THE UNIVERSITY of EDINBURGH
informatics

icsa Institute for Computing Systems Architecture

# A Mixture of Experts Approach for Runtime Mapping in Dynamic Environments


Murali Emani

*School of Informatics*

***University of Edinburgh***

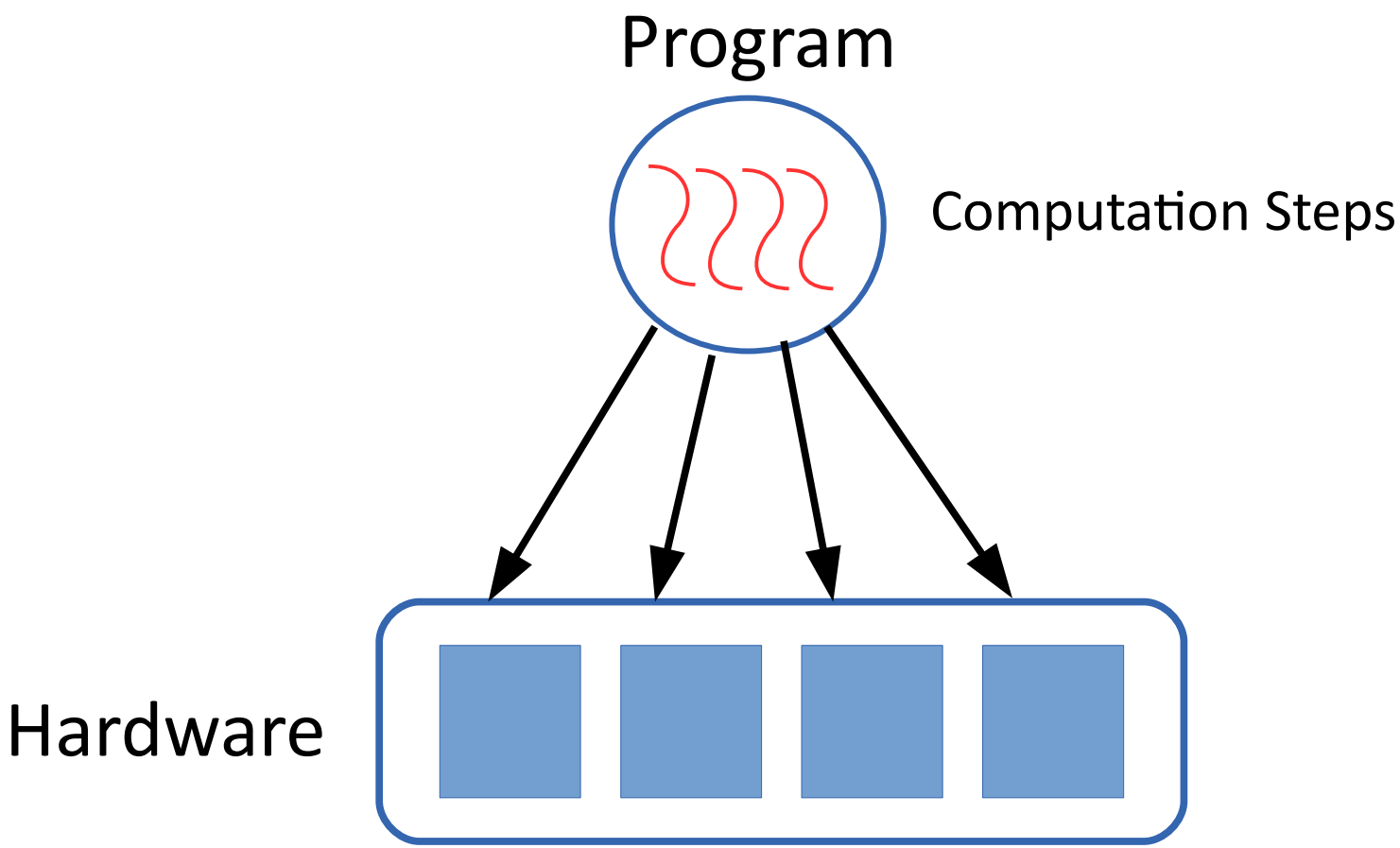

# Modern computing hardware

Diverse

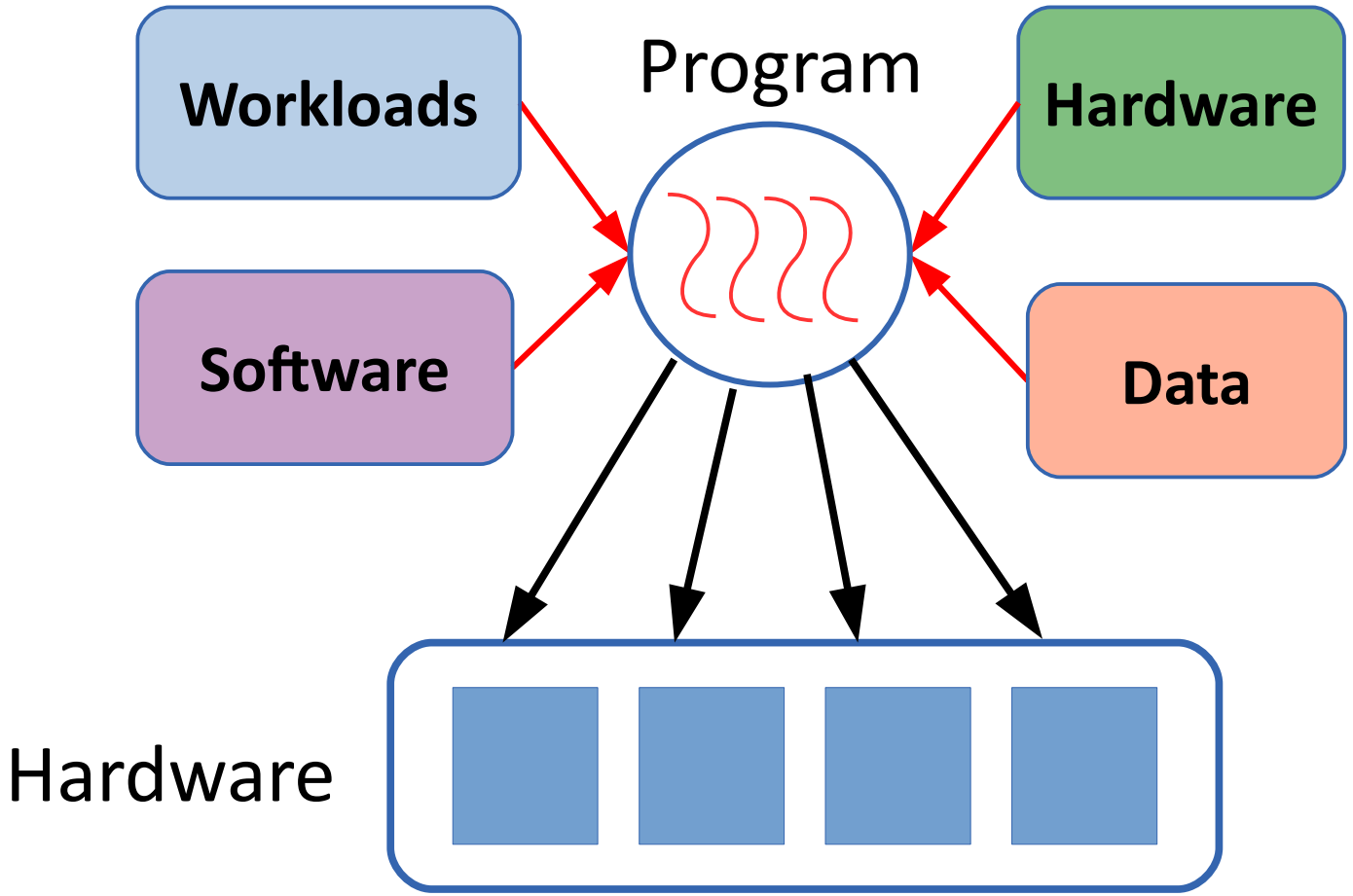
Stochastic

Evolving

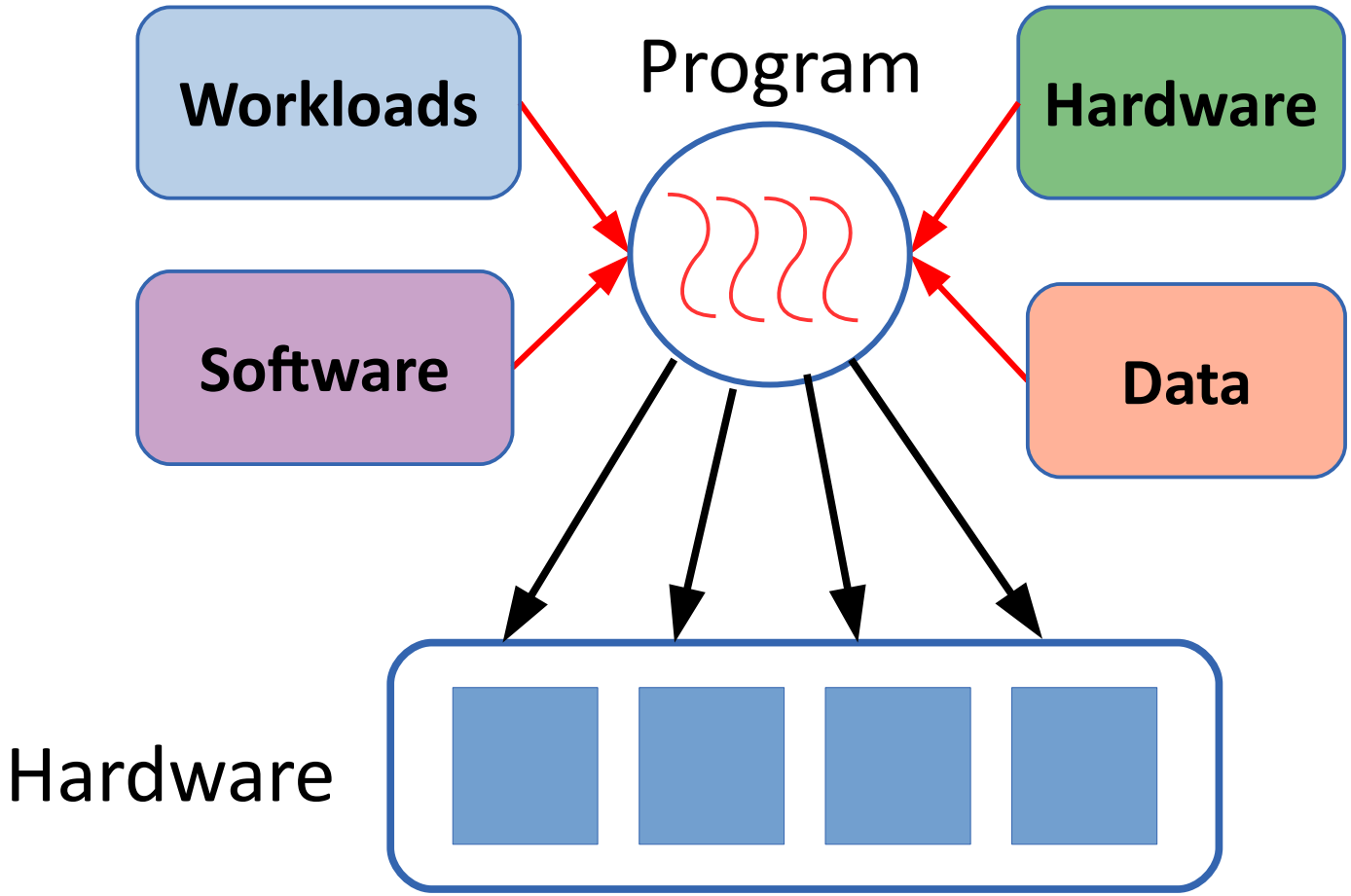
# Parallelism Mapping



# Parallelism Mapping



# Parallelism Mapping




Program performance is sensitive to the environment

# What exactly is the problem ?

Optimal partitioning of the parallel work is

*not static* *and* *non-trivial*

# What exactly is the problem ?

Existing approaches are based on

one-size-fits-all policy

# What exactly is the problem ?

Existing approaches are based on

one-size-fits-all policy

- Not suitable for dynamic environments
- Hard to extend and update

# Goals

- Determine optimal resources for a parallel program

  *Avoid under-subscription / over-subscription*

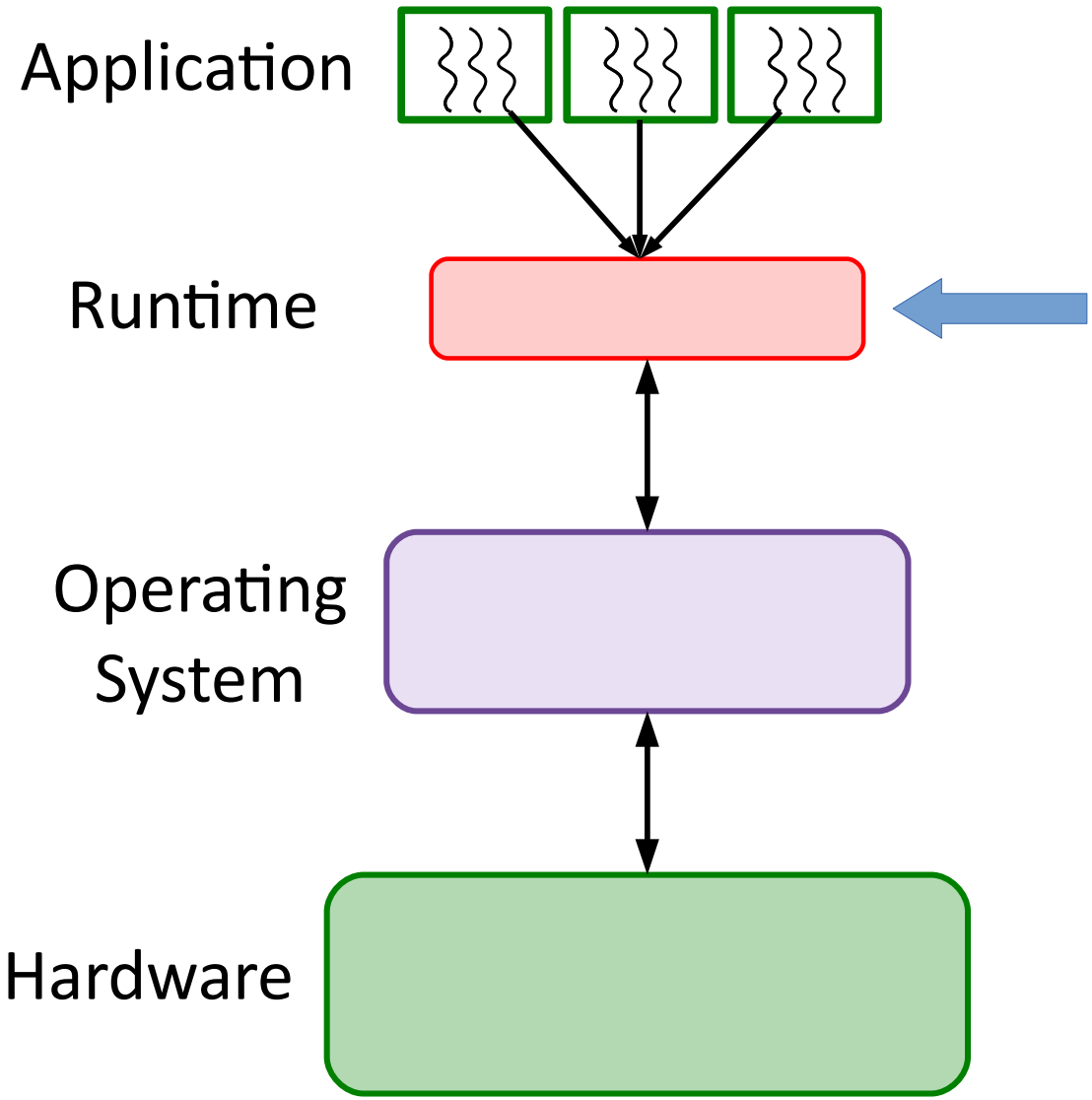
- Enable program auto-tuning

  *Adapt smartly to varying resources*

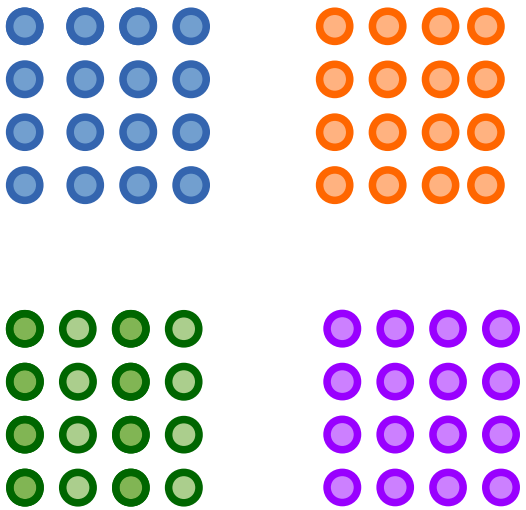
- Program and Platform aware

  *Generic and portable*

# Where does it fit in the stack

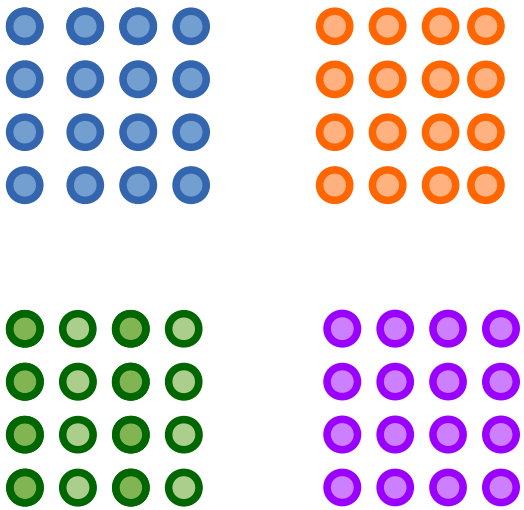


# State Space



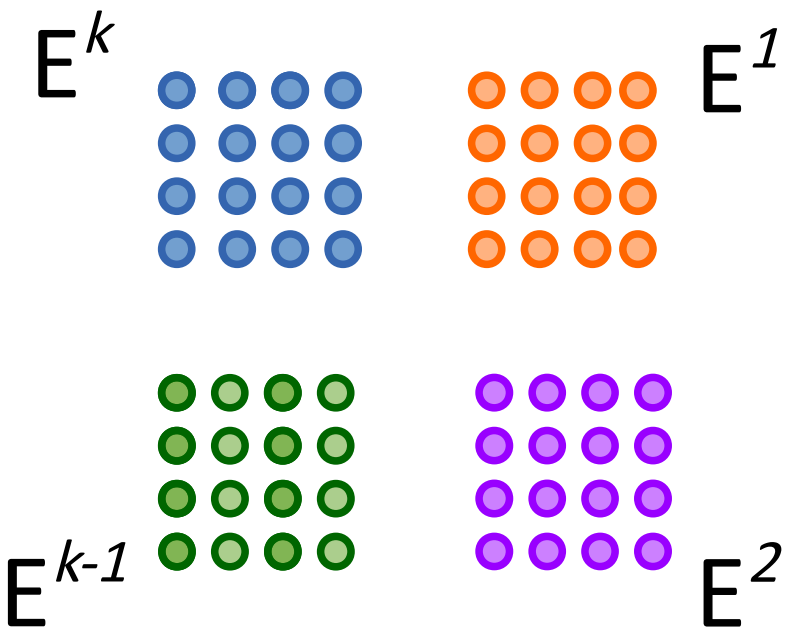
# Idea

- Identify best mapping policy in each set



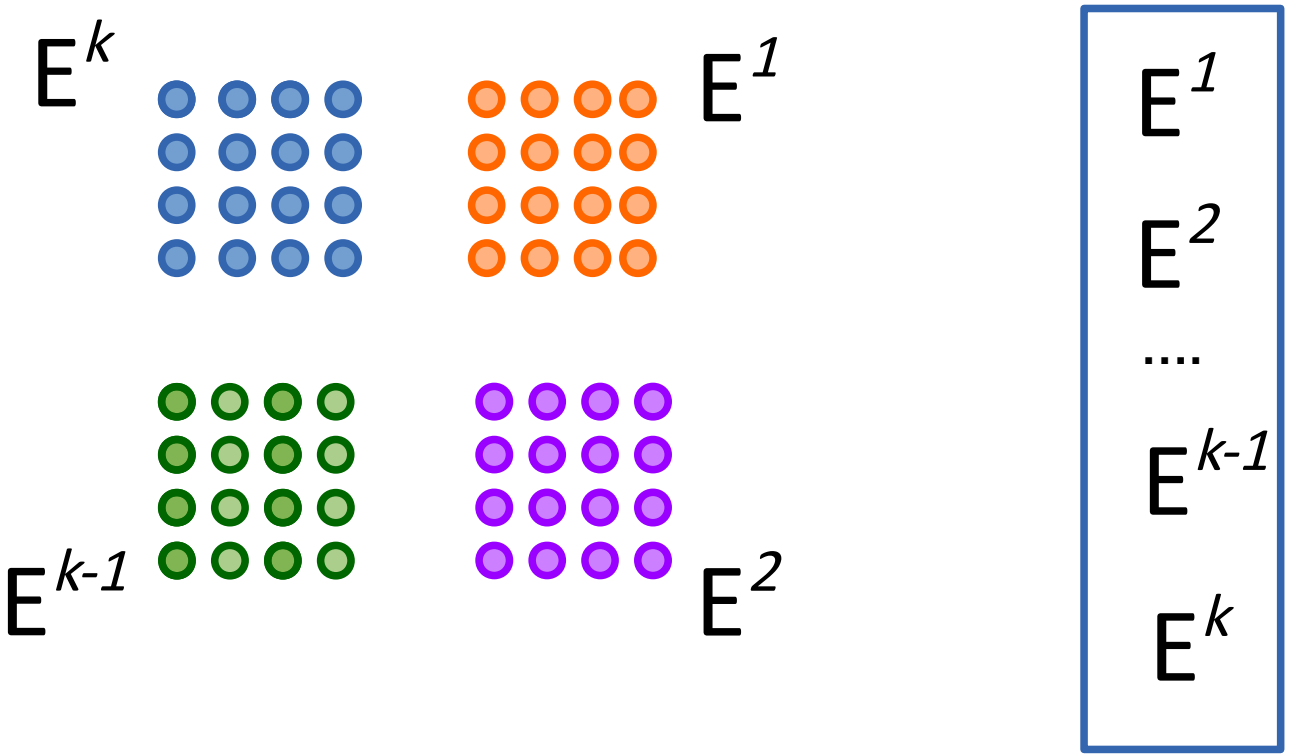
# Idea

- Identify best mapping policy in each set



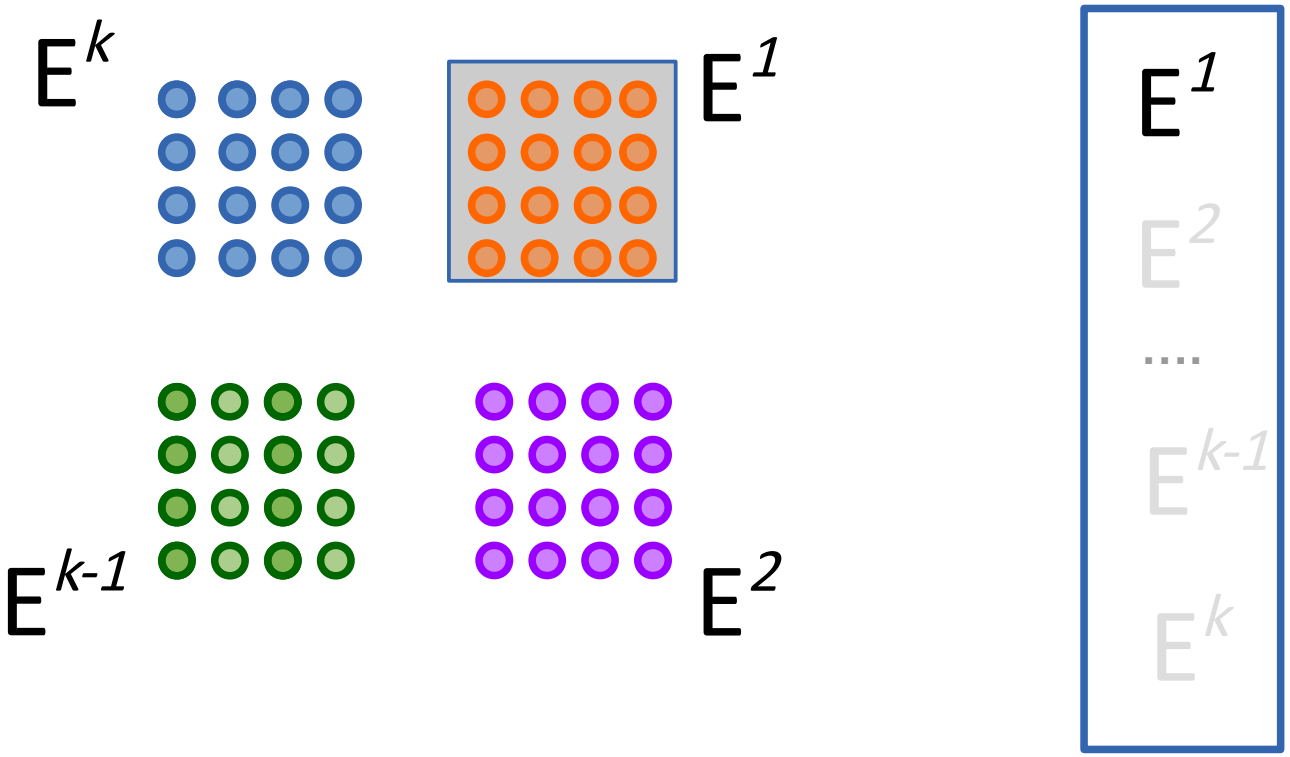
# Idea

- Collect these policies



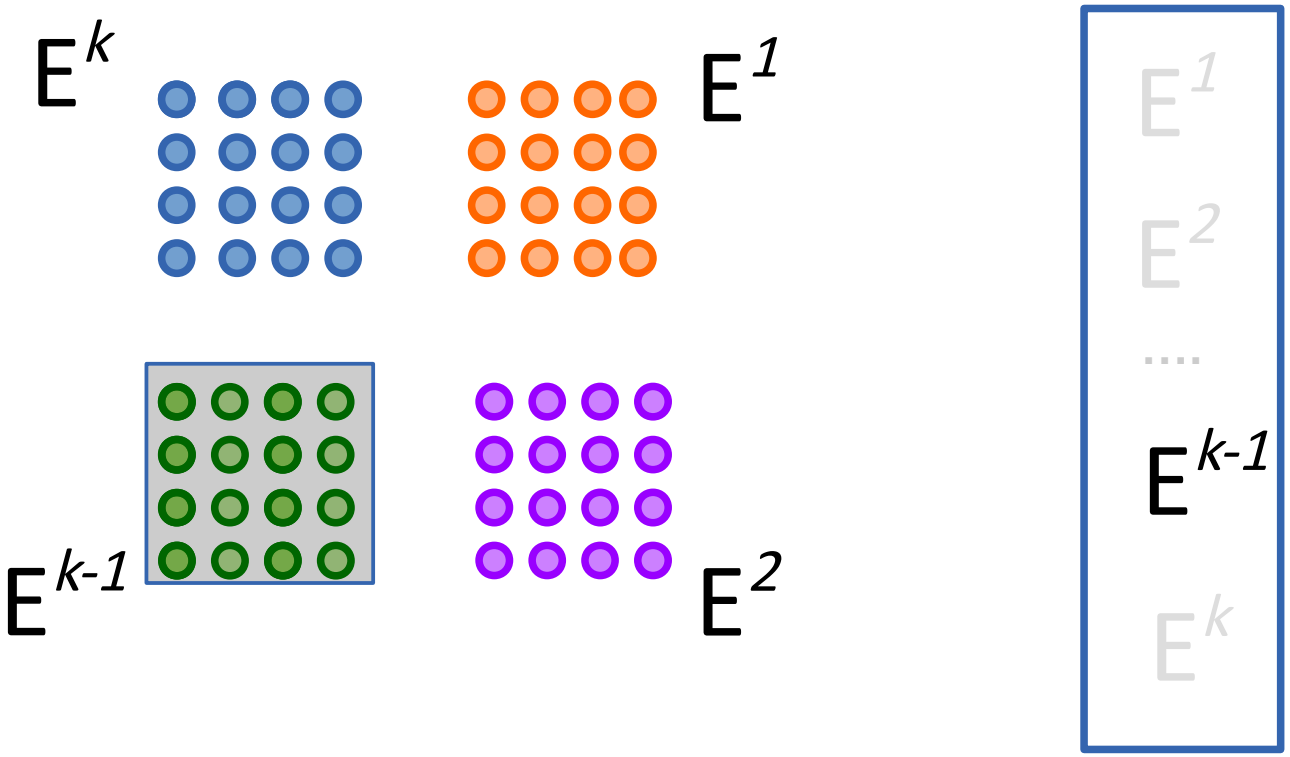
# Idea

- Choose the best policy based on current state



# Idea

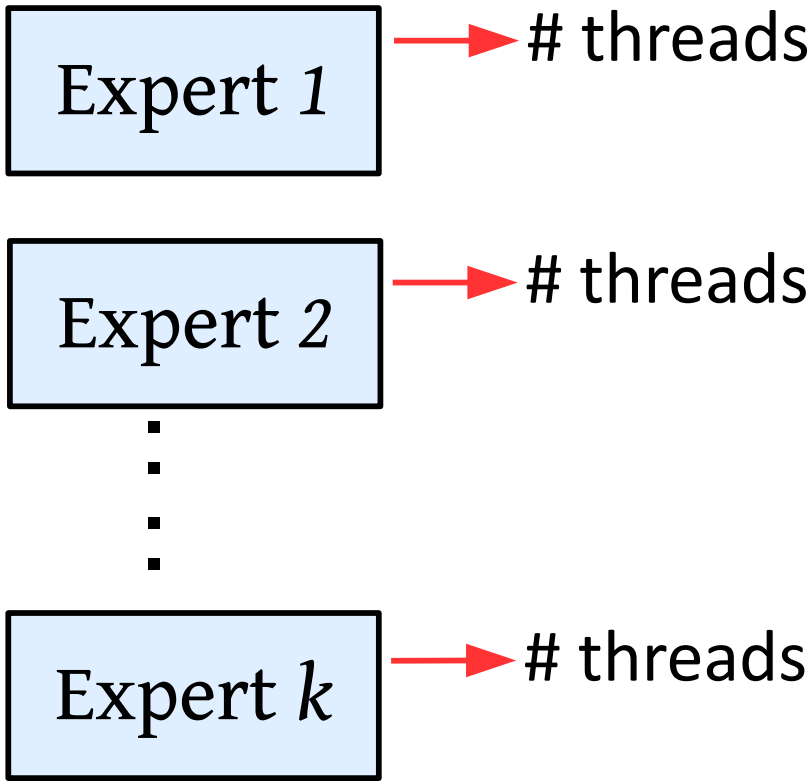
- Choose the best policy based on current state



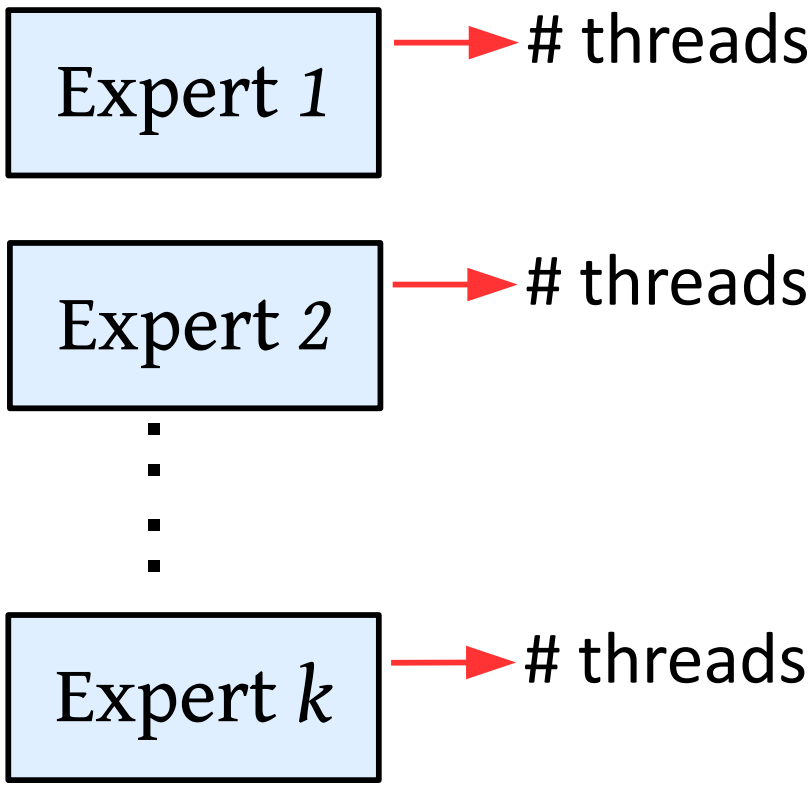
# Mixture of Experts based Mapping

- Ensemble of experts (mapping policies)
- Smart way to select the best expert at runtime
- Combine offline prior models with online learning

# Mixture of Experts based Mapping



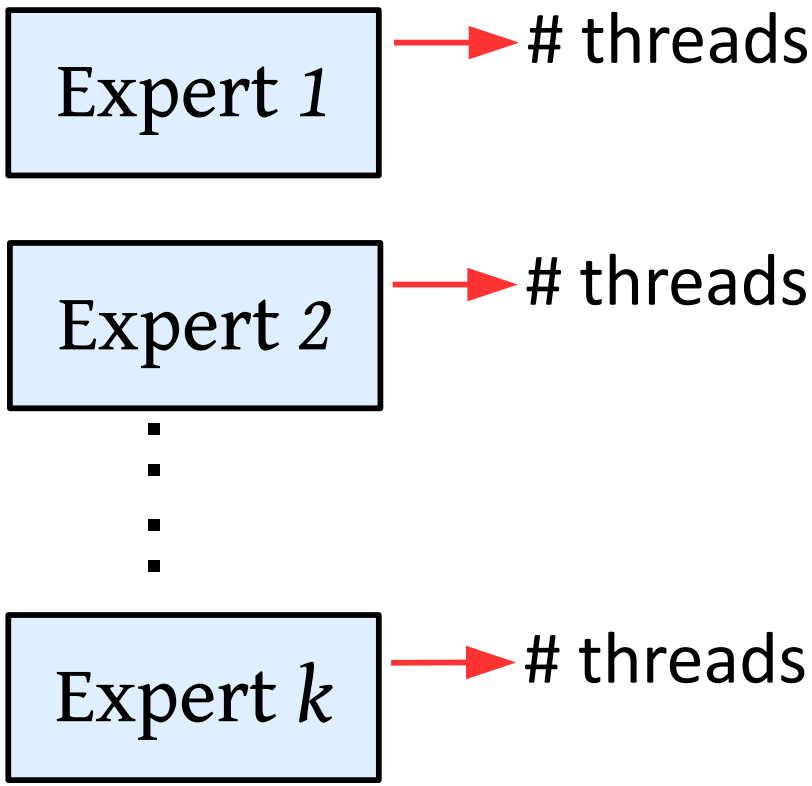
# Mixture of Experts based Mapping




How to select the best expert ?

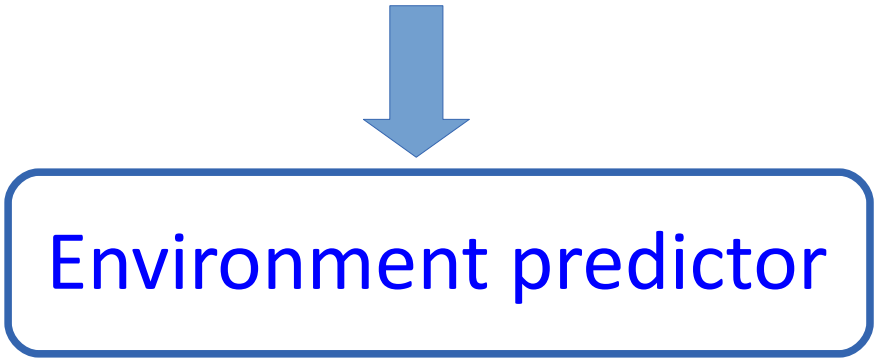
Expensive to evaluate with # threads of all experts

# Mixture of Experts based Mapping

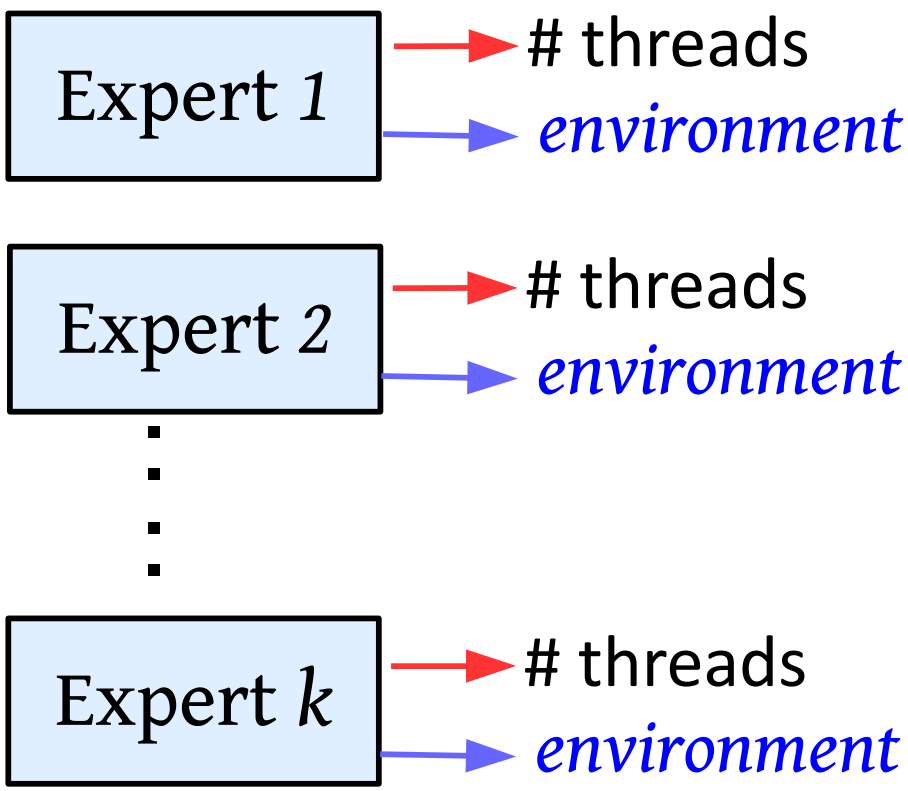



How to select the best expert ?

Expensive to evaluate with # threads of all experts

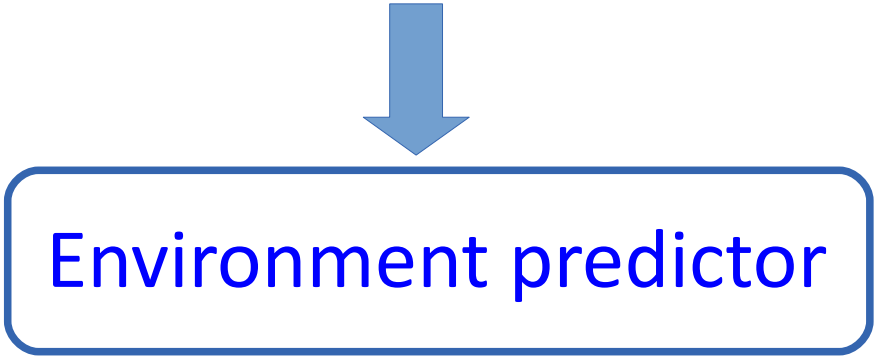


# Mixture of Experts based Mapping




How to select the best expert ?

Expensive to evaluate with # threads of all experts



# Predictive Modelling

Thread predictor

$$w(f_t) = \hat{n}_t$$

What is the best # threads

Environment predictor

$$m(f_t) = \hat{e_{t+1}}$$

What should the environment should look like

# Predictive Modelling

Thread predictor

$$w(f_t) = \hat{n}_t$$

What is the best # threads

Environment predictor

$$m(f_t) = \hat{e_{t+1}}$$

What should the environment should look like

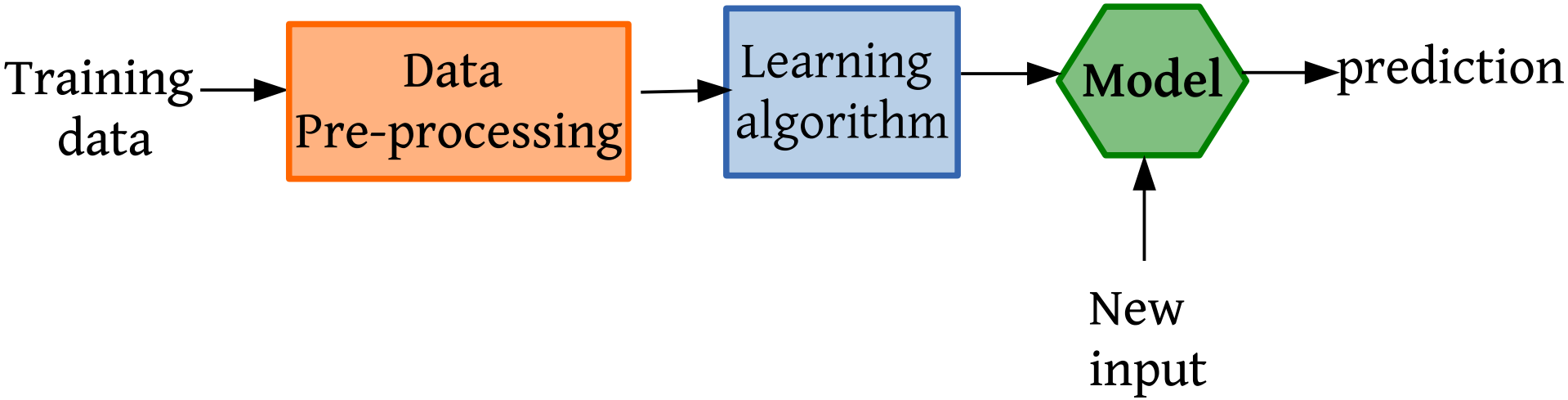
Input-feature-vector = < code, environment >

***f = (c,e)***

# Approach – Machine Learning

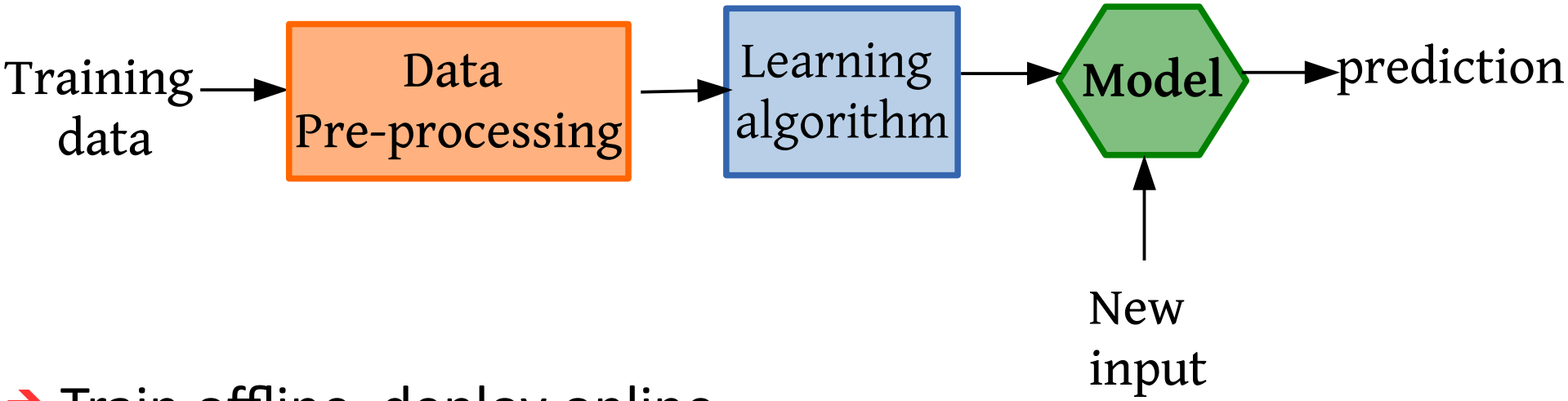
# Approach – Machine Learning

- Hand crafted solutions infeasible



# Approach – Machine Learning

- Hand crafted solutions infeasible




- Train offline, deploy online
- Supervised learning, Cross-validated
- Trained on NAS, evaluated on additional benchmarks

** Training overhead: one-off cost of 9216 experiments*

# Training phase

- Various configurations of program pairs and # threads

  9216 experiments ; 3 weeks for runs; 1.1 GB log

- Feature space dimensionality reduction : Information gain

  10 / 154 rich subset of features

- Linear Regression Models

# Features

| STATIC (code) | DYNAMIC (environment) |
|---|---|
| # instructions | # workload threads |
| # branches | # processors |
| # load/store | run queue size |
| | CPU load |
| | page free list rate |
| | cached memory |

# How to select the best expert

**Online Expert Selector**

Select expert *'k'*

$$k : min||\hat{e_k} - e_{actual}||$$

# How to select the best expert

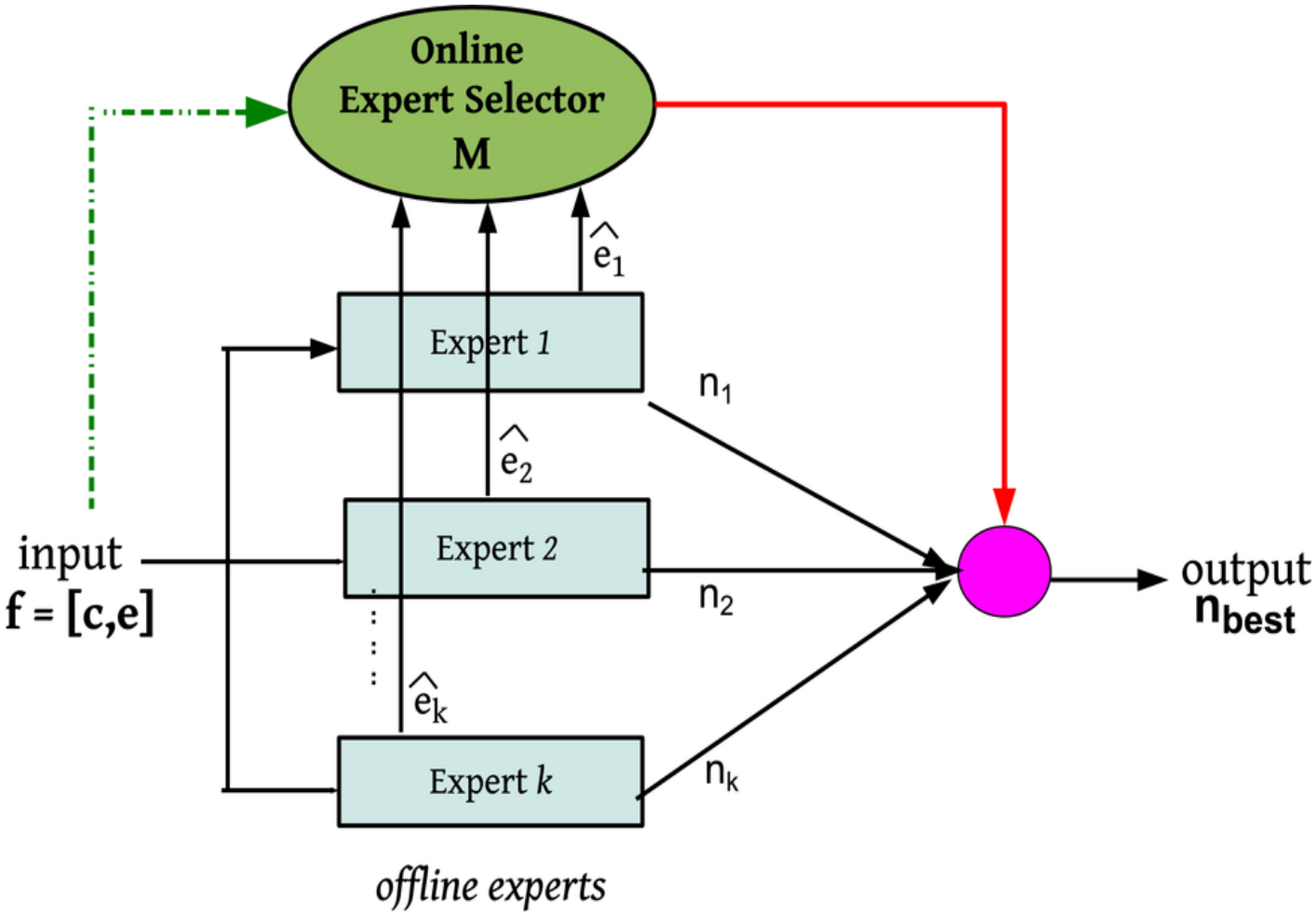
**Online Expert Selector**

Select expert *'k'*

$$k : min||\hat{e_k} - e_{actual}||$$

Use *'Environment predictor'* as a proxy to select the best mapping policy

# All put together...

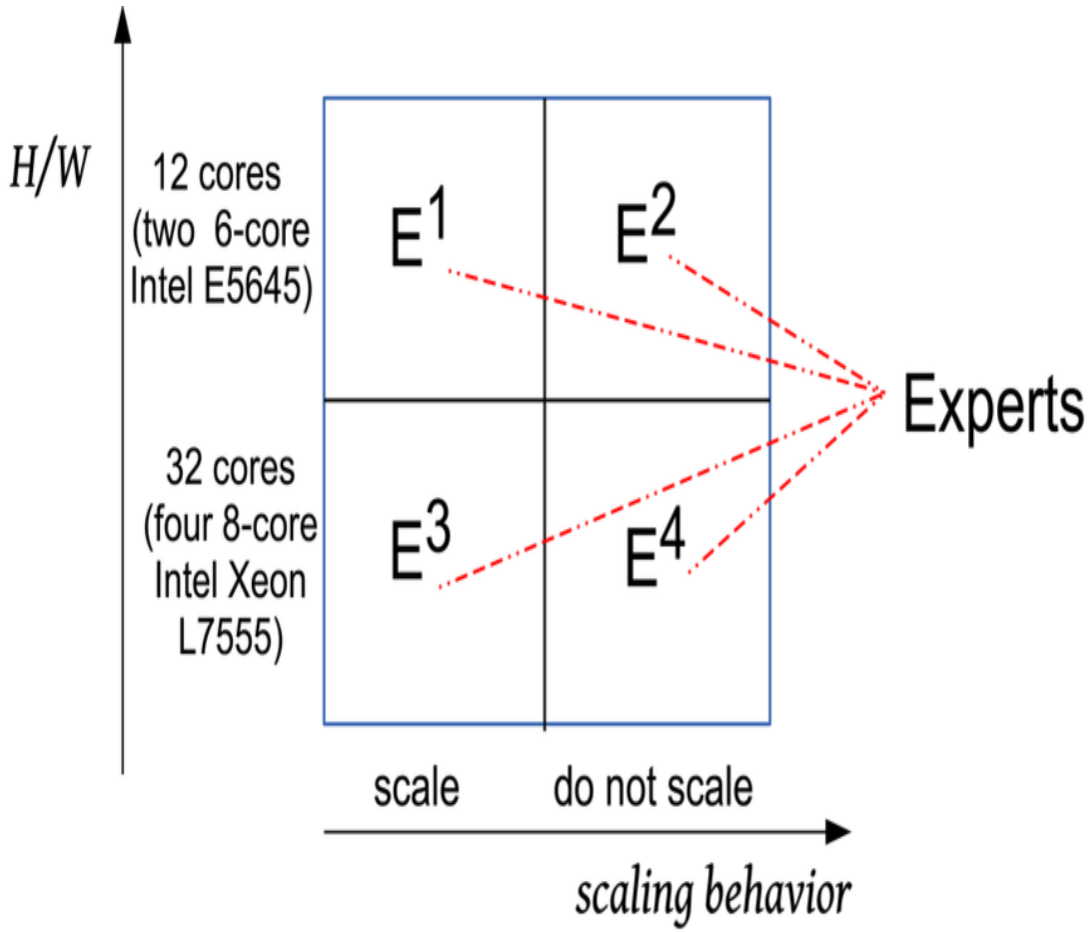


How many experts ?

# How many experts ?

open question

# Started with 4 experts



# Evaluation

*Platform* : 32-core Intel Xeon

*Benchmarks* : NAS, SpecOMP, Parsec (*OpenMP*)

*Comparison* : OpenMP default, Online, Offline, Analytic

*Workloads* : Small (*light*), large (*heavy*)

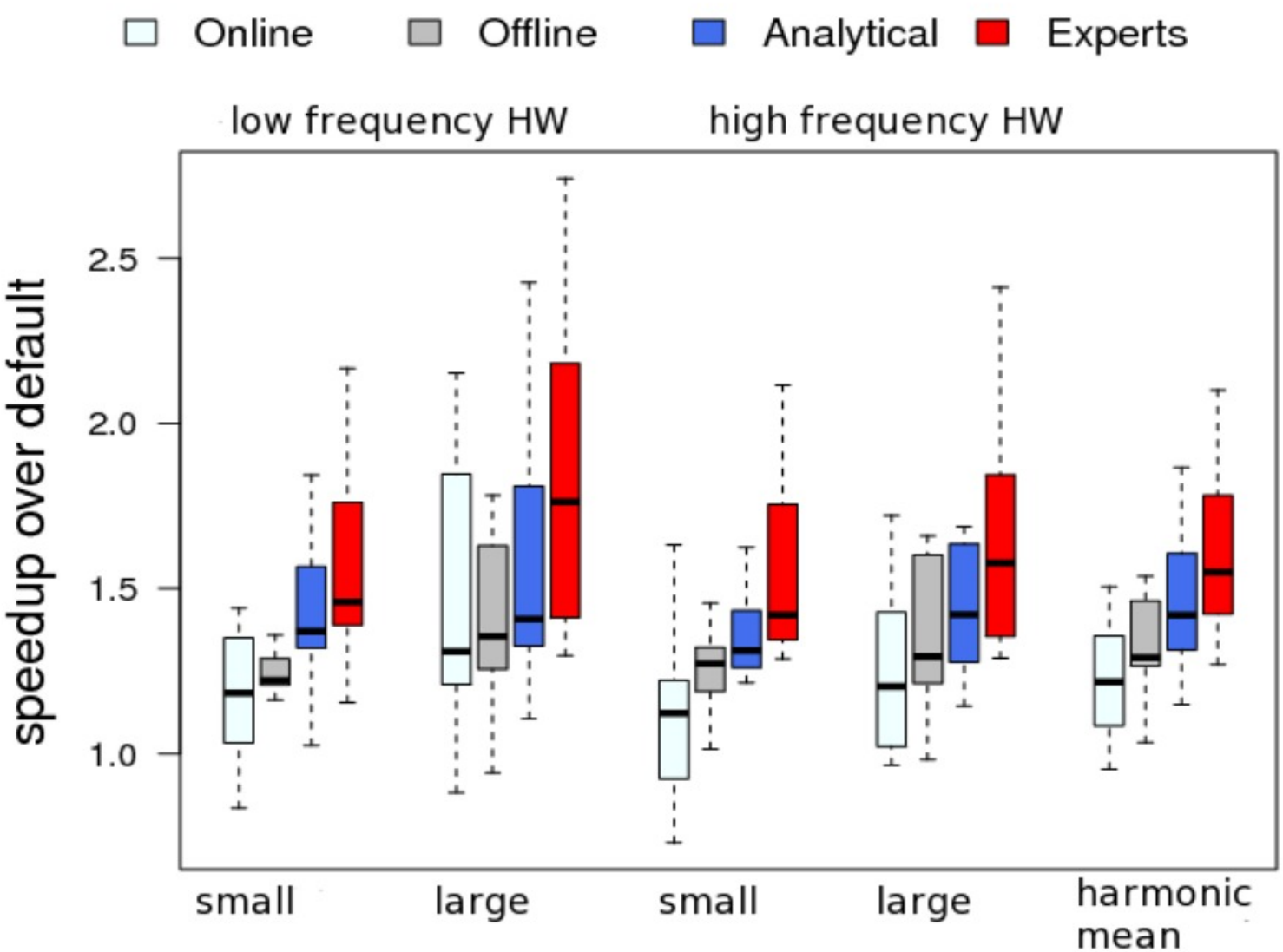
*Hardware* : Low, high frequent

**Online**: "Parcae: a system for flexible parallel execution", A. Raman, A. Zaks, J. W. Lee, and D. I. August. PLDI'12

**Offline**: "Smart, Adaptive Mapping of Parallelism in the Presence of External Workload", Murali Krishna Emani, Zheng Wang and Michael O'Boyle, CGO'13

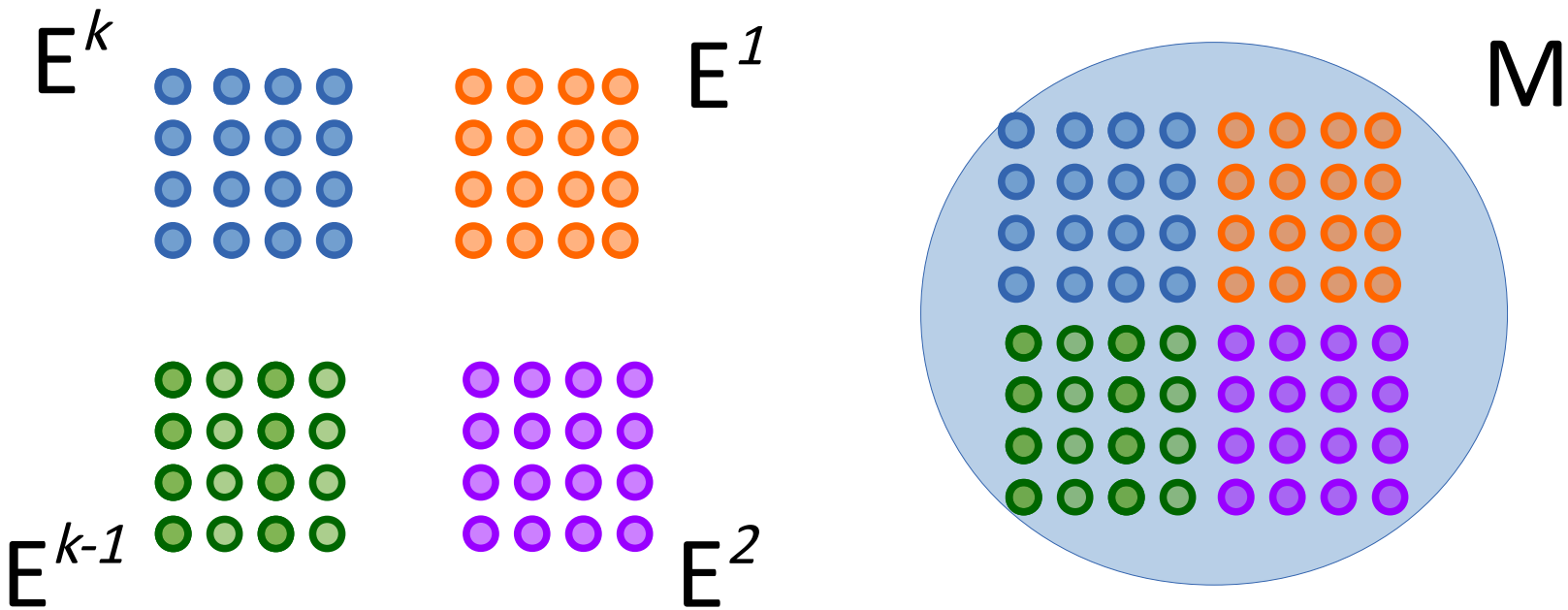
**Analytic**: "Adaptive, Efficient, Parallel Execution of Parallel Programs", S. Sridharan, G. Gupta, and G. S. Sohi. PLDI '14.

# Results

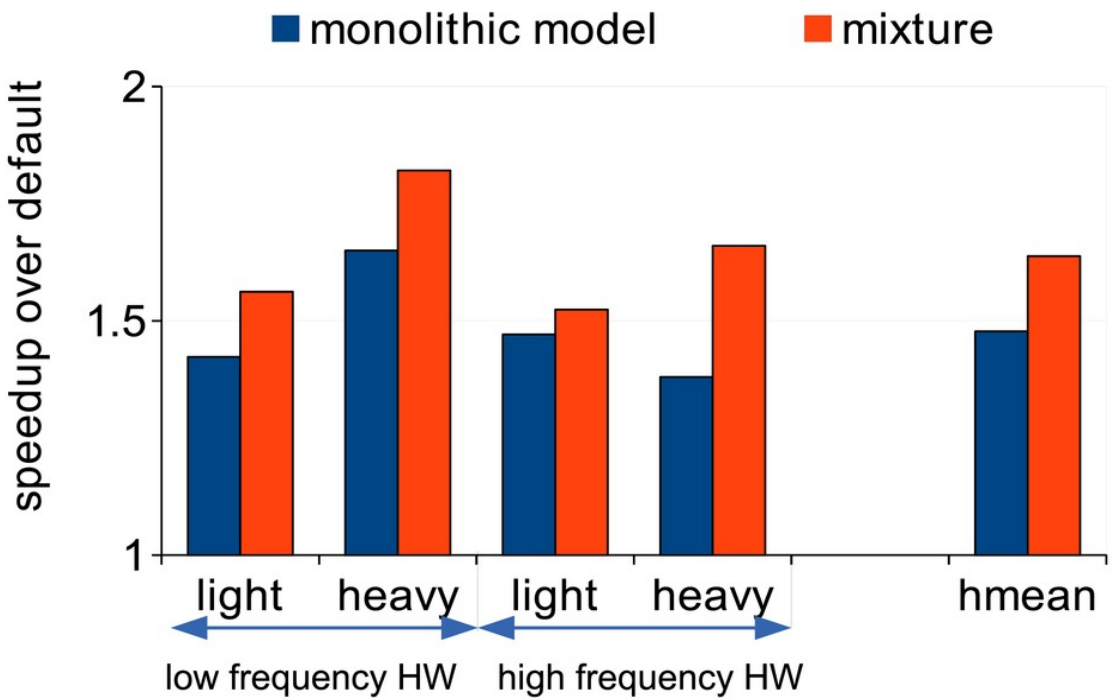



1.17x over analytic
1.26x over offline
1.38x over online

# Why multiple experts ? Why not a single model ?



# Why multiple experts ? Why not a single model ?

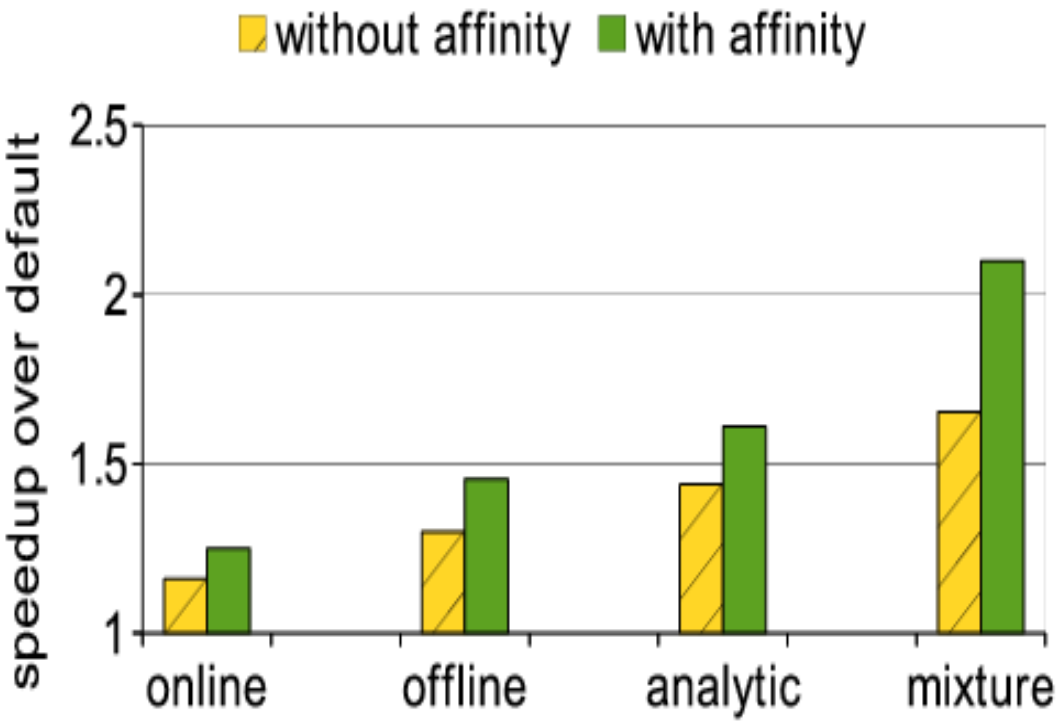



Multiple experts outperforms single model

Can this approach be used with other optimization techniques ?

Can this approach be used with other optimization techniques ?

Affinity-based scheduling



# To sum up...

Developed an approach for smart parallelism mapping

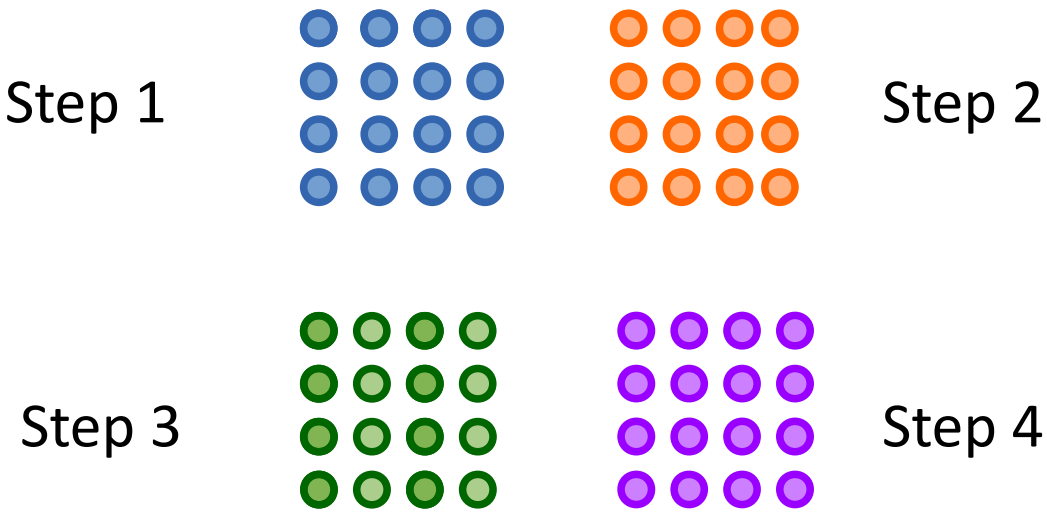
- Adaptive to dynamic environment
- Predictive modelling at its heart
- Environment predictor as a proxy to select the best mapping policy

# What next ?

- Integrating this concept in CnC
- Focus on **tuning** component
- Runtime and Application tuning
- Dynamic partitioning of resources to steps

# Idea

Instances of computations (steps)




- Varying resource requirements for steps
- Mapping depends on when data is ready

# Take away

- One-size-fits-*none*
- A bag of multiple policies is more practical than one
- Machine learning can be of help !!

# Thank you

Murali Emani
University of Edinburgh
m.k.emani@sms.ed.ac.uk

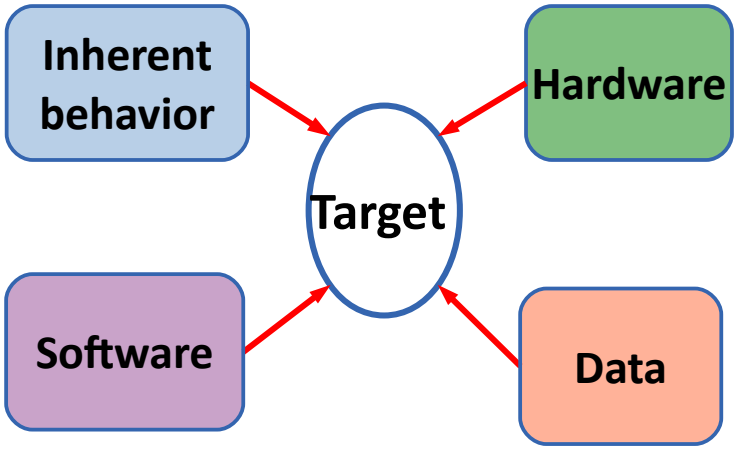
# Backup

# Adaptive Parallelism Mapping

➔ Program performance is sensitive to the environment

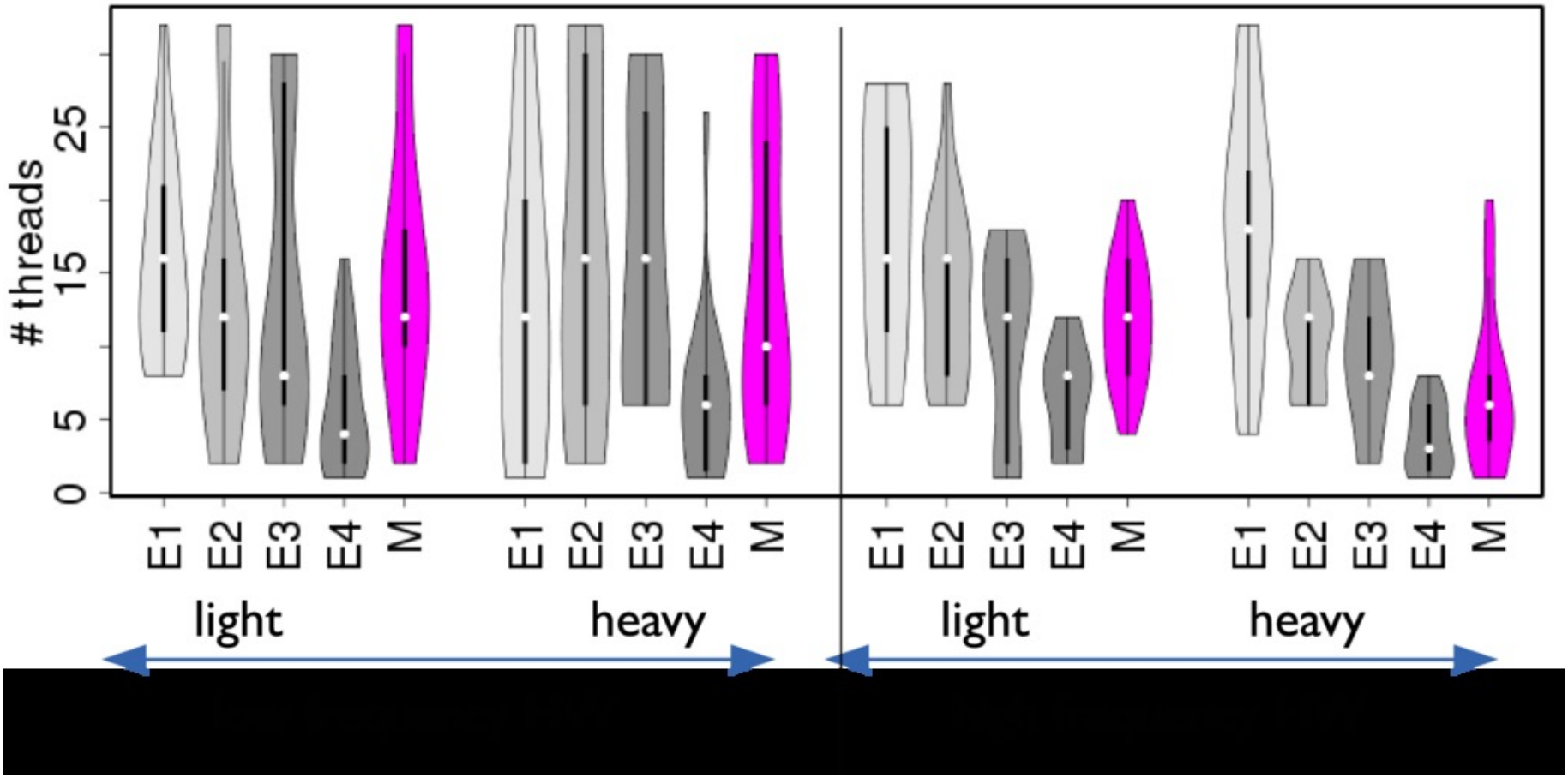
- Various characteristics
- Compute/memory/disk bound

- Large number of components
- Increased chances of failure

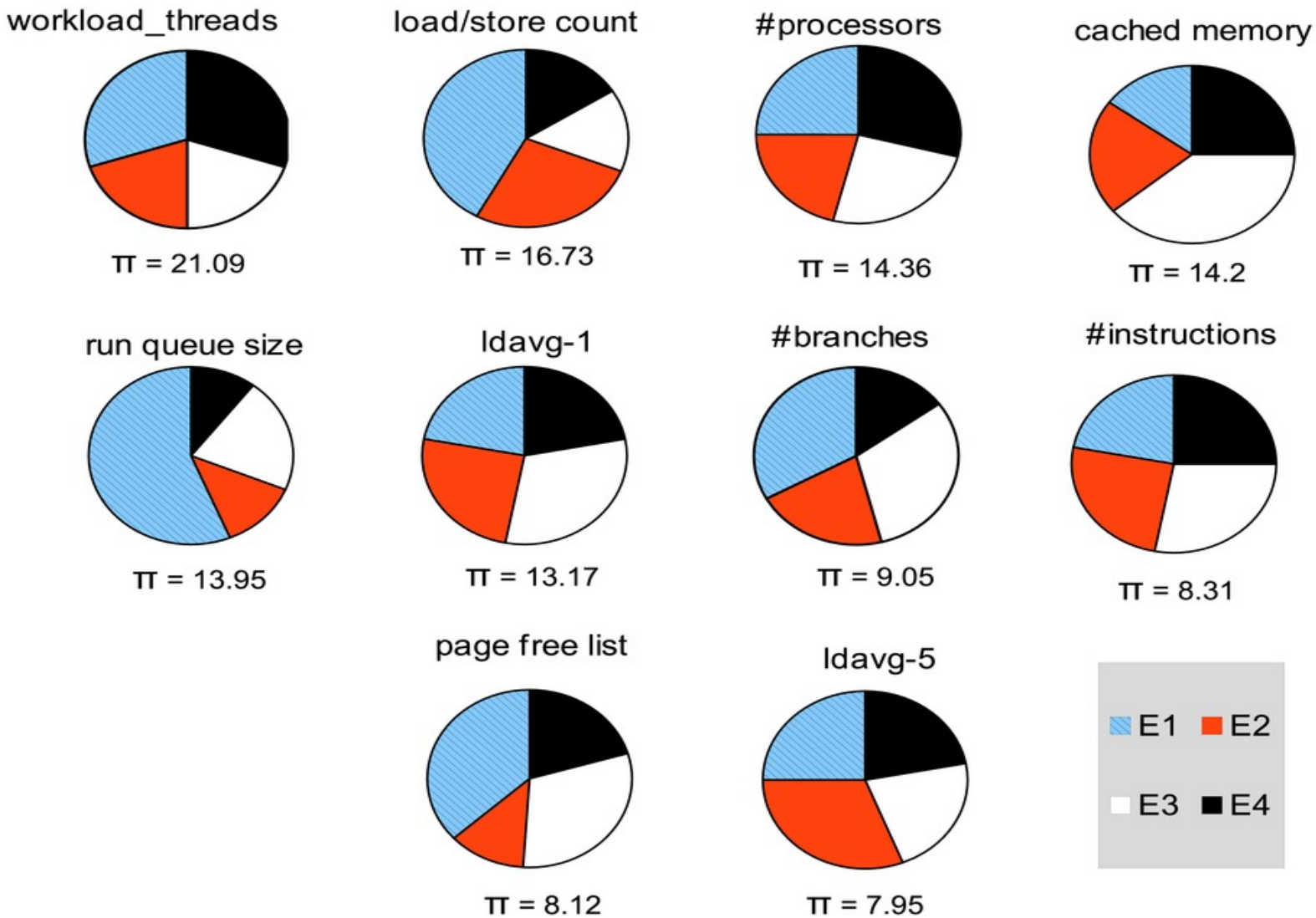


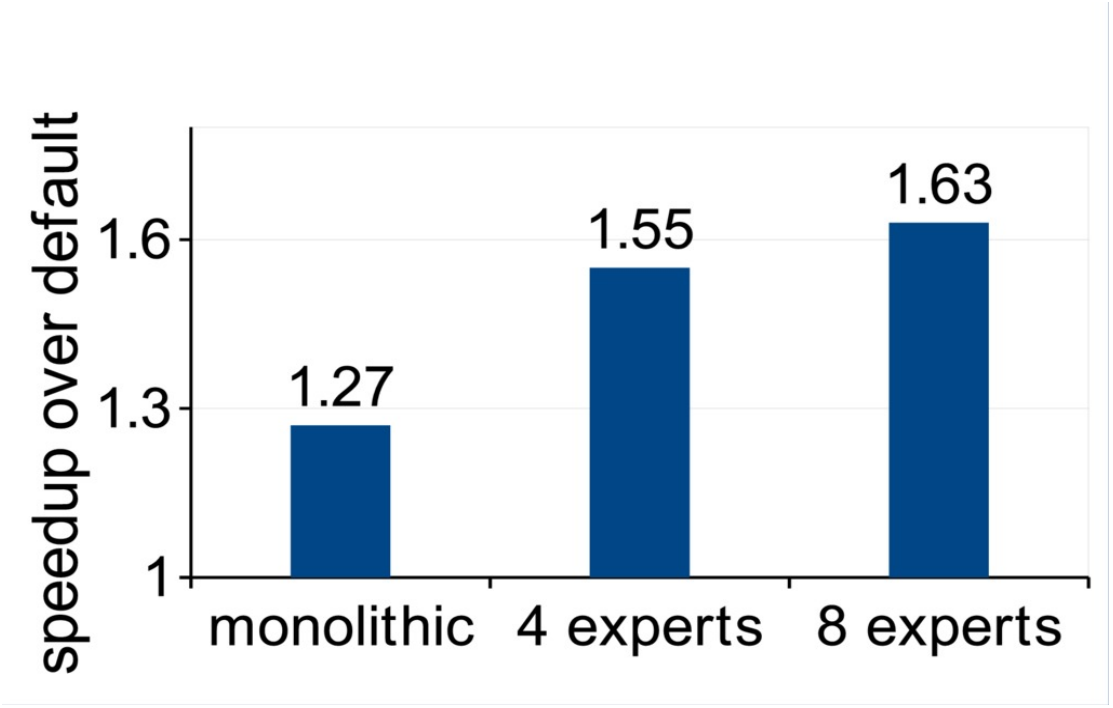

- Recurring upgrades
- Versions compatibility

- Varying amount of I/O
- Scalability issues


# threads
0
5
15
25
E1
E2
E3
E4
M
E1
E2
E3
E4
M
E1
E2
E3
E4
M
E1
E2
E3
E4
M
light
heavy
light
heavy

All experts use the same features, they vary in importance across each expert.




speedup over default
1.6
1.3
1
1.27
1.55
1.63
monolithic
4 experts
8 experts

# Evaluation

*Platform* : 32-core Intel Xeon 4 one-socket nodes, 8 cores/socket, 3.7.10 kernel

*Compiler* : gcc 4.6 "-O3 -fopenmp"

*Benchmarks* : NAS, SpecOMP, Parsec (*OpenMP*)

*Comparison* : OpenMP default, Online, Offline, Analytic

*Workloads* : Small (*light*), large (*heavy*)

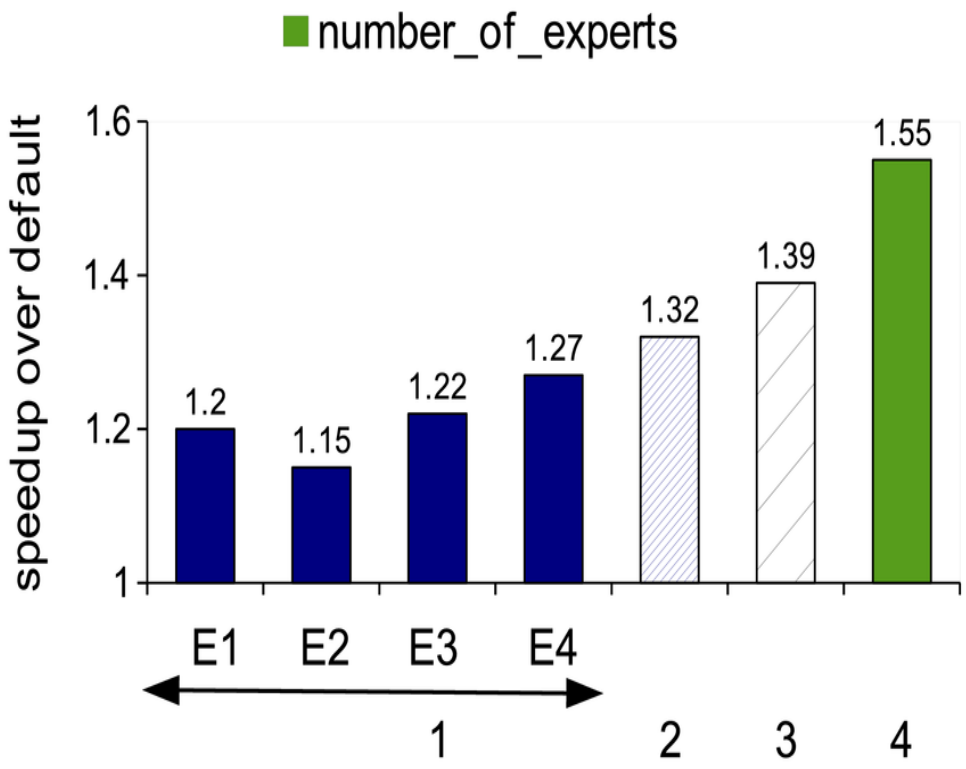
*Hardware* : Low, high frequent

**Online**: "Parcae: a system for flexible parallel execution", A. Raman, A. Zaks, J. W. Lee, and D. I. August. PLDI'12

**Offline**: "Smart, Adaptive Mapping of Parallelism in the Presence of External Workload", Murali Krishna Emani, Zheng Wang and Michael O'Boyle, CGO'13

**Analytic**: "Adaptive, Efficient, Parallel Execution of Parallel Programs", S. Sridharan, G. Gupta, and G. S. Sohi. PLDI '14.

## What is the effect of increasing # experts ?




Graceful addition of experts

What about # experts > 4 ? Needs more analysis