Chair of Communication Networks
Department of Electrical and Computer Engineering
Technical University of Munich

Adversarial Network Benchmarking

Andreas Blenk*

Joint work with:

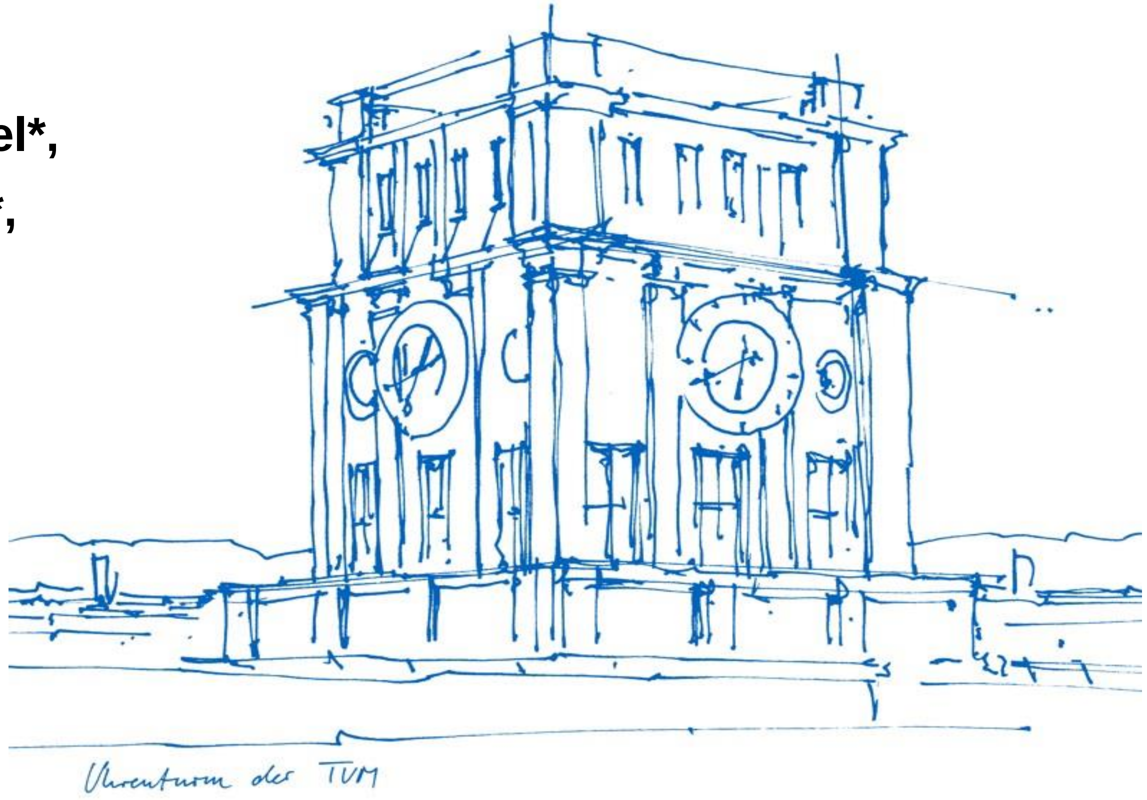
Johannes Zerwas*, Patrick Kalmbach*, Laurenz Henkel*, Sebastian Lettner, Gábor Rétvári^, Wolfgang Kellerer*, Stefan Schmid°

**Technical University of Munich, Germany*

^Budapest University of Technology and Economics, Hungary

°Faculty of Computer Science, University of Vienna, Austria

Communication Technologies Group,
Faculty of Computer Science, University of Vienna



Today’s Approach of Operating Networks?

Today's Approach of Operating Networks?

Monitors

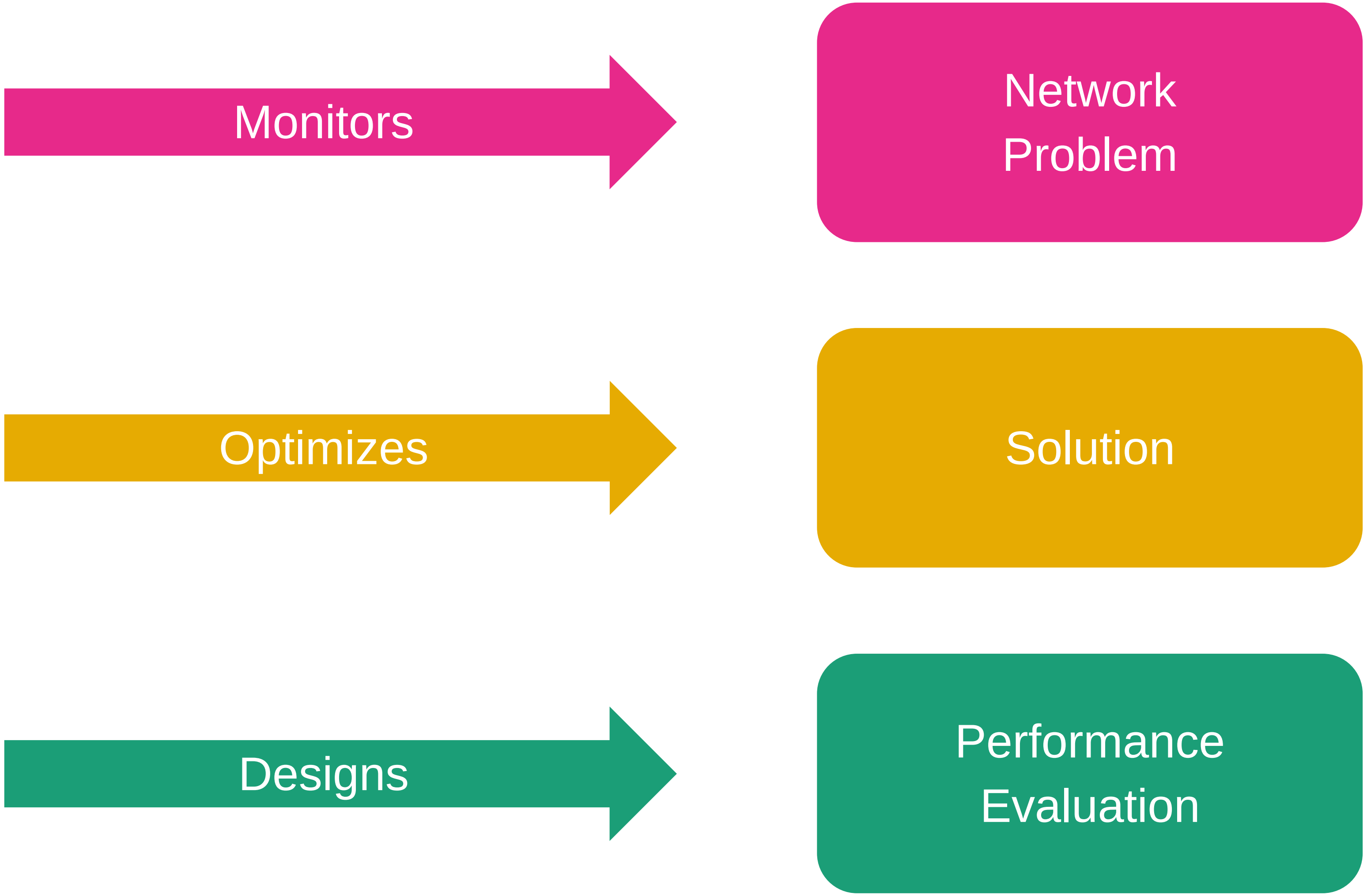
Network
Problem

Today’s Approach of Operating Networks?

Monitors → Network Problem

Optimizes → Solution

Today's Approach of Operating Networks?



Today's Approach of Operating Networks?

Monitors → Network Problem

Optimizes → Solution

Designs → Performance Evaluation

With more complex networks need for automation!

What Self-Driving Networks Should Do

What Self-Driving Networks Should Do

Source: https://www.pinterest.at/pin/318137161149129652/

What Self-Driving Networks Should Do

Source: https://www.pinterest.at/pin/318137161149129652/

Self-Monitoring

Network Problem

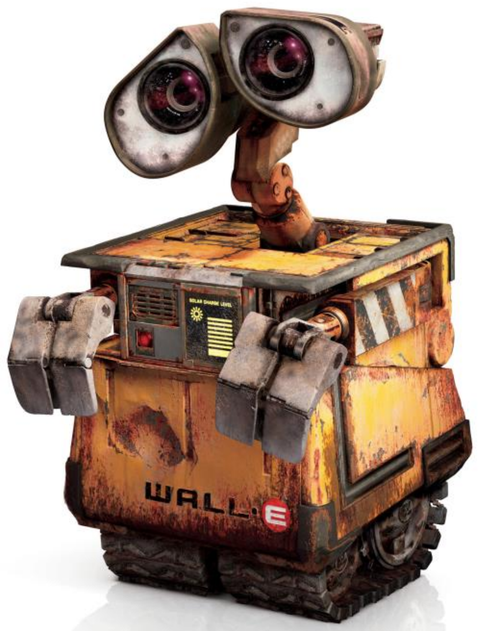
What Self-Driving Networks Should Do

Source: https://www.pinterest.at/pin/318137161149129652/

Self-Monitoring → Network Problem

Self-Optimizing → Solution

What Self-Driving Networks Should Do



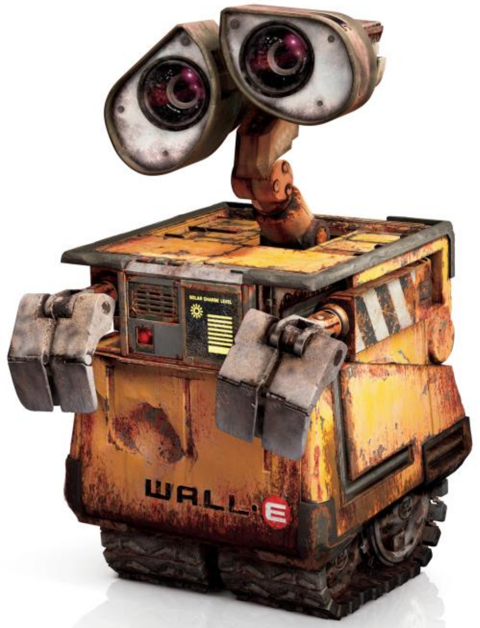
Source: https://www.pinterest.at/pin/318137161149129652/

Self-Monitoring → Network Problem

Self-Optimizing → Solution

Self-Benchmarking → Performance Evaluation

What Self-Driving Networks Should Do



Source: https://www.pinterest.at/pin/318137161149129652/

Self-Monitoring → Network Problem

Self-Optimizing → Solution

Self-Benchmarking → Performance Evaluation

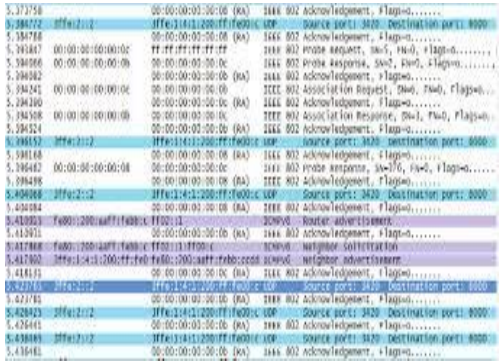
NetBOA (NetAI’19)

Benchmarking Network Algorithms, Architectures etc…

The Traditional Way …

Benchmarking Network Algorithms, Architectures etc…

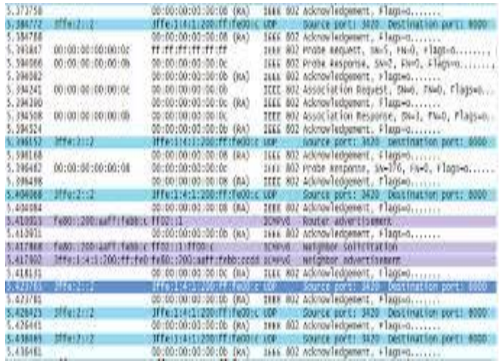
The Traditional Way …



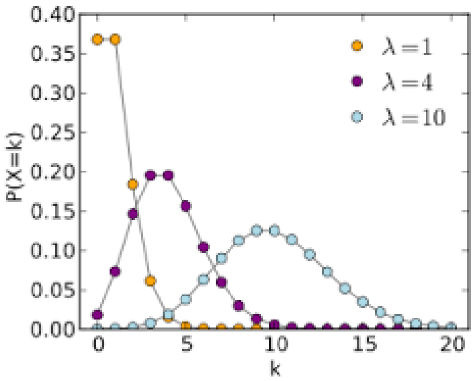
Traces

Benchmarking Network Algorithms, Architectures etc…

The Traditional Way …



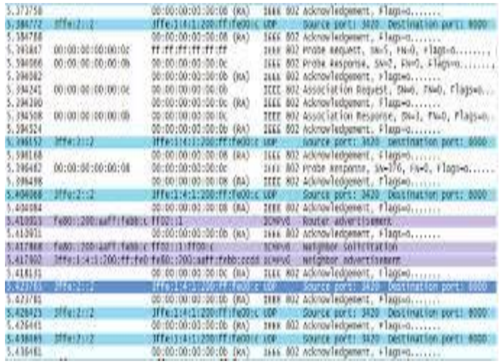
Traces



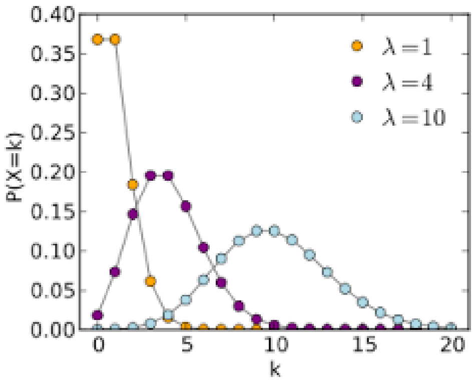
Models

Benchmarking Network Algorithms, Architectures etc…

The Traditional Way …



Traces



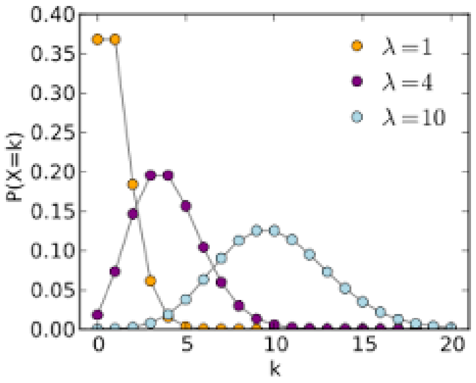
Models

Human‘s Best Guesses

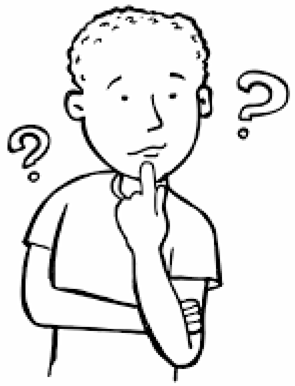
Benchmarking Network Algorithms, Architectures etc…

The Traditional Way …

Traces



Models

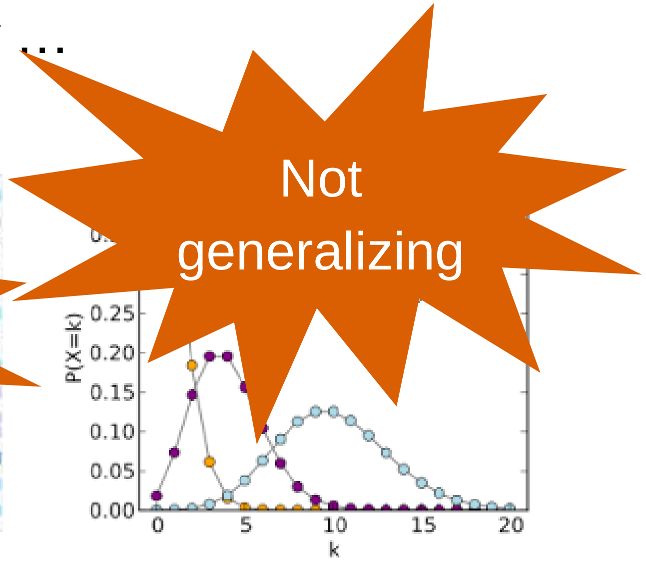


Human‘s Best Guesses

Benchmarking Network Algorithms, Architectures etc…

The Traditional Way …

Traces



Models

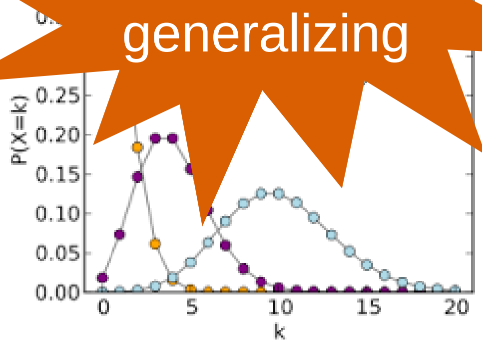
Human‘s Best Guesses

Benchmarking Network Algorithms, Architectures etc…

The Traditional Way …

Not always available

Traces



Not generalizing

Models

Hmm… Biased?

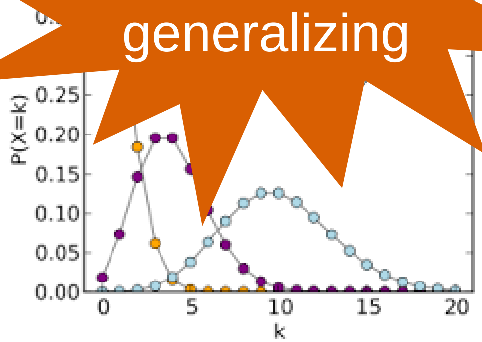
Human‘s Best Guesses

Benchmarking Network Algorithms, Architectures etc…

The Traditional Way …

Not always available

Traces



Not generalizing

Models

Hmm… Biased?

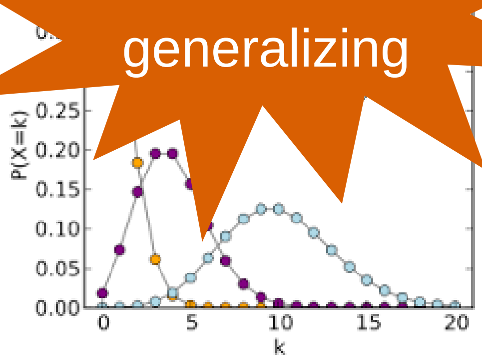
Human‘s Best Guesses

Data-Driven

Benchmarking Network Algorithms, Architectures etc…

The Traditional Way …

Traces



Models

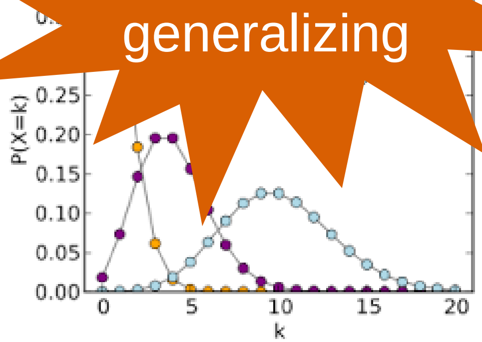
Human‘s Best Guesses

Data-Driven

Benchmarking Network Algorithms, Architectures etc…

The Traditional Way …

Traces



Models

Human's Best Guesses

Data-Driven

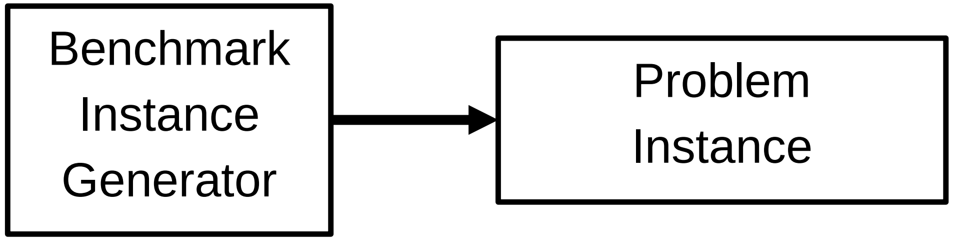
This Talk: Use Machine Learning to Benchmark Networks

Towards Automated Network Optimization and Design

Benchmark
Instance
Generator

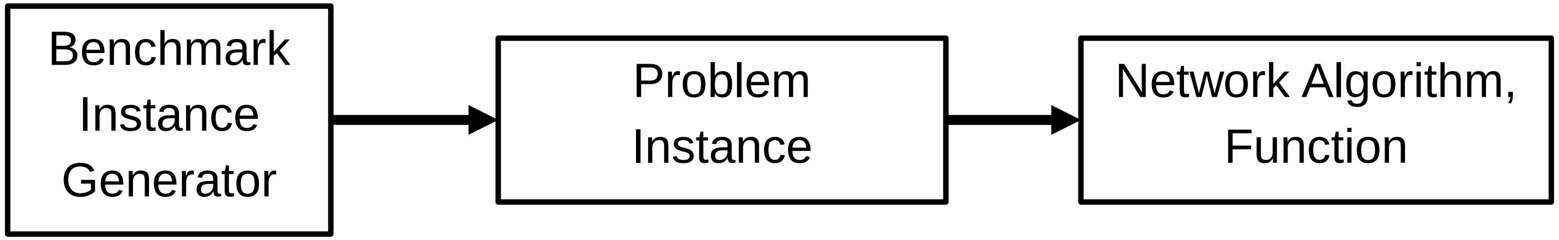
The Traditional Way!

Towards Automated Network Optimization and Design



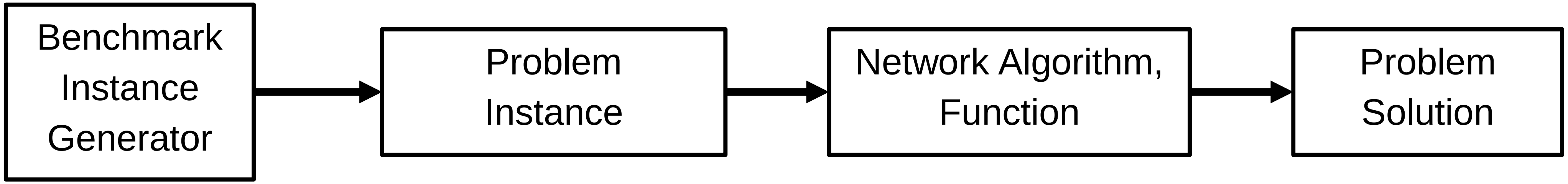
The Traditional Way!

Towards Automated Network Optimization and Design



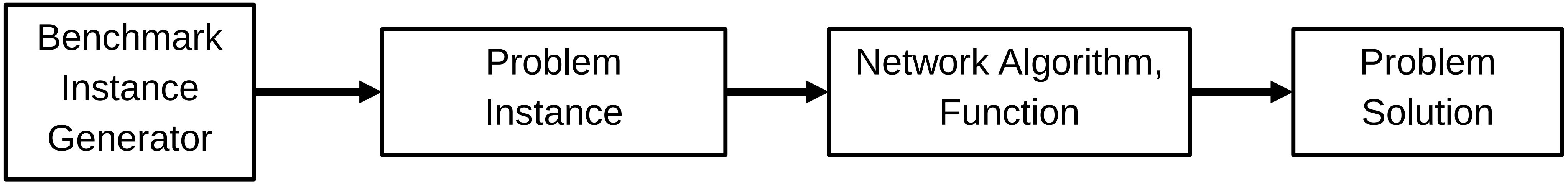
The Traditional Way!

Towards Automated Network Optimization and Design



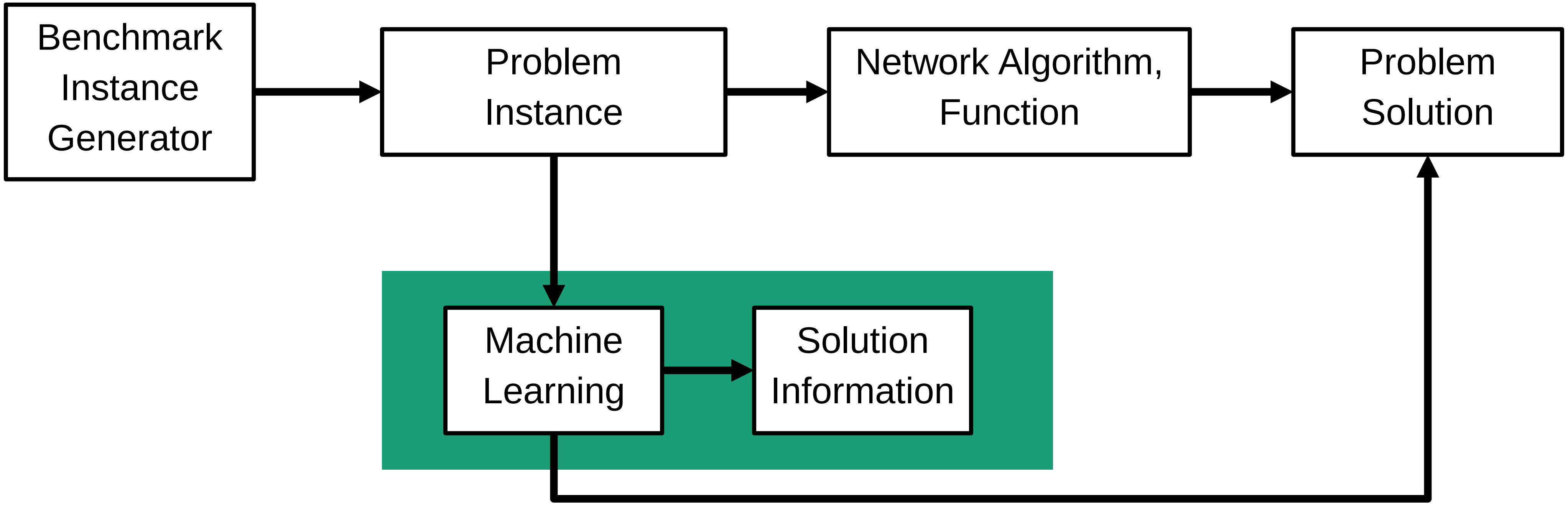
The Traditional Way!

Towards Automated Network Optimization and Design



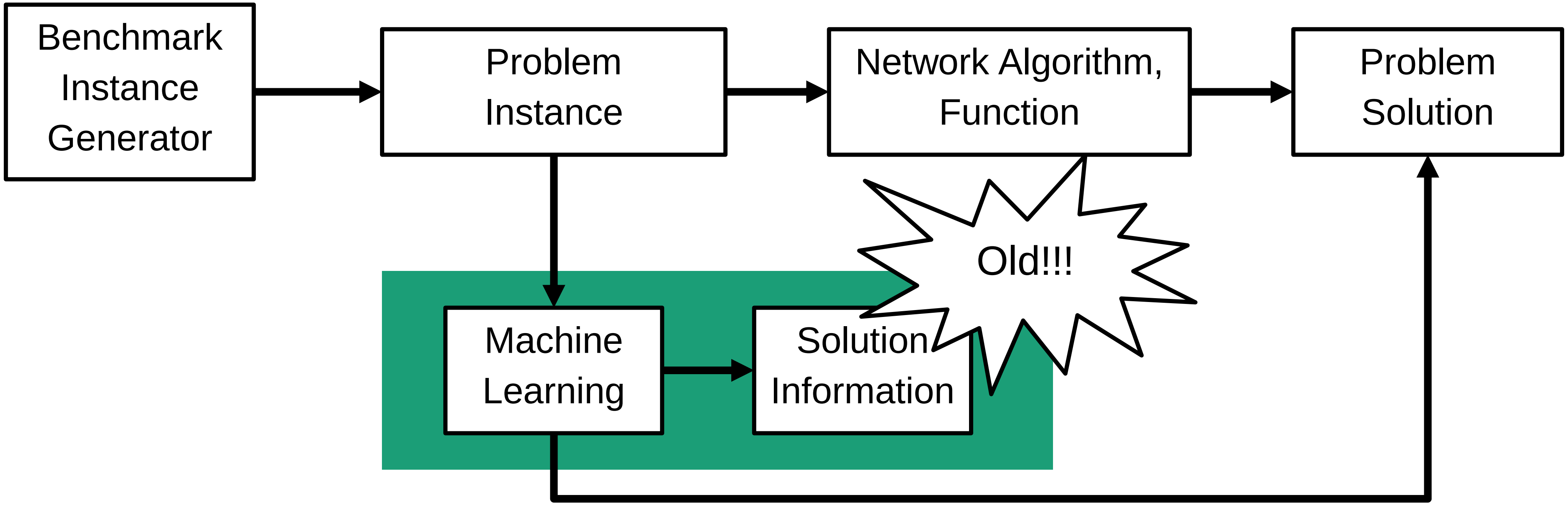
Our ML/AI Way!

Towards Automated Network Optimization and Design



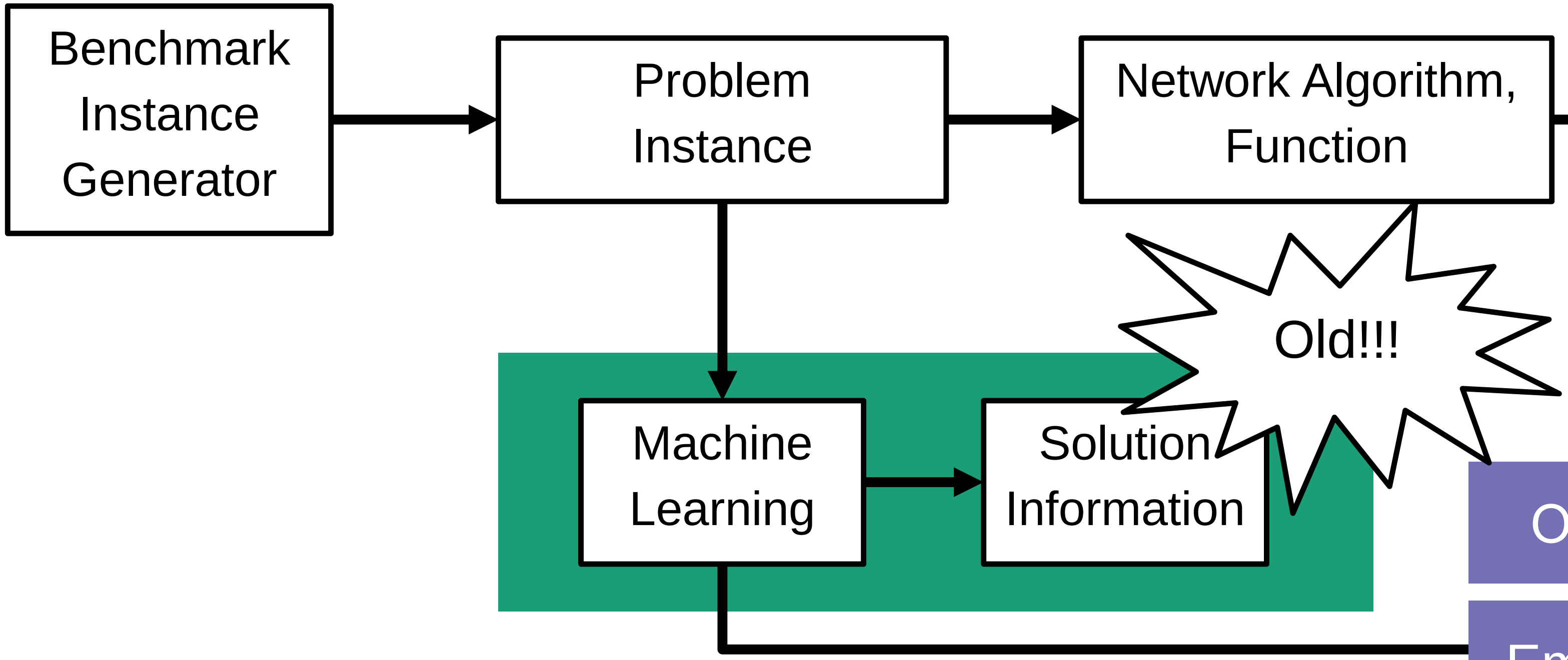
Our ML/AI Way!

Towards Automated Network Optimization and Design



Our ML/AI Way!

Towards Automated Network Optimization and Design



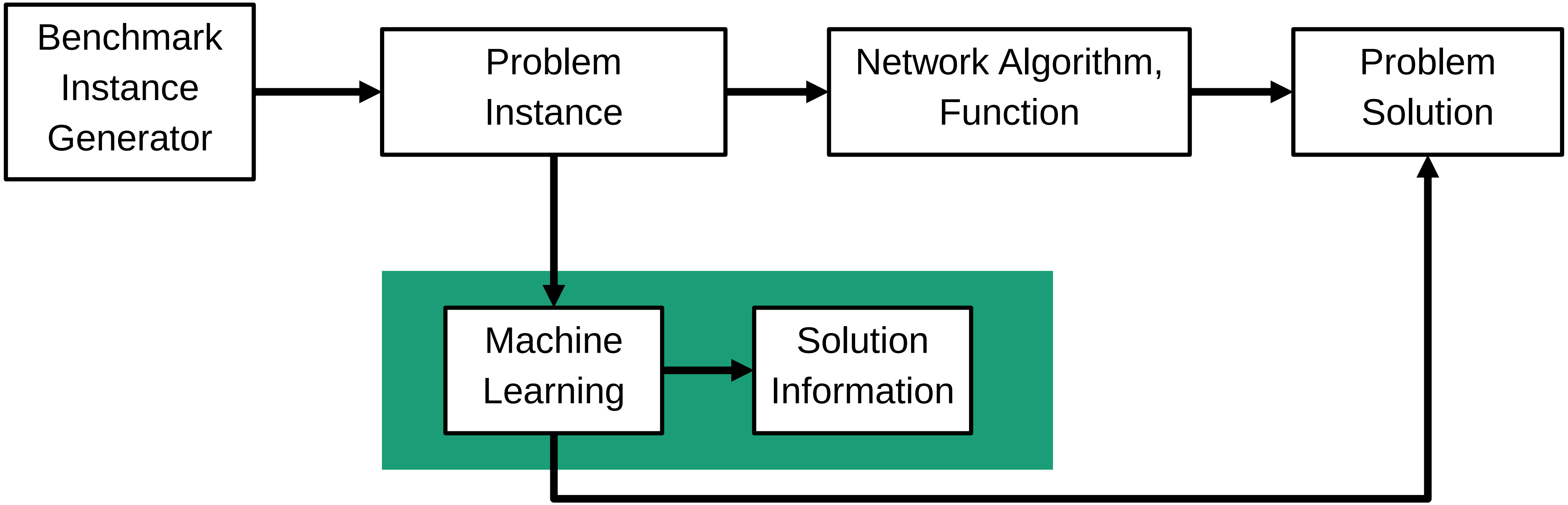
O’zapf t is [BIG DAMA’17]

Empowerement [SelfDN’18]

ISMAEL [TNSM’19]

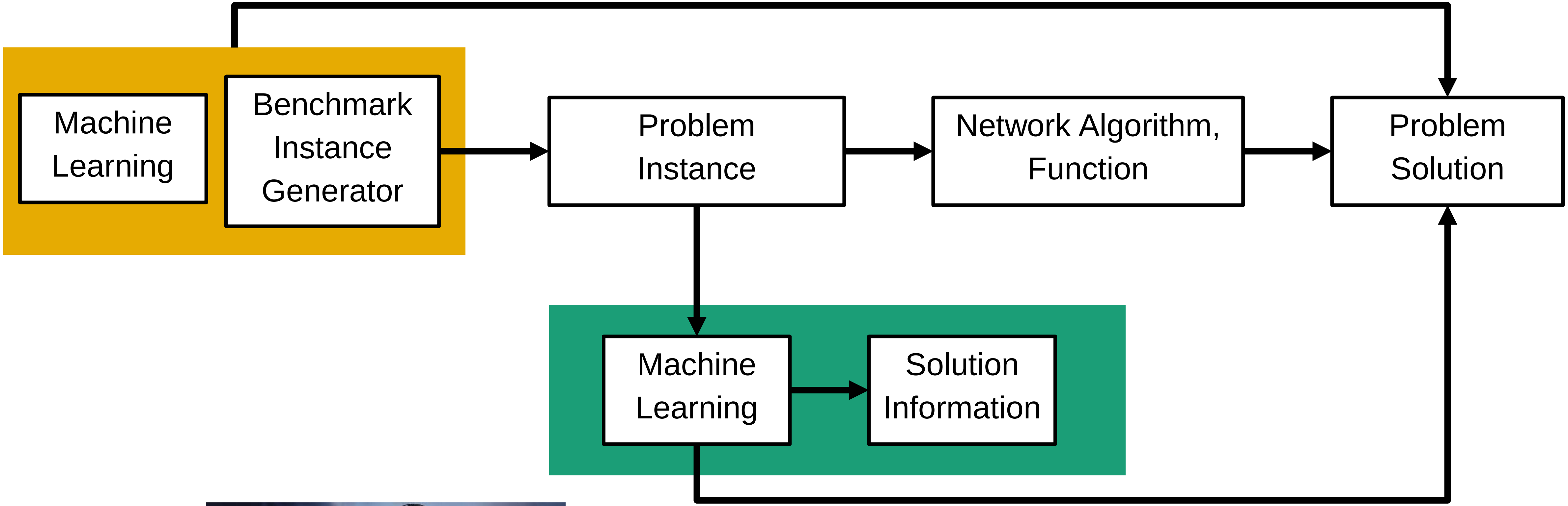
Our ML/AI Way!

Towards Automated Network Optimization and Design



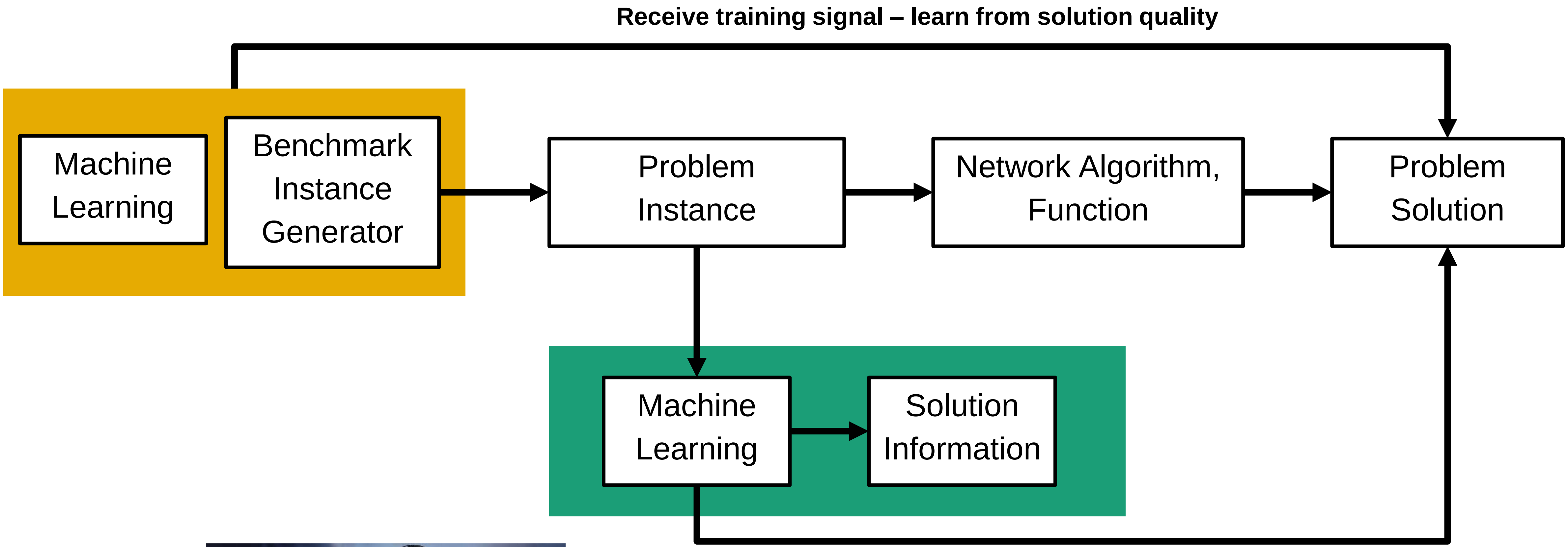
Our ML/AI Way!

Towards Automated Network Optimization and Design



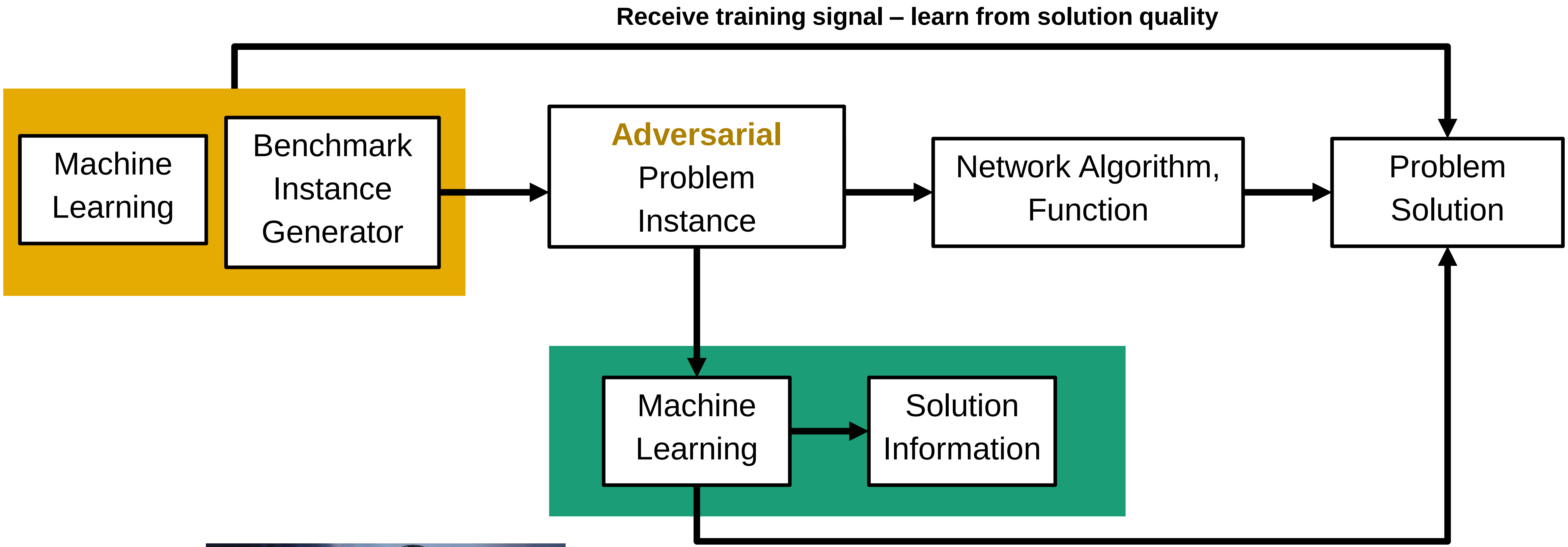
Our ML/AI Way!

Towards Automated Network Optimization and Design



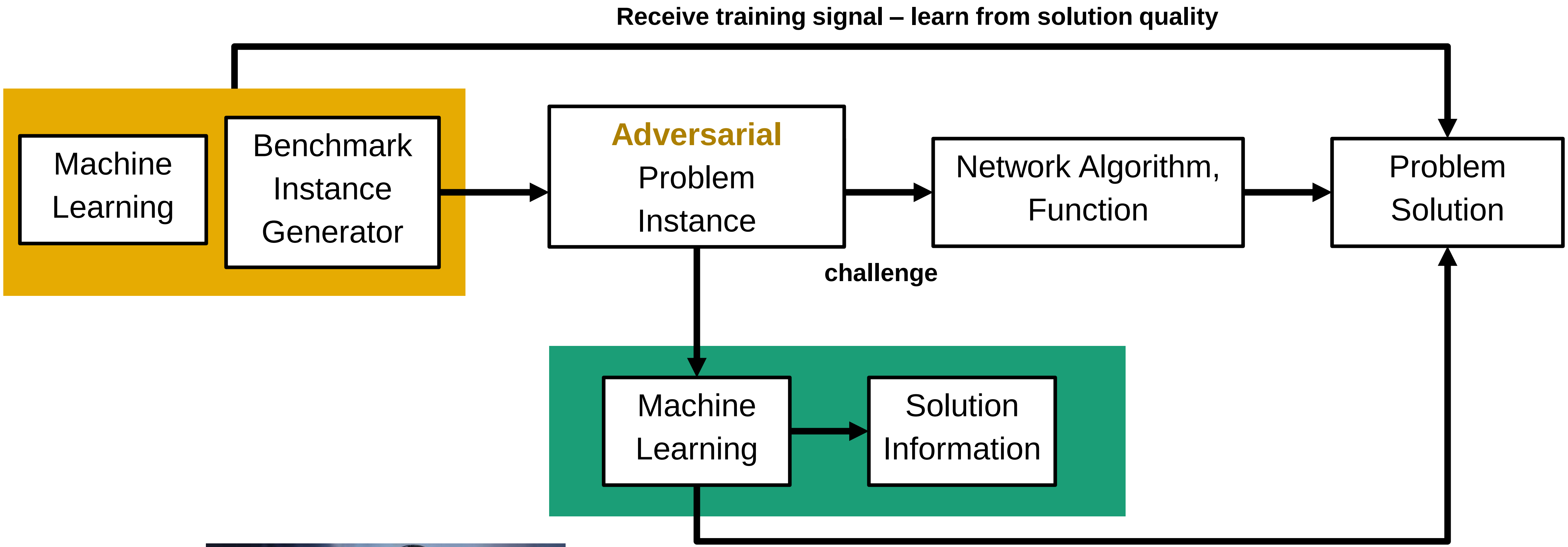
Our ML/AI Way!

Towards Automated Network Optimization and Design



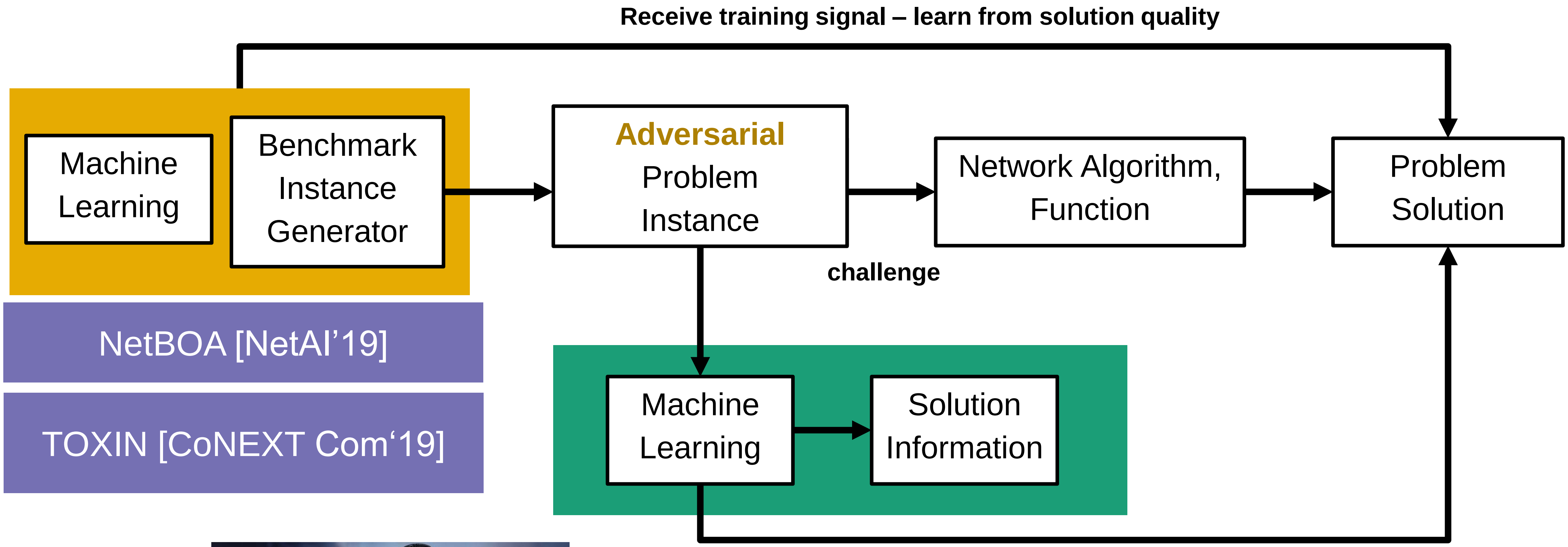
Our ML/AI Way!

Towards Automated Network Optimization and Design



Our ML/AI Way!

Towards Automated Network Optimization and Design



NetBOA [NetAI'19]

TOXIN [CoNEXT Com'19]

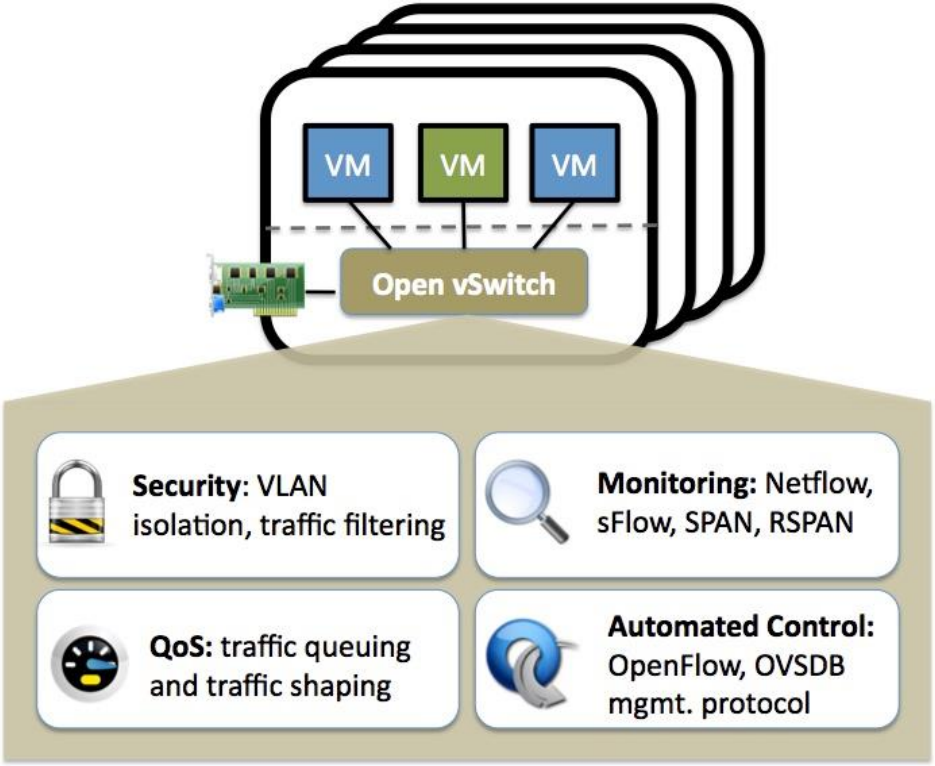
Our ML/AI Way!

ML/AI vs ML/AI

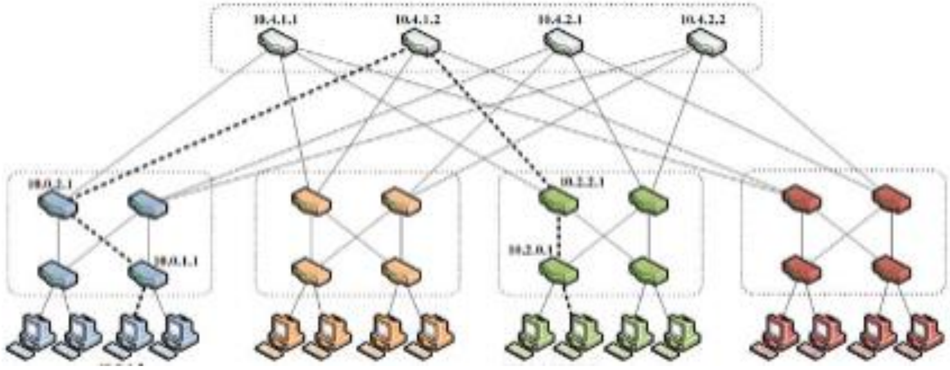
Adversarial Network Algorithm Benchmarking: Use Cases

Use Cases of this Talk

(1) Benchmarking Open vSwitch



(2) Benchmarking Data Center Traffic Scheduling Algorithms



(1) Benchmarking Open vSwitch

What Could be Seen as Related

- Algorithmic complexity attacks (software domain):
  - SlowFuzz
  - PerfFuzz

What Could be Seen as Related

- Algorithmic complexity attacks (software domain):
  - SlowFuzz
  - PerfFuzz
- *Automated Synthesis of Adversarial Workloads for Network Functions*, ACM Sigcomm 2018

What Could be Seen as Related

- Algorithmic complexity attacks (software domain):
  - SlowFuzz
  - PerfFuzz
- *Automated Synthesis of Adversarial Workloads for Network Functions*, ACM Sigcomm 2018
- **Policy Injection: A Cloud Dataplane DoS Attack**, ACM Sigcomm DEMO 2018

What Could be Seen as Related

- Algorithmic complexity attacks (software domain):
  - SlowFuzz
  - PerfFuzz
- *Automated Synthesis of Adversarial Workloads for Network Functions*, ACM Sigcomm 2018
- **Policy Injection: A Cloud Dataplane DoS Attack**, ACM Sigcomm DEMO 2018

Why Important?

What Could be Seen as Related

- Algorithmic complexity attacks (software domain):
  - SlowFuzz
  - PerfFuzz
- *Automated Synthesis of Adversarial Workloads for Network Functions*, ACM Sigcomm 2018
- **Policy Injection: A Cloud Dataplane DoS Attack**, ACM Sigcomm DEMO 2018

Why Important?

Implementation aspects can harm performance

Could even be used to attack your systems!

What Could be Seen as Related

- Algorithmic complexity attacks (software domain):
  - SlowFuzz
  - PerfFuzz
- *Automated Synthesis of Adversarial Workloads for Network Functions*, ACM Sigcomm 2018
- **Policy Injection: A Cloud Dataplane DoS Attack**, ACM Sigcomm DEMO 2018

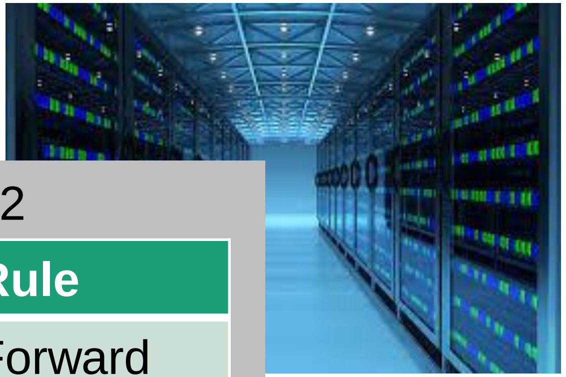
Why Important?

Implementation aspects can harm performance

Could even be used to attack your systems!

We propose NetBOA to automatically create network traffic input

Example: Benchmark Open vSwitch



Host 2

Match	Rule
	Forward
*	DROP

Open vSwitch

Example: Benchmark Open vSwitch

Host 1

Traffic Generator

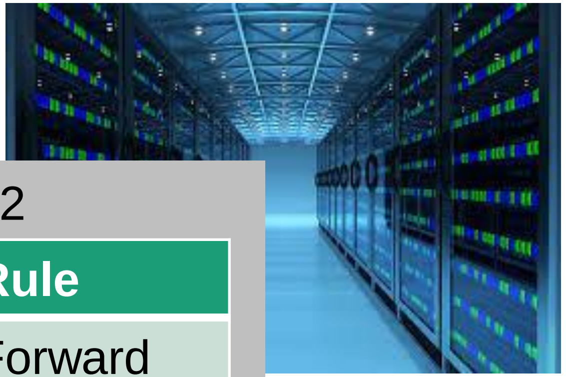
Config

Host 2

Match	Rule
	Forward
*	DROP

Open vSwitch

Example: Benchmark Open vSwitch



Host 1

Traffic Generator

Config

Host 2

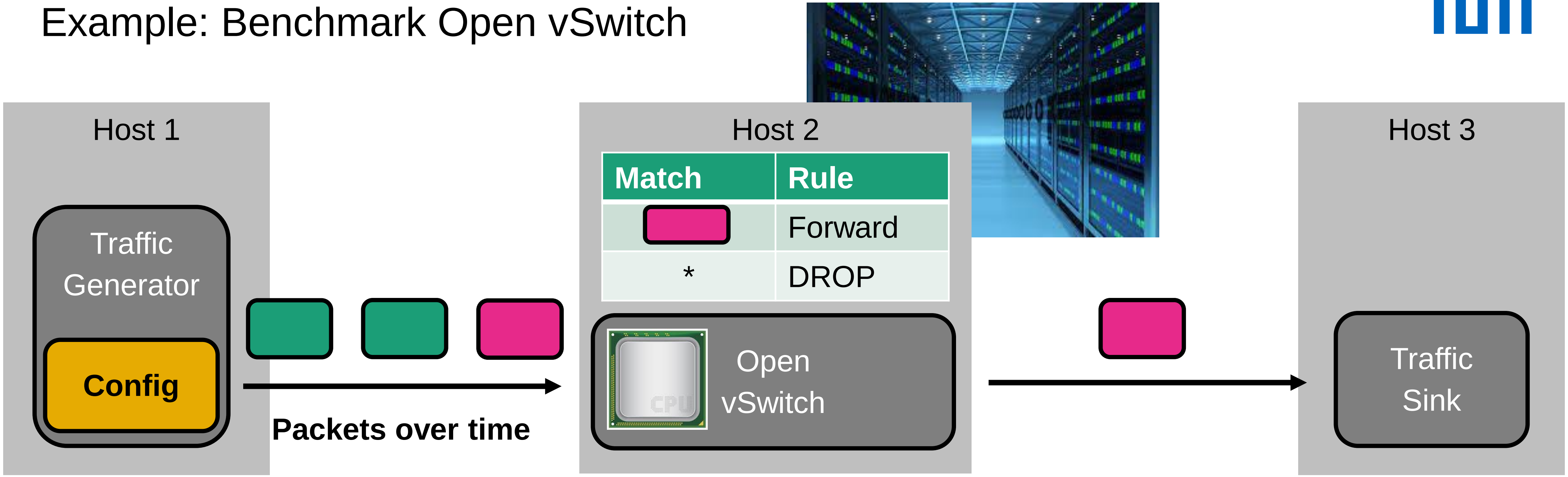
Match	**Rule**
	Forward
*	DROP

Open vSwitch

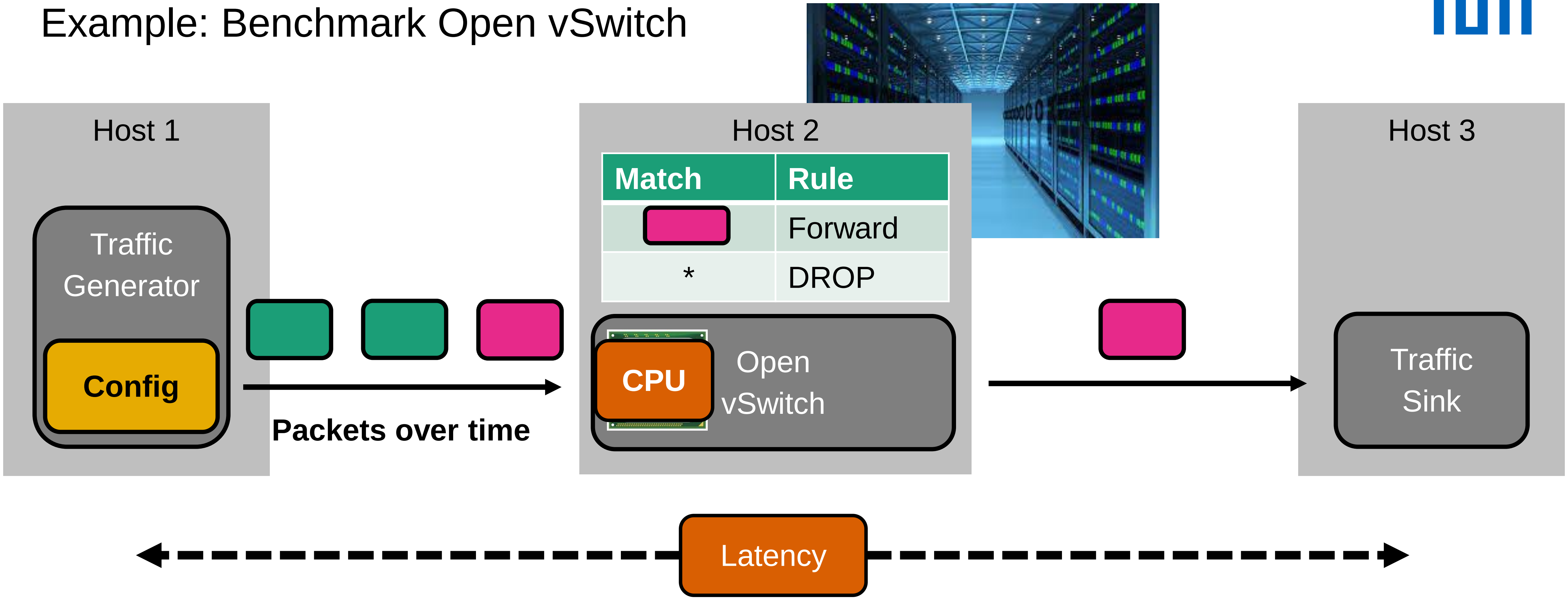
Host 3

Traffic Sink

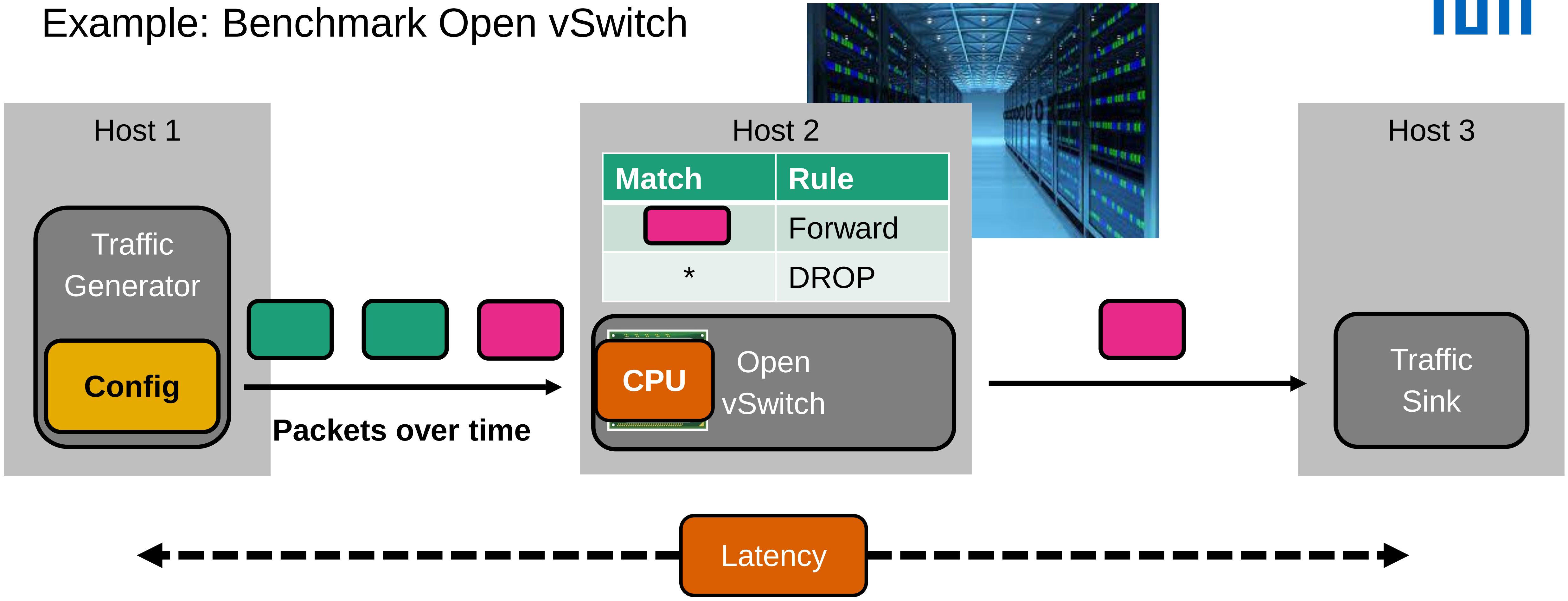
Example: Benchmark Open vSwitch



Example: Benchmark Open vSwitch



Example: Benchmark Open vSwitch



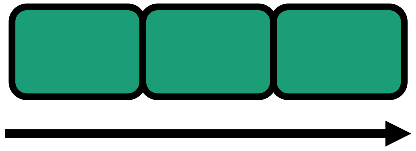
Goal: Find Network Traffic Configuration that Maximizes CPU/Latency

Network Benchmarking is Challenging: Complex and Huge Configuration Space

How many packets to send? How should headers look like? What protocol to use? When to send packets? Etc.

Network Benchmarking is Challenging: Complex and Huge Configuration Space

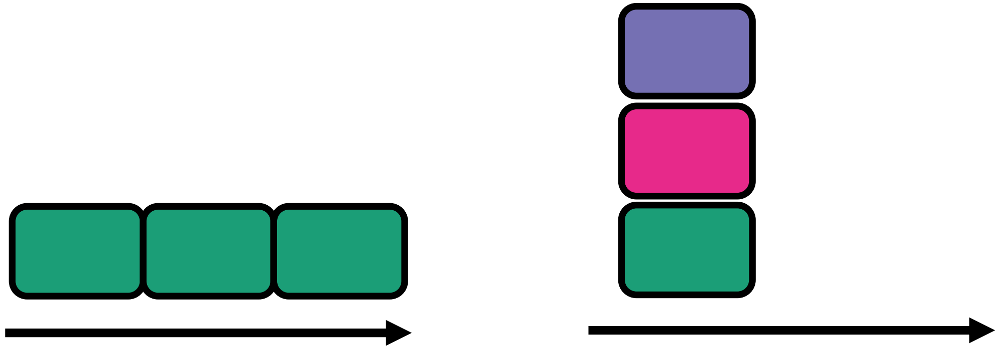
How many packets to send? How should headers look like? What protocol to use? When to send packets? Etc.



Number of Network Packets [1000 – 5000]

Network Benchmarking is Challenging: Complex and Huge Configuration Space

How many packets to send? How should headers look like? What protocol to use? When to send packets? Etc.

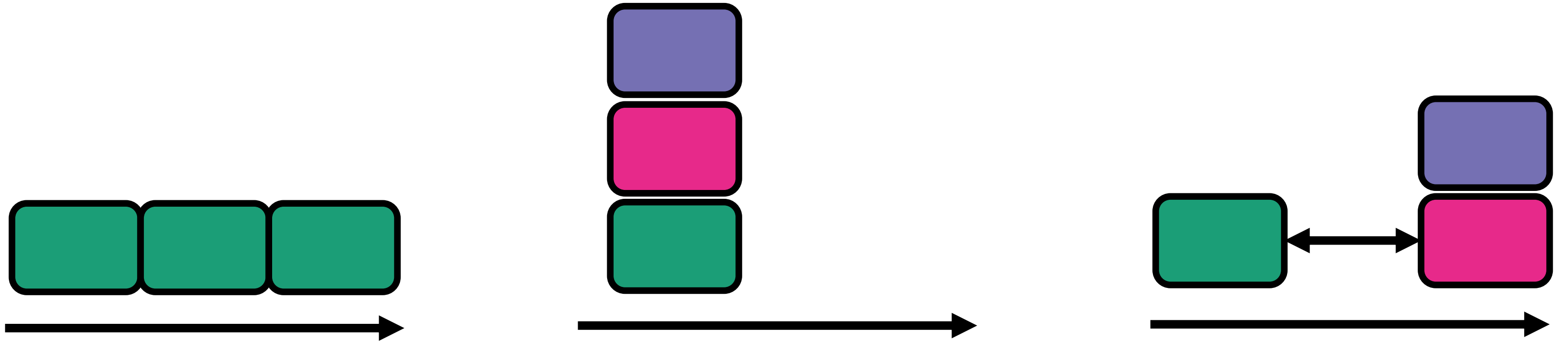


Number of Network Packets [1000 – 5000]

Batch Size [1-5]

Network Benchmarking is Challenging: Complex and Huge Configuration Space

How many packets to send? How should headers look like? What protocol to use? When to send packets? Etc.



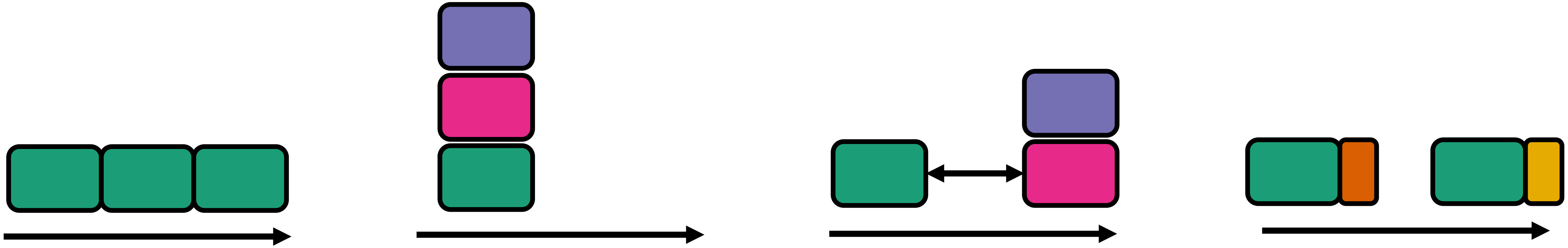
Number of Network Packets [1000 – 5000]

Batch Size [1-5]

Packet Inter Arrival Time [1ms – 13ms]

Network Benchmarking is Challenging: Complex and Huge Configuration Space

How many packets to send? How should headers look like? What protocol to use? When to send packets? Etc.



Number of Network Packets [1000 – 5000]

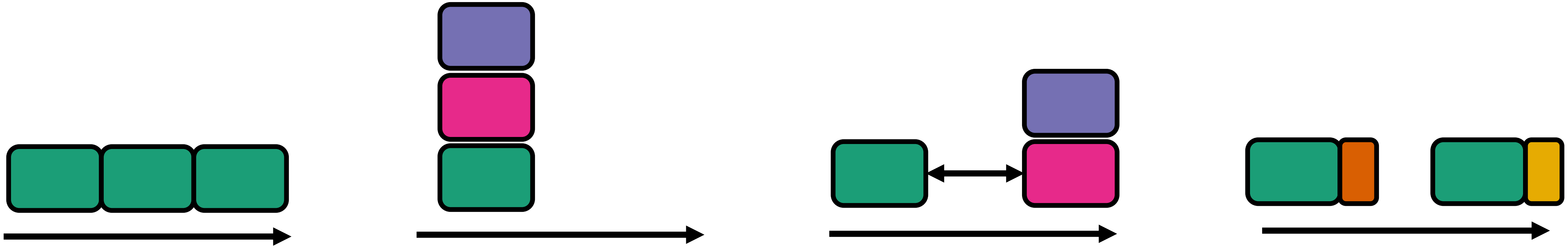
Batch Size [1-5]

Packet Inter Arrival Time [1ms – 13ms]

VLANs [1-5]

Network Benchmarking is Challenging: Complex and Huge Configuration Space

How many packets to send? How should headers look like? What protocol to use? When to send packets? Etc.



Number of Network Packets [1000 – 5000]

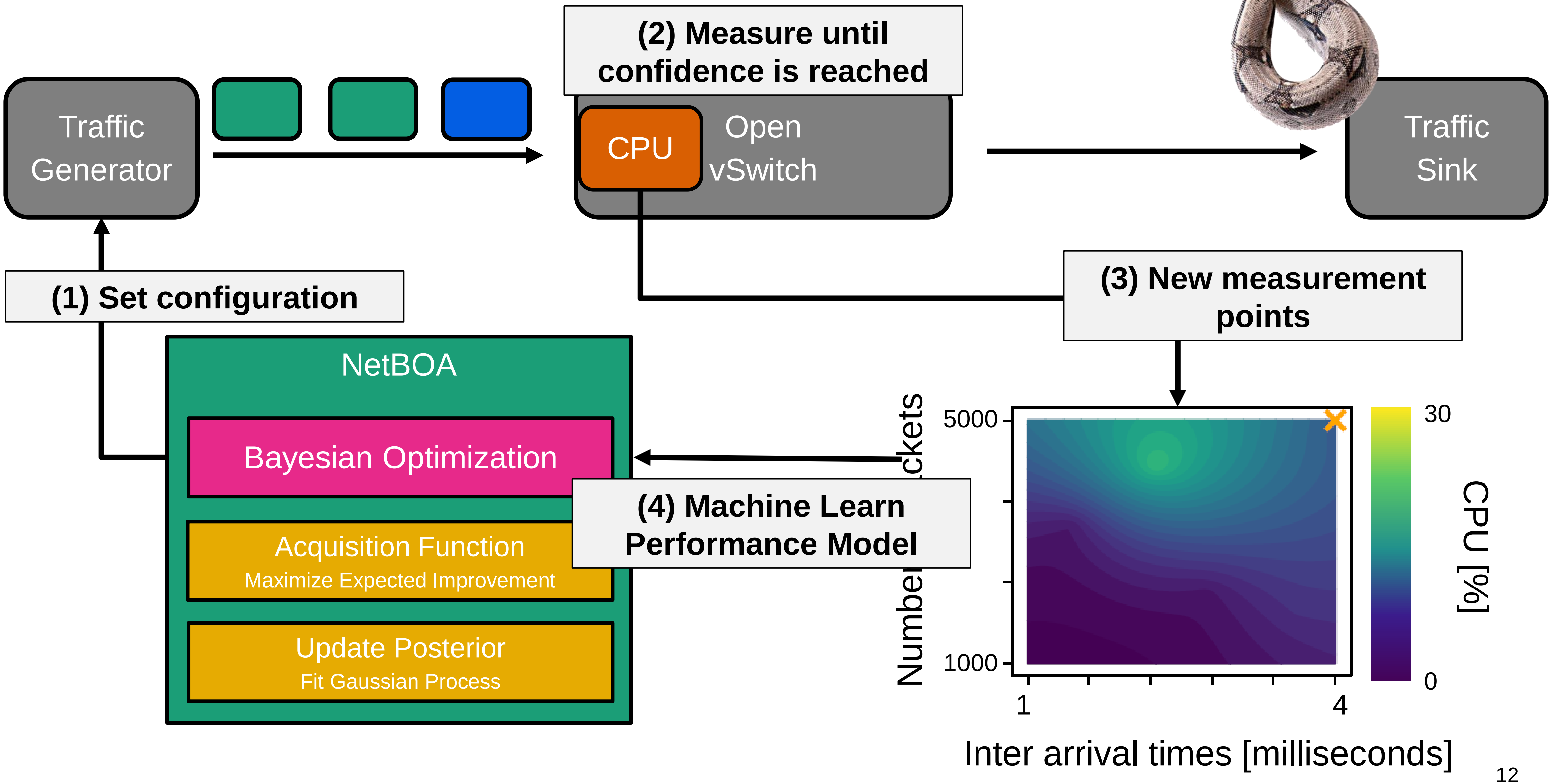
Batch Size [1-5]

Packet Inter Arrival Time [1ms – 13ms]

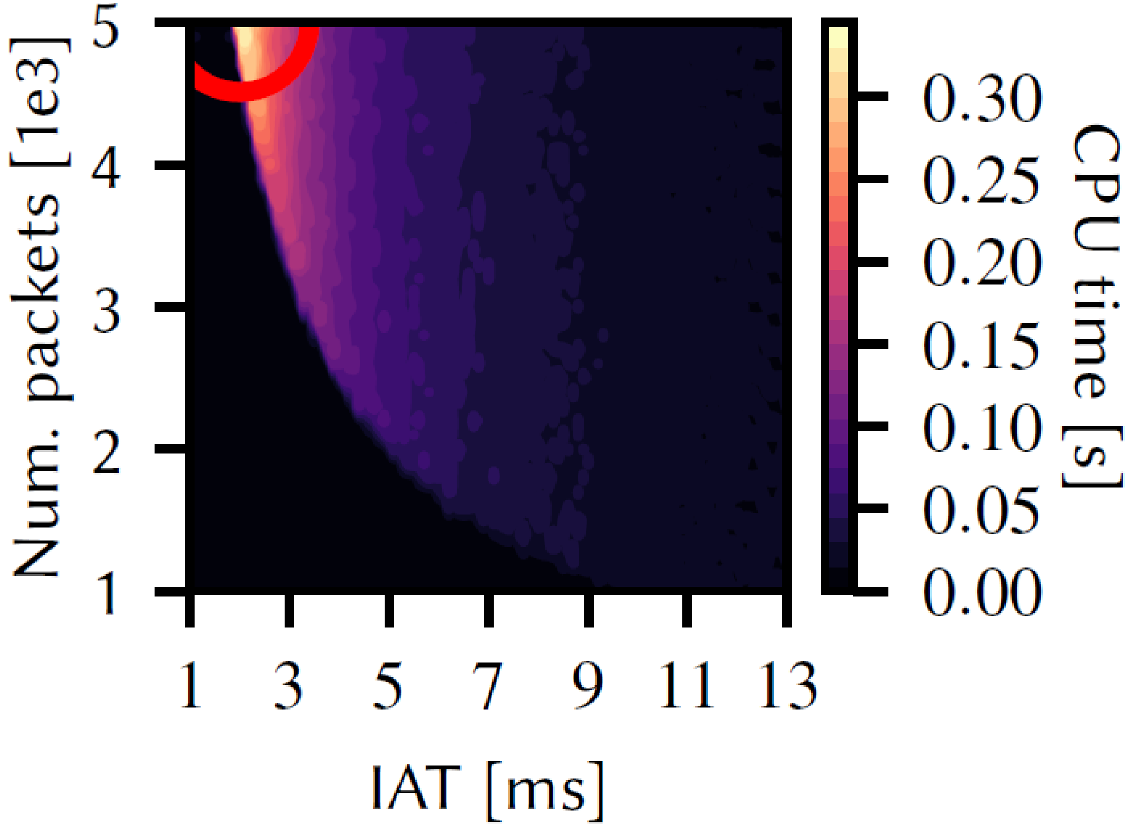
VLANs [1-5]

Source: https://de.kisspng.com/png-pc3d06/preview.html

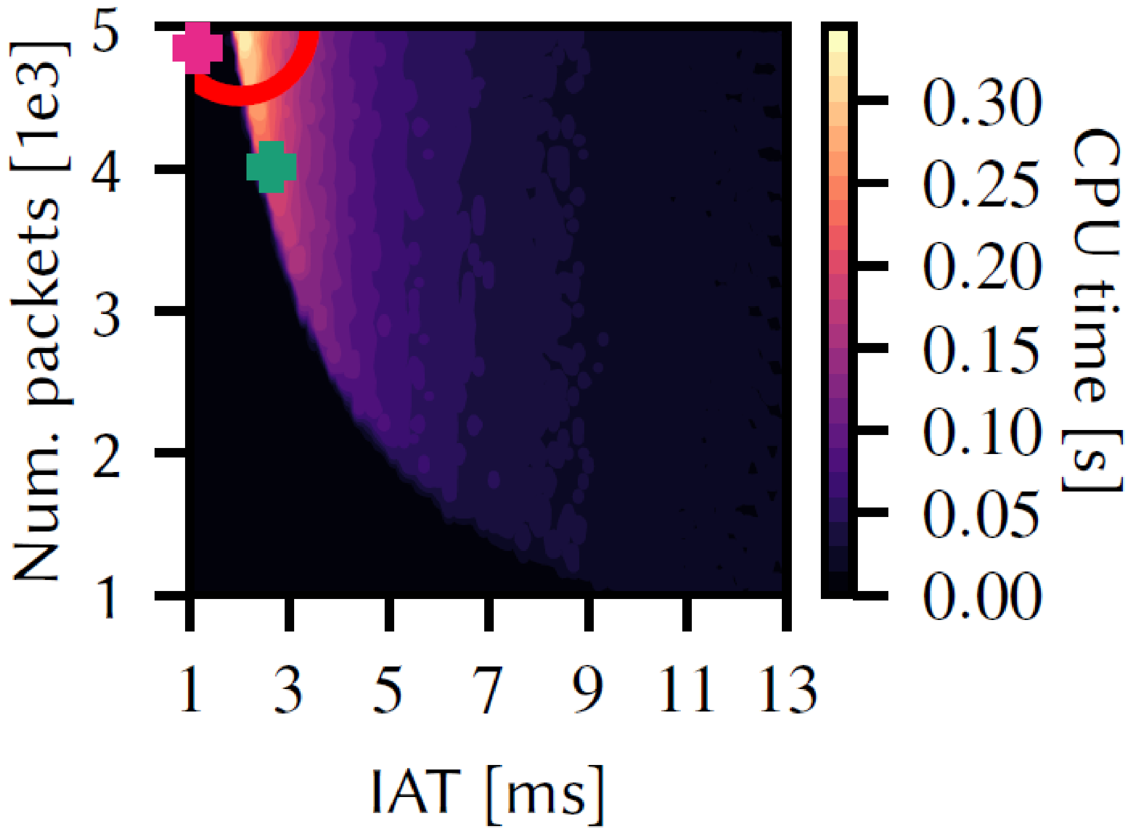
NetBOA: A Bayesian Optimization-based Approach



Grid Search for Two Parameters (Num. Packets and Inter Arrival Time)



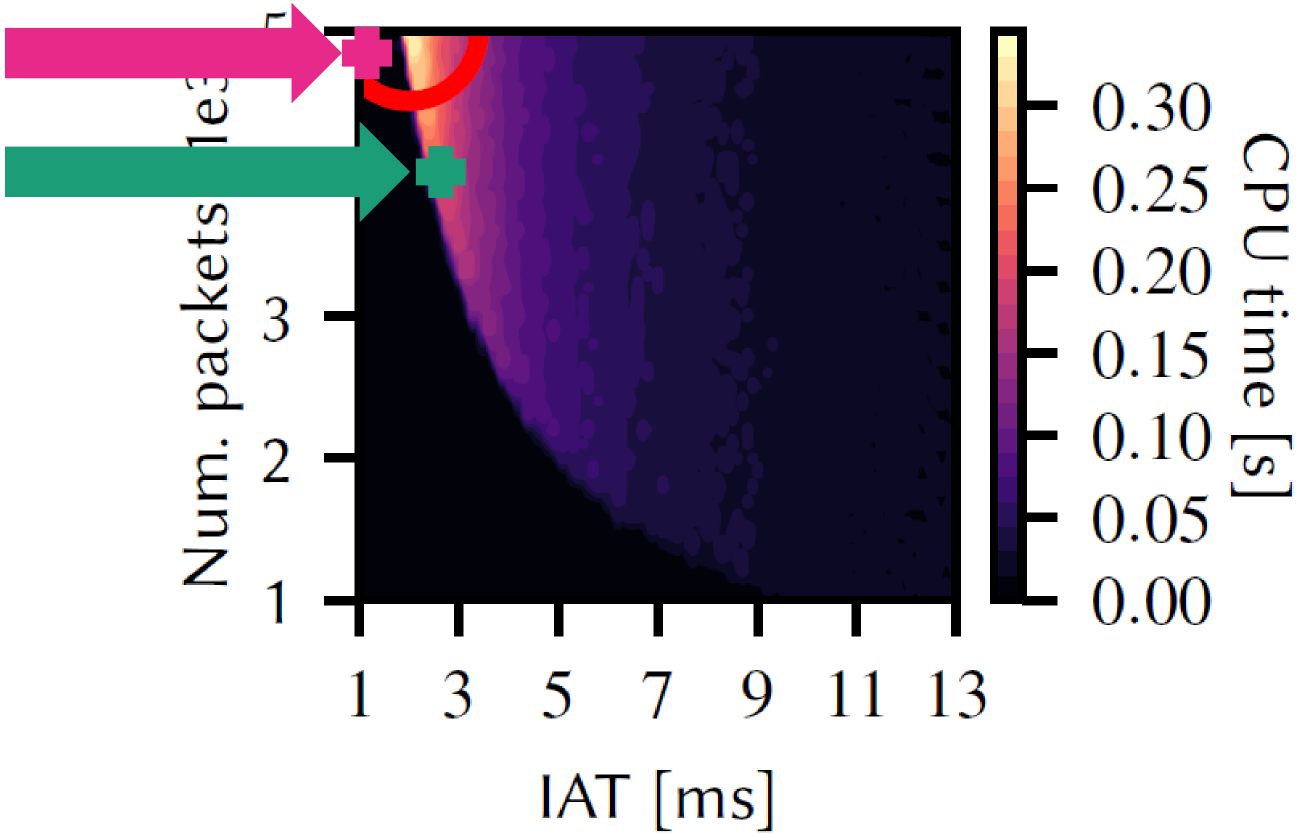
Grid Search for Two Parameters (Num. Packets and Inter Arrival Time)



Grid Search for Two Parameters (Num. Packets and Inter Arrival Time)

5000 packets, IAT 1 ms

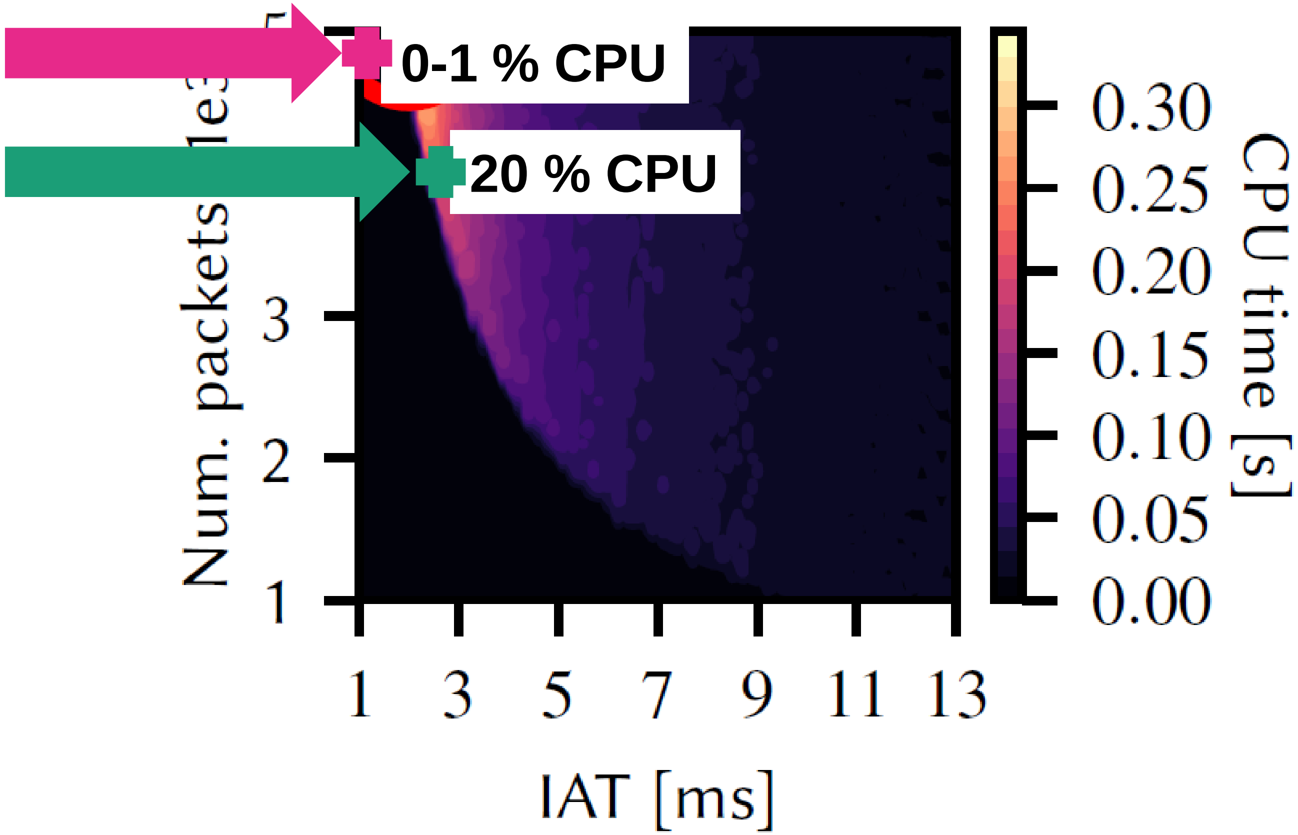
4000 packets, IAT 3 ms



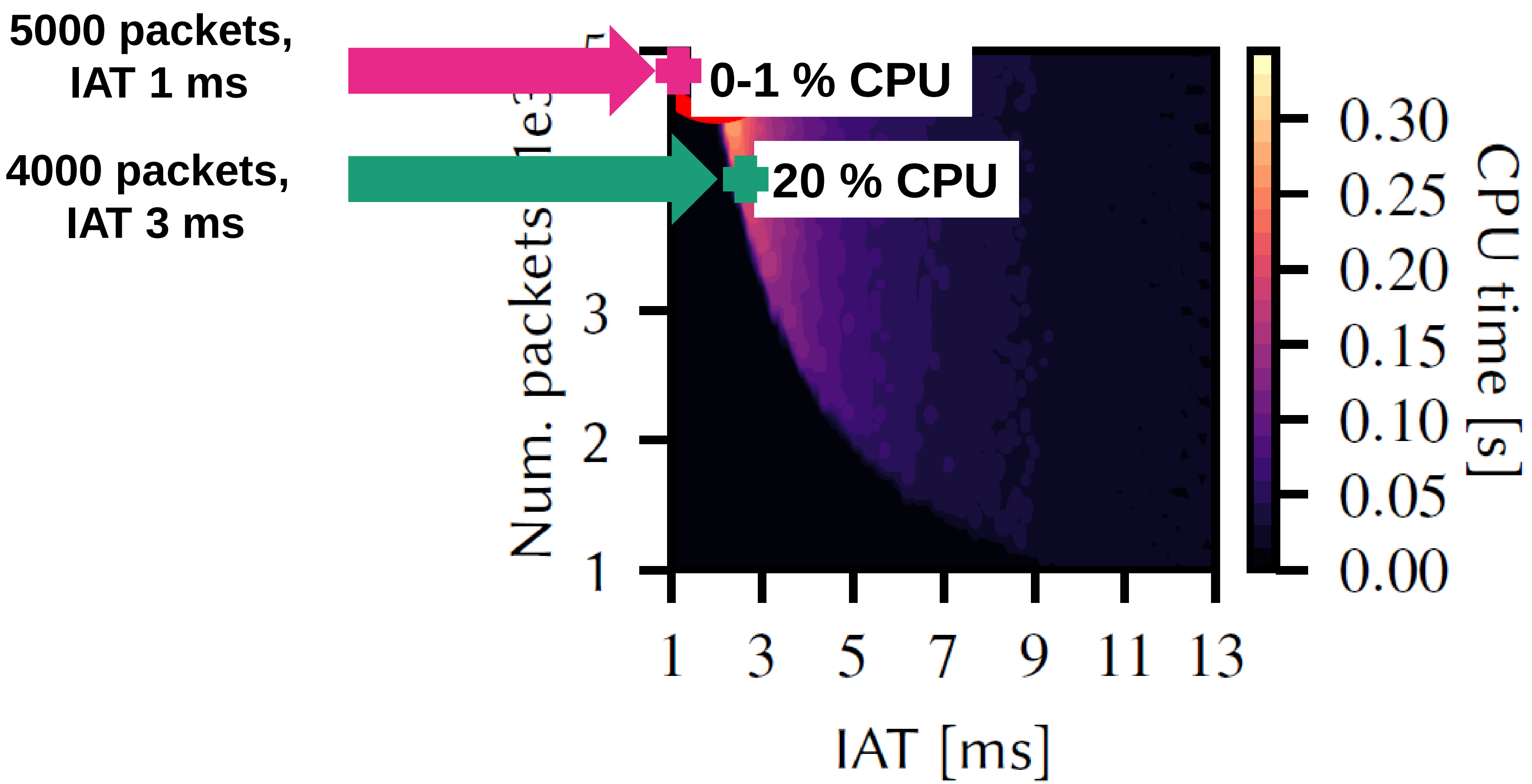
Grid Search for Two Parameters (Num. Packets and Inter Arrival Time)

5000 packets,
IAT 1 ms

4000 packets,
IAT 3 ms

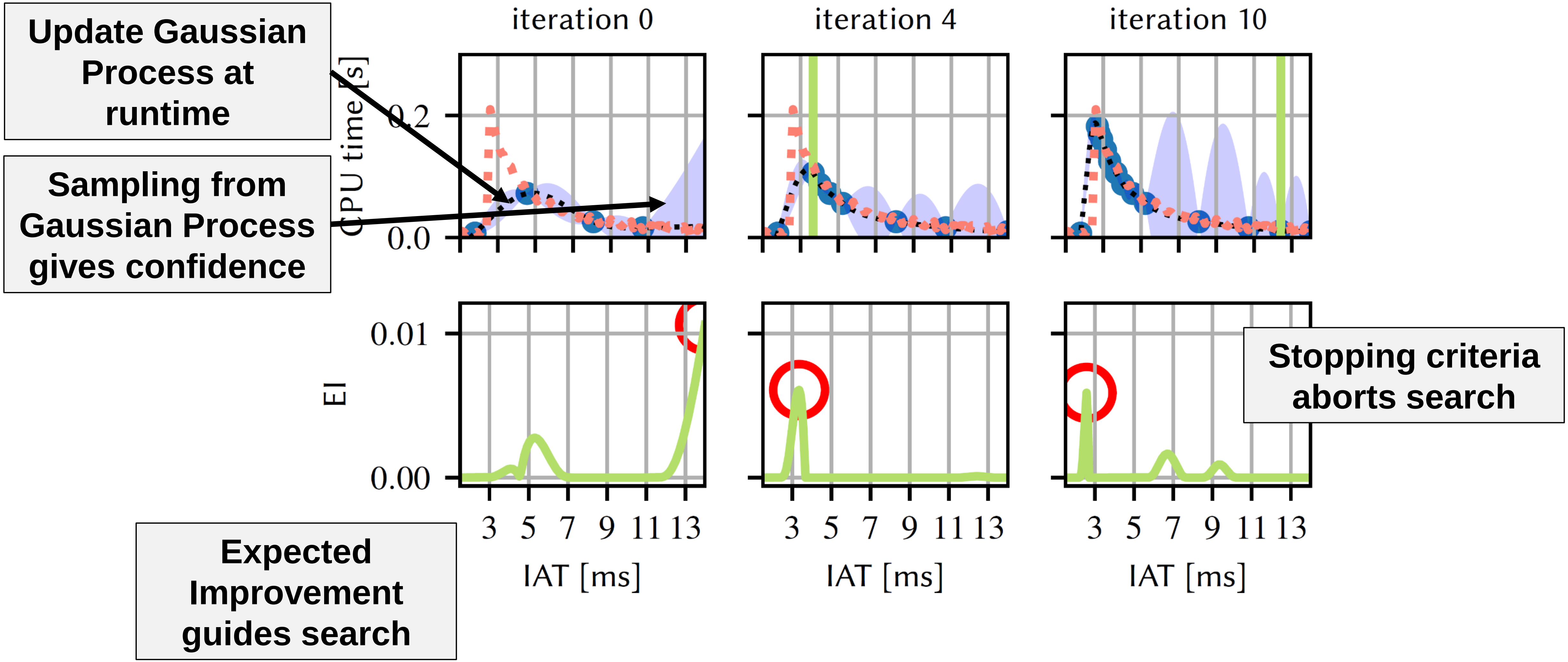


Grid Search for Two Parameters (Num. Packets and Inter Arrival Time)



- Performance models are non-trivial
- **Surprising:** Sending less network packets over time can lead to significantly higher CPU
- **But**: Can we find such weak-spots automatically?

Bayesian Optimization: NetBOA for Inter Arrival Time (IAT) Parameter



Why? Let Us Look At OvS Behavior!

Match	
1	Drop

OvS rule timeout 10 seconds

- We are using the OvS switch with the **Megaflow Cache enabled**

Why? Let Us Look At OvS Behavior!

Match	
1	Drop

Match	
1	Drop
2	Drop

OvS rule timeout 10 seconds

- We are using the OvS switch with the **Megaflow Cache enabled**

Why? Let Us Look At OvS Behavior!

Match	
1	Drop

Match	
1	Drop
2	Drop

Match	
1	Drop
2	Drop
3	Drop
…	…
N	Drop

- We are using the OvS switch with the **Megaflow Cache enabled**

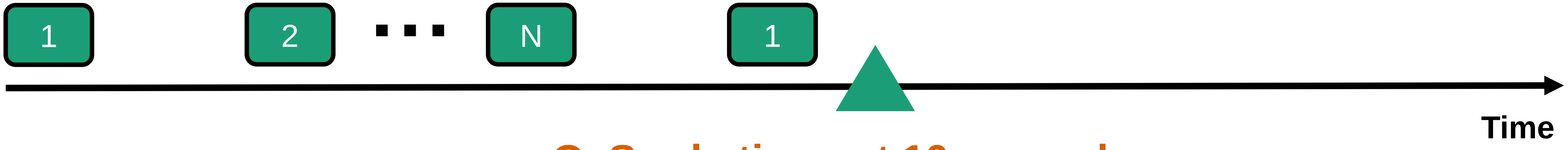
Why? Let Us Look At OvS Behavior!

Match	
1	Drop

Match	
1	Drop
2	Drop

Match	
1	Drop
2	Drop
3	Drop
…	…
N	Drop

Match	
1	Drop
2	Drop
3	Drop
…	…
N	Drop



OvS rule timeout 10 seconds

- We are using the OvS switch with the **Megaflow Cache enabled**

Why? Let Us Look At OvS Behavior!

Match	
1	Drop

Match	
1	Drop
2	Drop

Match	
1	Drop
2	Drop
3	Drop
…	…
N	Drop

OvS rule timeout 10 seconds

- We are using the OvS switch with the **Megaflow Cache enabled**

Why? Let Us Look At OvS Behavior!

Match	
1	Drop

Match	
1	Drop
2	Drop

Match	
1	Drop
2	Drop
3	Drop
…	…
N	Drop

Match	
2	Drop
3	Drop
…	…
N	Drop

OvS rule timeout 10 seconds

- We are using the OvS switch with the **Megaflow Cache enabled**

Why? Let Us Look At OvS Behavior!

Match	
1	Drop

Match	
1	Drop
2	Drop

Match	
1	Drop
2	Drop
3	Drop
…	…
N	Drop

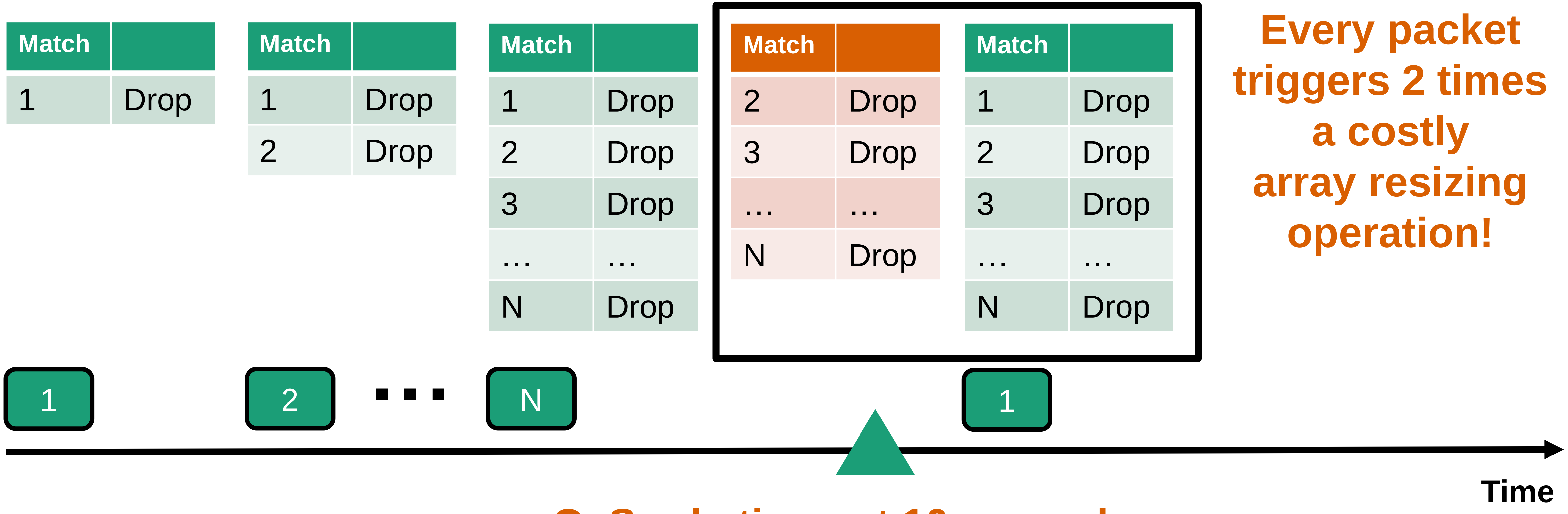
Match	
2	Drop
3	Drop
…	…
N	Drop

Match	
1	Drop
2	Drop
3	Drop
…	…
N	Drop

OvS rule timeout 10 seconds

- We are using the OvS switch with the **Megaflow Cache enabled**

Why? Let Us Look At OvS Behavior!

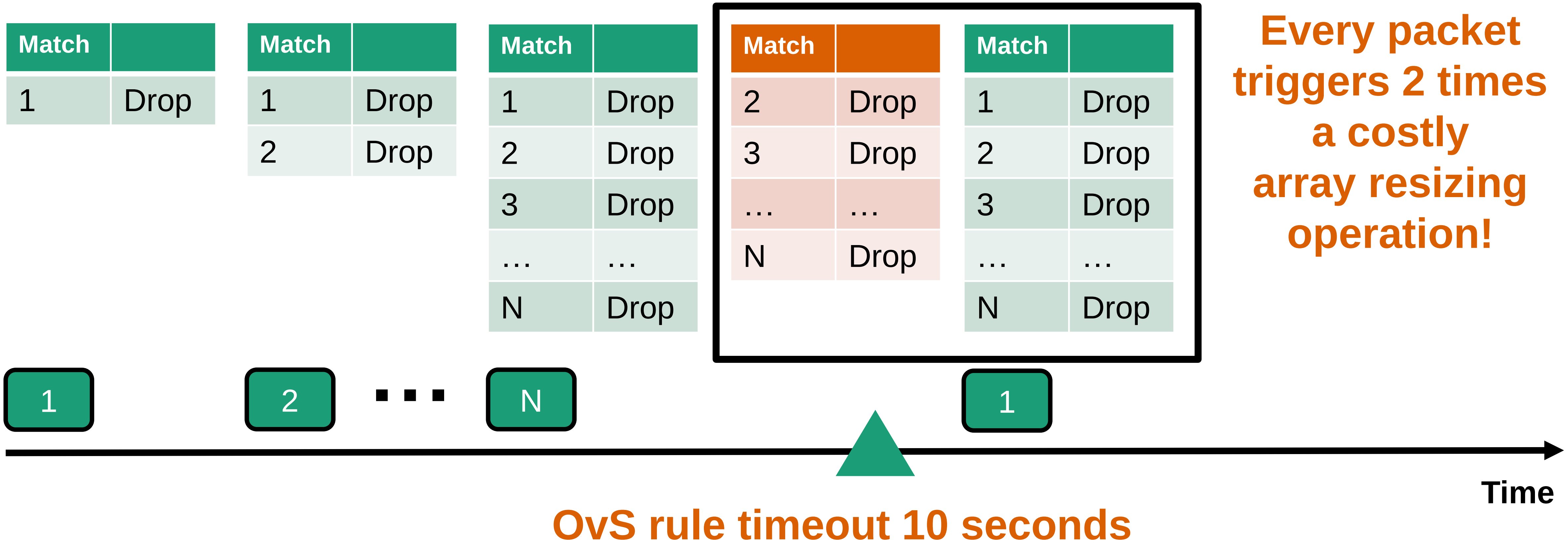


Every packet triggers 2 times a costly array resizing operation!

OvS rule timeout 10 seconds

- We are using the OvS switch with the **Megaflow Cache enabled**

Why? Let Us Look At OvS Behavior!



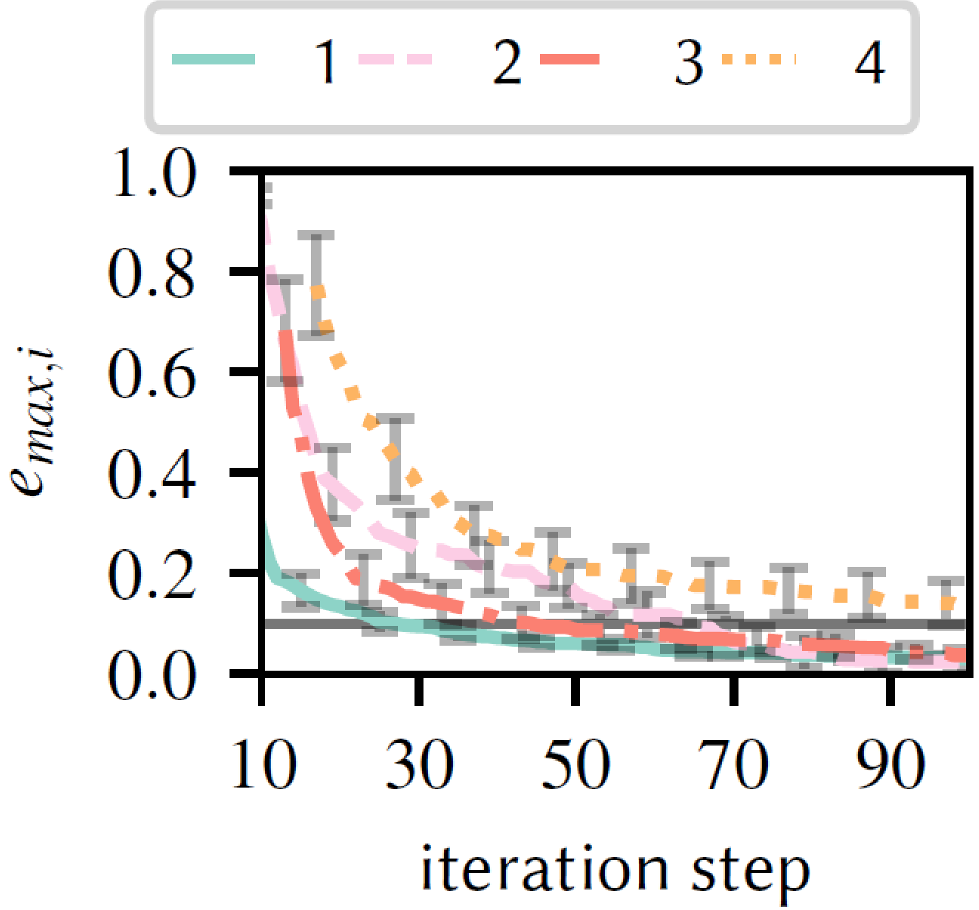
Every packet triggers 2 times a costly array resizing operation!

- We are using the OvS switch with the **Megaflow Cache enabled**
- For instance for 5000 packets: We trigger roughly every >2 ms a flow insertion + removal

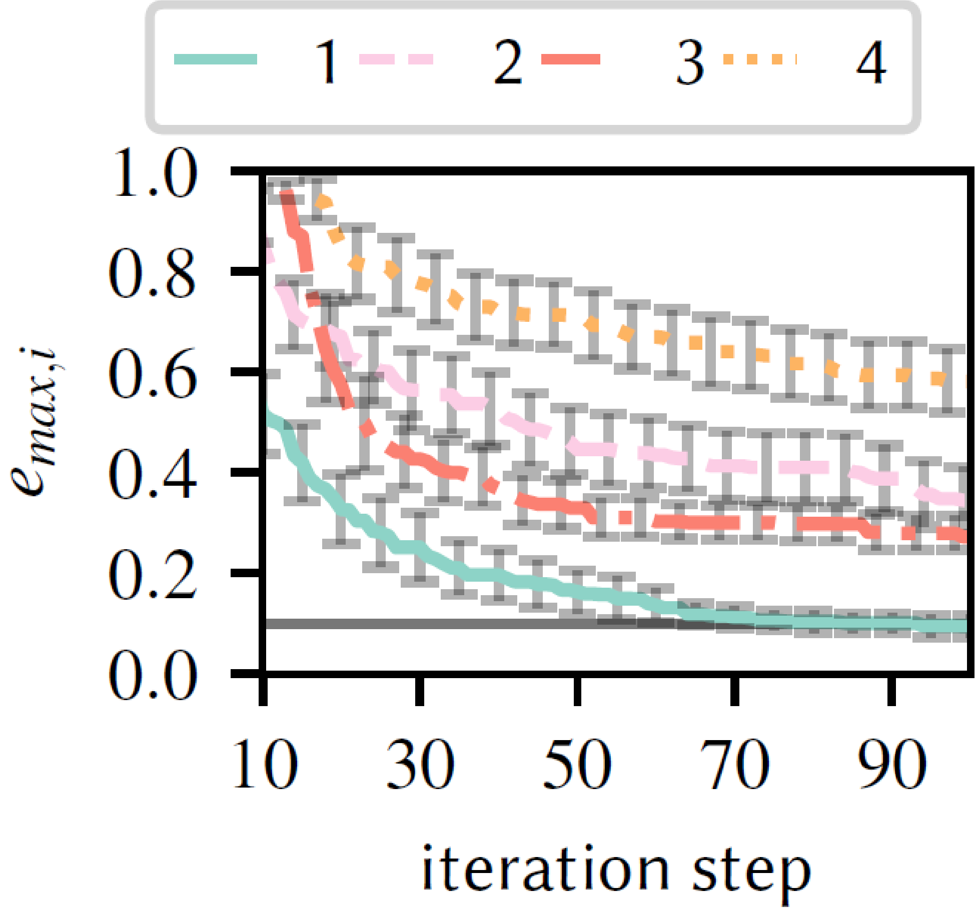
→ **Forcing OvS to continuously run through the array + resizing it**

NetBOA vs Random Search

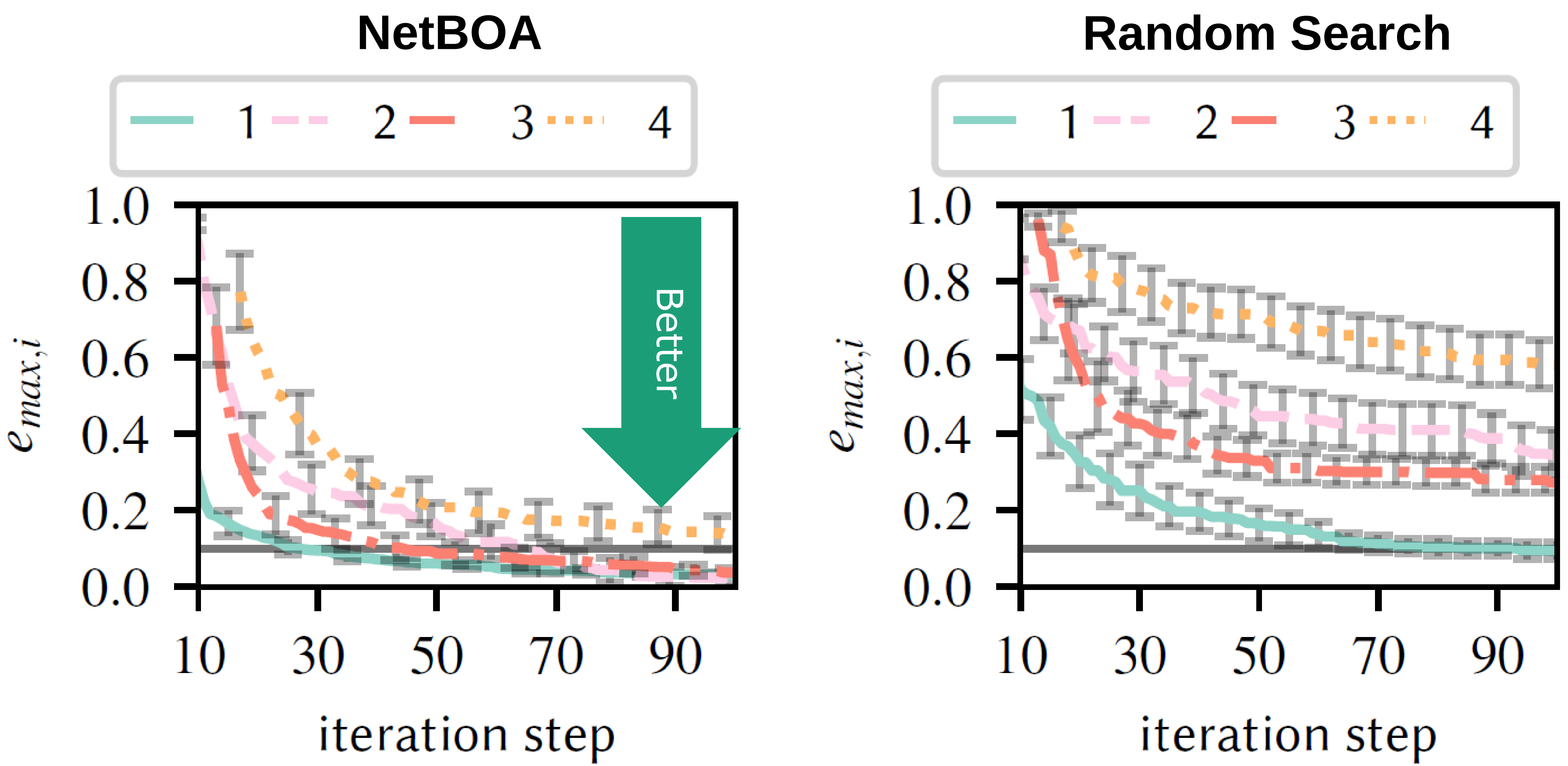
NetBOA



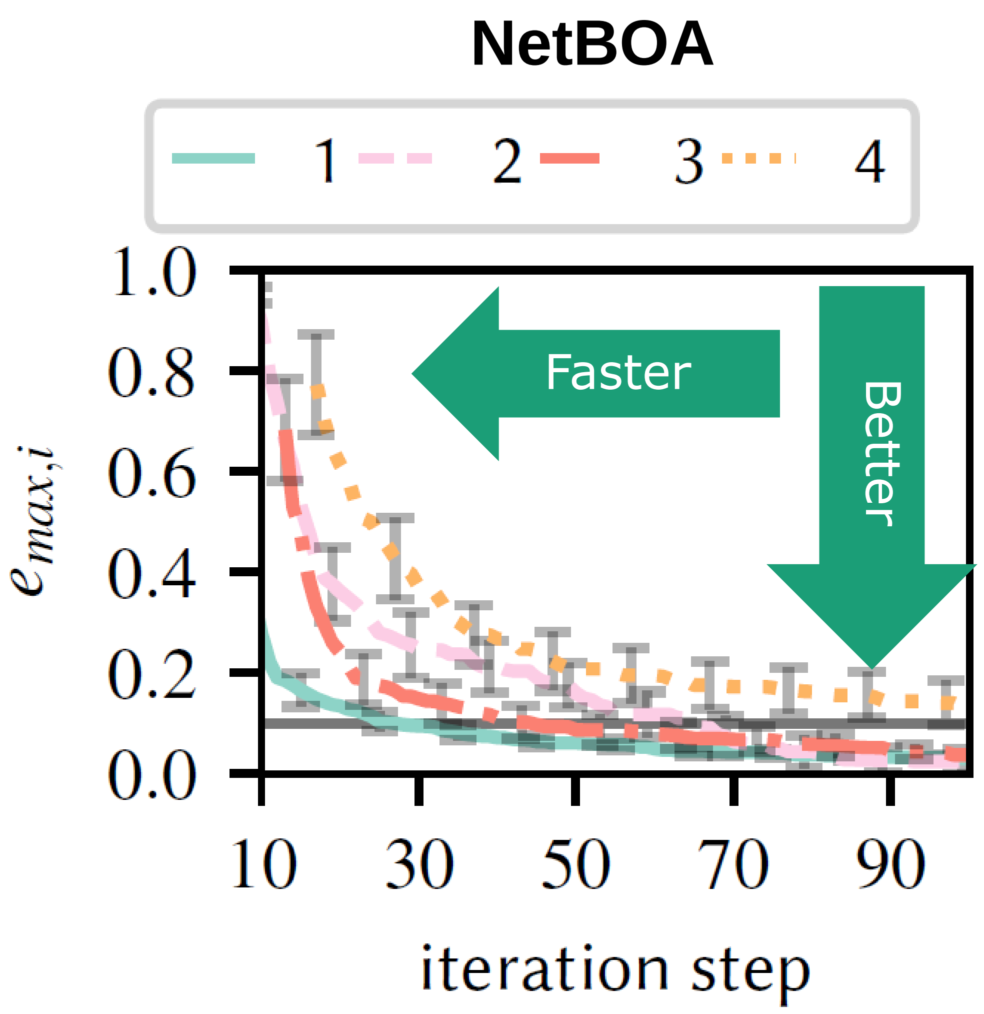
Random Search

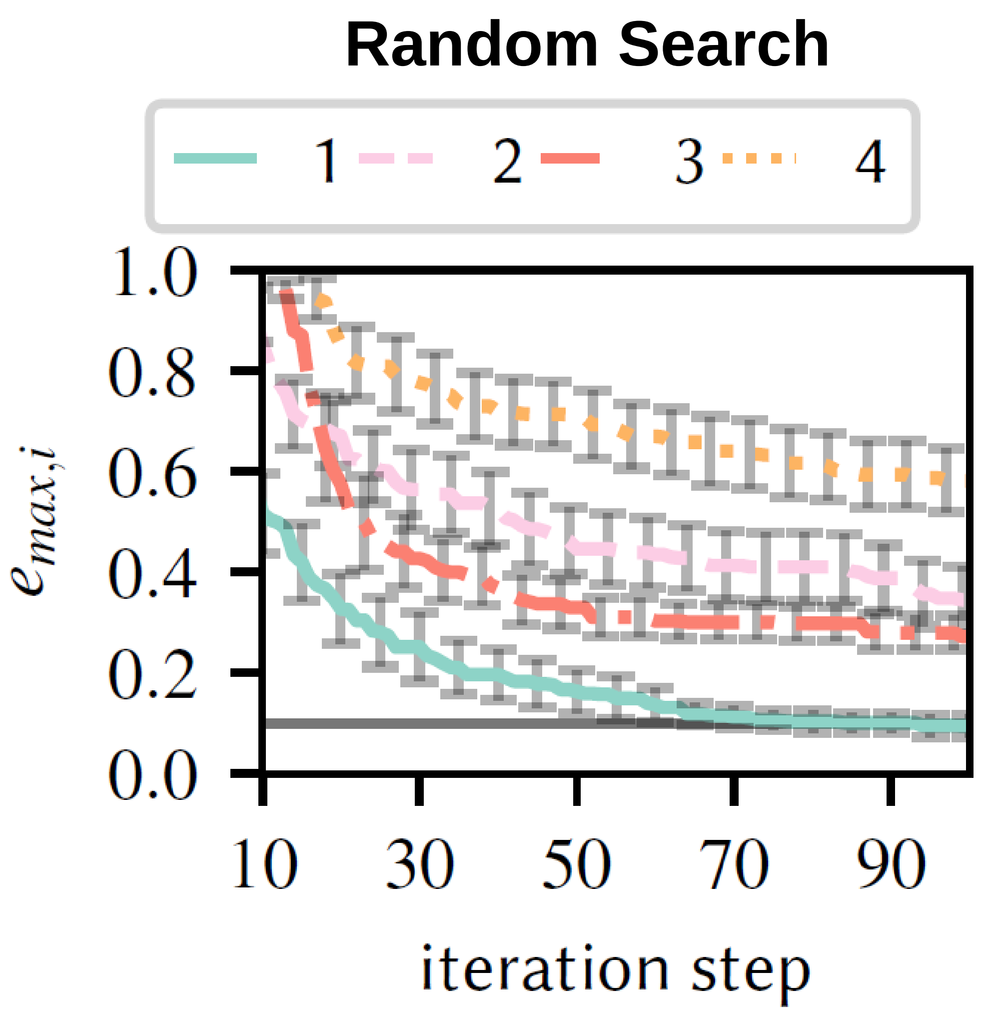


NetBOA vs Random Search

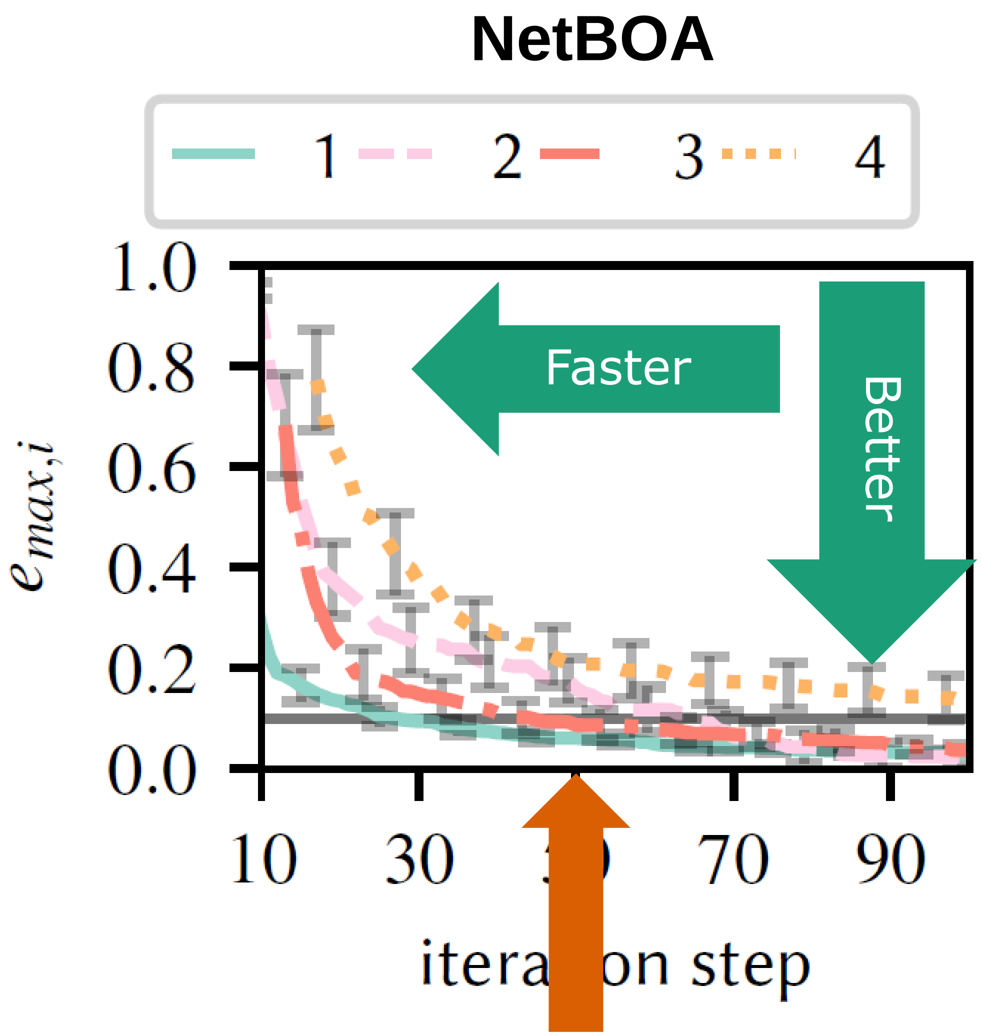


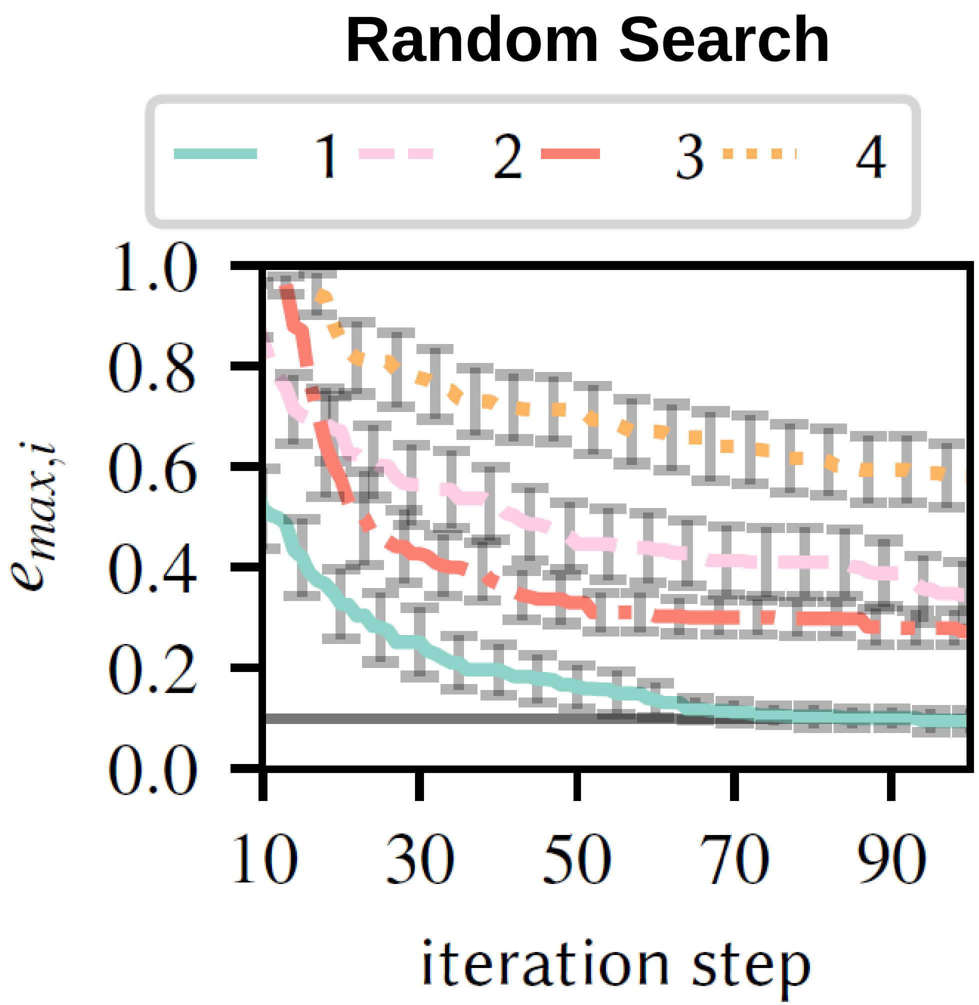
NetBOA vs Random Search





NetBOA vs Random Search





24 % higher CPU utilization

Part 1: Conclusion

- Adversarial input generation to find weak spots, security holes … to make your systems bullet-proof?

Part 1: Conclusion

- Adversarial input generation to find weak spots, security holes … to make your systems bullet-proof?

- Use case: NetBOA is a Bayesian Optimization-based data-driven approach to generate network traffic configurations for benchmarking network function implementations

Part 1: Conclusion

- Adversarial input generation to find weak spots, security holes … to make your systems bullet-proof?

- Use case: NetBOA is a Bayesian Optimization-based data-driven approach to generate network traffic configurations for benchmarking network function implementations

→NetBOA can efficiently find challenging network traffic configurations (maximize CPU/Latency)
→NetBOA can also be used to minimize, e.g., CPU or Latency

Part 1: Conclusion

- Adversarial input generation to find weak spots, security holes … to make your systems bullet-proof?

- Use case: NetBOA is a Bayesian Optimization-based data-driven approach to generate network traffic configurations for benchmarking network function implementations

→NetBOA can efficiently find challenging network traffic configurations (maximize CPU/Latency)
→NetBOA can also be used to minimize, e.g., CPU or Latency

- Open questions and problems:
  - Does beating the machine means it generalizes?
  - Does it scale?
  - Alternatives?
  - Bayesian Optimization needs also tuning!

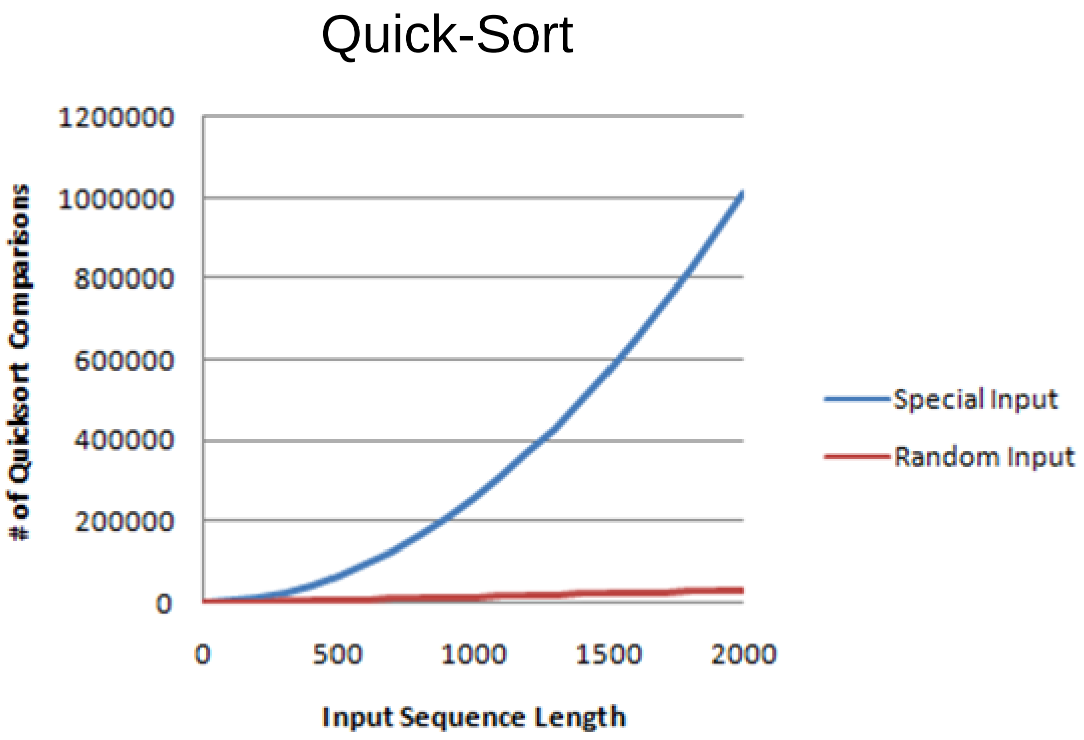
(2) Benchmarking Data Center Traffic Scheduling Algorithms

Motivation: Automation Helps Finding Weak-Spots

- O(n * log(n)) on average case

- O(n * n) on worst case (e.g. inversely sorted list for pivot on last element)

→Worst case can be calculated

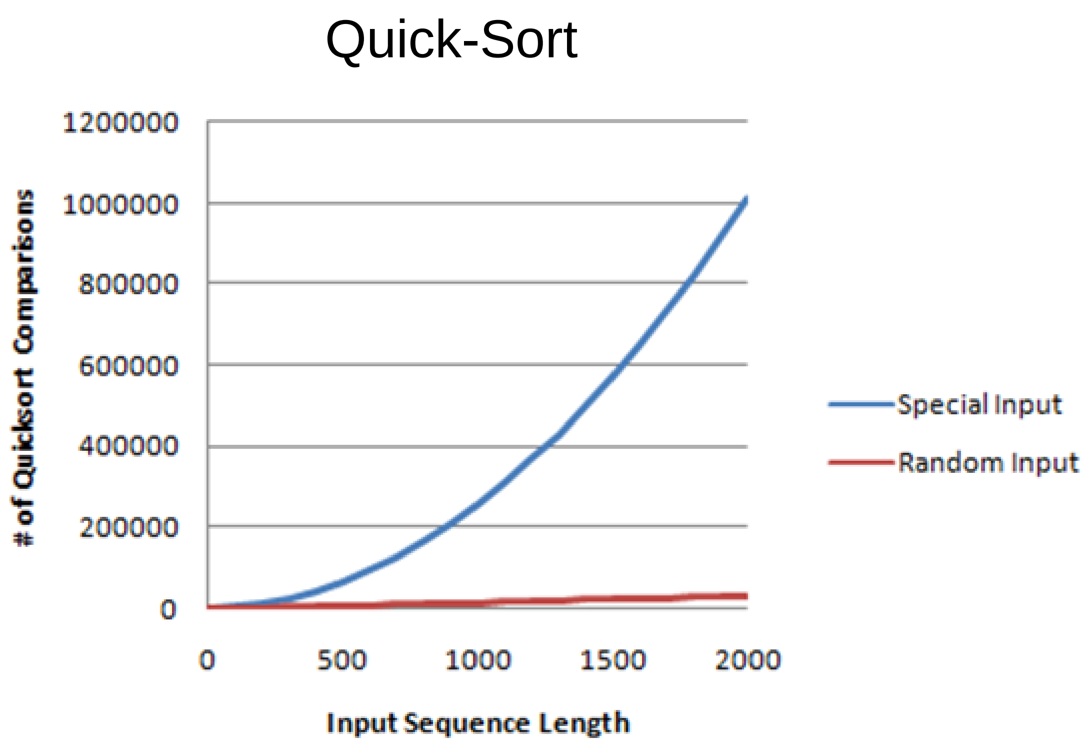


Motivation: Automation Helps Finding Weak-Spots

- O(n * log(n)) on average case

- O(n * n) on worst case (e.g. inversely sorted list for pivot on last element)

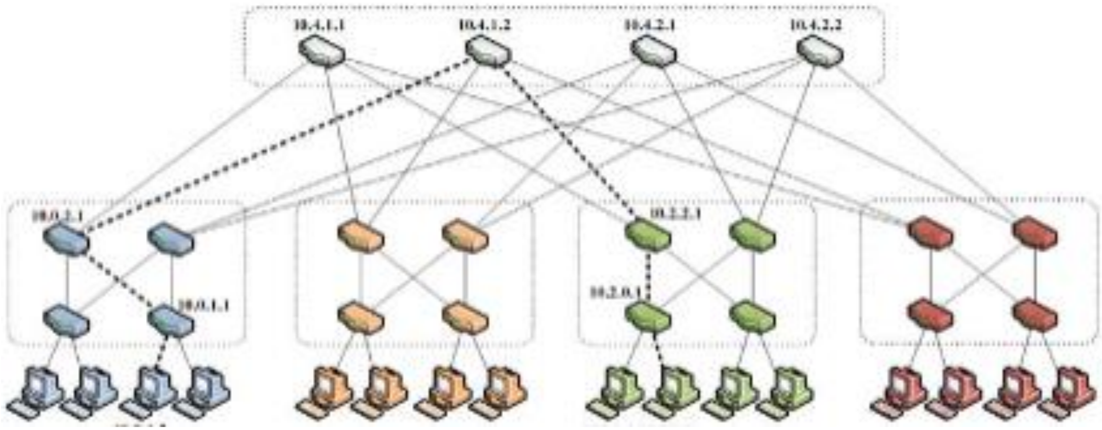
→Worst case can be calculated



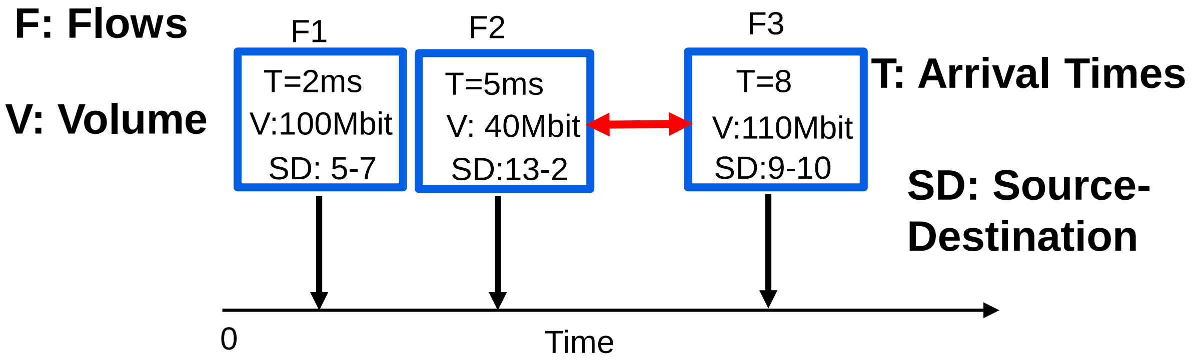
Question: How to apply automation to data center traffic?

[1] https://igoro.com/archive/quicksort-killer/

Data Center Scenario

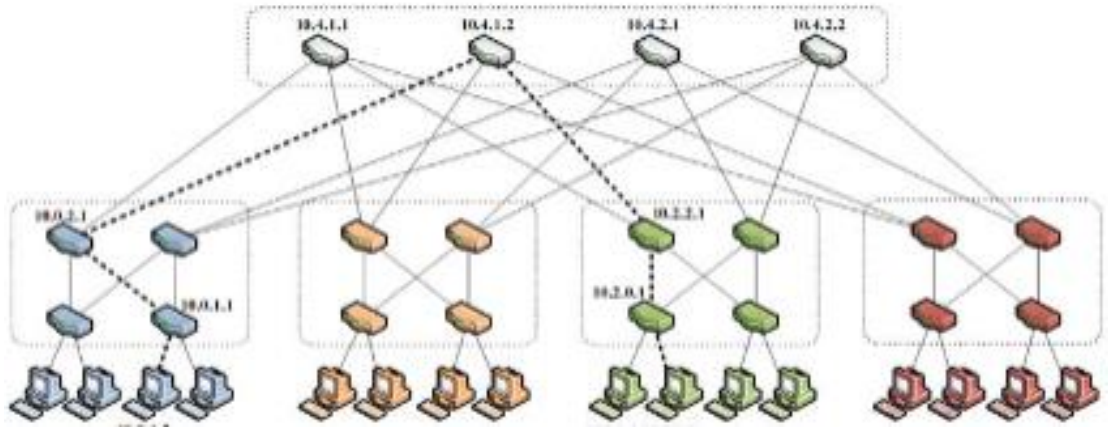


What can be changed?



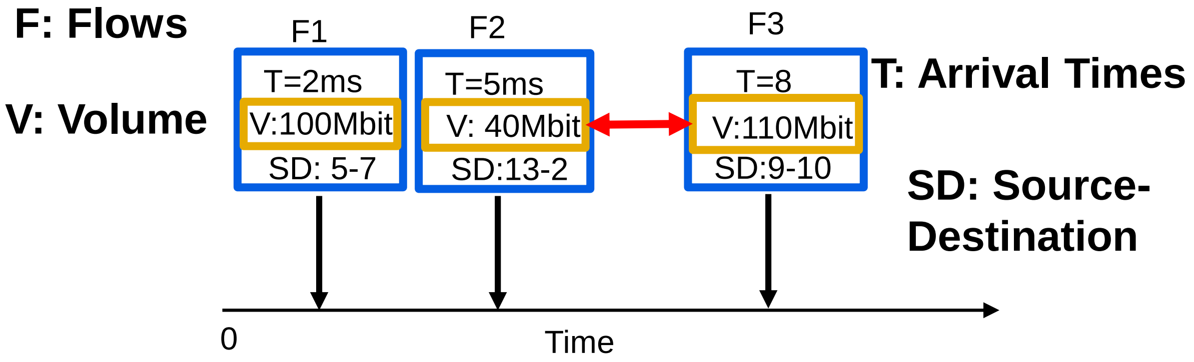
- Small flows are scheduled first
- Shortest-Path-Routing
- K=4 Fattree

Data Center Scenario



- Small flows are scheduled first
- Shortest-Path-Routing
- K=4 Fattree

What can be changed?

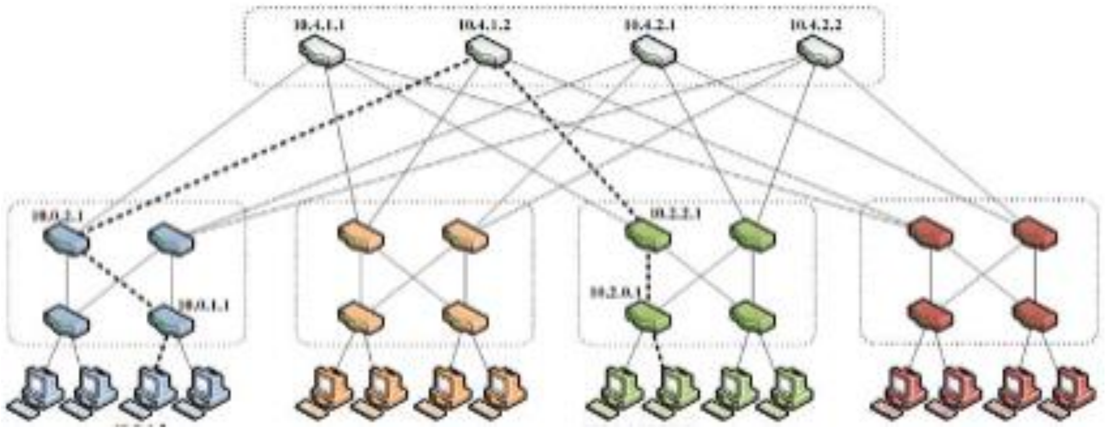


What we change

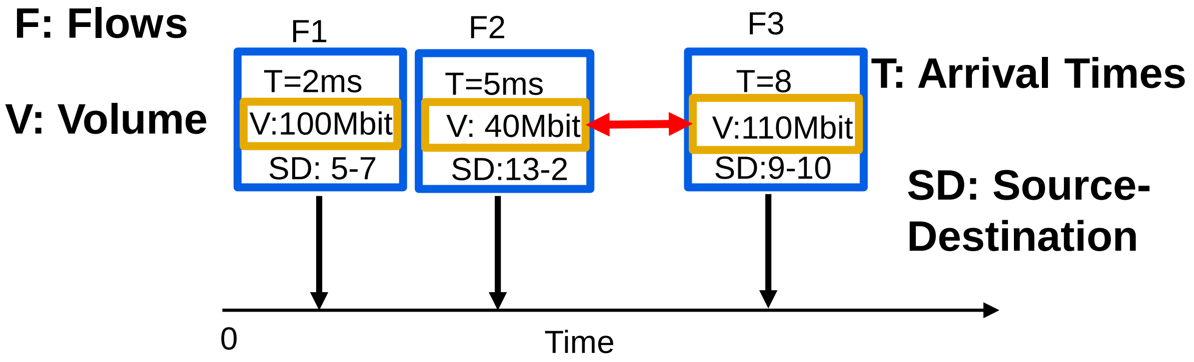
Assignment from volumes to flows

Set of flow volumes stays constant

Data Center Scenario



What can be changed?



What we change

Assignment from volumes to flows

Set of flow volumes stays constant

- Small flows are scheduled first
- Shortest-Path-Routing
- K=4 Fattree

Flow-Level Simulator evaluates traffic loads

Problem Definition

Given Set of Flows:

	F1	F2	F3	F4	F5	F6
Arrival Time	12ms	14ms	17ms	18ms	21ms	24ms
Source	3	4	13	2	3	12
Destination	14	12	7	7	1	6
Volume	10Mbit	400Mbit	90Mbit	200MBit	9Mbit	110Mbit

Find the order of Volumes such that:

$$\underset{F_N}{\operatorname{argmax}} \frac{1}{N} \sum_{i=1}^{N} FCT(f_i)$$

Problem Definition

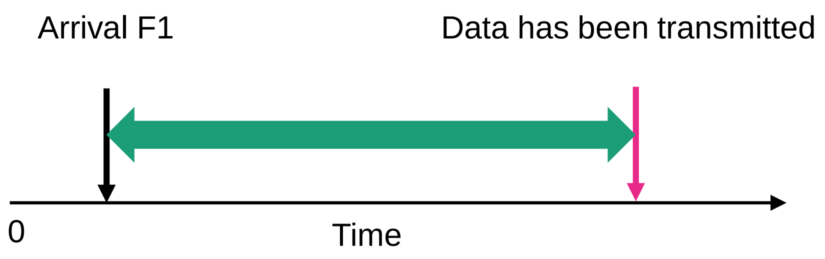
Given Set of Flows:

	F1	F2	F3	F4	F5	F6
Arrival Time	12ms	14ms	17ms	18ms	21ms	24ms
Source	3	4	13	2	3	12
Destination	14	12	7	7	1	6
Volume	10Mbit	400Mbit	90Mbit	200MBit	9Mbit	110Mbit

Find the order of Volumes such that:

$$\underset{F_N}{\operatorname{argmax}} \frac{1}{N} \sum_{i=1}^{N} FCT(f_i)$$

FCT: Flow Completion Time



Approach: Genetic Algorithm

Pseudo Code

12ms	14ms	17ms	18ms	21ms	24ms
3	4	13	2	3	12
14	12	7	7	1	6
10Mbit	400Mbit	90Mbit	200MBit	9Mbit	110Mbit

Approach: Genetic Algorithm

Pseudo Code

1. Sample N flows

12ms	14ms	17ms	18ms	21ms	24ms
3	4	13	2	3	12
14	12	7	7	1	6
10Mbit	400Mbit	90Mbit	200MBit	9Mbit	110Mbit

Approach: Genetic Algorithm

<u>Pseudo Code</u>

1. Sample N flows

x	x	x	x	x	x
x	x	x	x	x	x
x	x	x	x	x	x
10Mbit	400Mbit	90Mbit	200MBit	9Mbit	110Mbit

Approach: Genetic Algorithm

<u>Pseudo Code</u>

1. Sample N flows
2. Extract the sequence of volumes V

x	x	x	x	x	x
x	x	x	x	x	x
x	x	x	x	x	x
10Mbit	400Mbit	90Mbit	200MBit	9Mbit	110Mbit

Approach: Genetic Algorithm

<u>Pseudo Code</u>

1. Sample N flows
2. Extract the sequence of volumes V
3. Generate permutations of V by changing is order (=initial population)

x	x	x	x	x	x
x	x	x	x	x	x
x	x	x	x	x	x
10Mbit	400Mbit	90Mbit	200MBit	9Mbit	110Mbit

Approach: Genetic Algorithm

<u>Pseudo Code</u>

1. Sample N flows
2. Extract the sequence of volumes V
3. Generate permutations of V by changing is order (=initial population)
4. Repeat (until convergence)

x	x	x	x	x	x
x	x	x	x	x	x
x	x	x	x	x	x
10Mbit	400Mbit	90Mbit	200MBit	9Mbit	110Mbit

Approach: Genetic Algorithm

<u>Pseudo Code</u>

1. Sample N flows
2. Extract the sequence of volumes V
3. Generate permutations of V by changing is order (=initial population)
4. Repeat (until convergence)

   Crossover

x	x	x	x	x	x
x	x	x	x	x	x
x	x	x	x	x	x
10Mbit	400Mbit	90Mbit	200MBit	9Mbit	110Mbit

Approach: Genetic Algorithm

<u>Pseudo Code</u>

1. Sample N flows
2. Extract the sequence of volumes V
3. Generate permutations of V by changing is order (=initial population)
4. Repeat (until convergence)

    Crossover

    Mutation

x	x	x	x	x	x
x	x	x	x	x	x
x	x	x	x	x	x
10Mbit	400Mbit	90Mbit	200MBit	9Mbit	110Mbit

Approach: Genetic Algorithm

Pseudo Code

1. Sample N flows
2. Extract the sequence of volumes V
3. Generate permutations of V by changing is order (=initial population)
4. Repeat (until convergence)

   Crossover

   Mutation

   Fitness

x	x	x	x	x	x
x	x	x	x	x	x
x	x	x	x	x	x
10Mbit	400Mbit	90Mbit	200MBit	9Mbit	110Mbit

Approach: Genetic Algorithm

Pseudo Code

1. Sample N flows
2. Extract the sequence of volumes V
3. Generate permutations of V by changing is order (=initial population)
4. Repeat (until convergence)

   Crossover

   Mutation

   Fitness

   Selection

x	x	x	x	x	x
x	x	x	x	x	x
x	x	x	x	x	x
10Mbit	400Mbit	90Mbit	200MBit	9Mbit	110Mbit

Fitness Function

x	x	x	x	x	x
x	x	x	x	x	x
x	x	x	x	x	x
10	30	12	55	4	7

Flow Completion Time
(Fitness)

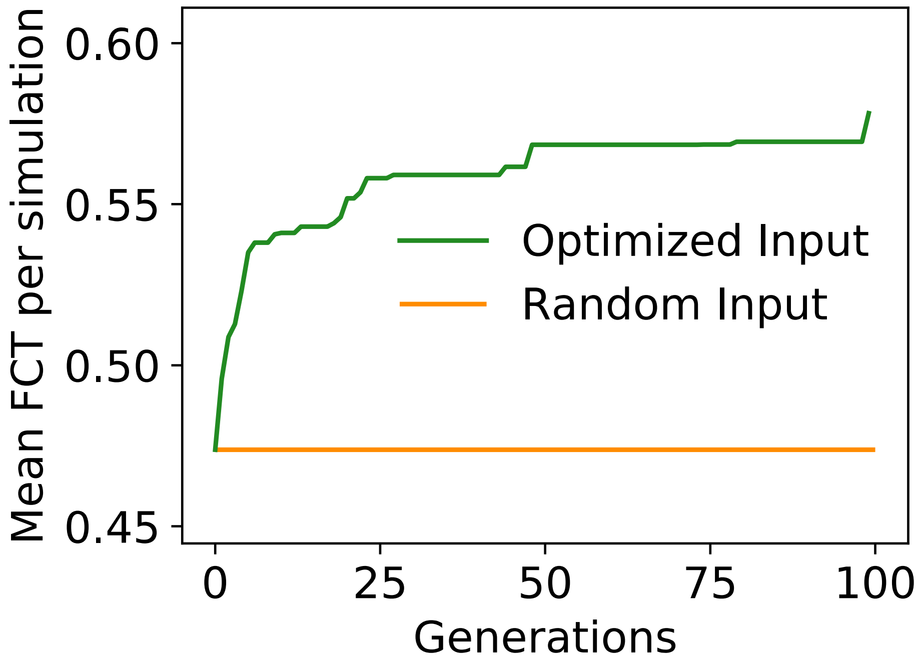
Data Center
Simulator

Simulator:

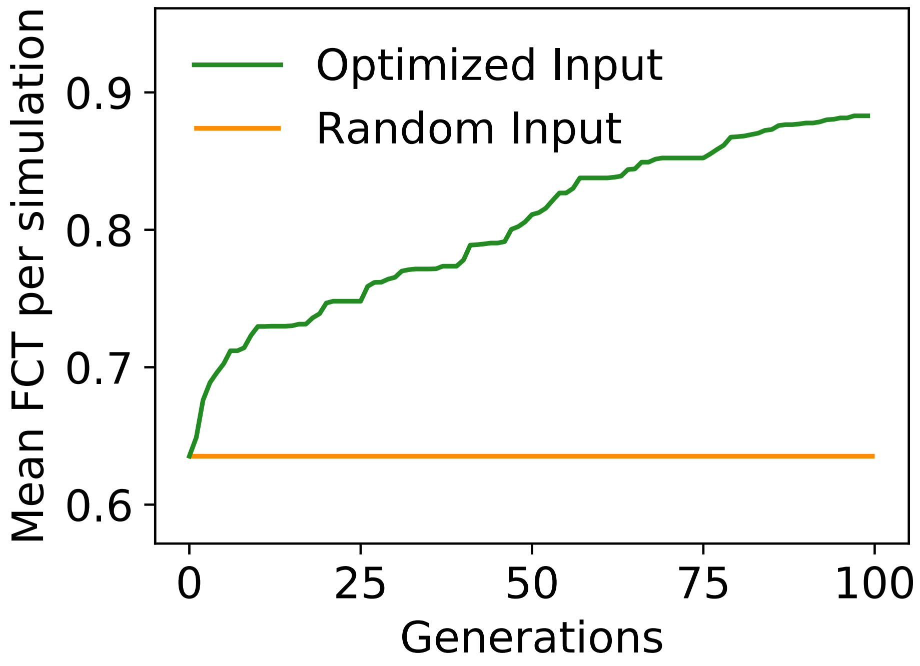
- 500 Mbit Links
- Uniform Volume between 1 and 500 Mbit
- Poisson Arrival Times (mean 0.7 sec)
- Uniform Src-Dst pairs

GA Behavior over Generations for Different Population Sizes

Population with N = 10 flows

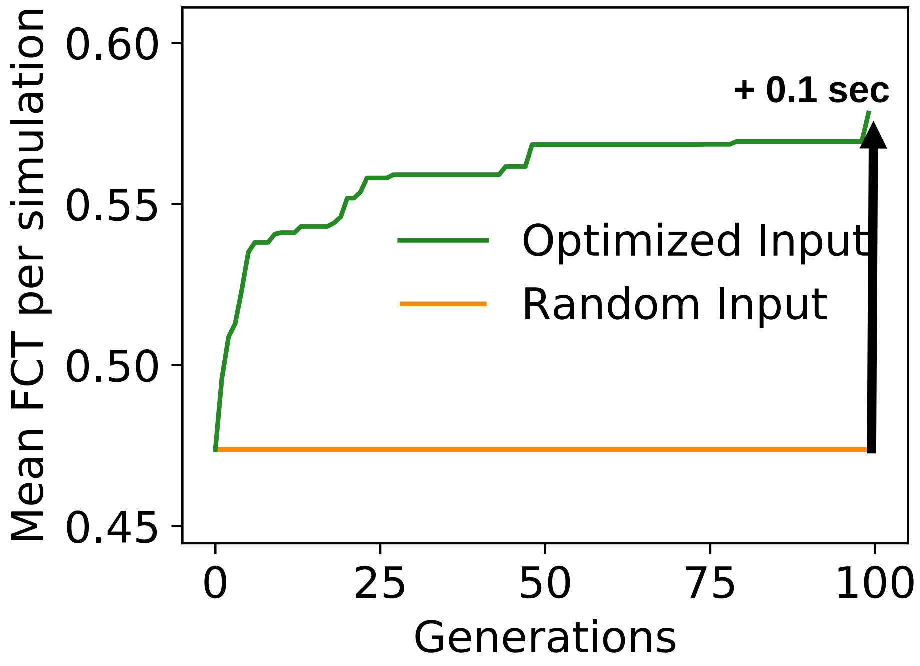


Population with N = 30 flows

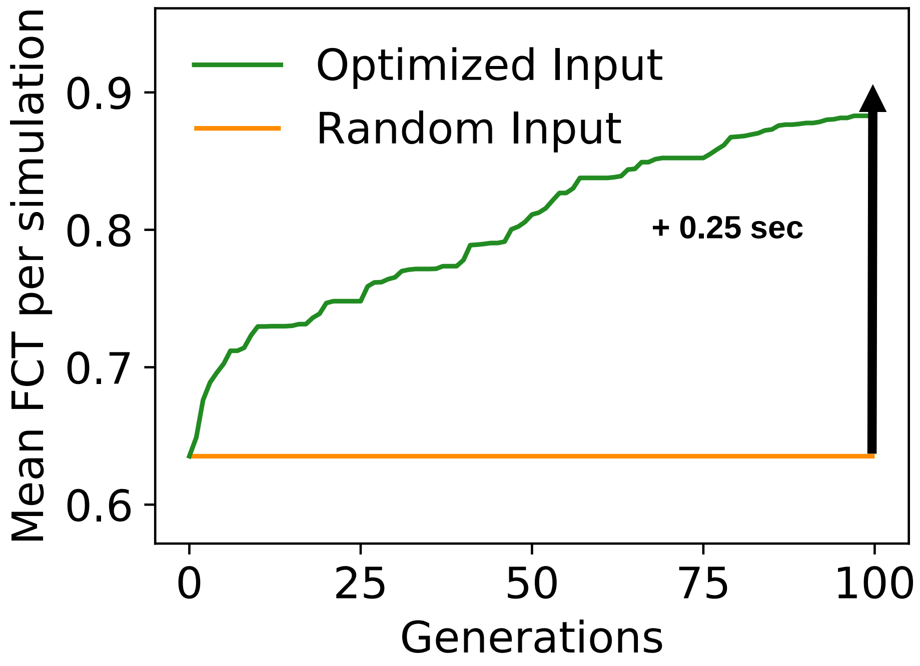


GA Behavior over Generations for Different Population Sizes

Population with N = 10 flows



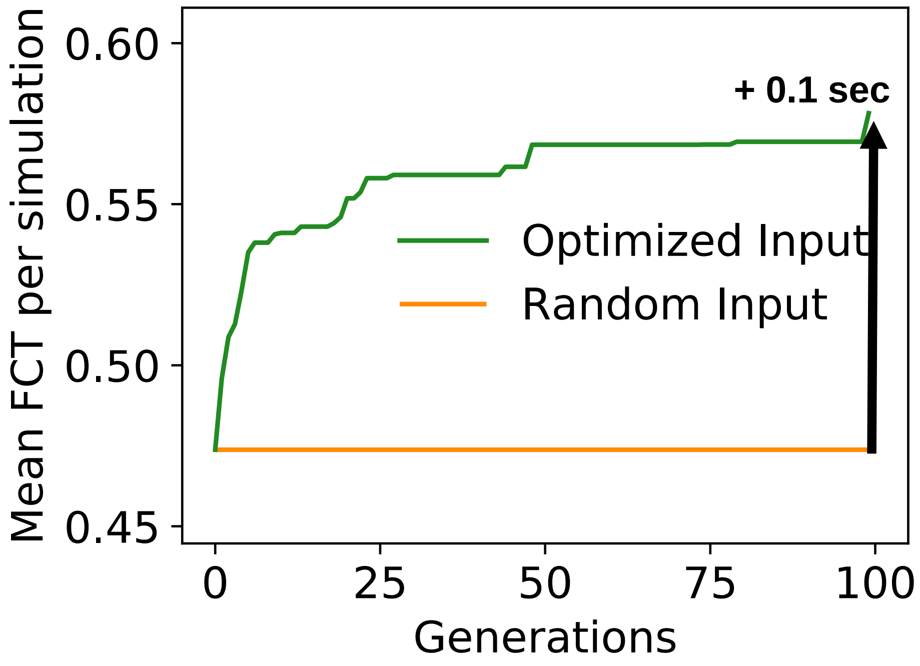
Population with N = 30 flows



GA Behavior over Generations for Different Population Sizes

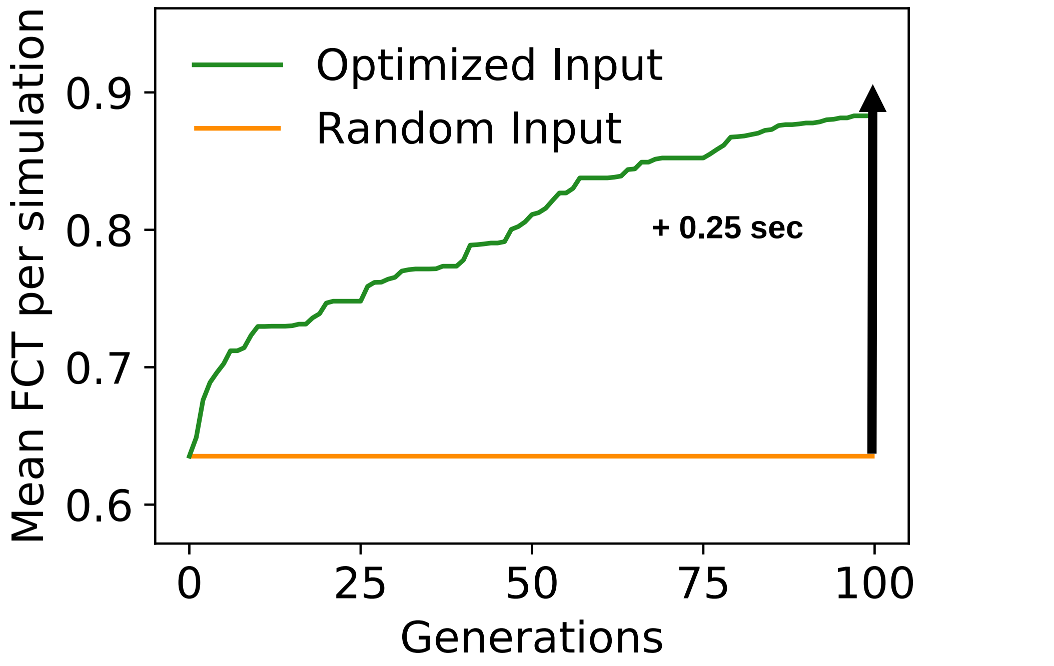
Population with N = 10 flows

17% more challenging



Population with N = 30 flows

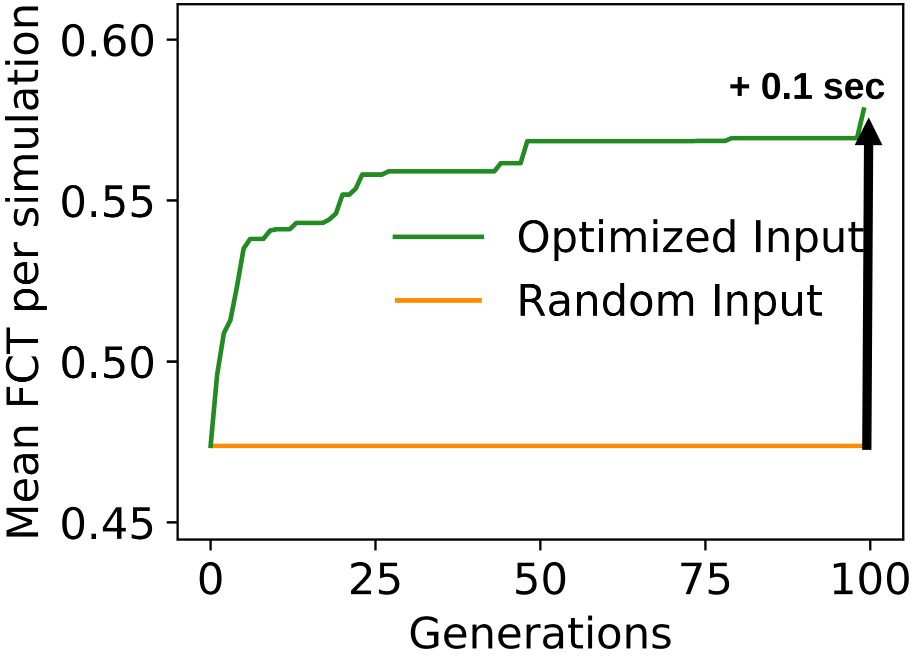
35% more challenging



GA Behavior over Generations for Different Population Sizes

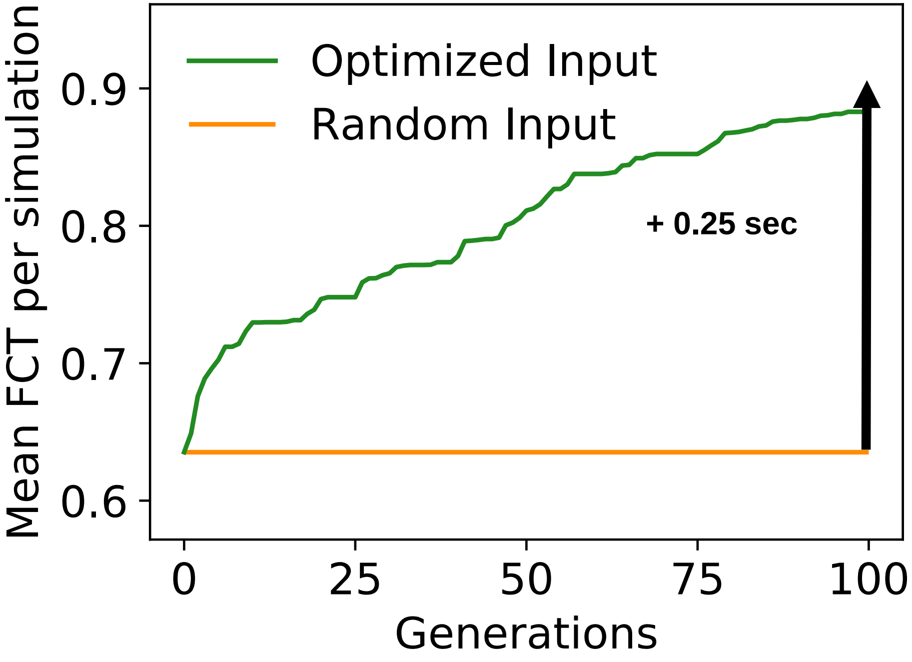
Population with N = 10 flows

17% more challenging



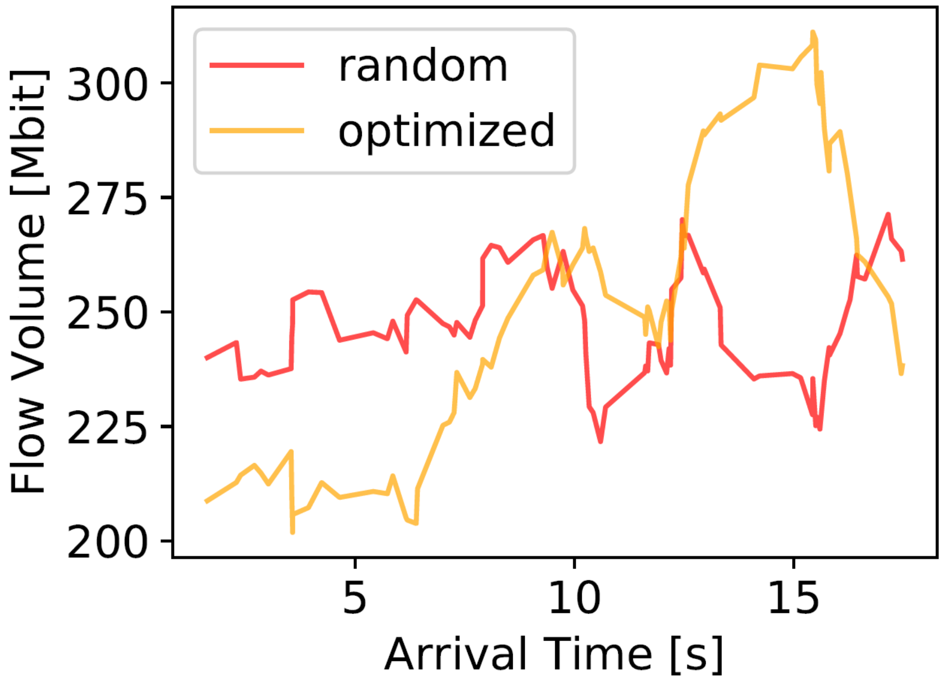
Population with N = 30 flows

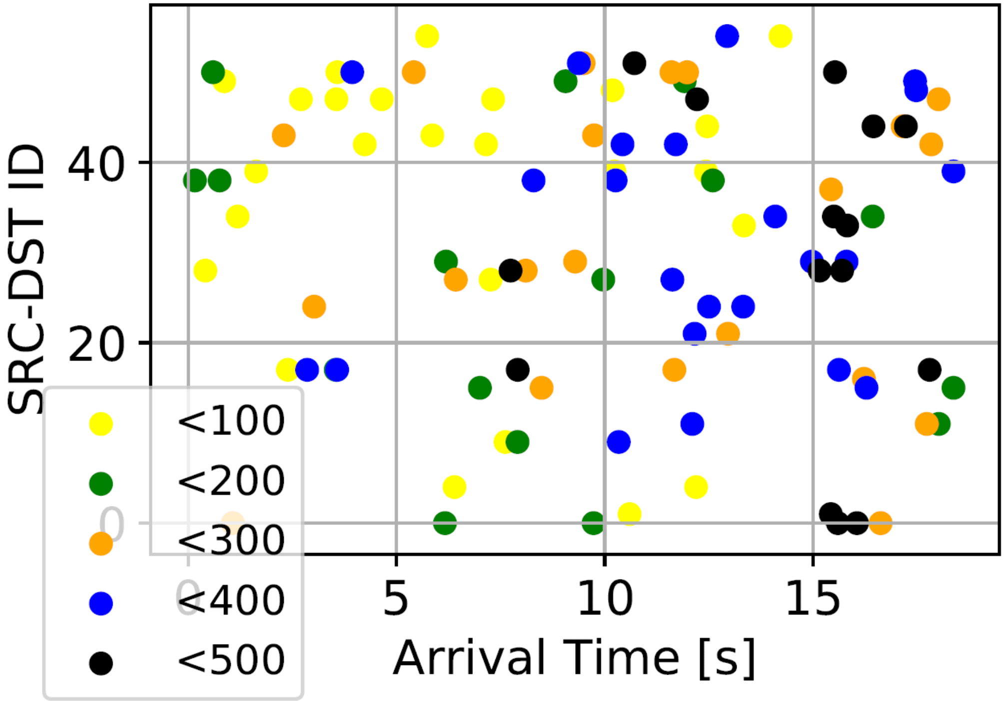
35% more challenging



More Flows → Higher margin of optimization

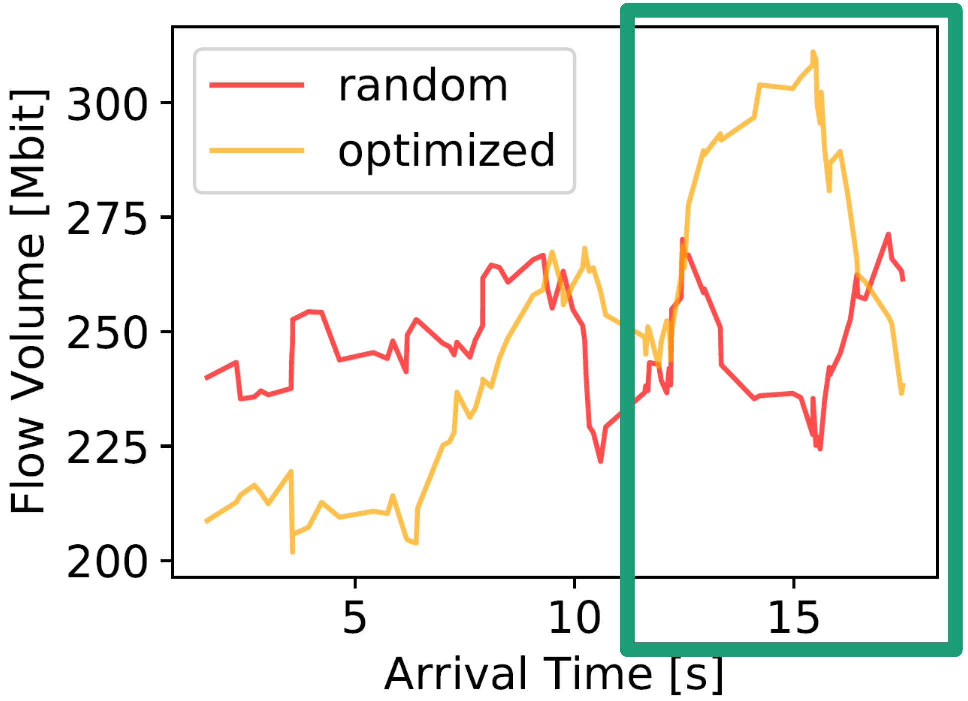
Flow Volume over Time (N=100)

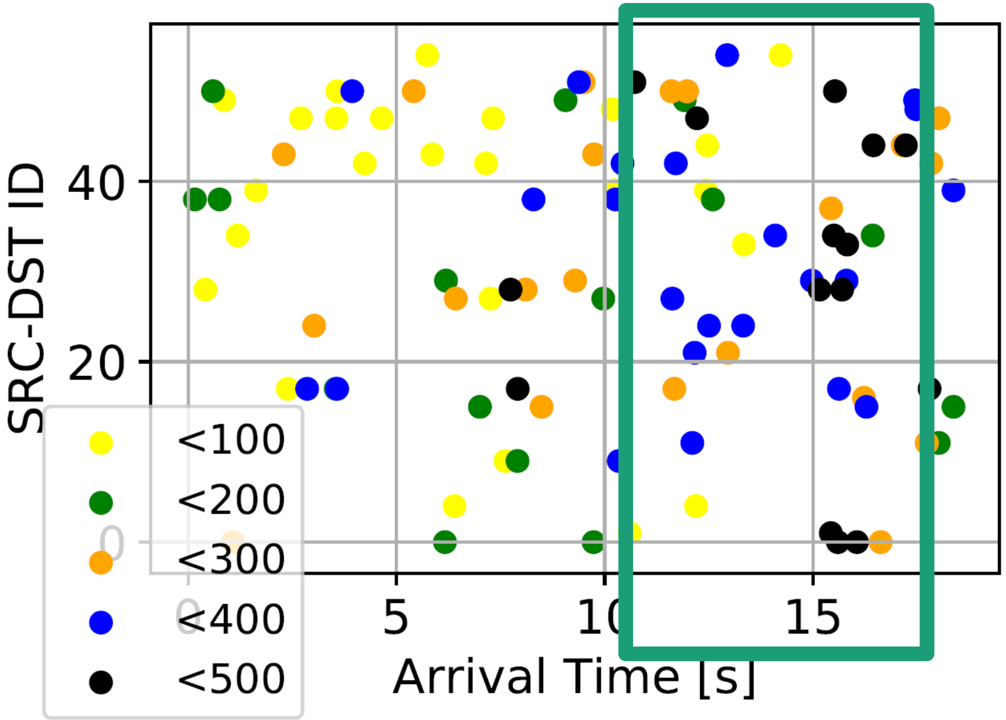




- Concentrate larger flows together
- Place large flows on the same link for close arrivals

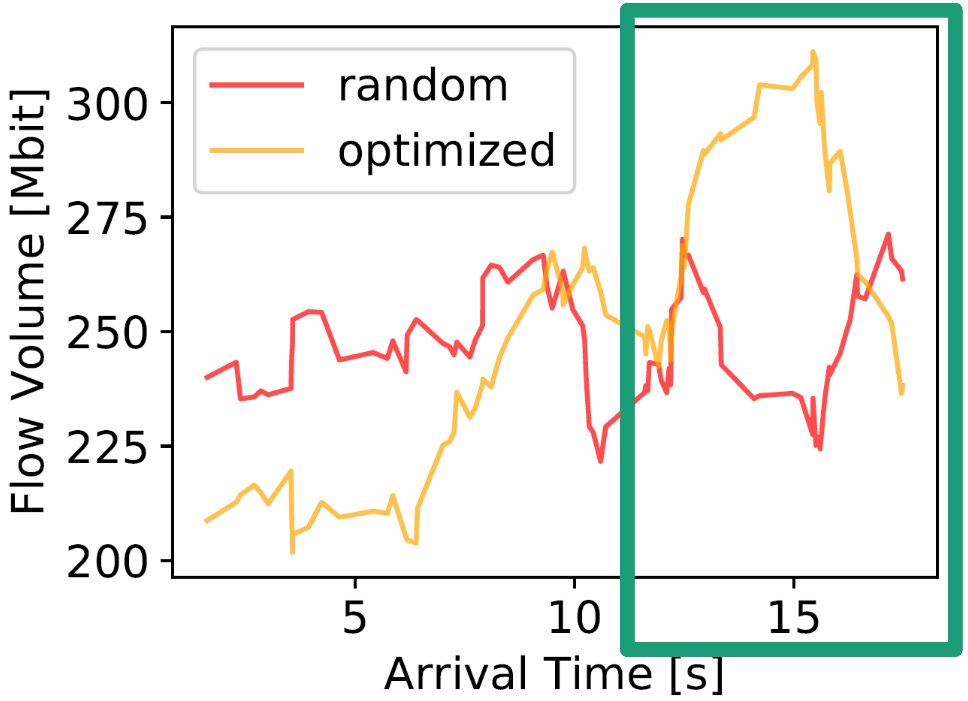
Flow Volume over Time (N=100)

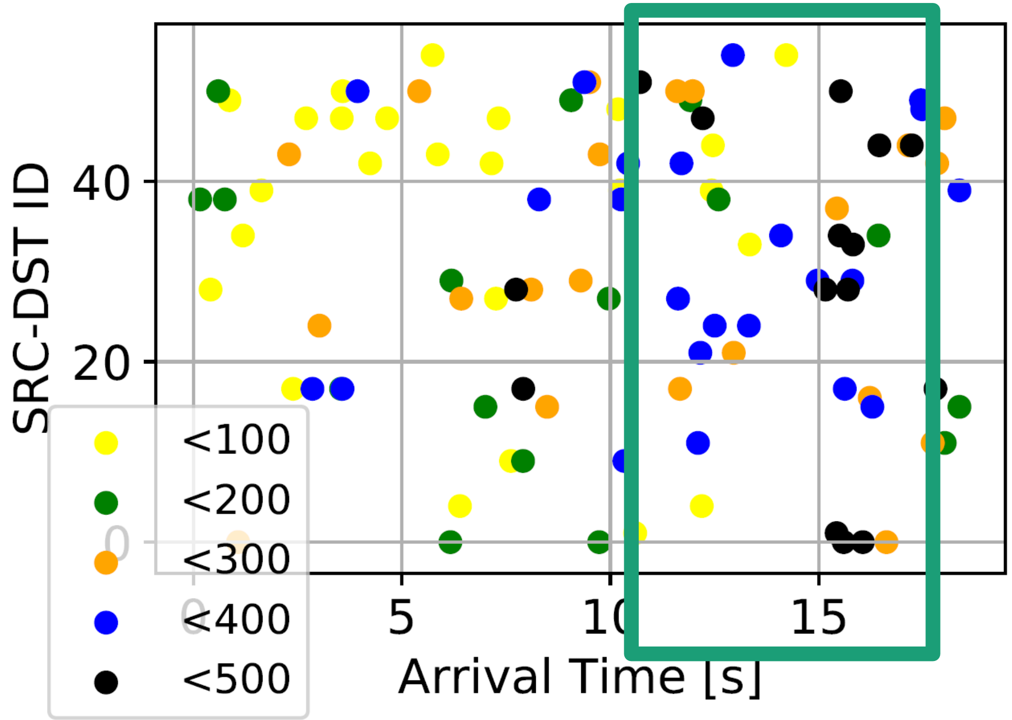




- Concentrate larger flows together
- Place large flows on the same link for close arrivals

Flow Volume over Time (N=100)

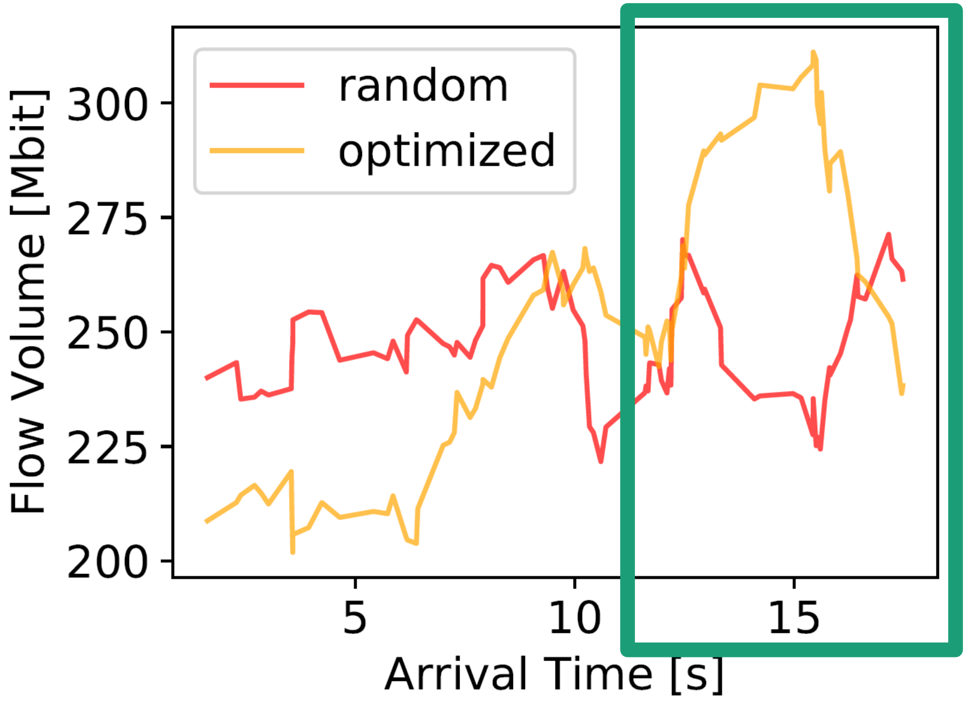


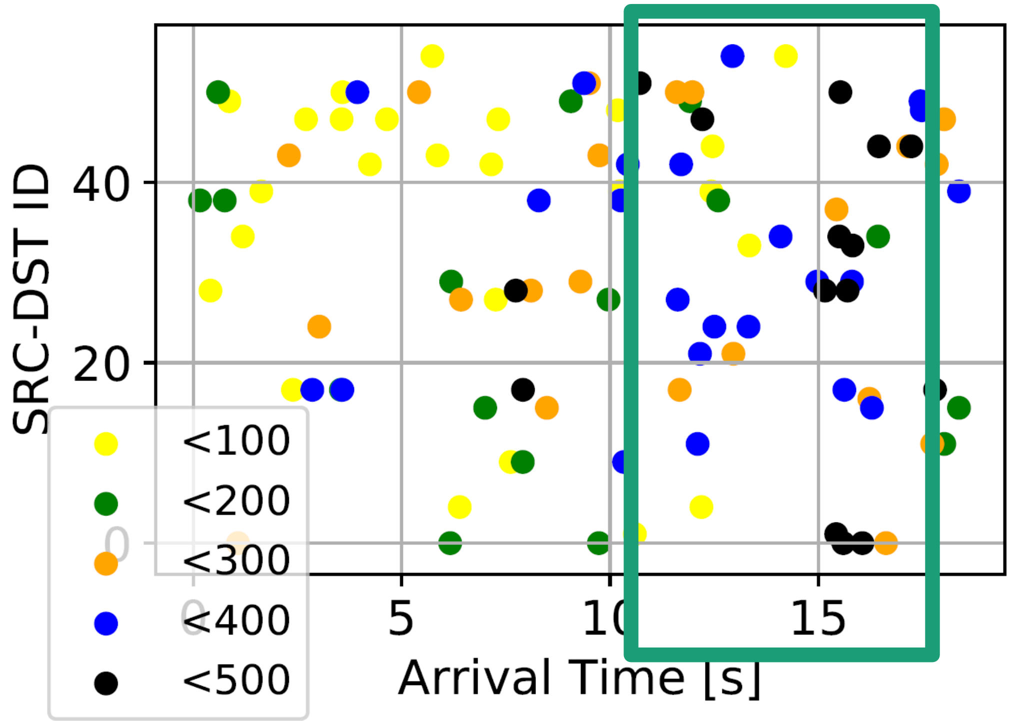


- Concentrate larger flows together
- Place large flows on the same link for close arrivals

BUT: Simulations consume a lot of time!

Flow Volume over Time (N=100)





- Concentrate larger flows together
- Place large flows on the same link for close arrivals

BUT: Simulations consume a lot of time!

Idea: Use Machine Learning in Genetic Algorithm [Bha13]

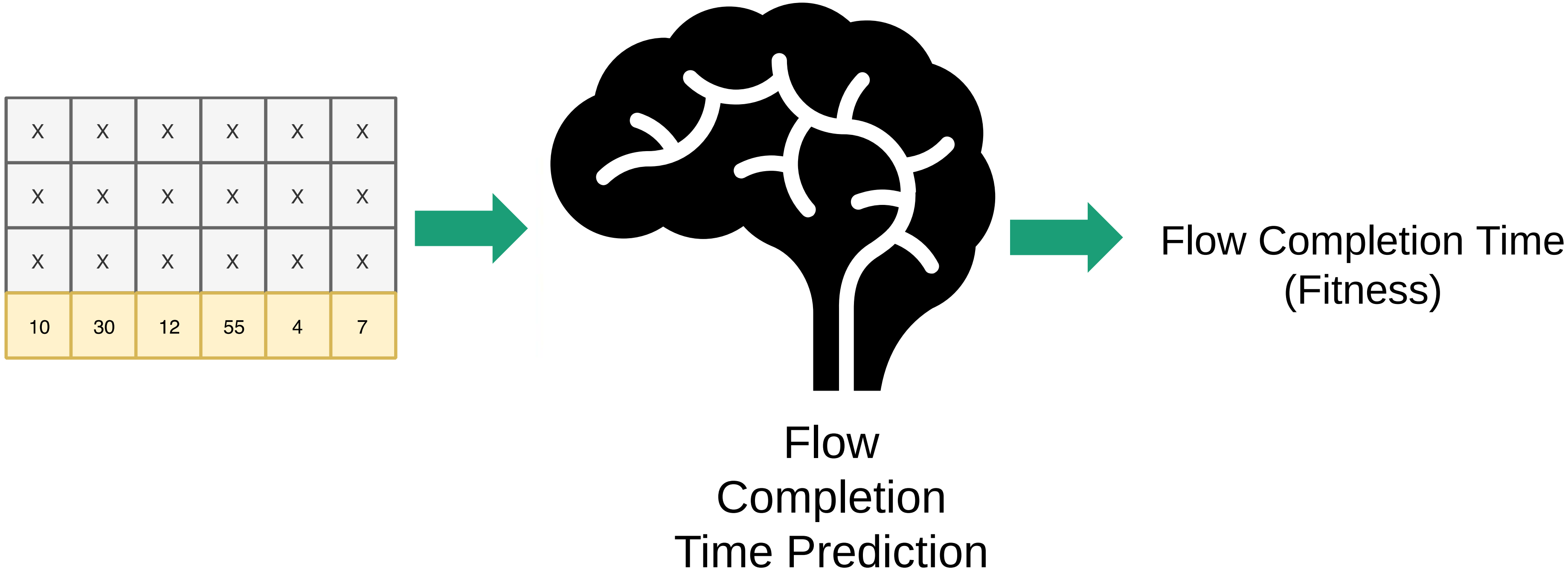
GA Acceleration – Deep Learning

X	X	X	X	X	X
X	X	X	X	X	X
X	X	X	X	X	X
10	30	12	55	4	7

Data Center Simulation

Flow Completion Time (Fitness)

GA Acceleration – Deep Learning



GA Acceleration – Deep Learning

X	X	X	X	X	X
X	X	X	X	X	X
X	X	X	X	X	X
10	30	12	55	4	7

Flow
Completion
Time Prediction

Flow Completion Time
(Fitness)

→ Needs to be evaluated very frequently
→ slow, does not scale

GA Acceleration – Deep Learning

X	X	X	X	X	X
X	X	X	X	X	X
X	X	X	X	X	X
10	30	12	55	4	7

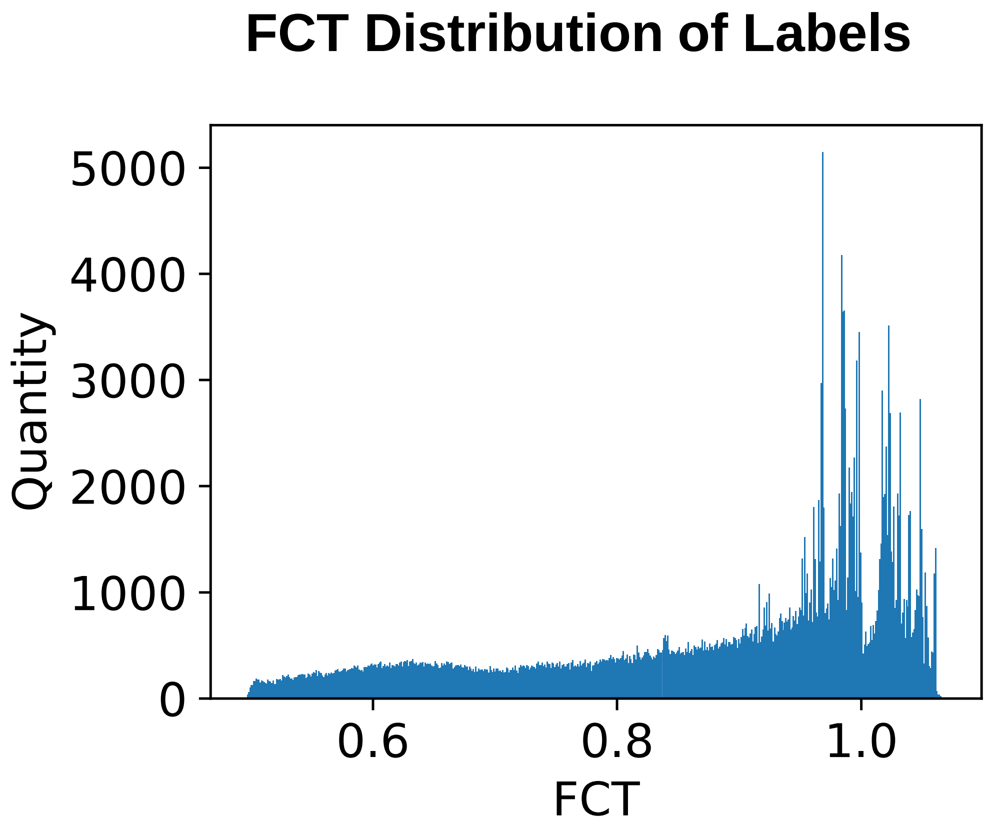
Flow Completion Time Prediction

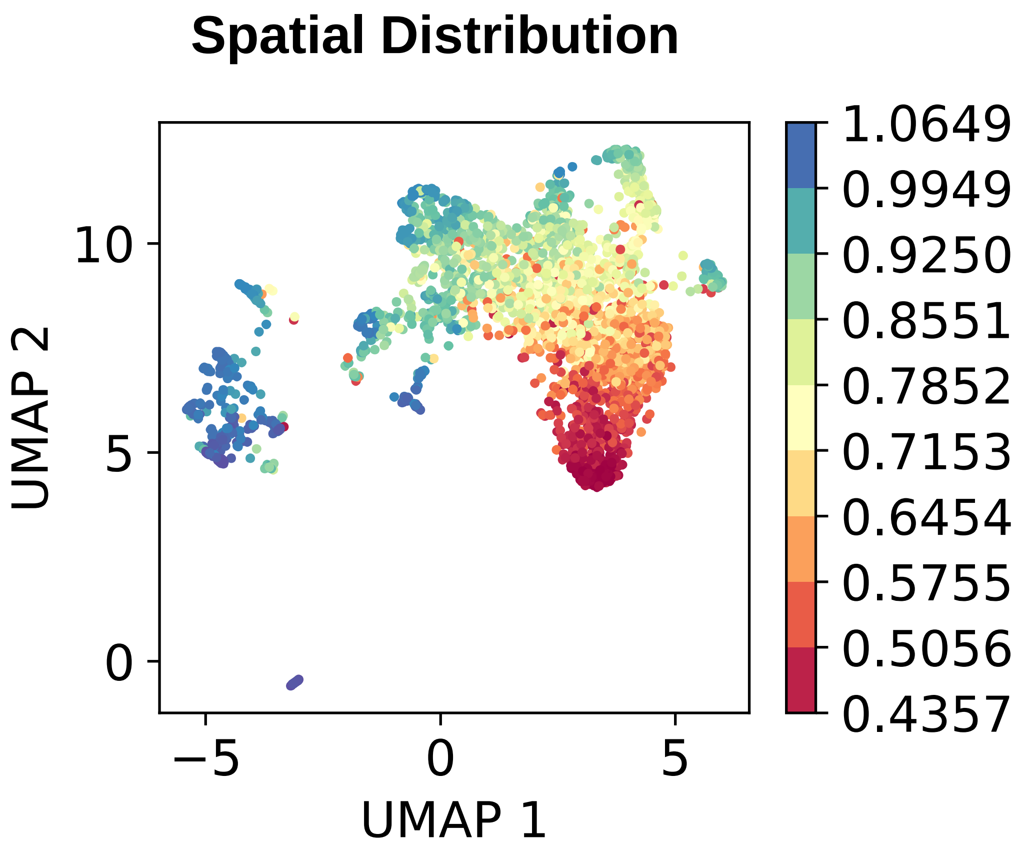
Flow Completion Time (Fitness)

→ Needs to be evaluated very frequently
→ slow, does not scale

Approximate Fitness Function with Deep Neural Network

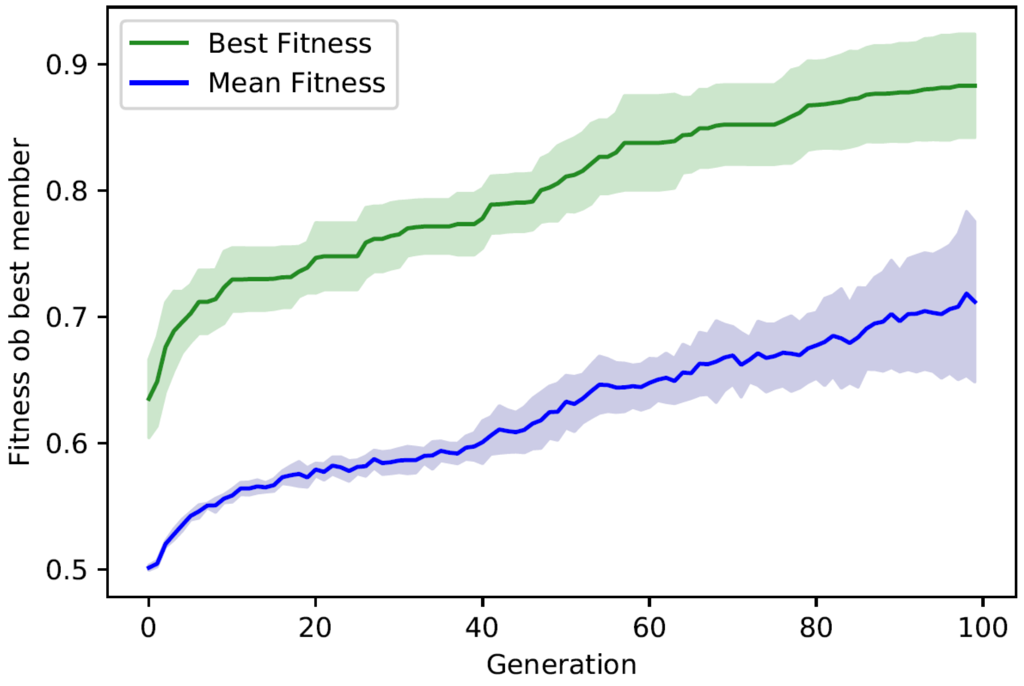
The Training Data

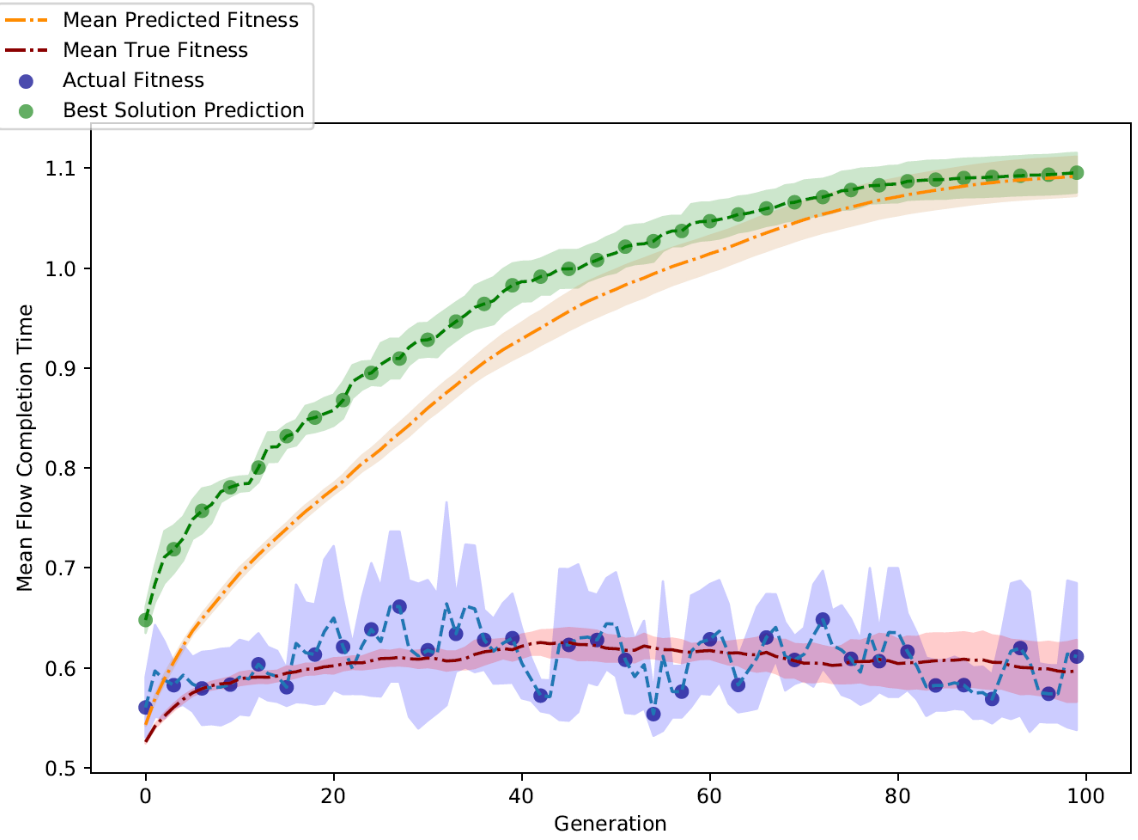




- Test Set Score: 87% of the samples achieved a relative error of less then 5%

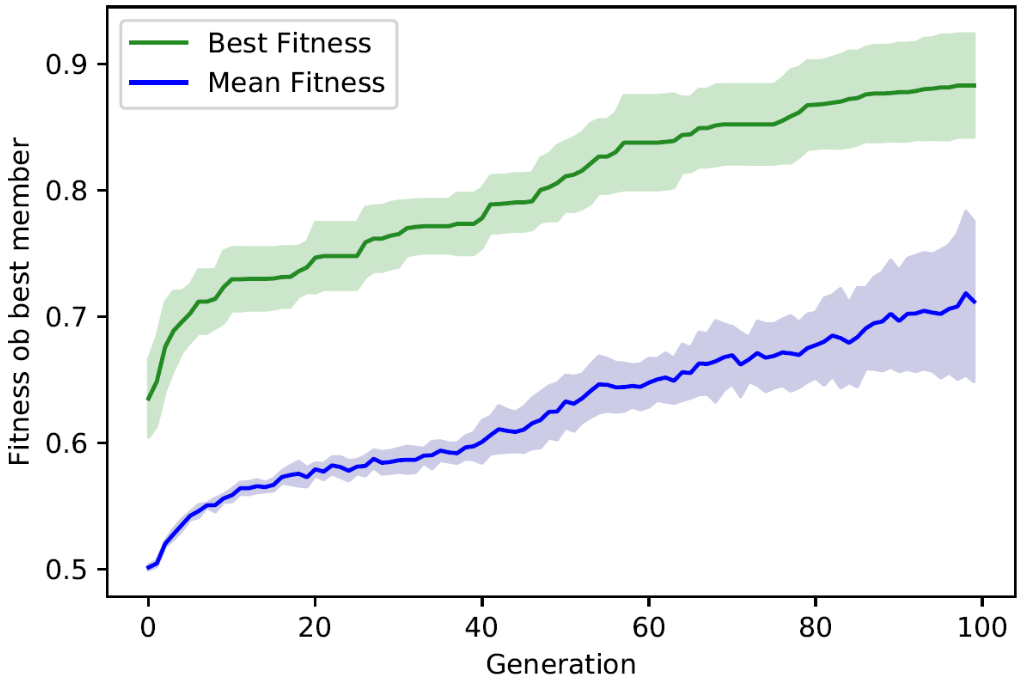
Comparison of Simulation vs Neural Network

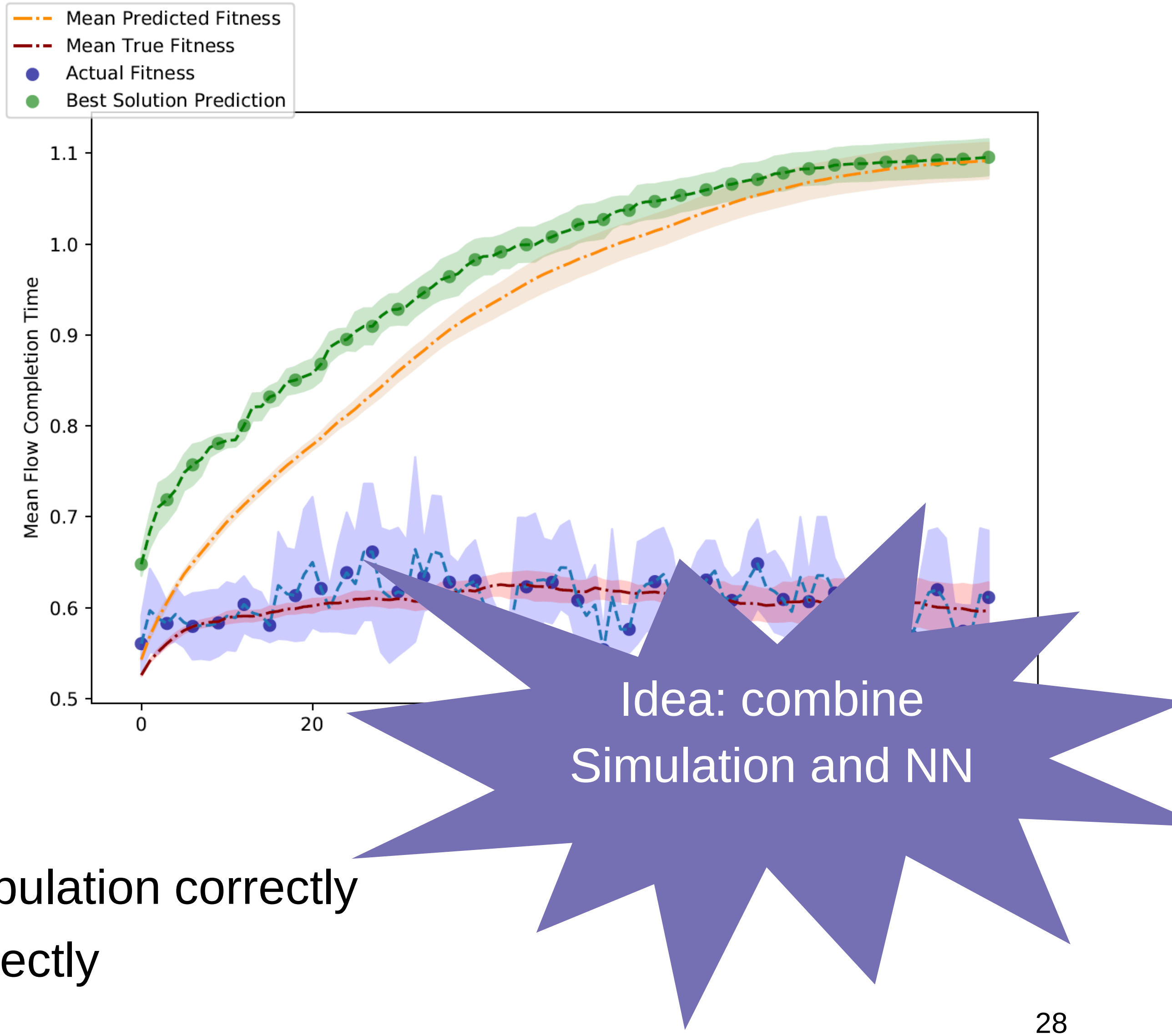




- Neural network cannot predict the fitness of population correctly
- Best population members cannot be found correctly

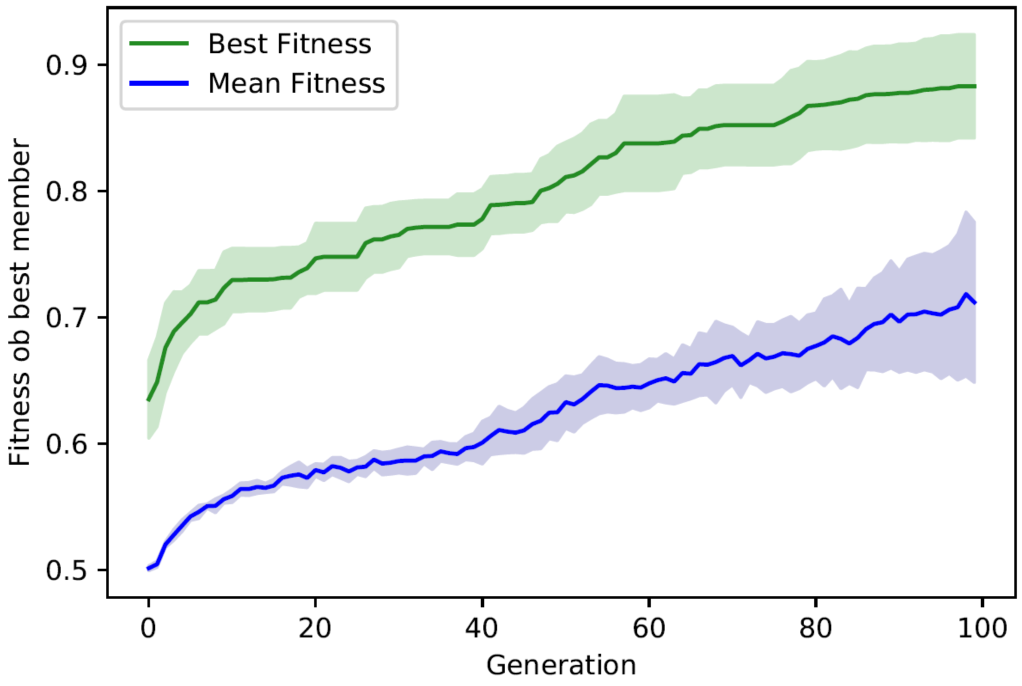
Comparison of Simulation vs Neural Network

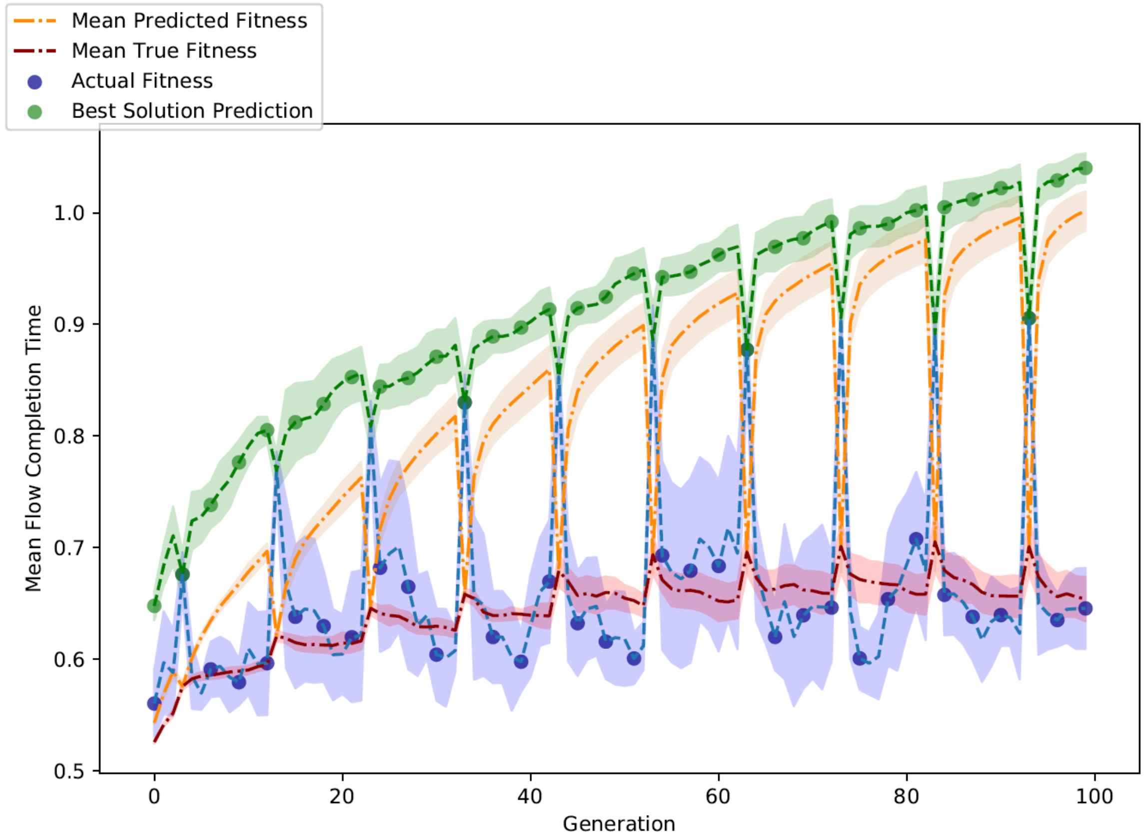




- Neural network cannot predict the fitness of population correctly
- Best population members cannot be found correctly

Comparison of Simulation vs Simulation-enhanced Neural Network Approach

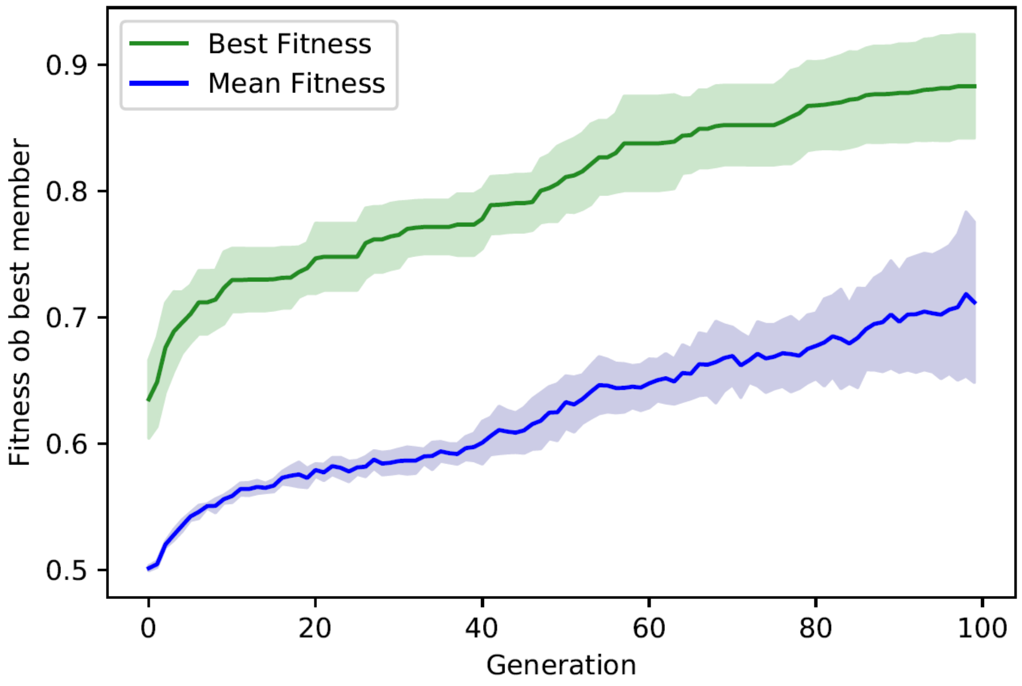


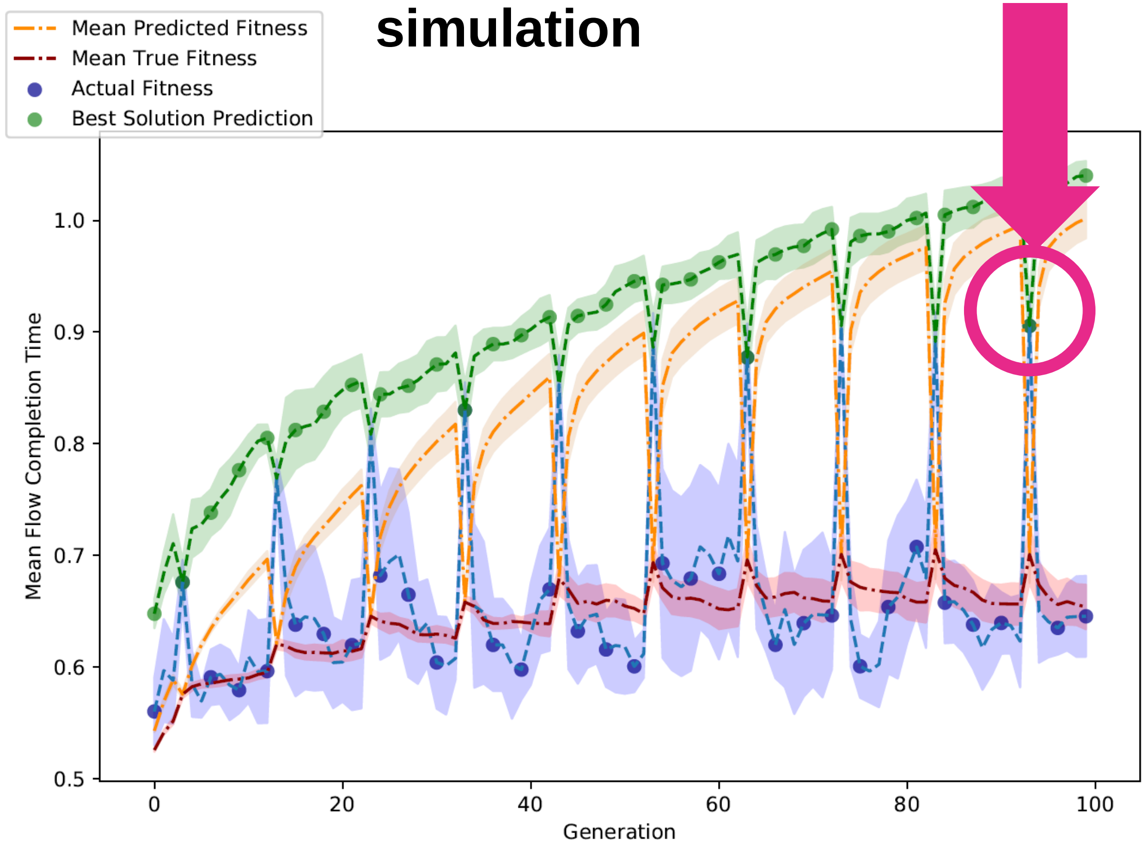


- Simulations can be used to determine current best simulation members
- More than one simulation needed to improve population

Comparison of Simulation vs Simulation-enhanced Neural Network Approach

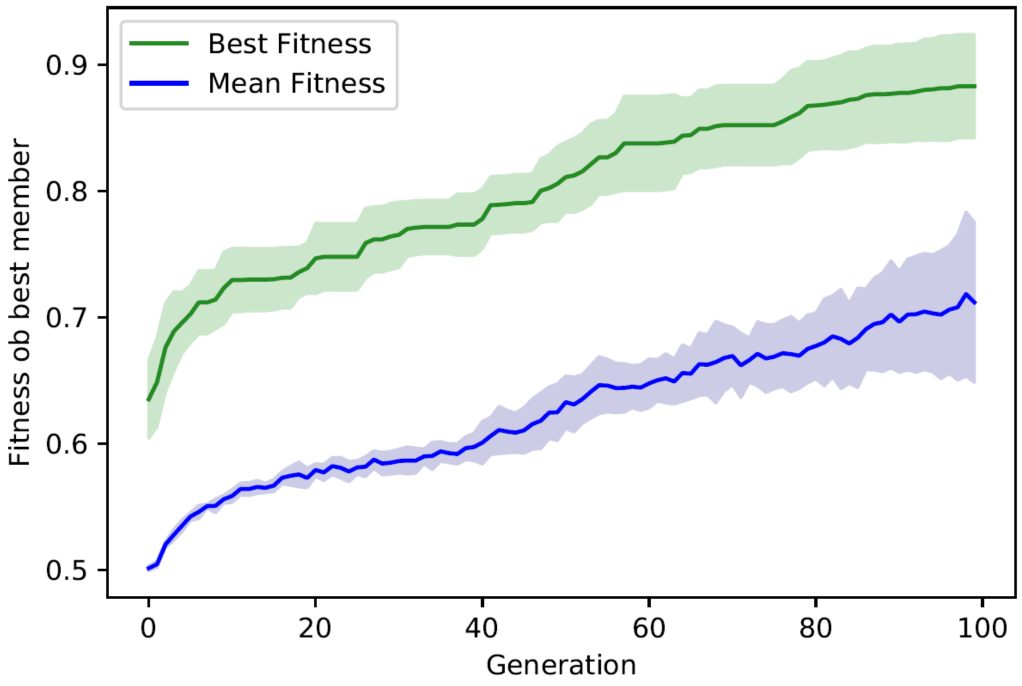
Performance even better than simulation



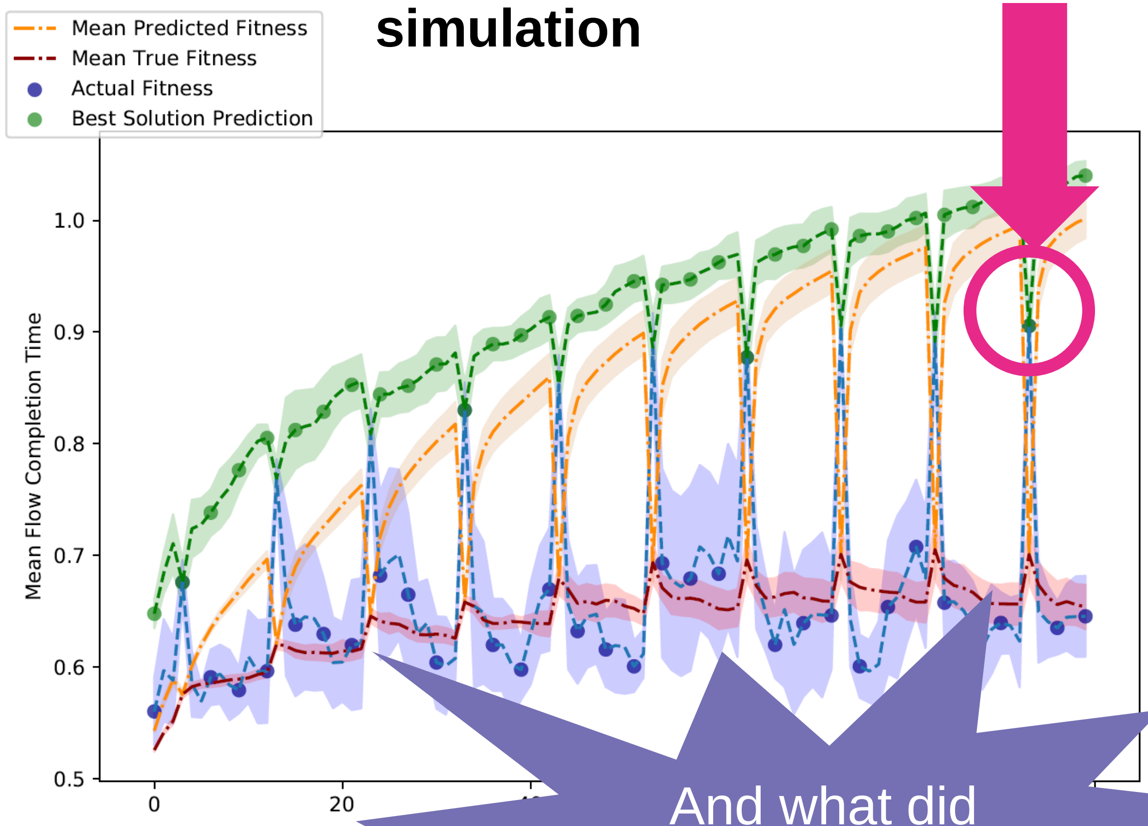


- Simulations can be used to determine current best simulation members
- More than one simulation needed to improve population

Comparison of Simulation vs Simulation-enhanced Neural Network Approach



Performance even better than simulation



And what did we save?

- Simulations can be used to determine current best simulation members
- More than one simulation needed to improve population

Runtime Comparison (10 runs, CI 95%)

Mixed 4 times slower than NeuralNet

Population	10	30	50	100
Simulation	106 (7.1)	369 (7.7)	638 (24.2)	1187 (17.9)
NeuralNet	23 (1.8)	29 (1.5)	32 (1.6)	43 (1.1)
Mixed	32 (0.9)	73 (1.0)	118 (2.6)	210 (3.1)

… but Mixed 4 times faster than Simulation

Neural Net < Mixed < Simulation

Part 2: Conclusion

- Genetic Algorithm can automate adversary Traffic Generation

→Automated Benchmarking

- Neural Network can significantly accelerate Genetic Algorithms

→Scalability

Limitations:

- Long training time of GA
- Accelerator trades-off solution quality and compute time
- Accelerator needs to be re-trained when fitness function changes

Potentials and Future Work:

- Utilize current network state (e.g., demand matrix)
- Make a prediction for the next arrival(s) – e.g., investigate existing network traces

Thank you!

Questions?

References

[BIG DAMA’17] Blenk, Andreas; Kalmbach, Patrick; Schmid, Stefan; Kellerer, Wolfgang: o'zapft is: Tap Your Network Algorithm's Big Data! ACM SIGCOMM 2017 Workshop on Big Data Analytics and Machine Learning for Data Communication Networks (Big-DAMA), 2017

[SelfDN’18] Kalmbach, Patrick; Zerwas, Johannes; Babarczi, Péter; Blenk, Andreas; Kellerer, Wolfgang; Schmid, Stefan: Empowering Self-Driving Networks. Proceedings of the Afternoon Workshop on Self-Driving Networks - SelfDN 2018, ACM Press, 2018

[NetAI’19] Zerwas, Johannes; Kalmbach, Patrick; Henkel, Laurenz; Retvari, Gabor; Kellerer, Wolfgang; Blenk, Andreas; Schmid, Stefan: NetBOA: Self-Driving Network Benchmarking. ACM SIGCOMM 2019 Workshop on Network Meets AI & ML (NetAI '19), 2019

[CoNEXT Com‘19] Lettner, Sebastian; Blenk, Andreas: Adversarial Network Algorithm Benchmarking. The 15th International Conference on emerging Networking EXperiments and Technologies (CoNEXT ’19 Companion), ACM, 2019

[TNSM‘19] Zerwas, Johannes; Kalmbach, Patrick; Schmid, Stefan; Blenk, Andreas: Ismael: Using Machine Learning To Predict Acceptance of Virtual Clusters in Data Centers. IEEE Transactions on Network and Service Management, 2019

[Bha13] Maumita Bhattacharya. 2013. Evolutionary Approaches to Expensive Optimisation. Arxiv - Computers & Society 2, 3 (2013), 53–59. DOI:http://dx.doi.org/10.14569/IJARAI.2013.020308