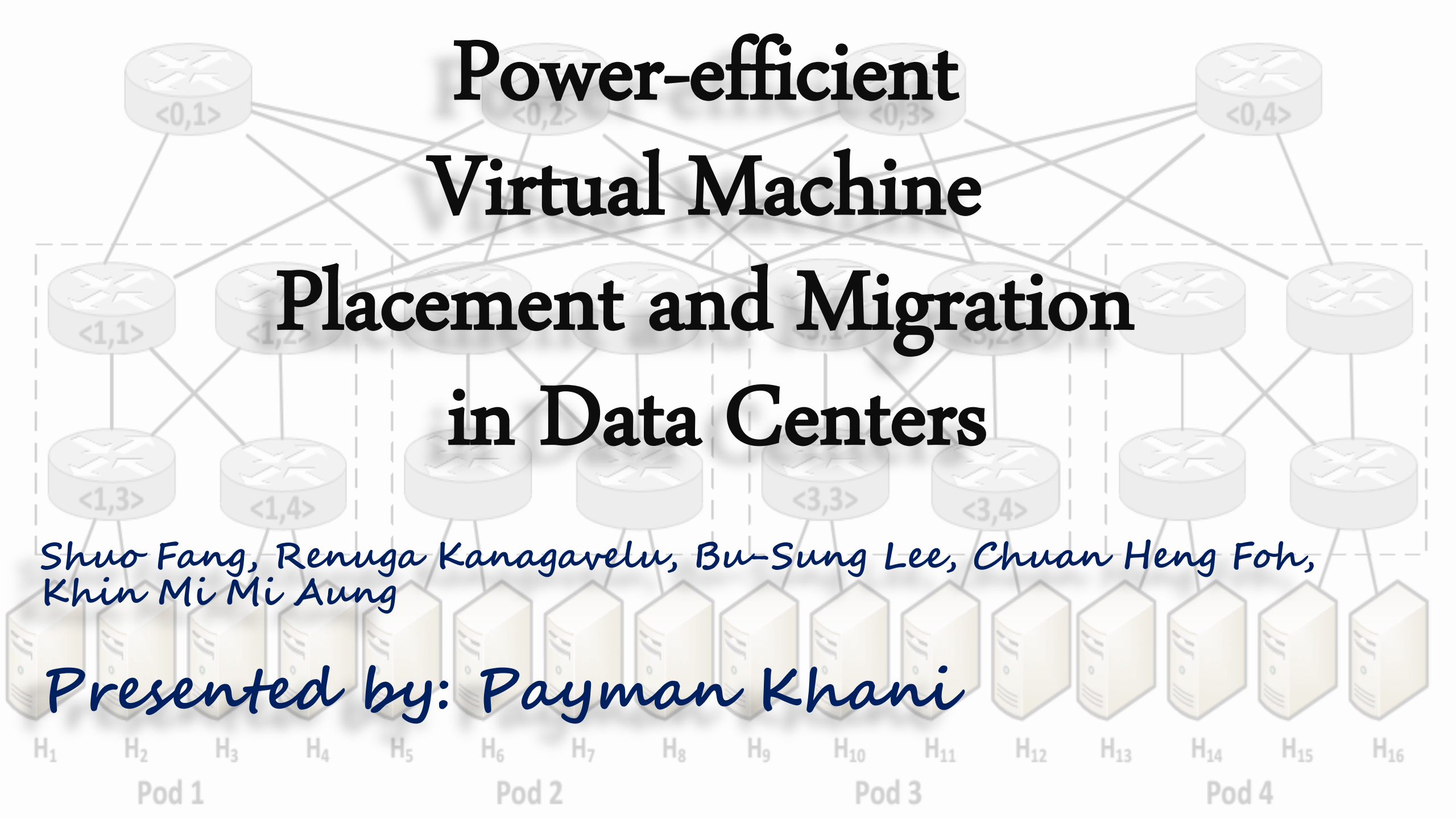
Power-efficient Virtual Machine Placement and Migration in Data Centers

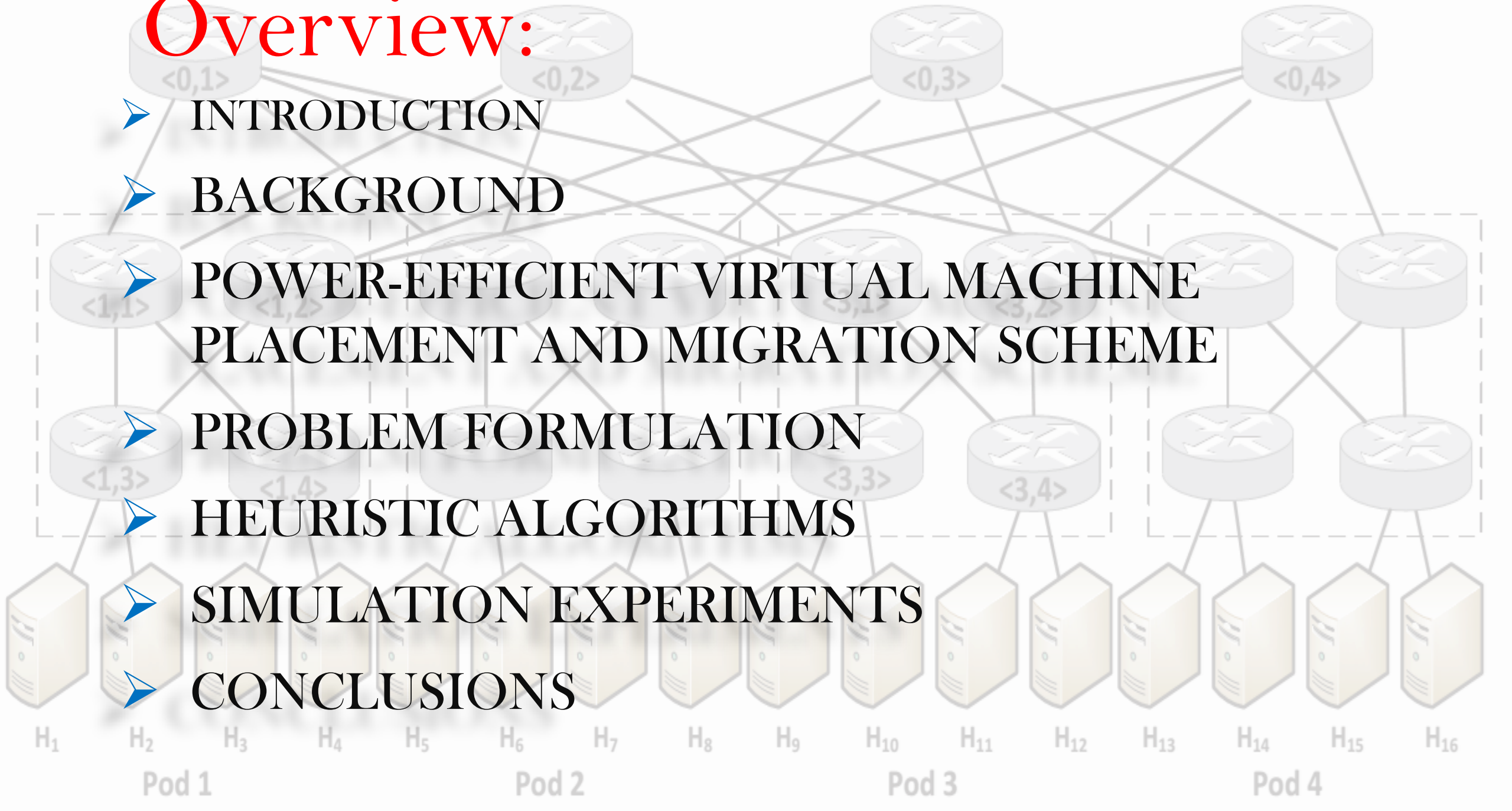
Shuo Fang, Renuga Kanagavelu, Bu-Sung Lee, Chuan Heng Foh, Khin Mi Mi Aung

Presented by: Payman Khani



Overview:

- INTRODUCTION
- BACKGROUND
- POWER-EFFICIENT VIRTUAL MACHINE PLACEMENT AND MIGRATION SCHEME
- PROBLEM FORMULATION
- HEURISTIC ALGORITHMS
- SIMULATION EXPERIMENTS
- CONCLUSIONS



INTRODUCTION

- In a data center, virtualization concerns three tiers:

server virtualization, network virtualization and storage virtualization.

- Multiple VMs may run concurrently on a single physical machine (PM).
- Two criteria:
  - ✓ Hardware resource usage(limits the number of VMs that can be allocated and their placement)
  - ✓ Power consumptions(influences the operating cost of a data center)
- Servers, storages and network devices consume around 45% of the total power in data centers.
- Another 45% of the power are consumed for cooling down these equipment.
- Server virtualization not only saves energy on powering equipment but also reduces cooling cost

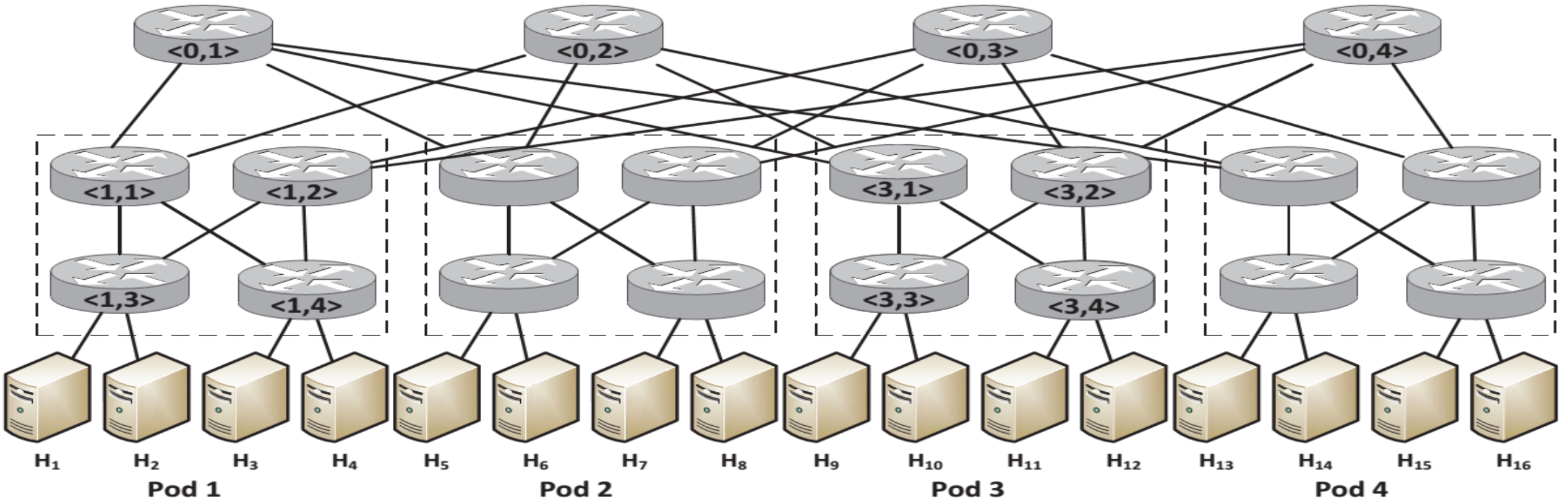
INTRODUCTION

- From the view of a whole data center, minimizing the number of active servers and switches saves the major equipment power consumption as well as cooling cost.
- A typical data center consists of three layers of switches,

edge switches, aggregate switches and core switches:

- A source-destination VM pair that close to each other can communicate through edge switches instead of travelling through core switches.
- By migrating VMs of communicating parties to a close location, core switches usage can be reduced.
- Thus, data center can save energy with the same workloads.

INTRODUCTION



- Example: Source-destination VMs are at PMs of H1and H9
- This paper aims to design a VM placement and migration solution that ensures power efficiency in data centers.

BACKGROUND

- The proposal is based on:

1 - Fat tree topology(an attractive solution for modern data centers)

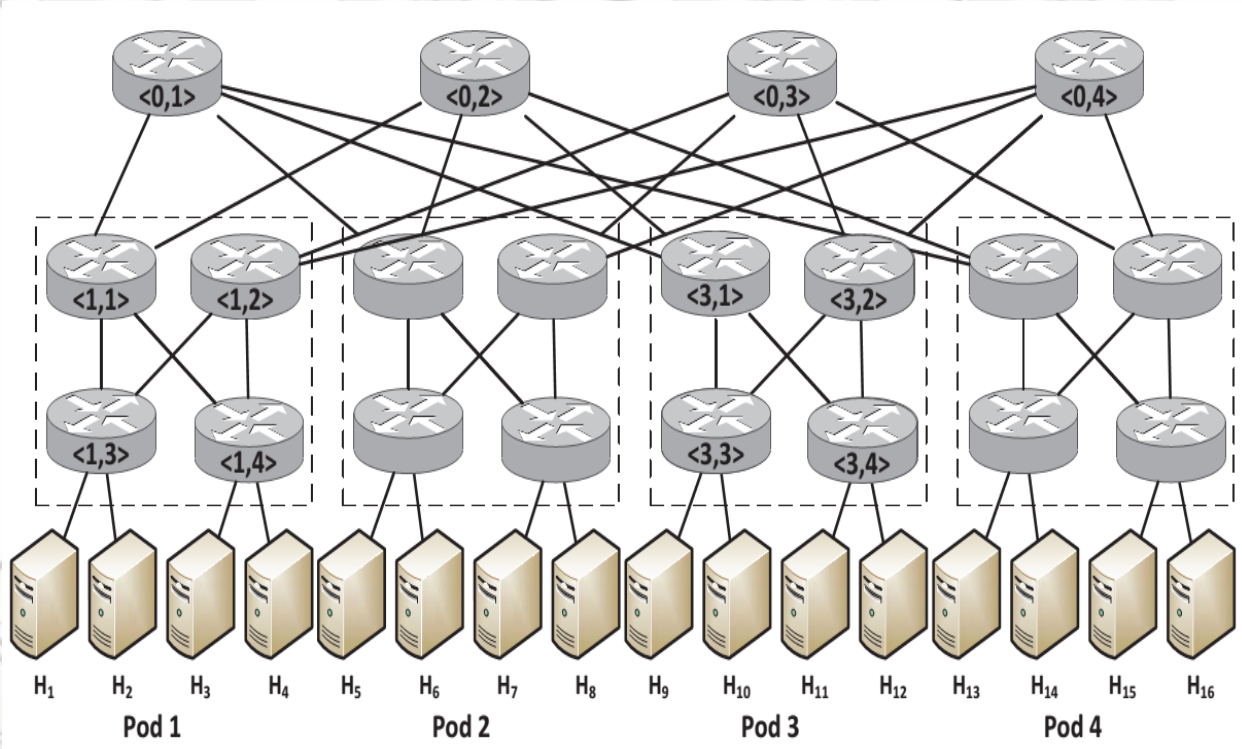
N–ary fat tree:

- Contains three layers of switches (edge layer, aggregation layer and core layer)

- The lower two layers are separated into 2N pods, each containing two layers of N switches, lower edge switches and upper layer aggregation switches.

- In the upmost layer, there are N2 core switches, each core switch has a connection towards each of the2N pods.

- Since each edge switch connects to N end stations and each pod has N edge switches, their areN^2 end stations in a pod and 2N^3 end stations in an N=2 data center fat tree topology.



BACKGROUND

2 - OpenFlow Protocol(a software defined networking technology for customize network protocols)

- Different from traditional network in which data plane and control plane are implemented on hardware, SDN decouples intelligence that controls the flow to a distinct software application and handles packet traffic on hardware.

- Hence, SDN provides much easier modification to switch algorithms, as well as faster control over network status changes.

- OpenFlow project, the most popular enabler for SDN.

Installing a small piece of OpenFlow firmware gives engineers access to flow tables, rules for routing traffic.

POWER-EFFICIENT VIRTUAL MACHINE PLACEMENT AND MIGRATION SCHEME

- The proposal aims to schedule VMs with power efficient concerns.
- It attempts to utilize fewer PMs and schedule VMs to migrate when necessary.
- TERMS:
- **Job:** The work load in a data center. (Ex: Email service, VoIP service)
  - ✓ To complete a job, different types of VMs are required.
  - ✓ Job information is usually described with three entries:
  - ✓ Job ID, Job Priority and VM List (VM list lists number of VMs on each type that are required by this job.)
- **Virtual Machine (VM)** : A working unit which requires several system resources.
  - ✓ Different types of VM demands various system resource.
  - ✓ A type of VM is defined by parameters such as VM Type ID, CPU, RAM, storage and bandwidth.
- **Hypervisor :** A piece of computer software, firmware or hardware that creates and runs virtual machines. A hypervisor is running on a PM and creates VMs on it. Hypervisors track PM resources in a structure including Hypervisor ID, CPU,RAM, storage and bandwidth.

POWER-EFFICIENT VIRTUAL MACHINE PLACEMENT AND MIGRATION SCHEME

- Job Requirements and Process:
- To process a job, the OpenFlow controller deploys all the VMs required.
- To deploy a VM on a PM, current resources on PM must be sufficient to meet VM requirement.
- A job can only work when all its VMs are deployed.
- If any one of the VM cannot be deployed, this job will suspend for a random time before checking to execute again.
- A job is processed through two stages:
  - ✓ VM placement: Happens when a job is launched. In this stage,VMs are initially scheduled and deployed.
  - ✓ VM migration: Happens during job execution when flows are initiated, VMs might be rescheduled to form more inner pod flows thus less core switches are needed

PROBLEM FORMULATION

- Goal : reducing the network energy and the end-to-end delay.
- Problem Statement:
  - ✓ A directed graph G=(V,E) [v ∈ V represents a network switch, PM or an external client.
  - ✓ Each directed edge (u, v) ∈E represents a physical link between the switches and between a switches and a host.
  - ✓ Considering a set of t hosts are available and their resource capacities given along CPU, memory, storage and Network bandwidth dimensions.
  - ✓ We have to find a mapping between VMs and PMs that satisfies the VMs' resource requirements while minimizing the number of PMs used, minimizing data center network energy, and minimizing the end-to-end delay.

PROBLEM FORMULATION

- Input:
  - ✓ *t* PMs S ={S1,S2...St} and each host Si ={ci,ri,si,bi,ui}with different resource capacity attributes: ci for CPU, ri for RAM, si for storage, and bi for bandwidth. Each server has a Capped Resource Utilization value, ui (default at 95%) beyond which no more resources are allocated.
  - ✓ *m* different VM types depend on their resource needed (VMs) V={V1,V2...Vm}. Each VM type Vi ={cvi,rvi,svi,bvi} .
  - ✓ Set of n jobs need to be assigned to a data center, J={J1,J2..,Jn}
  - ✓ Each job Ji with a list of VM types, Ji ={x1,x2...xm}
  - ✓ Number of requested VMs for that type,∀k, xk is the number of VM type Vk needed by Job Ji.
  - ✓ Hosti ={hosti1, ..., hostim} for each server Si that denotes the set of VMs assigned to that server.
  - ✓ The power consumed by server Si depends on its physical component as well as on the set of VMs present: P(Si)=P(Si,Hosti)

HEURISTICALGORITHMS

1. Virtual Machine Placement:

This stage is invoked when a job is launched. Two mechanisms:

- ✓ Random Placement: randomly selects a hypervisor; if the selected hypervisor cannot satisfy the VM requirements, the mechanism continuously explore other hypervisors.
- ✓ Power-efficient Placement: it firstly checks active hypervisors and consults if the VM can be assigned to any of them. A new PM only starts when all the active hypervisors are unable to accommodate the VM.
  - It not only aggregate traffic to a smaller number of active PMs, but also assures the active PMs are close enough to fully utilization before activating a new PM.
  - Overall system utilization rate is also higher than Random mechanism.

HEURISTICALGORITHMS

- In Power-efficient mechanism, three algorithms are discussed:

First Fit (FF), Best Fit (BF) and Worst Fit (WF).

  - ✓ First Fit (FF): the controller deploys the VM directly to the first suitable PM.
  - ✓ Best Fit (BF): the controller deploys the VM in the PM with lowest weight
  - ✓ Worst Fit (WF): the controller deploys the VM in the PM with highest weight

- wi =Δci/ci+Δri/ri+Δsi/si+Δbi/bi [CPU, RAM, storage and bandwidth]

- When a job is completed, it signals the OpenFlow controller to schedule a VM release event.
- After VM release, active hypervisors check the VMs assigned to its hosting PM.
- In case of none VMs assigned, the hypervisor instructs the PM to switch into the inactive mode.

HEURISTICALGORITHMS

2. Virtual Machine Migration:

This stage is invoked when a flow starts.

- ✓ Since each pod contains N2 hosts, we use the PM index to check whether they are in the same pod.
- ✓ The mechanism first checks if the source and destination VM are in the same pod. Otherwise, they attempts to migrate either source VM or destination VM to an active PM in the same pod of the other.
- ✓ If neither source pod nor destination pod can accommodate the other VM, the algorithm tries to allocate either VM to an inactive PM.

SIMULATION EXPERIMENTS

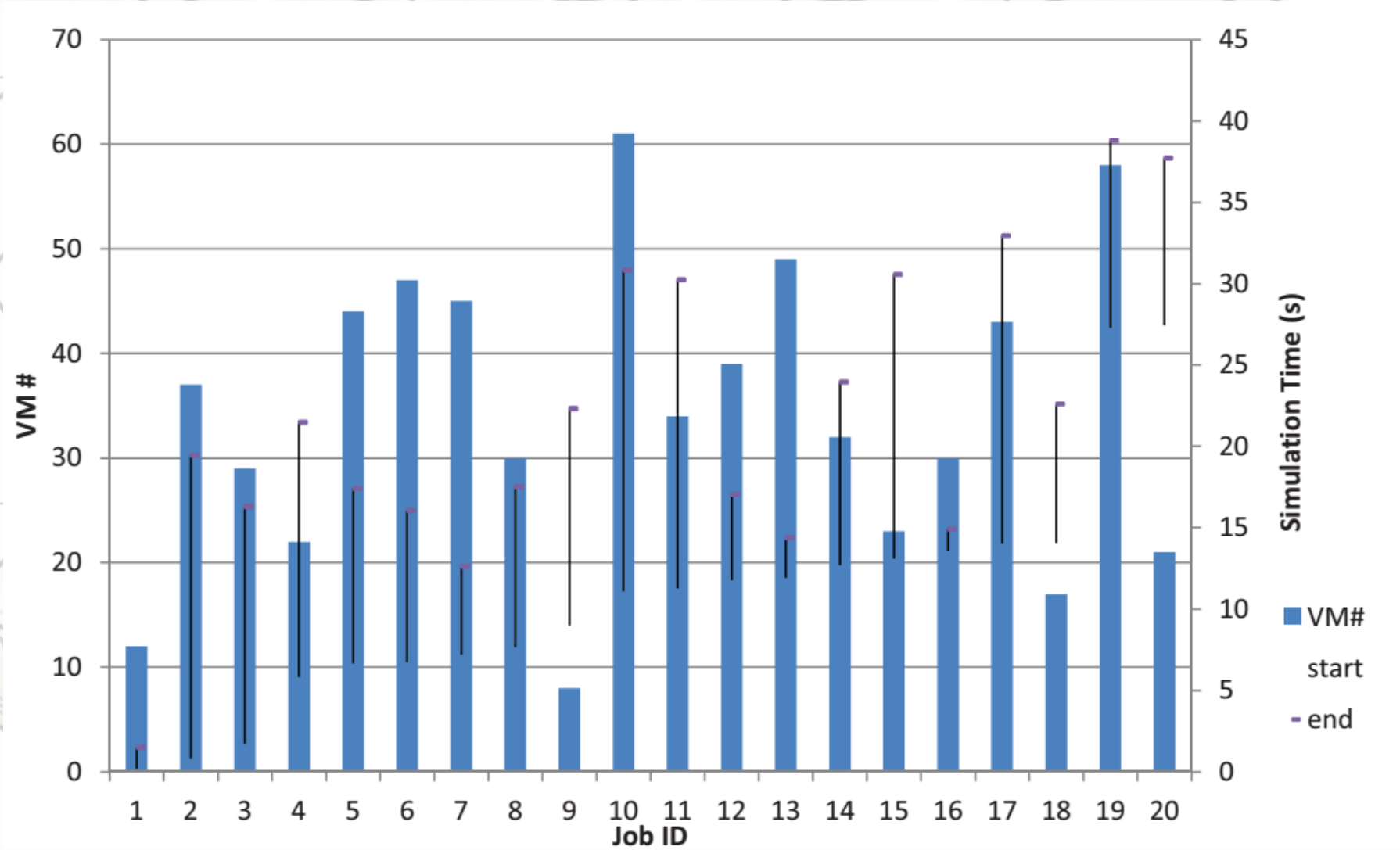
- Two sets of experiments:
  - ✓ N=4 fat tree topology where hosts 128 PMs.
  - ✓ N=8 fat tree topology where1024 PMs are hosted.
- Two perspectives:

1) Active PM number: Indicates the server power consumption level in a data center.
2) Inner pod flow percentage: shows the percentage of flows that travels within a pod instead of traveling across different pods.

- A higher inner pod flow percentage indicates less core switch power consumption.

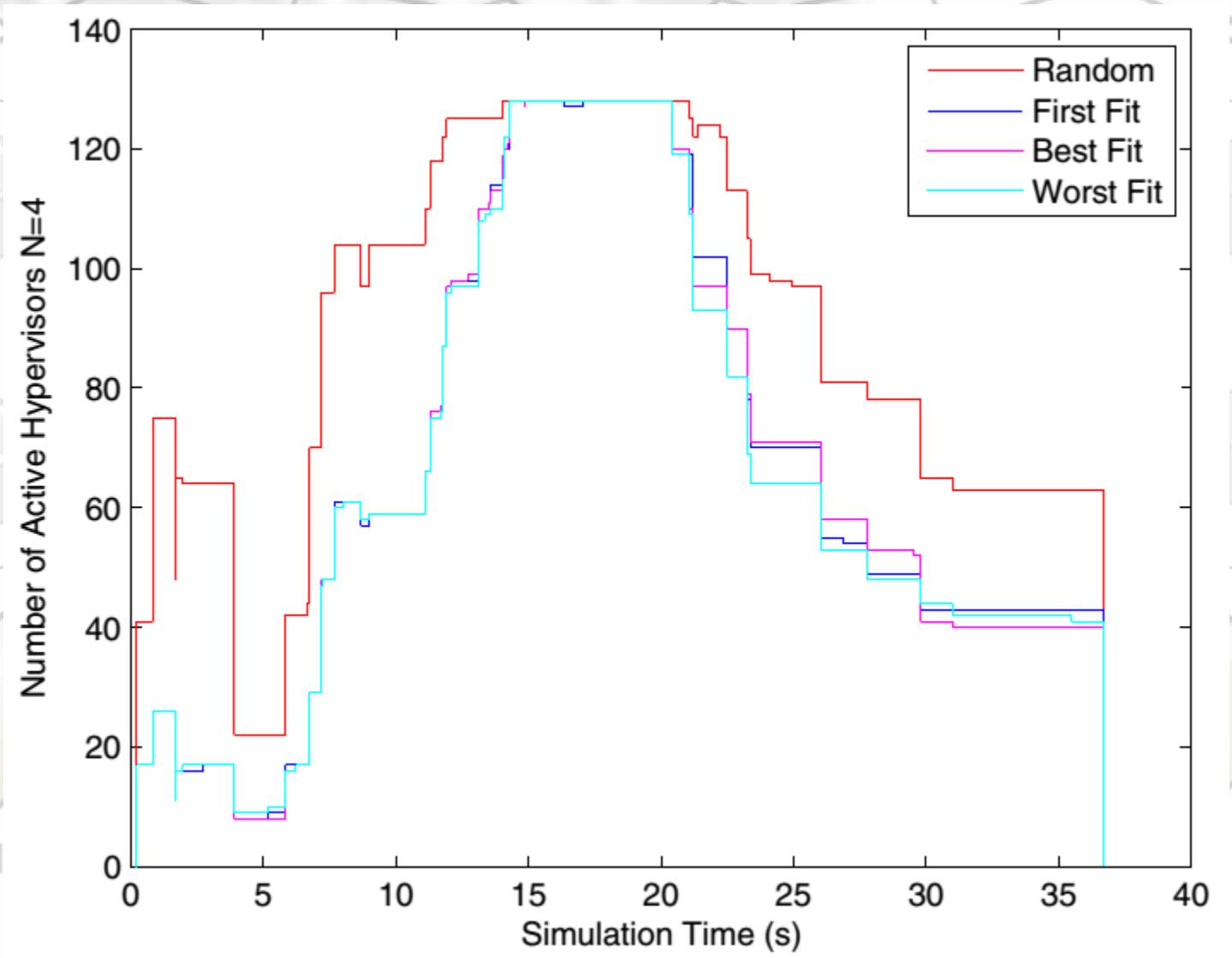
SIMULATION EXPERIMENTS

- Job start/end time and VM number illustration



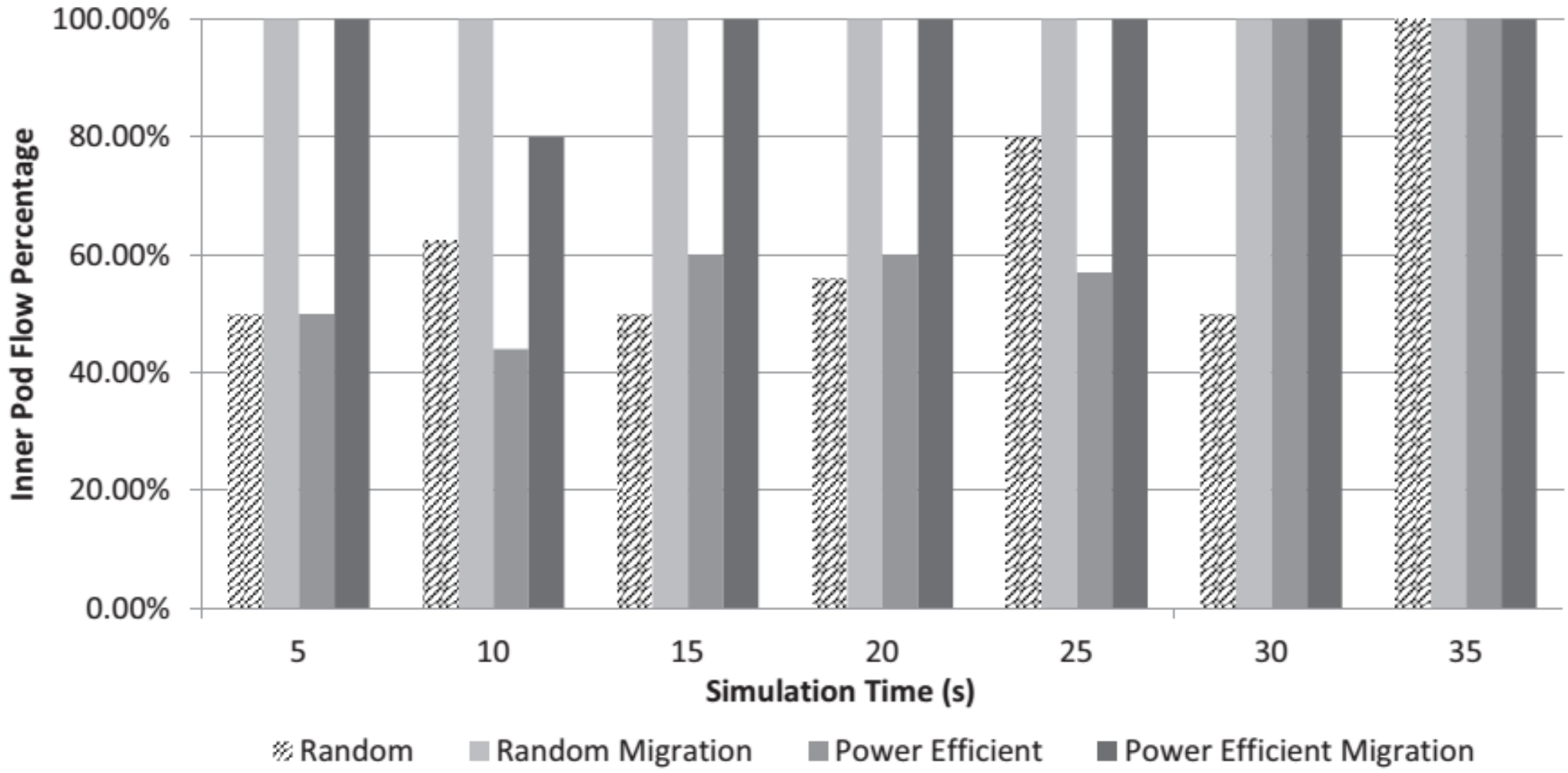
SIMULATION EXPERIMENTS

- Active PM number for N=4 fat tree



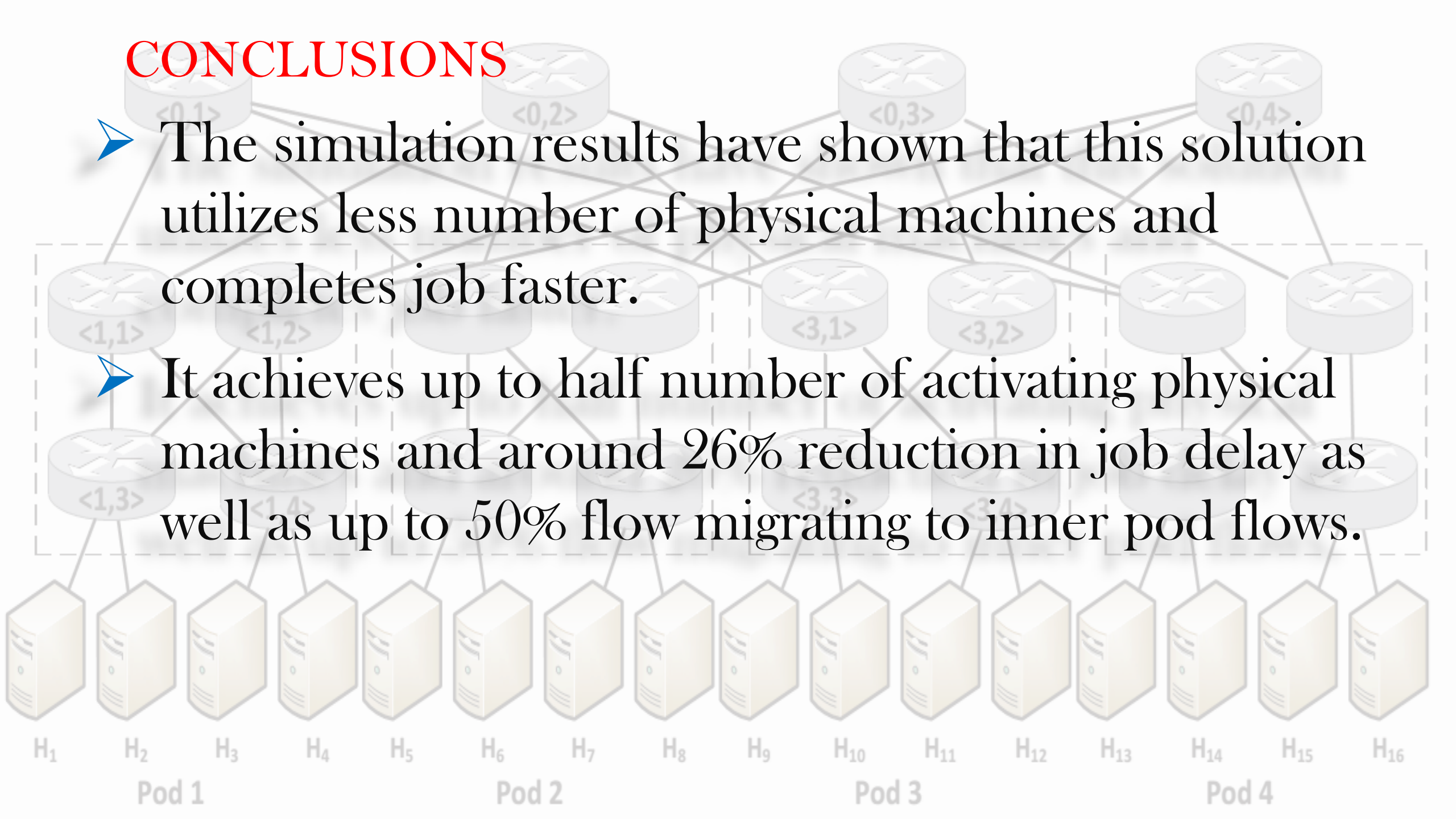
SIMULATION EXPERIMENTS

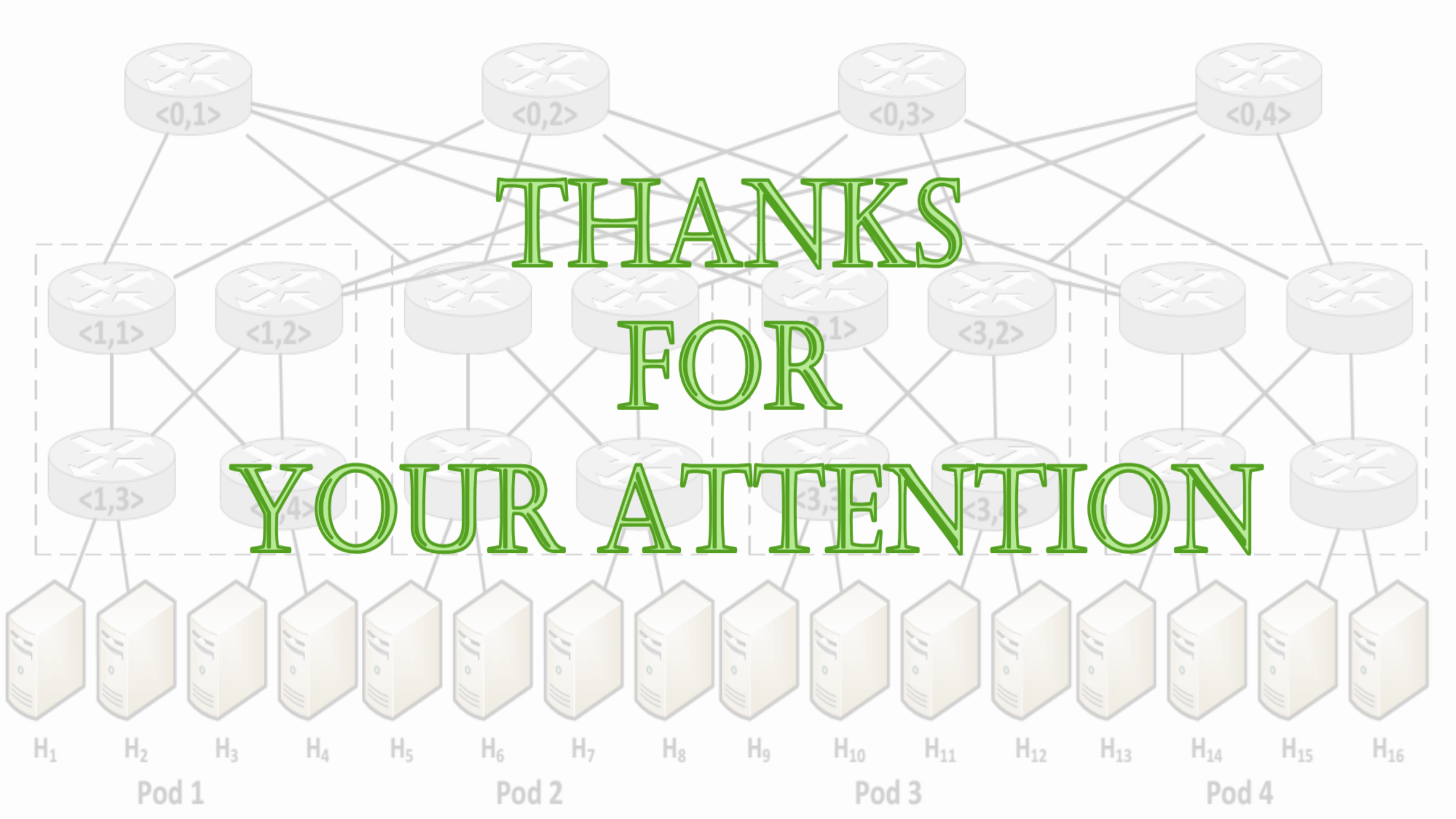
- Inner pod flow percentage when N=4



CONCLUSIONS

- The simulation results have shown that this solution utilizes less number of physical machines and completes job faster.
- It achieves up to half number of activating physical machines and around 26% reduction in job delay as well as up to 50% flow migrating to inner pod flows.





THANKS FOR YOUR ATTENTION