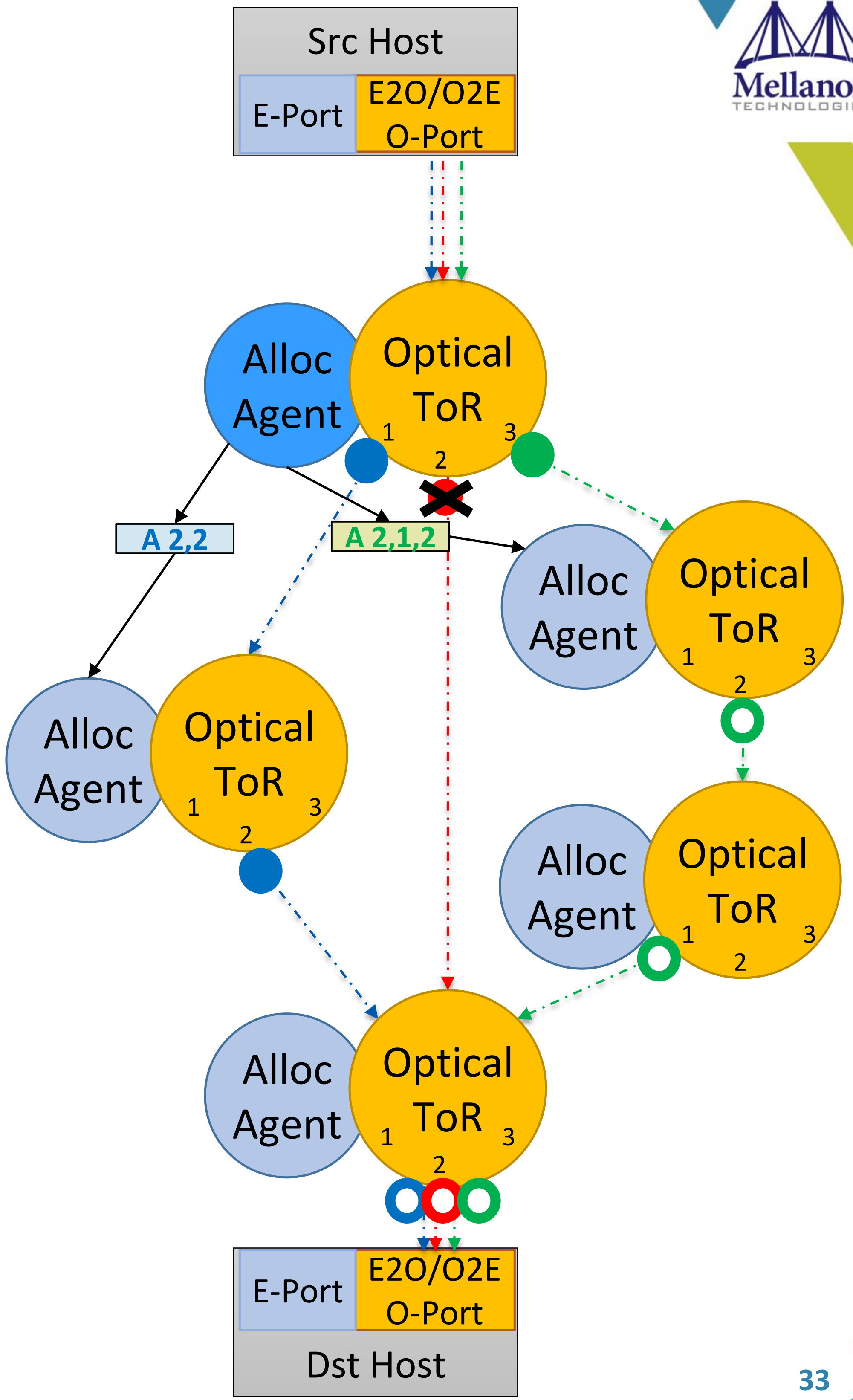
SoX Scheduling

- Distributed and Scalable
  - K Optical switches are managed by a single Allocation Agent (AA)
  - The more Optical Switches the more agents
- High Level Algorithm
  - Try N pre-calculated paths from source to destination
  - Paths colors are preassigned
  - AAs tracks the used colors on each output of the Optical Switches it manages
  - When a request arrives to an AA it looks to see if the Output/Color pair are free
    - If they are free
      - Reserve the color and send the request to next Agent or back to the origin if none
      - Configure the optical switch accordingly
    - If not free send Backward Cancellation to previous AA
  - When AA receive Backward/Forward Cancellation request
    - It send it to previous/next AA (if not last)
  - Requestor uses the first reserved path



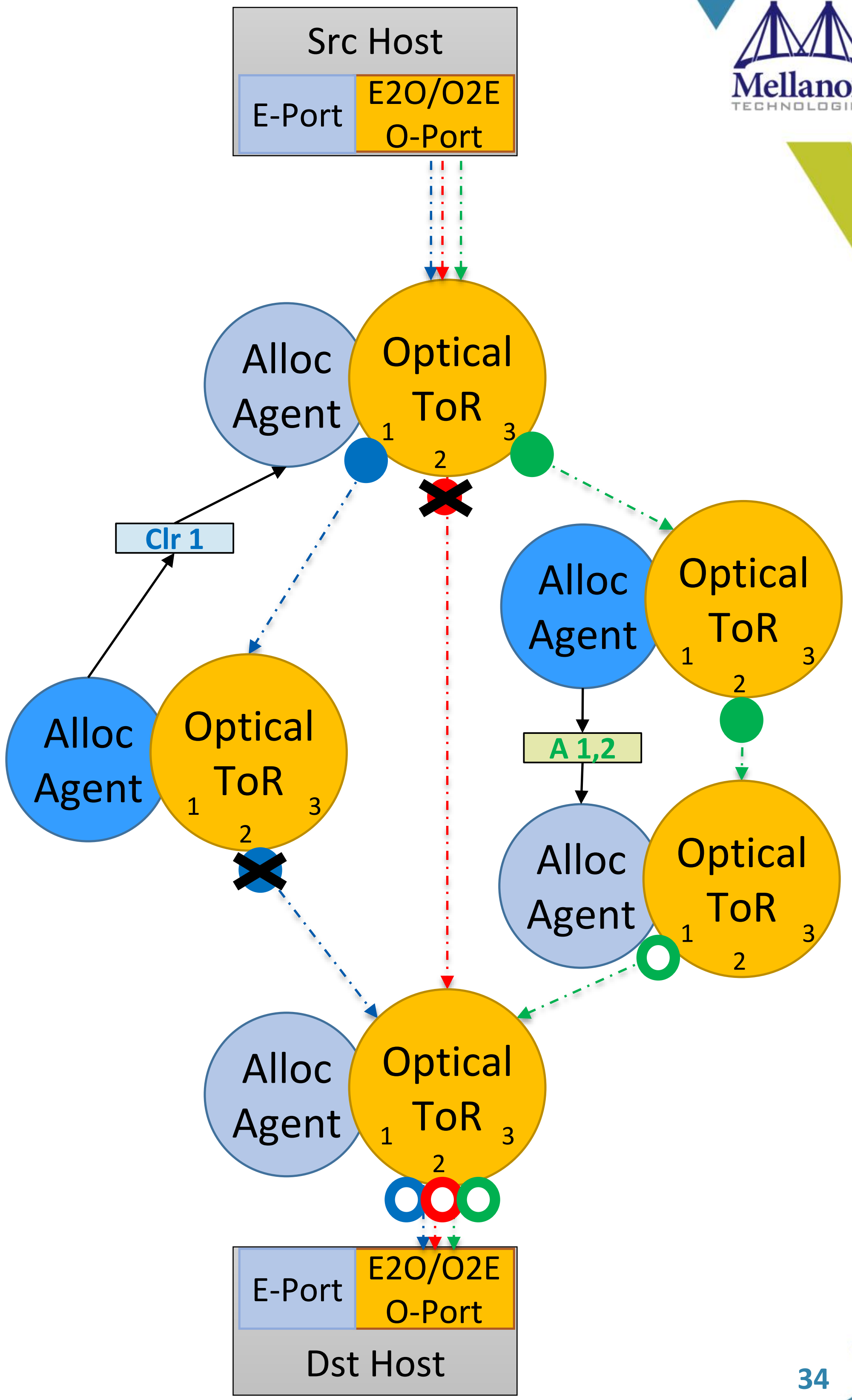
SoX Scheduling

- Distributed and Scalable
  - K Optical switches are managed by a single Allocation Agent (AA)
  - The more Optical Switches the more agents
- High Level Algorithm
  - Try N pre-calculated paths from source to destination
  - Paths colors are preassigned
  - AAs tracks the used colors on each output of the Optical Switches it manages
  - When a request arrives to an AA it looks to see if the Output/Color pair are free
    - If they are free
      - Reserve the color and send the request to next Agent or back to the origin if none
      - Configure the optical switch accordingly
    - If not free send Backward Cancellation to previous AA
  - When AA receive Backward/Forward Cancellation request
    - It send it to previous/next AA (if not last)
  - Requestor uses the first reserved path



SoX Scheduling

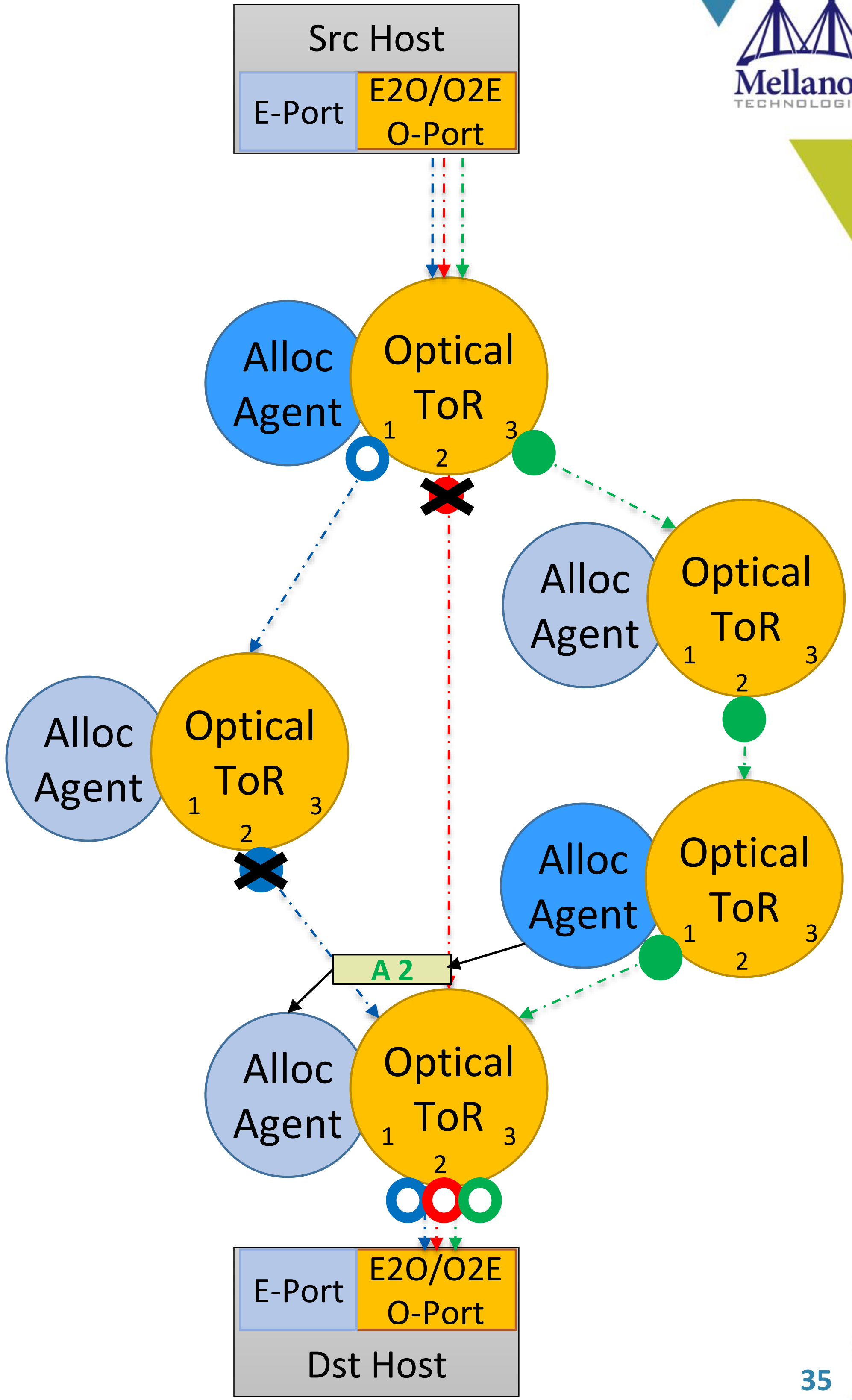
- Distributed and Scalable
  - K Optical switches are managed by a single Allocation Agent (AA)
  - The more Optical Switches the more agents
- High Level Algorithm
  - Try N pre-calculated paths from source to destination
  - Paths colors are preassigned
  - AAs tracks the used colors on each output of the Optical Switches it manages
  - When a request arrives to an AA it looks to see if the Output/Color pair are free
    - If they are free
      - Reserve the color and send the request to next Agent or back to the origin if none
      - Configure the optical switch accordingly
    - If not free send Backward Cancellation to previous AA
  - When AA receive Backward/Forward Cancellation request
    - It send it to previous/next AA (if not last)
  - Requestor uses the first reserved path



SoX Scheduling

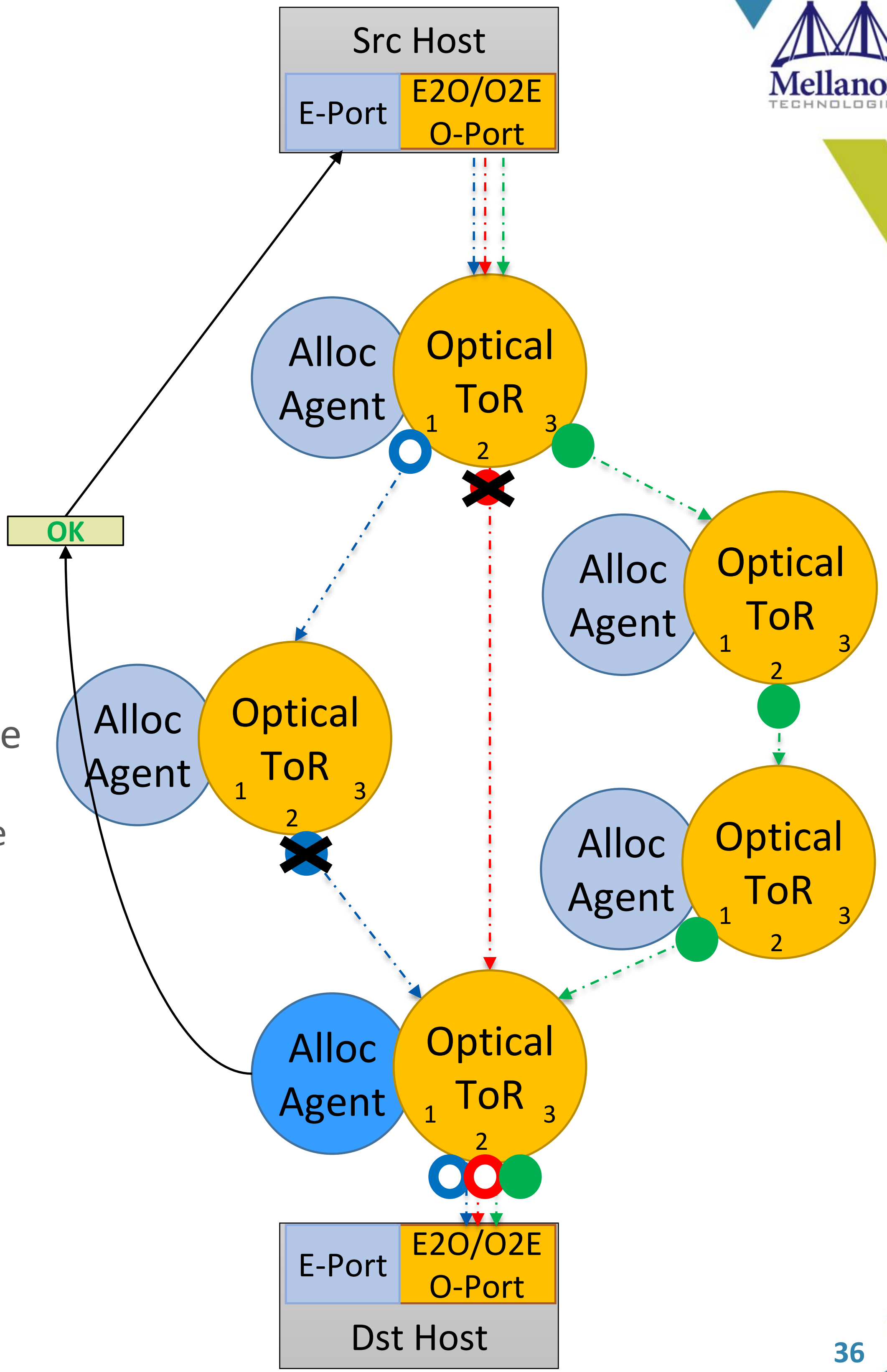
- Distributed and Scalable
  - K Optical switches are managed by a single Allocation Agent (AA)
  - The more Optical Switches the more agents

- High Level Algorithm
  - Try N pre-calculated paths from source to destination
  - Paths colors are preassigned
  - AAs tracks the used colors on each output of the Optical Switches it manages
  - When a request arrives to an AA it looks to see if the Output/Color pair are free
    - If they are free
      - Reserve the color and send the request to next Agent or back to the origin if none
      - Configure the optical switch accordingly
    - If not free send Backward Cancellation to previous AA
  - When AA receive Backward/Forward Cancellation request
    - It send it to previous/next AA (if not last)
  - Requestor uses the first reserved path



SoX Scheduling

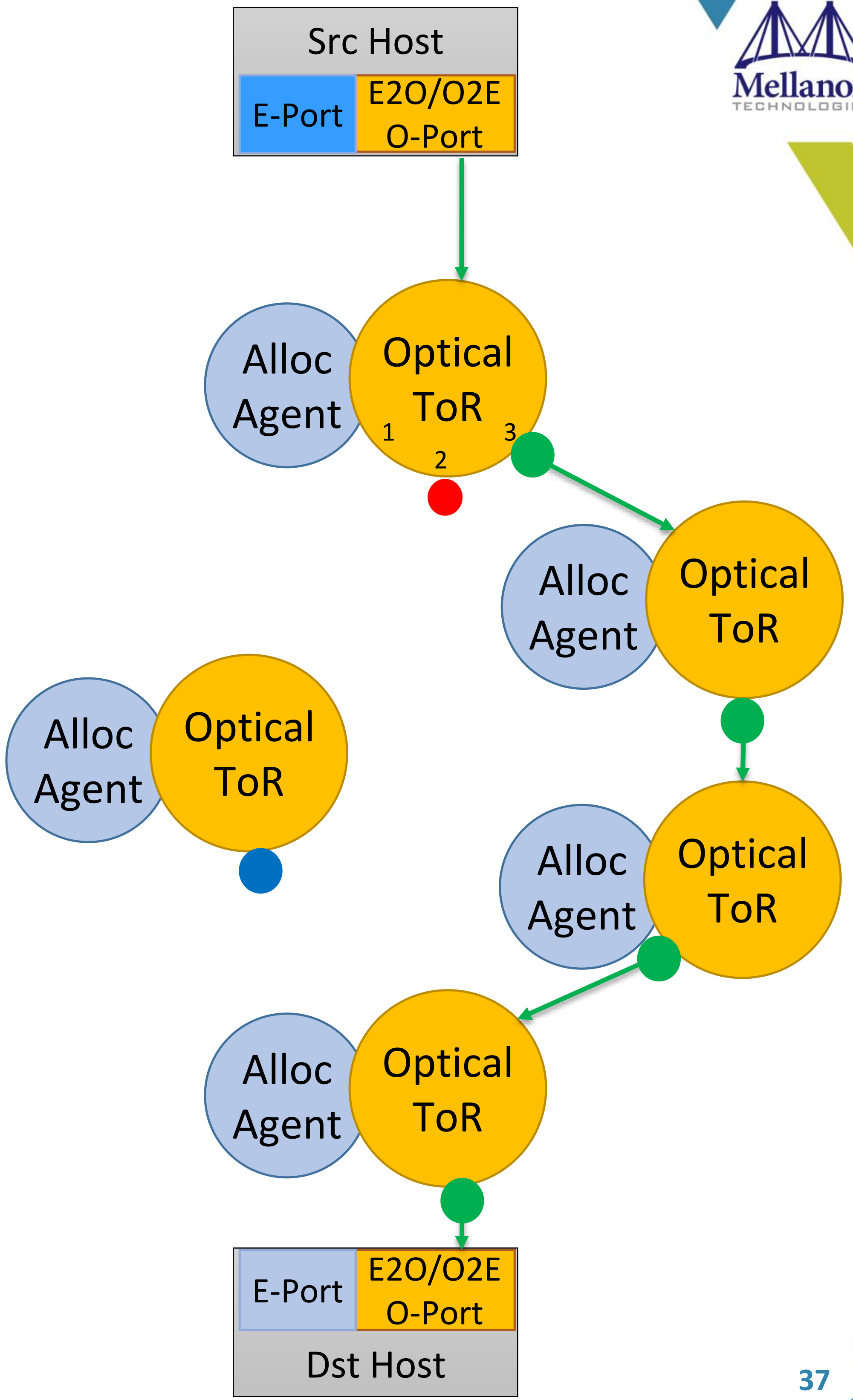
- Distributed and Scalable
  - K Optical switches are managed by a single Allocation Agent (AA)
  - The more Optical Switches the more agents
- High Level Algorithm
  - Try N pre-calculated paths from source to destination
  - Paths colors are preassigned
  - AAs tracks the used colors on each output of the Optical Switches it manages
  - When a request arrives to an AA it looks to see if the Output/Color pair are free
    - If they are free
      - Reserve the color and send the request to next Agent or back to the origin if none
      - Configure the optical switch accordingly
    - If not free send Backward Cancellation to previous AA
  - When AA receive Backward/Forward Cancellation request
    - It send it to previous/next AA (if not last)
  - Requestor uses the first reserved path



SoX Scheduling

- Distributed and Scalable
  - K Optical switches are managed by a single Allocation Agent (AA)
  - The more Optical Switches the more agents

- High Level Algorithm
  - Try N pre-calculated paths from source to destination
  - Paths colors are preassigned
  - AAs tracks the used colors on each output of the Optical Switches it manages
  - When a request arrives to an AA it looks to see if the Output/Color pair are free
    - If they are free
      - Reserve the color and send the request to next Agent or back to the origin if none
      - Configure the optical switch accordingly
    - If not free send Backward Cancellation to previous AA
  - When AA receive Backward/Forward Cancellation request
    - It send it to previous/next AA (if not last)
  - Requestor uses the first reserved path



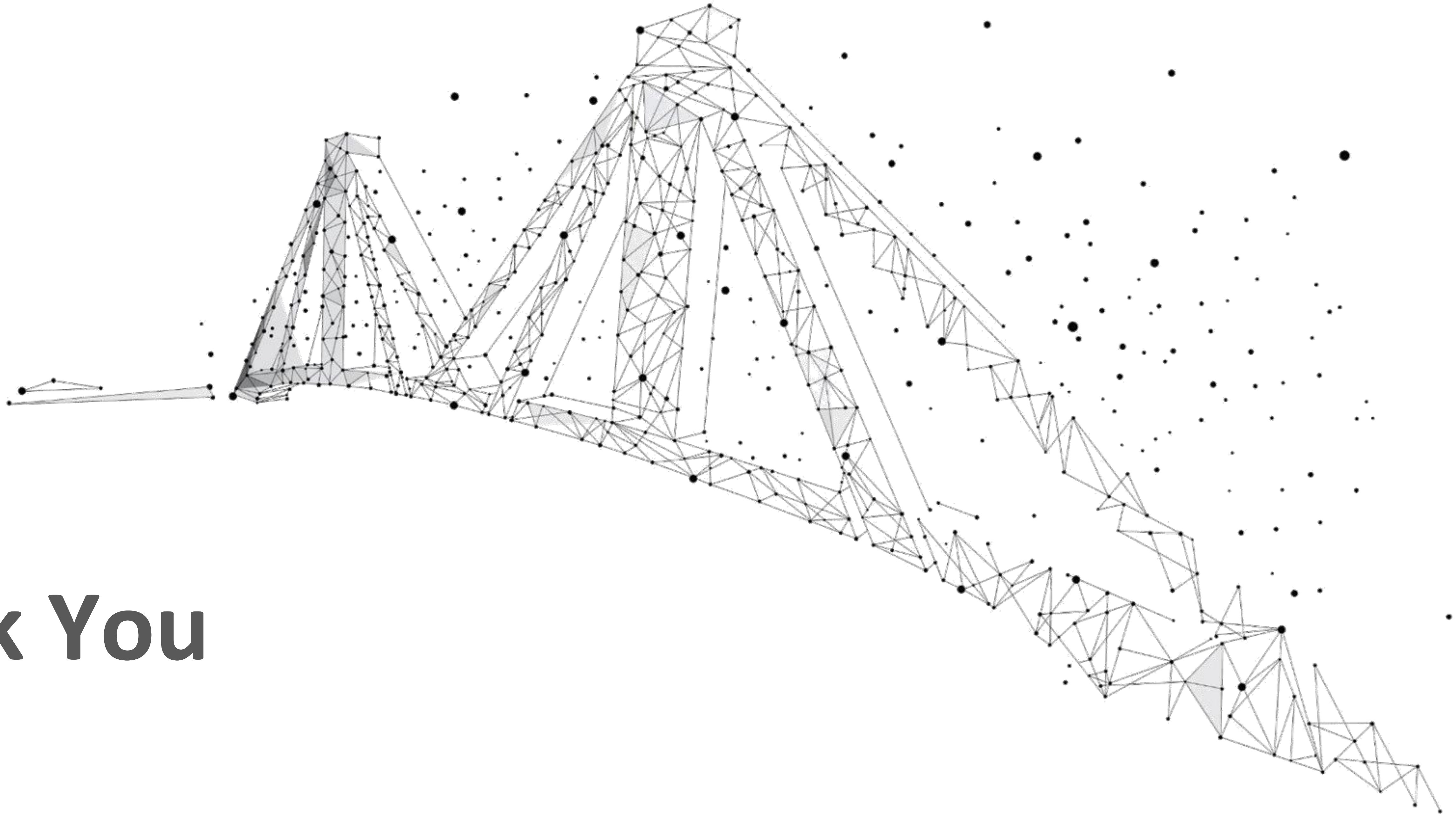
Conclusions

- Central Scheduler Architectures reach a Dead End
  - New architectural innovations overcome that
    - RotorNet – fixed schedule
    - Distributed Scheduling
      - However, using Electrical Switches as ToRs is contradicting to our main ODCN motivation
        - Optical Network Directly attached to the Host is avoiding the bottleneck

SOX = Server attached Optical Xpander

No bandwidth bottleneck

Distributed Scheduling



Thank You