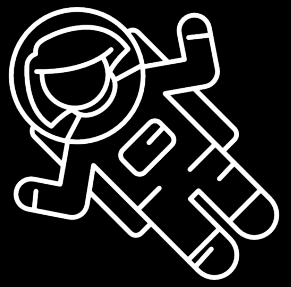
ACM/IEEE CODES+ISSS 2017, Seoul, South Korea



COSMOS: Coordination of High-Level Synthesis and Memory Optimization for Hardware Accelerators

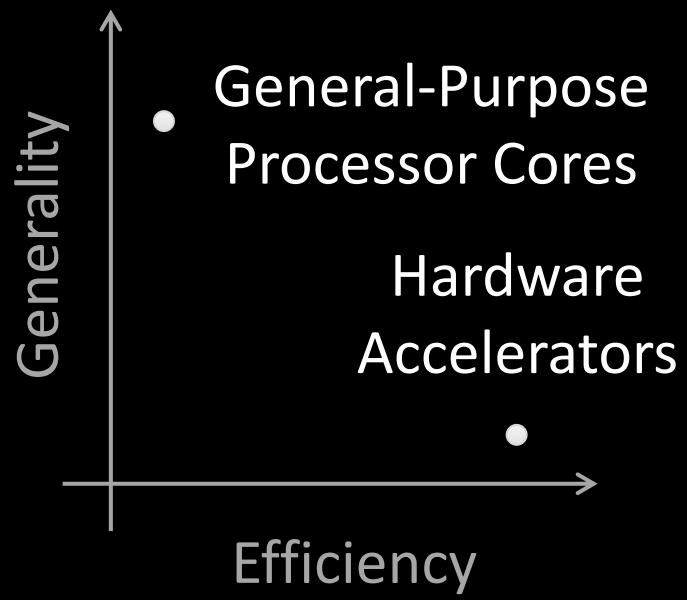
Luca Piccolboni, Paolo Mantovani,
Giuseppe Di Guglielmo, Luca Carloni
Columbia University, New York, USA

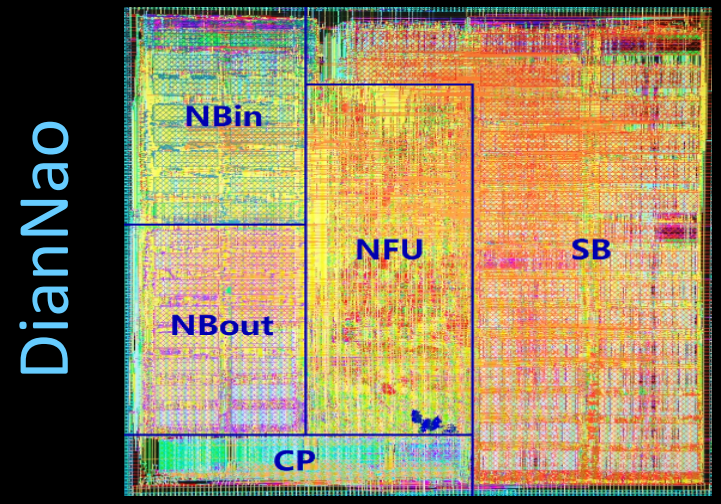
COLUMBIA UNIVERSITY

Hardware Accelerators

Motivations

- Hardware accelerators are devices designed and optimized to realize very **specific functionalities**

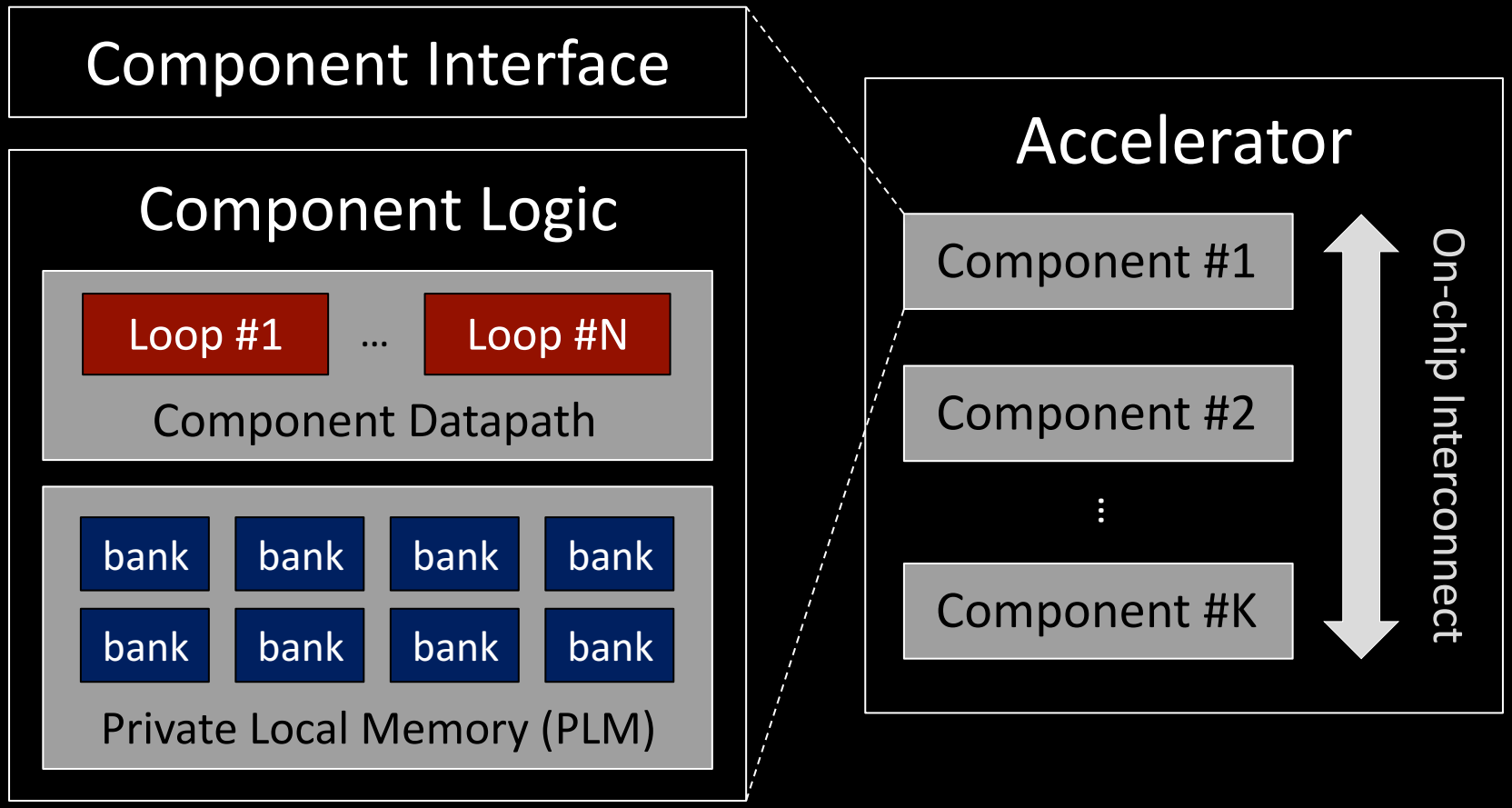




[T. Chen et al., ASPLOS'14]

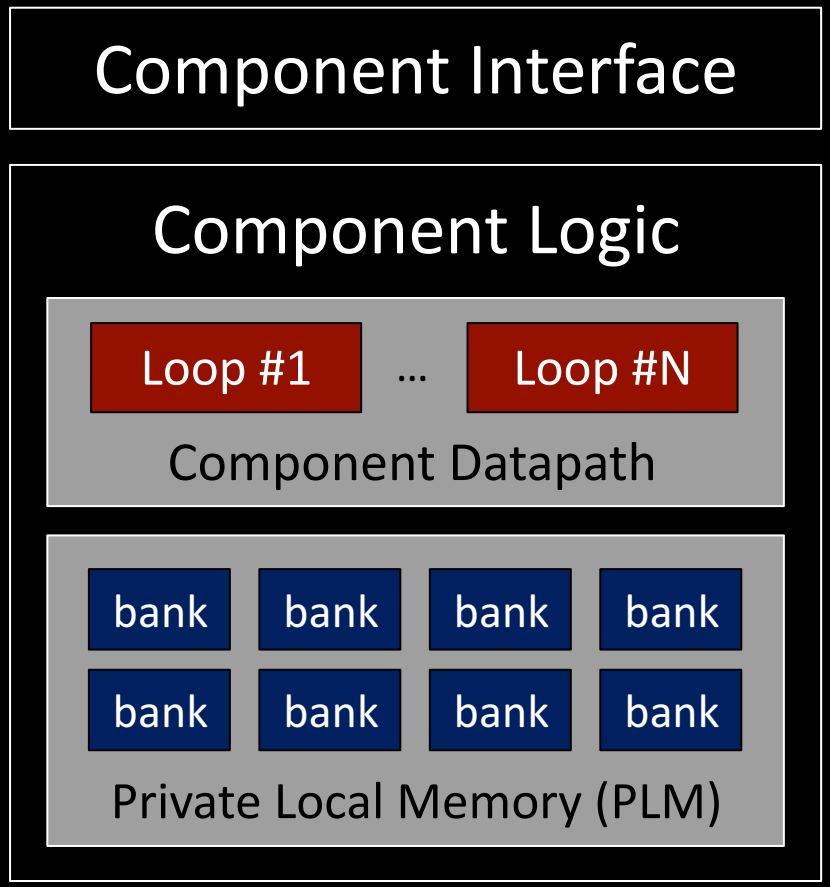
Hardware Accelerators

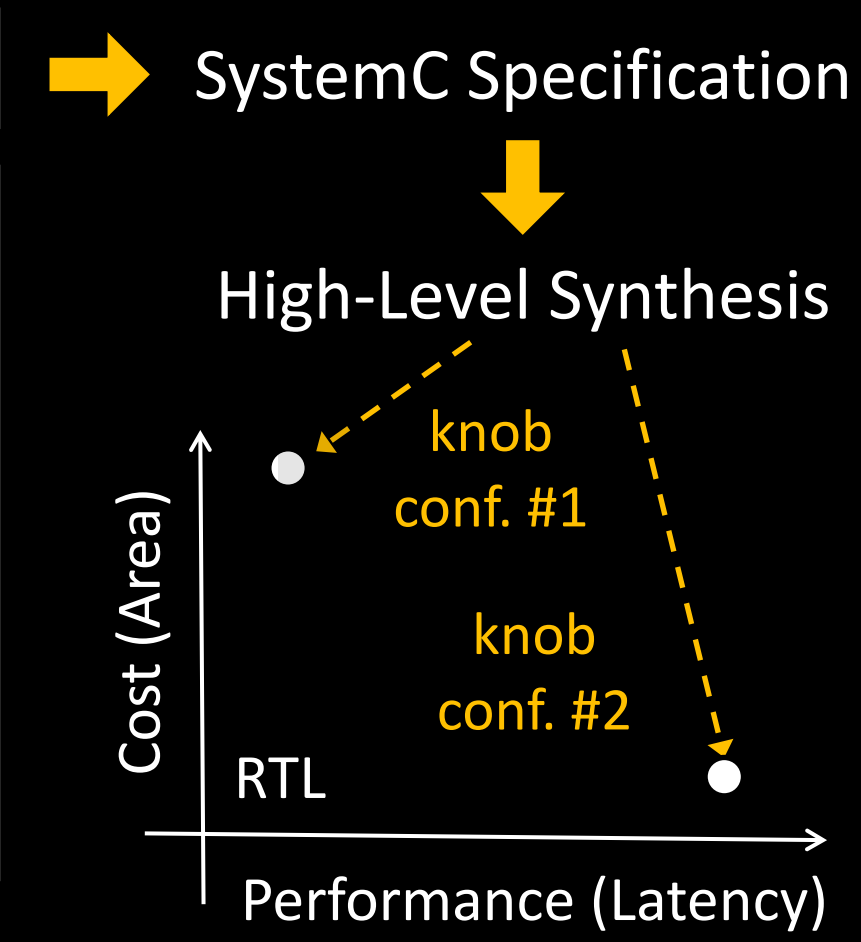
Architecture



Hardware Accelerators

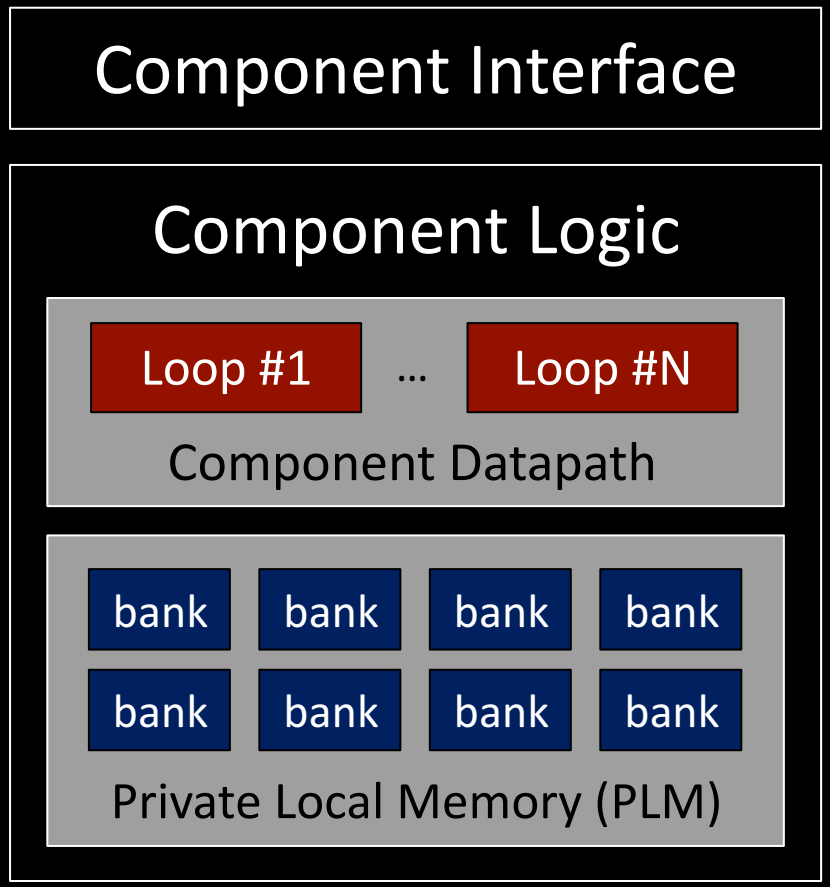
High-Level Synthesis (HLS)





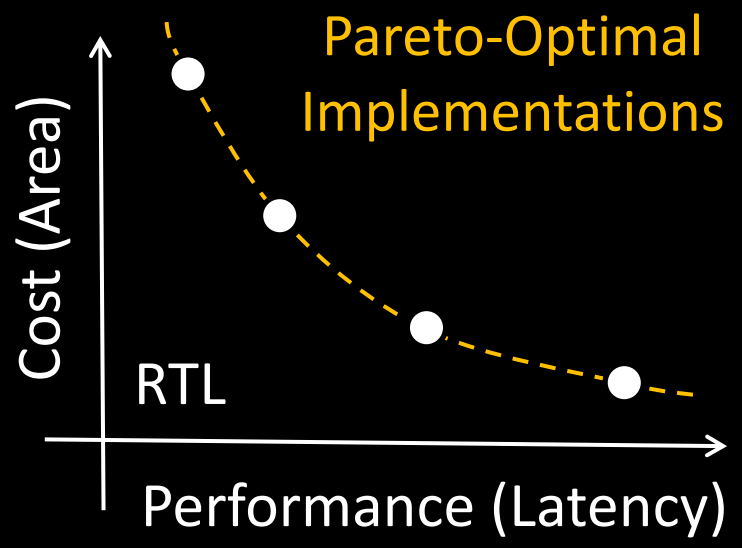
Hardware Accelerators

High-Level Synthesis (HLS)



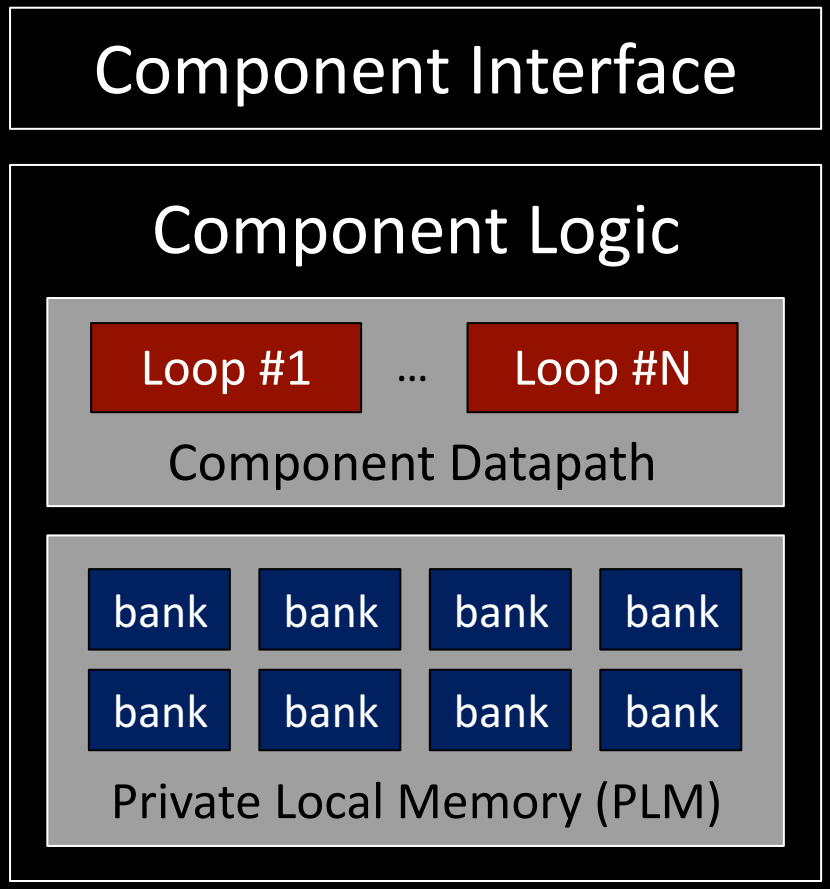
SystemC Specification

High-Level Synthesis



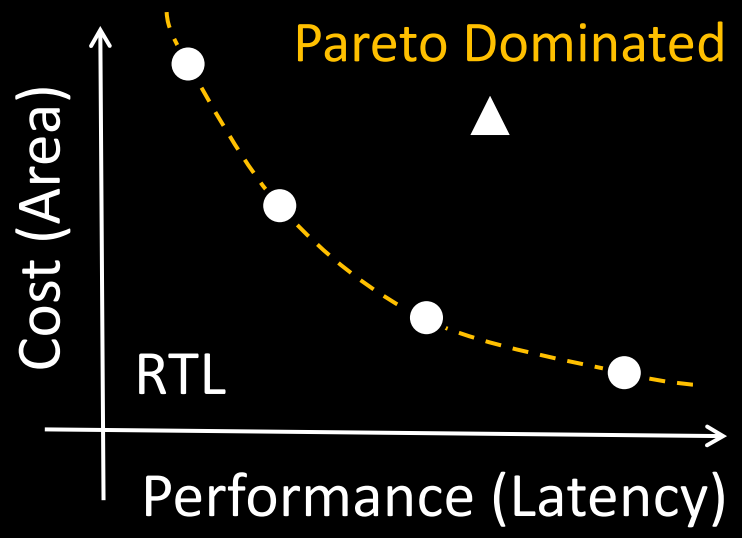
Hardware Accelerators

High-Level Synthesis (HLS)



SystemC Specification

High-Level Synthesis

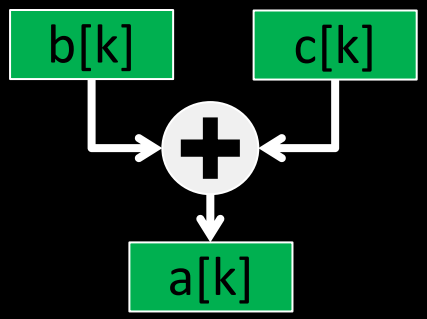


Hardware Accelerators

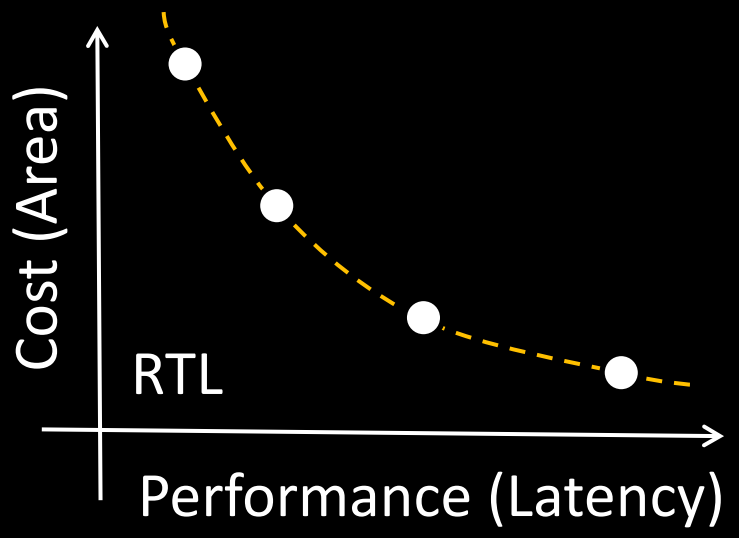
High-Level Synthesis (HLS)

1. Loop unrolling

```
for (k = 0; k < N; ++k)
  a[k] = b[k] + c[k];
```



Which knobs can be used to obtain several RTL implementations?

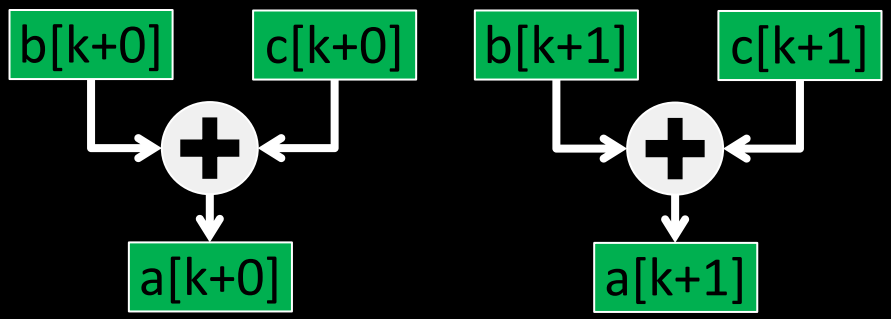


Hardware Accelerators

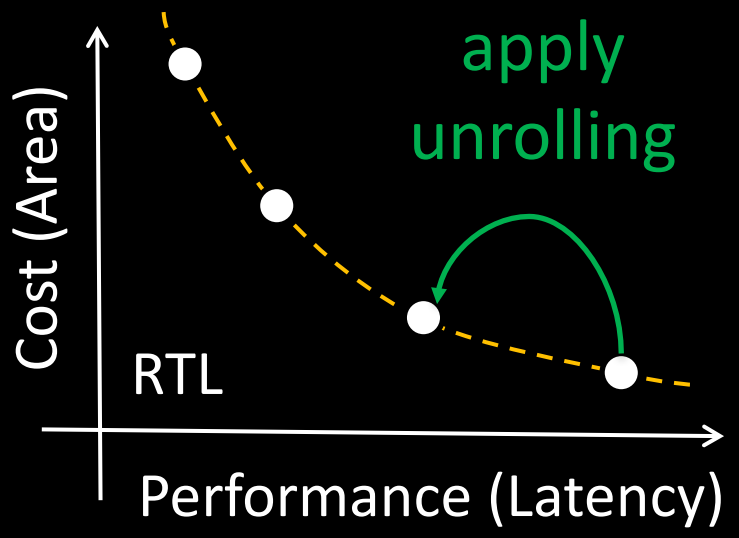
High-Level Synthesis (HLS)

1. Loop unrolling

```
for (k = 0; k < N; k += 2)
  a[k+0] = b[k+0] + c[k+0];
  a[k+1] = b[k+1] + c[k+1];
```



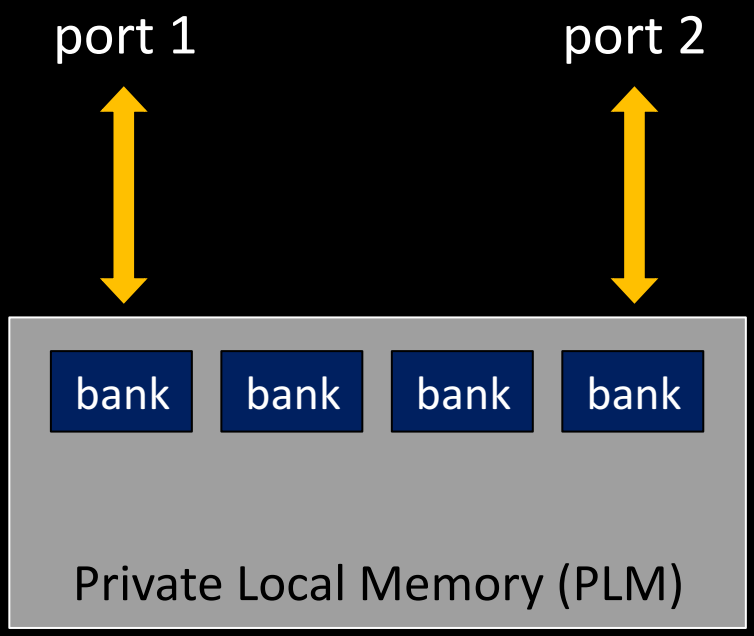
Which knobs can be used to obtain several RTL implementations?



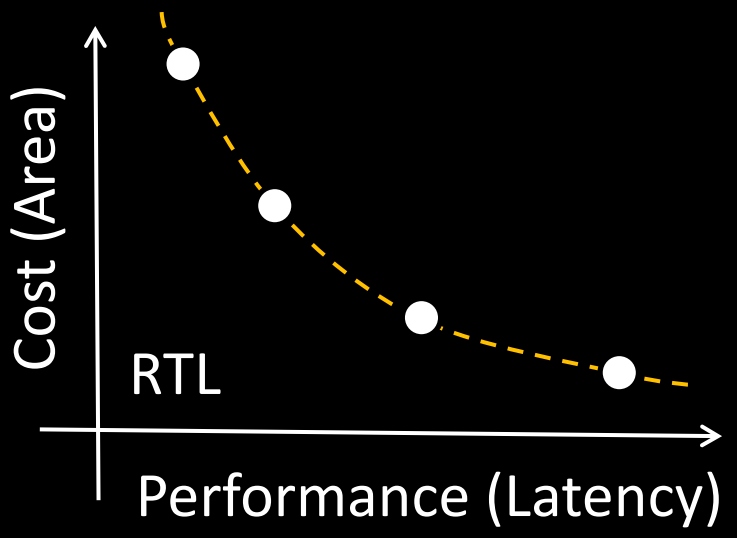
Hardware Accelerators

High-Level Synthesis (HLS)

2. Memory Ports



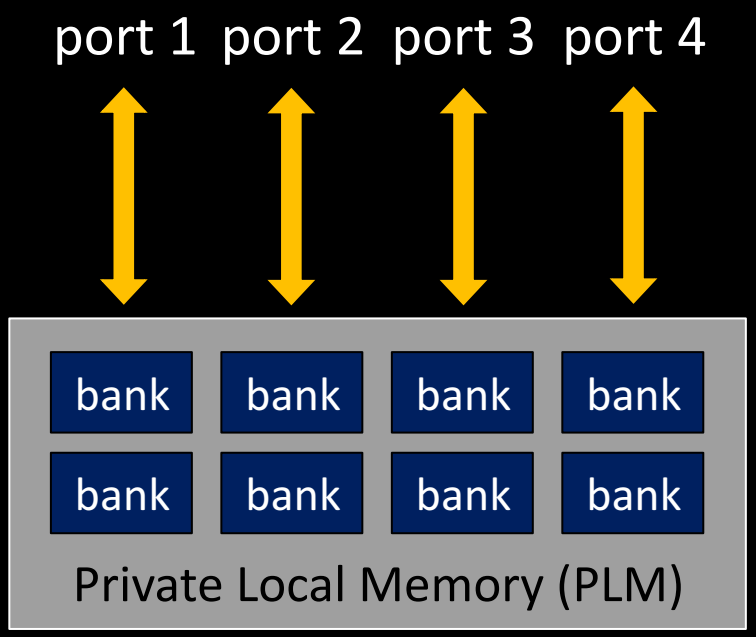
Which knobs can be used to obtain several RTL implementations?



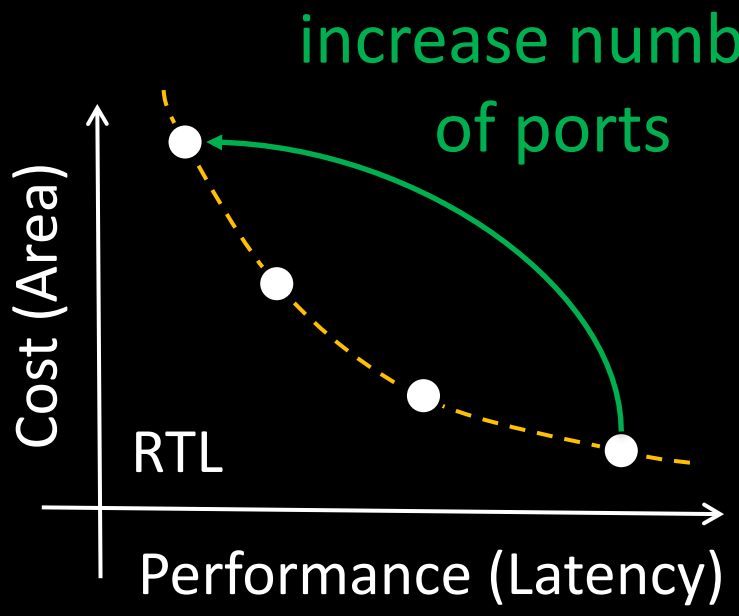
Hardware Accelerators

High-Level Synthesis (HLS)

2. Memory Ports



Which knobs can be used to obtain several RTL implementations?



Motivational Examples

- Performing an accurate and exhaustive design-space exploration for a hardware accelerator is complex:

Motivational Examples

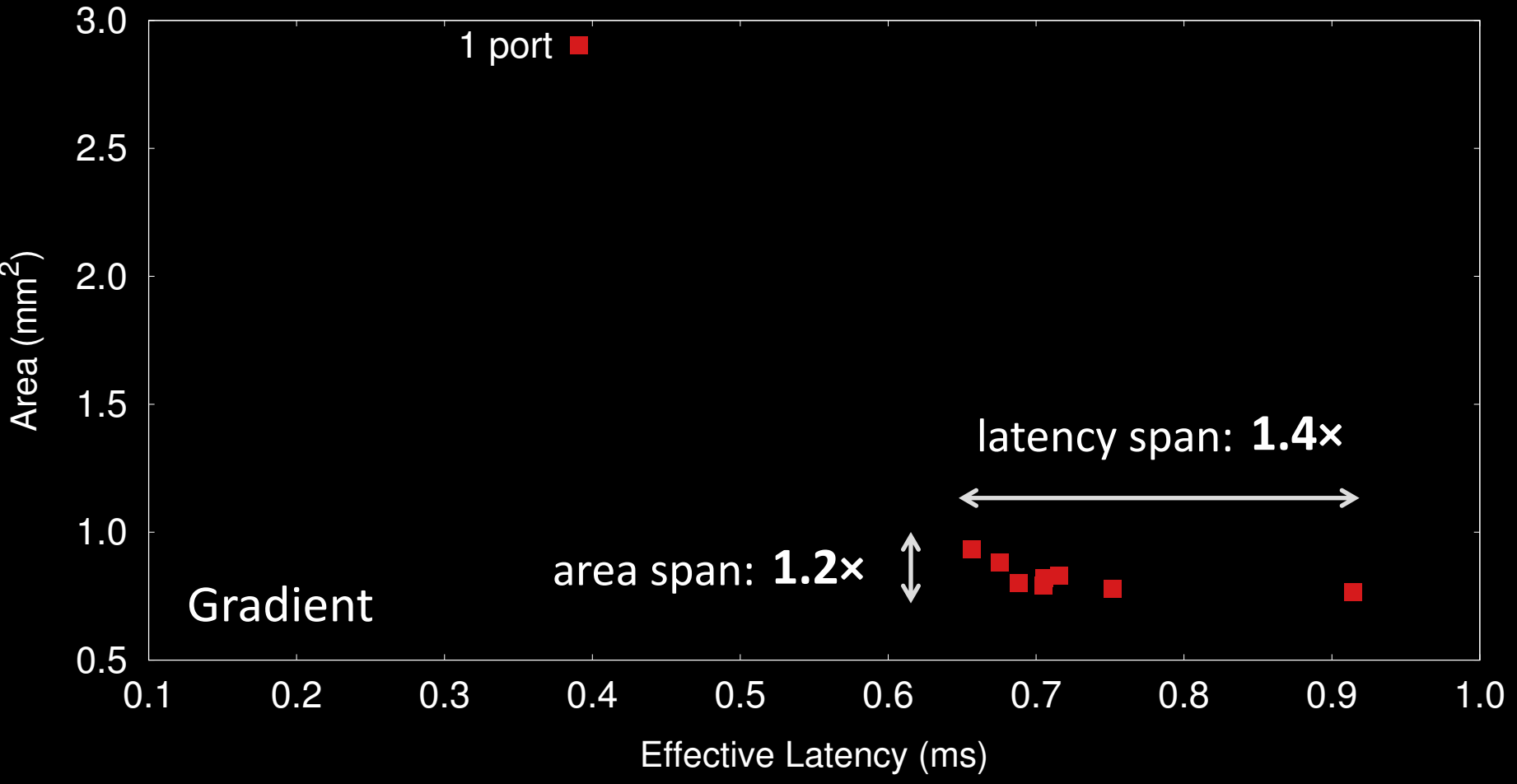
- Performing an accurate and exhaustive design-space exploration for a hardware accelerator is complex:

  1. HLS tools do not always support the generation (and optimization) of the private local memories

Motivational Examples

Need of multi-port memories

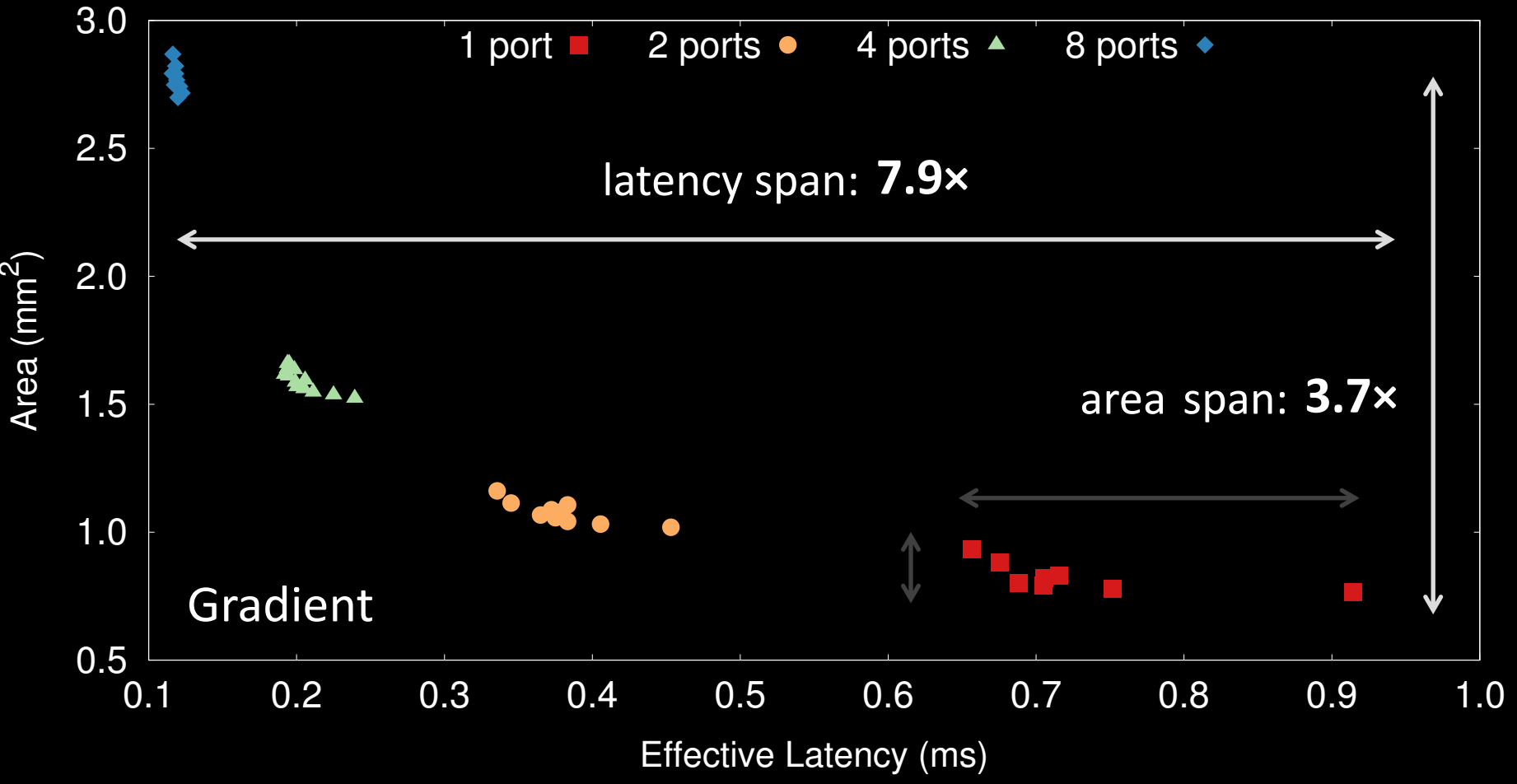
using standard memories



Motivational Examples

Need of multi-port memories

using multi-port memories



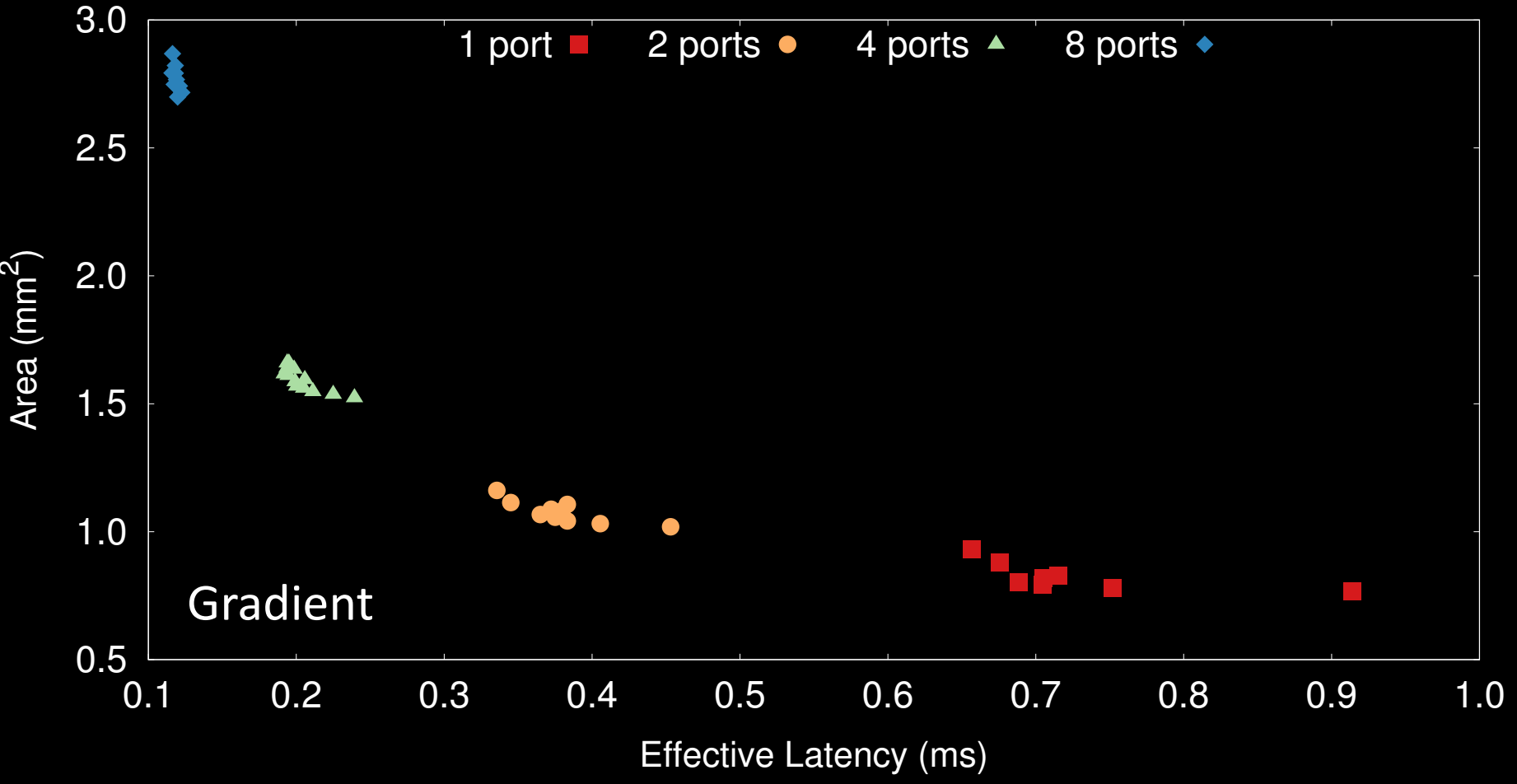
Motivational Examples

- Performing an accurate and exhaustive design-space exploration for a hardware accelerator is complex:

1. HLS tools do not always support the generation (and optimization) of the private local memories
2. The algorithms adopted by HLS tools are based on heuristics that make it hard to set the knobs

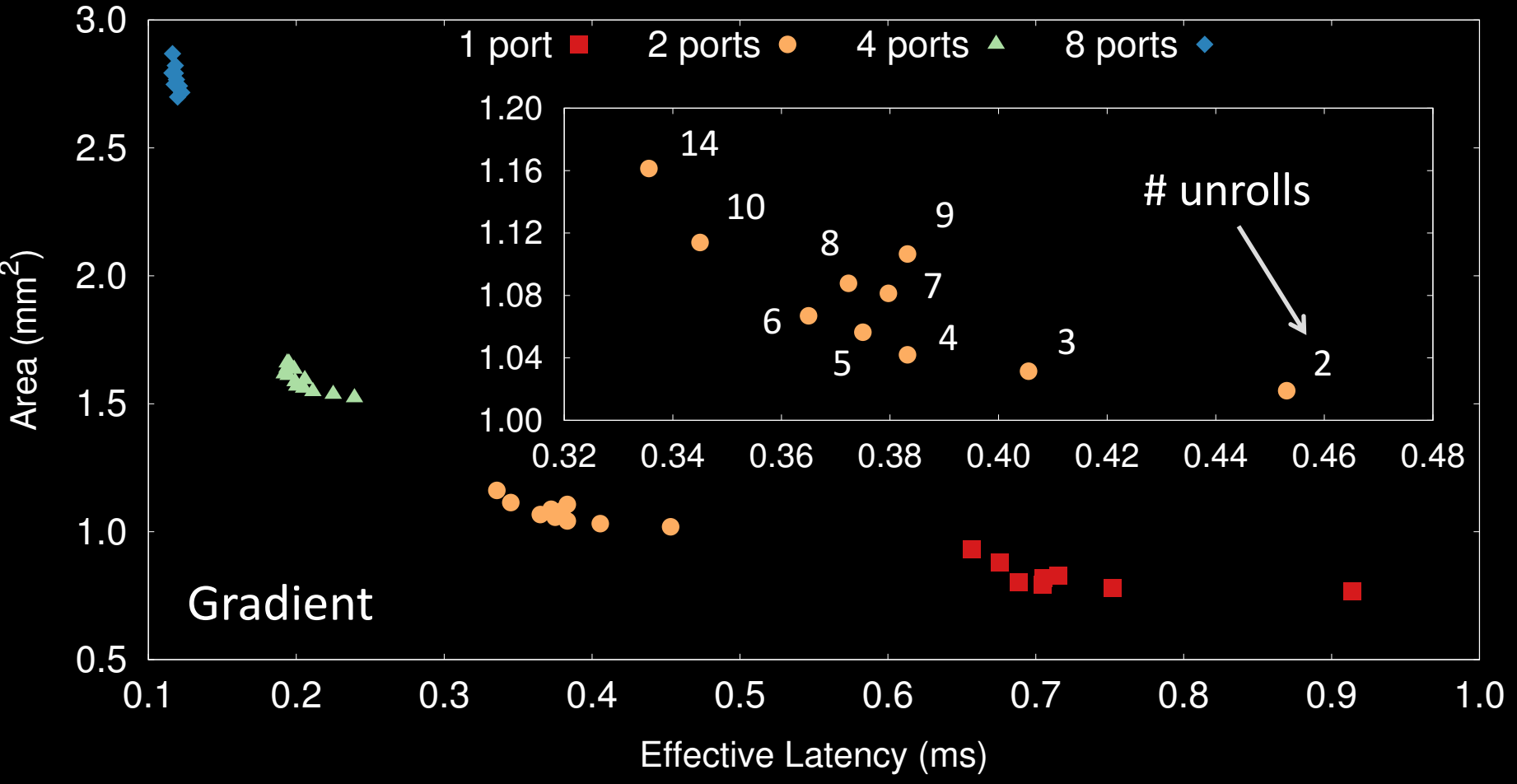
Motivational Examples

Unpredictability of HLS tools



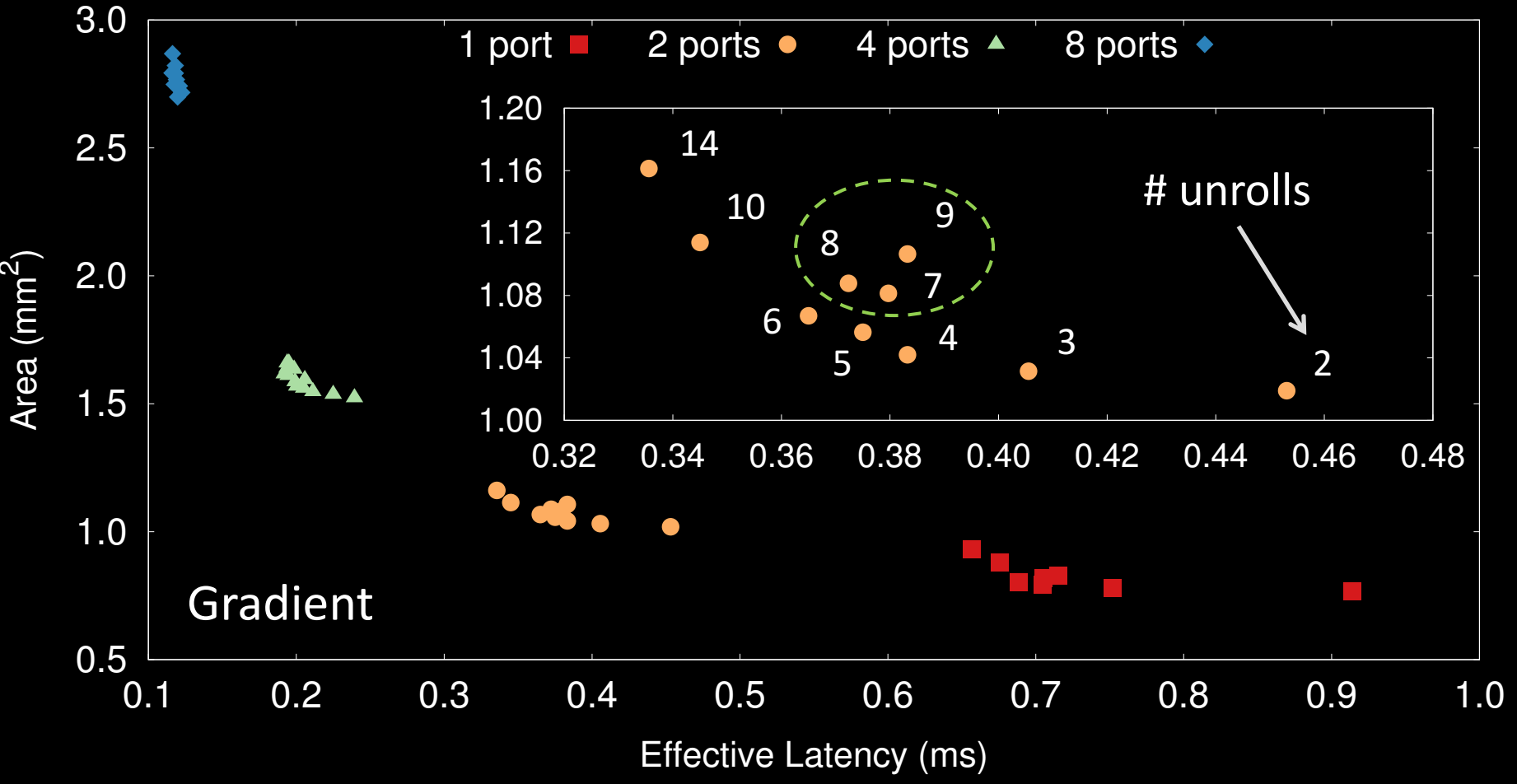
Motivational Examples

Unpredictability of HLS tools



Motivational Examples

Unpredictability of HLS tools



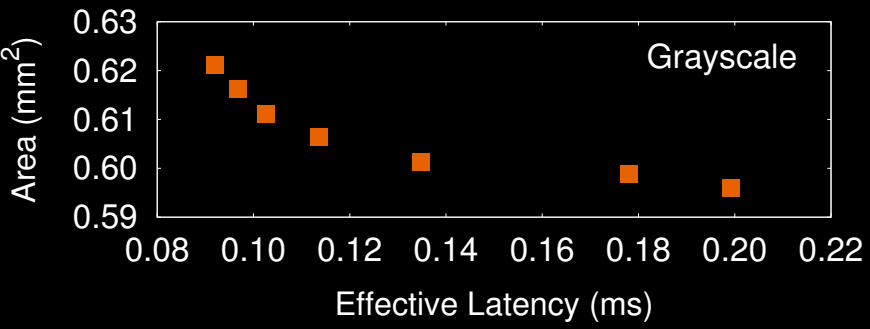
Motivational Examples

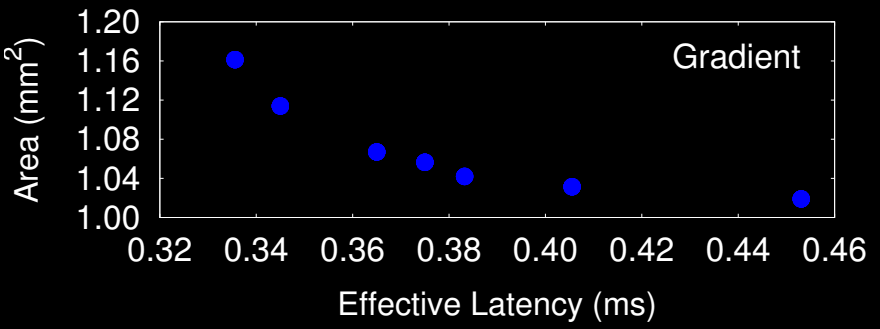
- Performing an accurate and exhaustive design-space exploration for a hardware accelerator is complex:

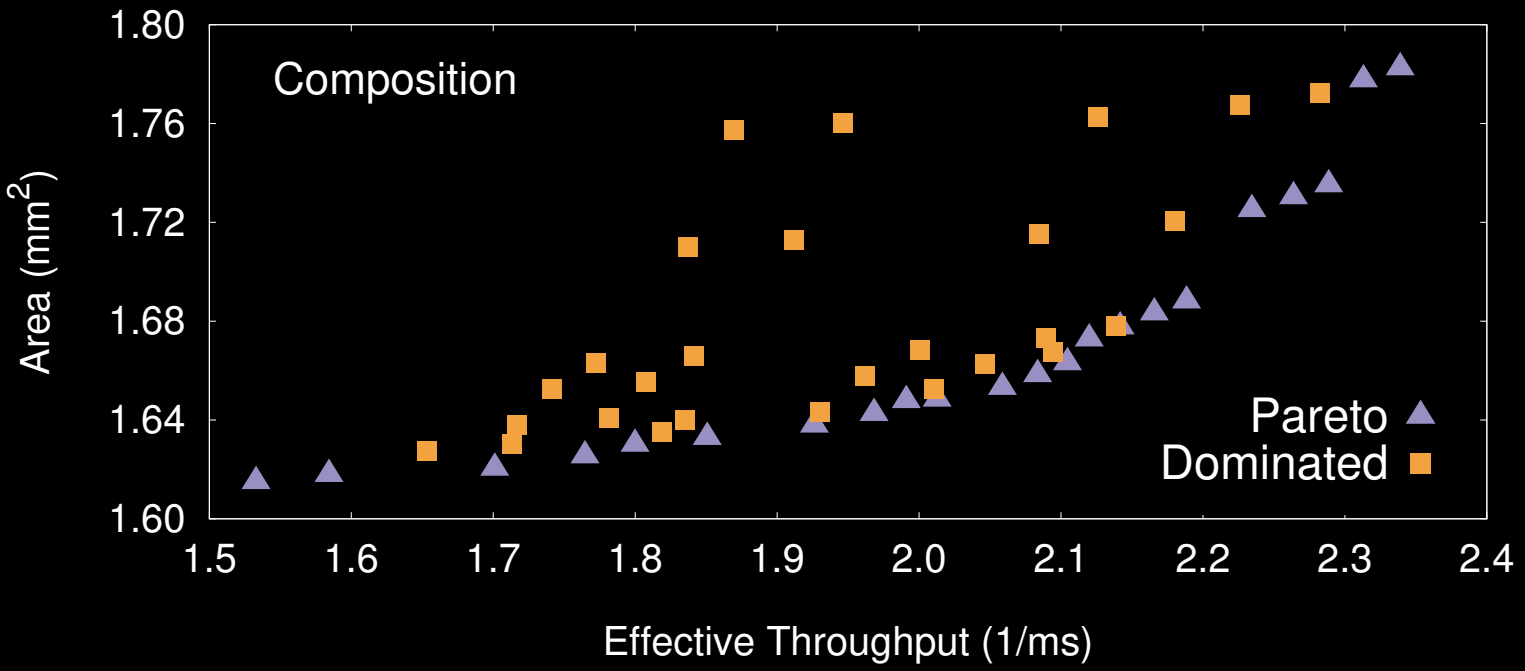
1. HLS tools do not always support the generation (and optimization) of the private local memories
2. The algorithms adopted by HLS tools are based on heuristics that make it hard to set the knobs
3. HLS tools do not handle the simultaneous optimization of multiple components

Motivational Examples

Need of compositionality







Contributions

- We propose COSMOS, an automatic methodology for the design-space exploration of complex accelerators

  1. COSMOS is able to efficiently coordinate high-level synthesis and memory generator tools

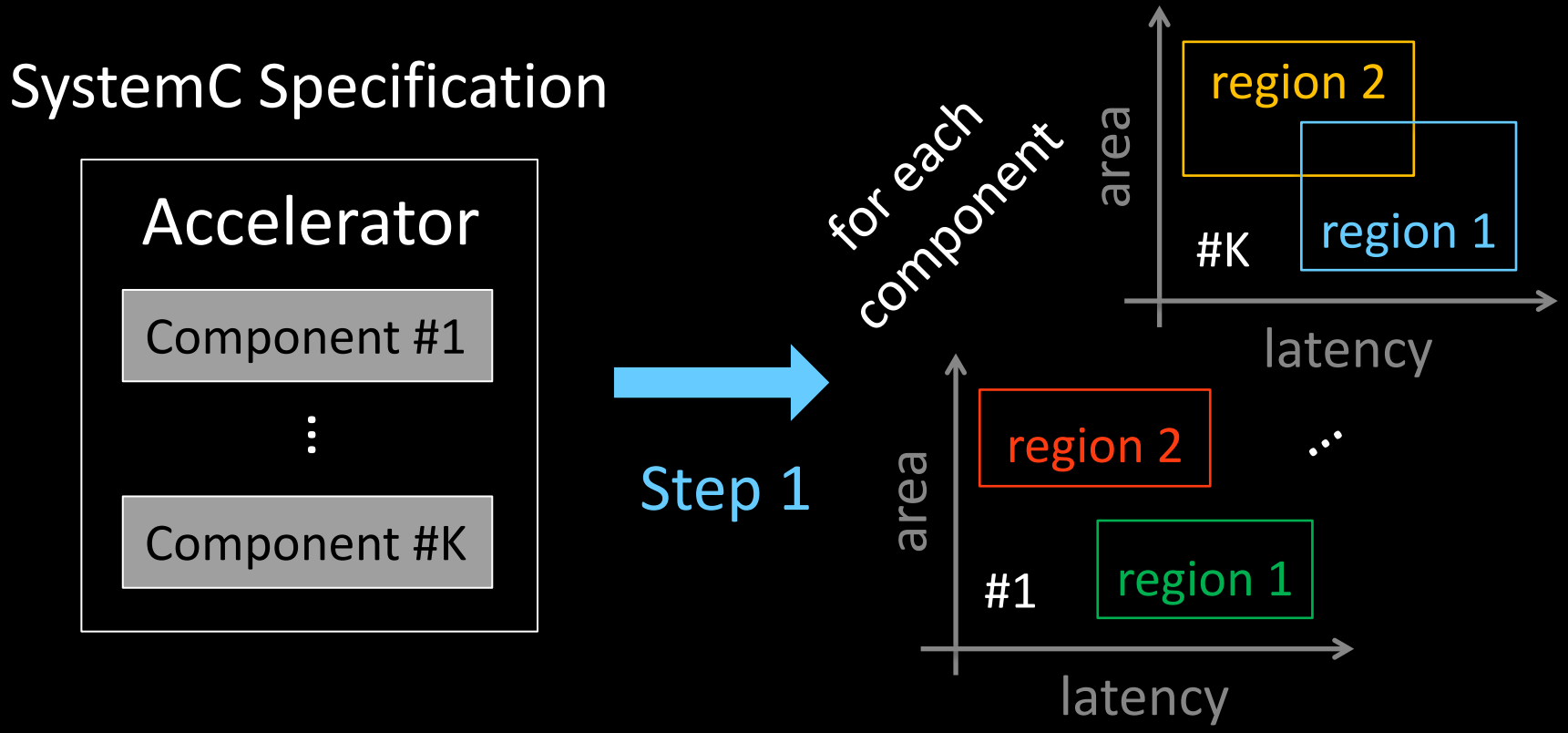
Contributions

- We propose COSMOS, an automatic methodology for the design-space exploration of complex accelerators

  1. COSMOS is able to efficiently coordinate high-level synthesis and memory generator tools
  2. COSMOS leverages a scalable compositional design-space exploration methodology

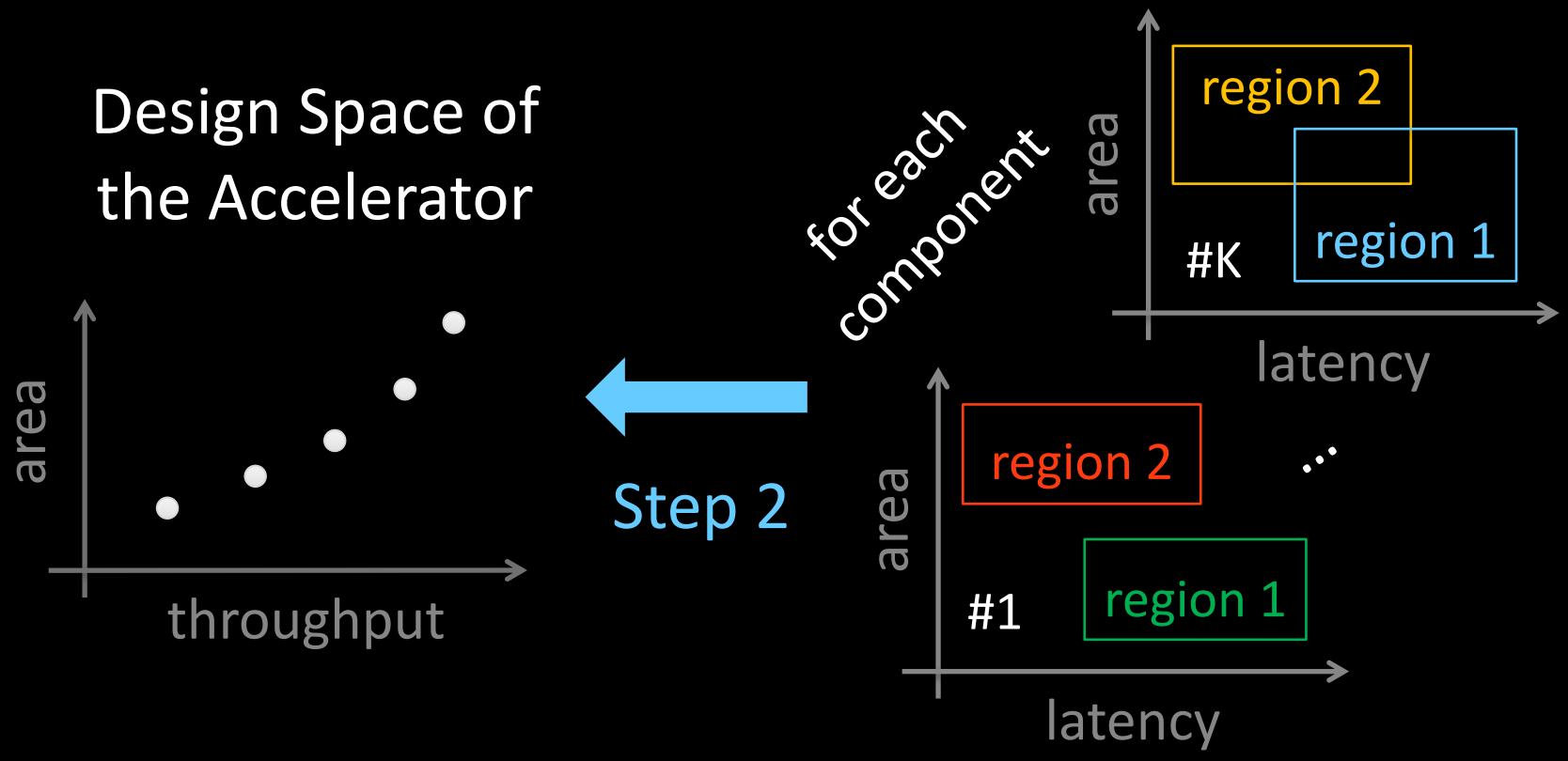
Contributions

- We propose COSMOS, an automatic methodology for the design-space exploration of complex accelerators
  - Step 1: Component Characterization



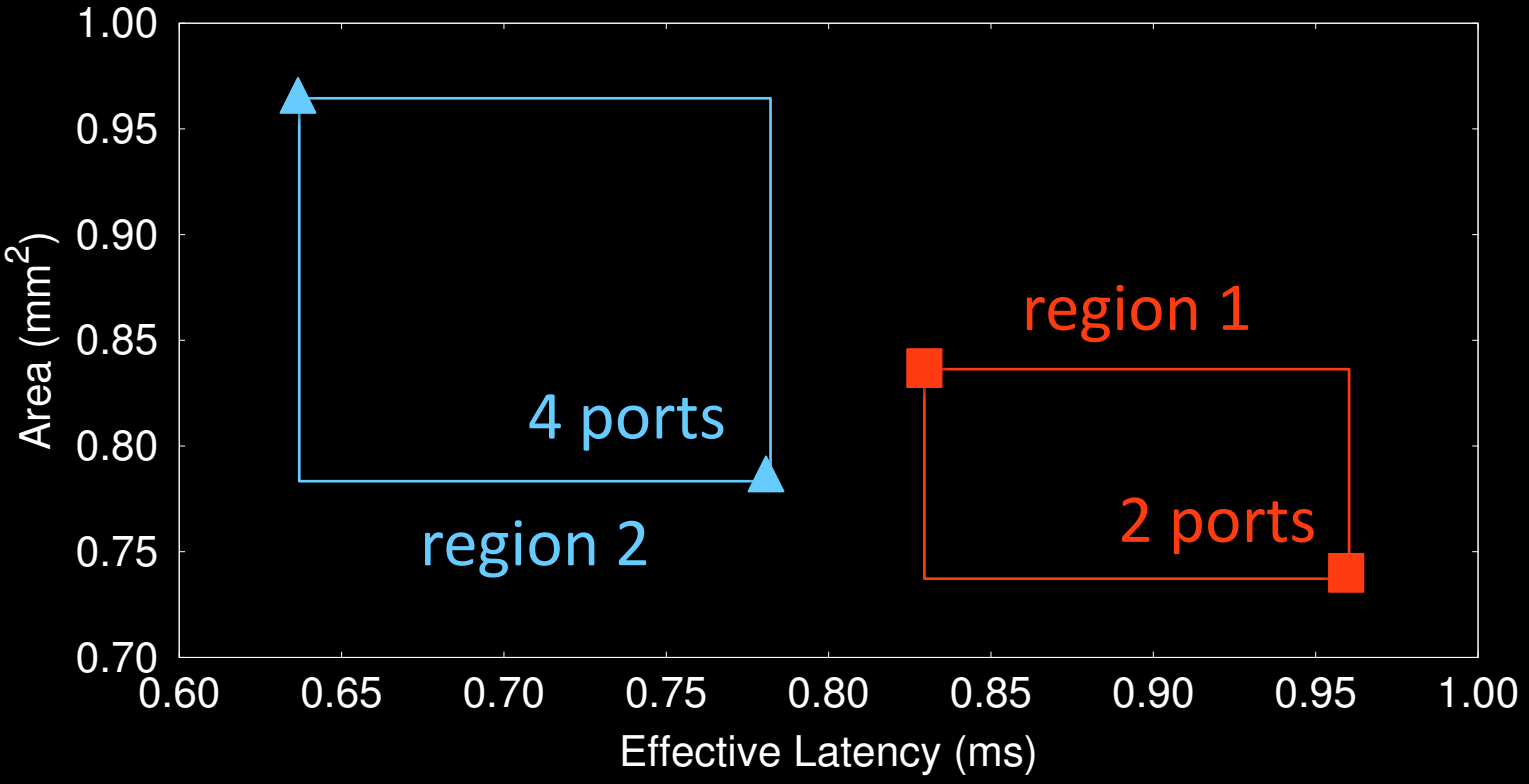
Contributions

- We propose COSMOS, an automatic methodology for the design-space exploration of complex accelerators
  - Step 2: Design-Space Exploration



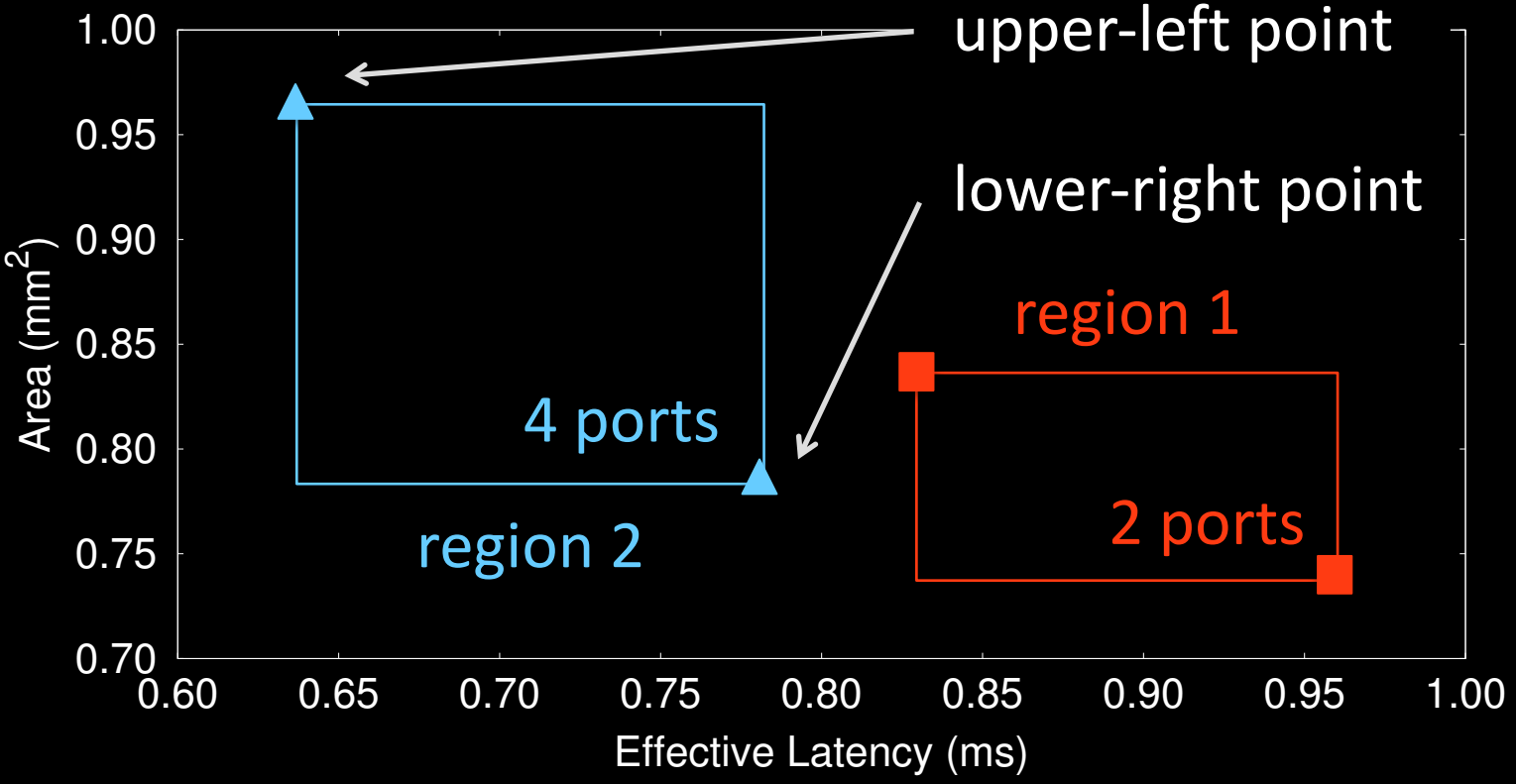
Component Characterization

- Goal: for each component of the accelerator identify the regions with the Pareto-optimal implementations



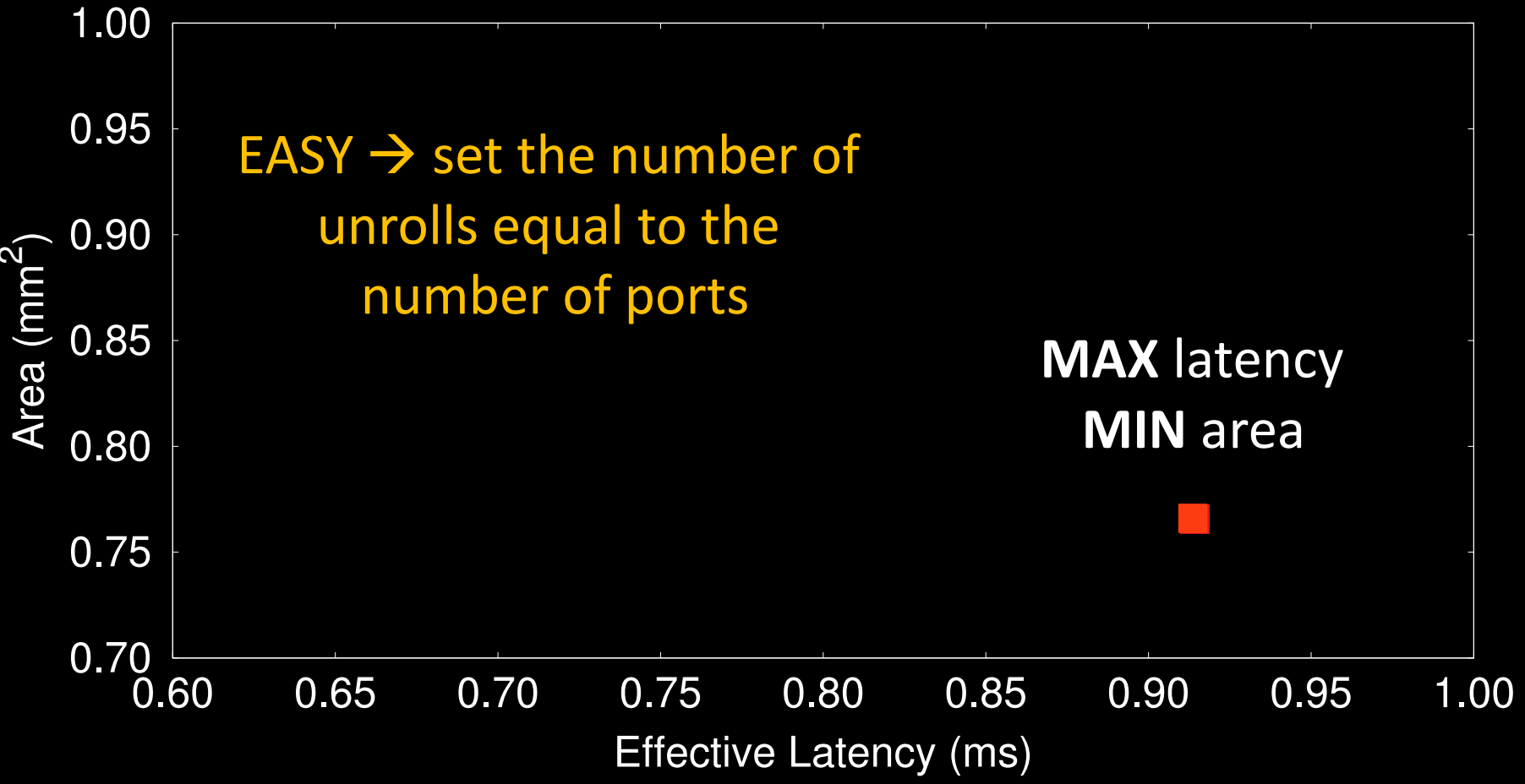
Component Characterization

- Goal: for each component of the accelerator identify the regions with the Pareto-optimal implementations



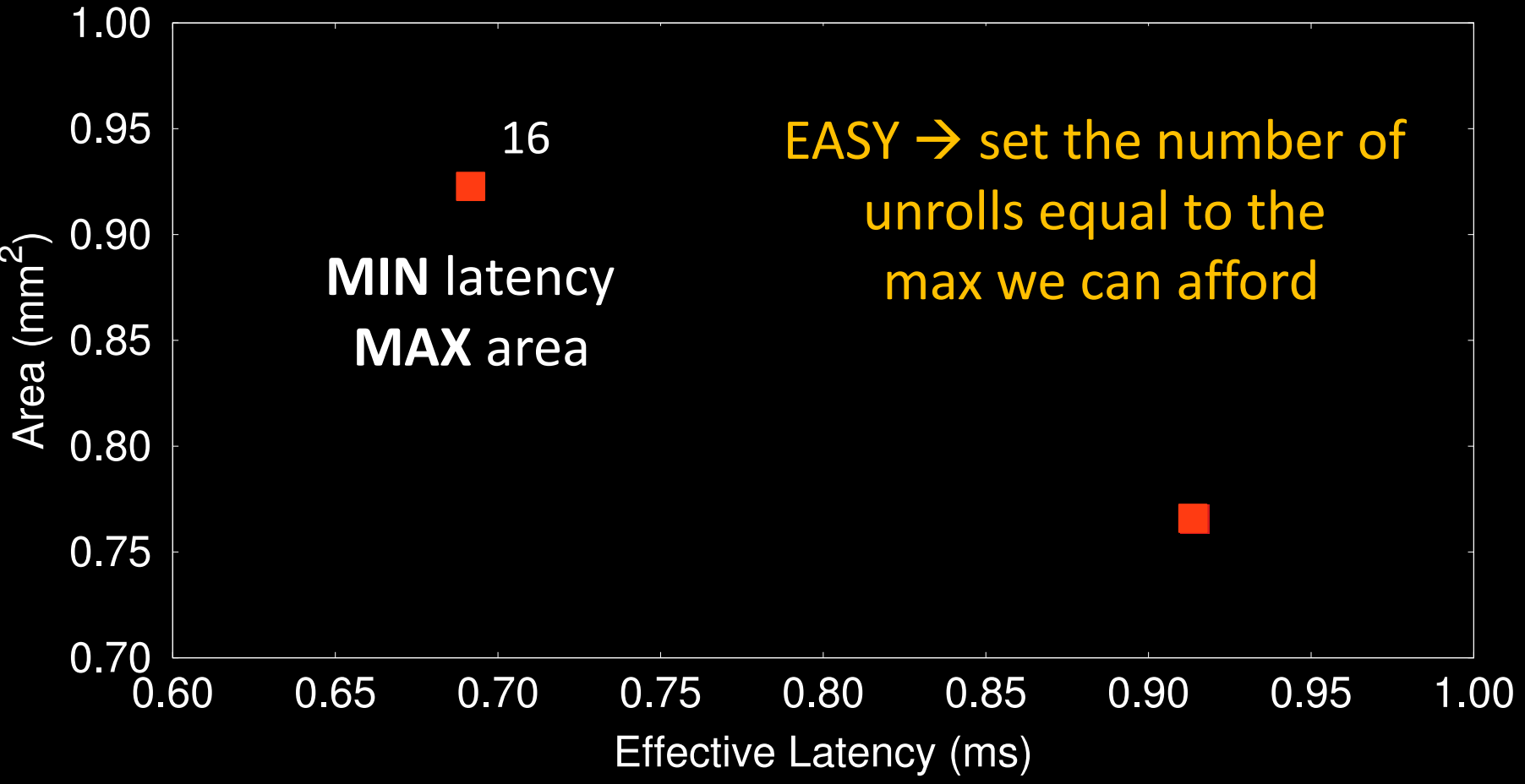
Component Characterization

How to identify the lower-right point



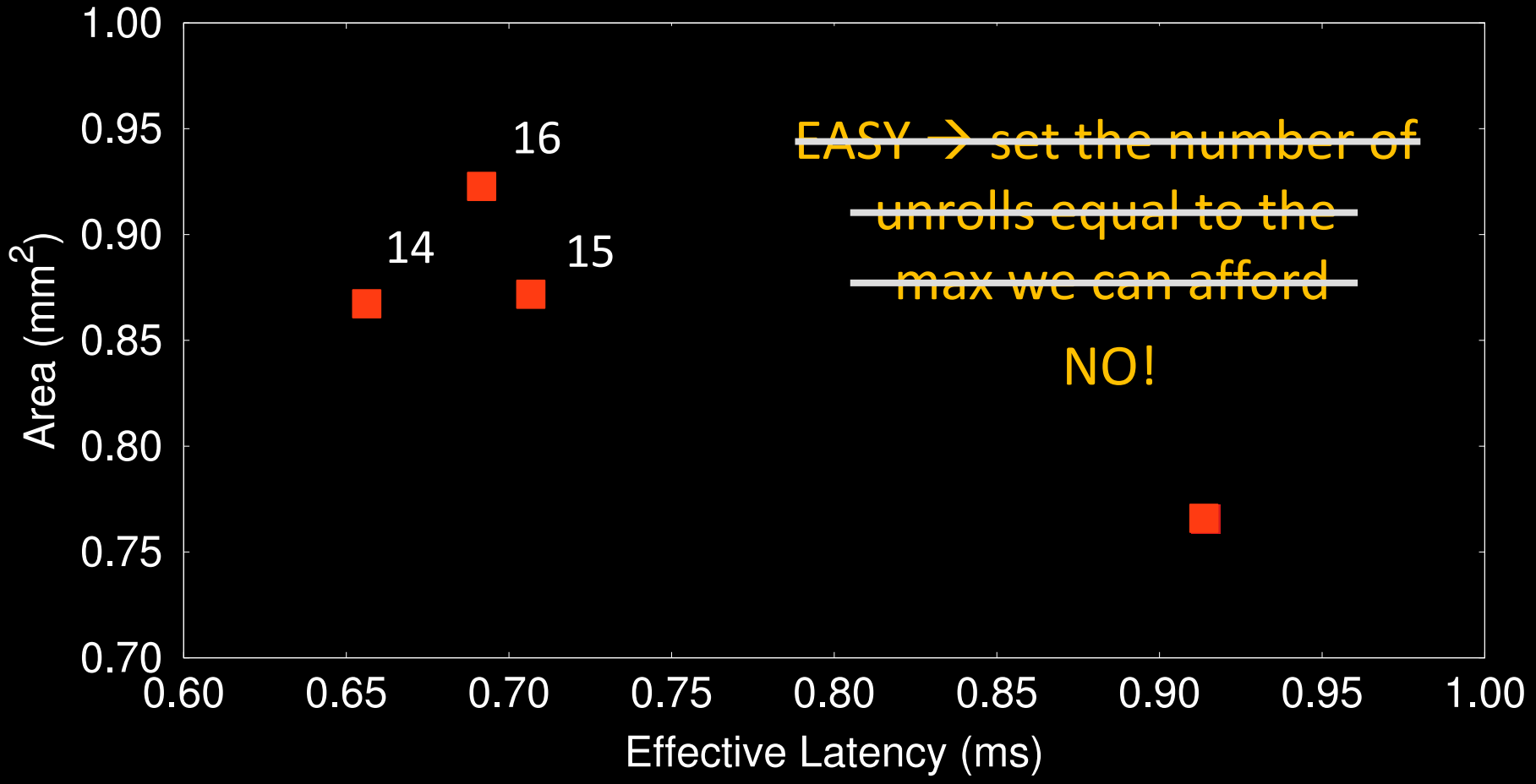
Component Characterization

How to identify the upper-left point



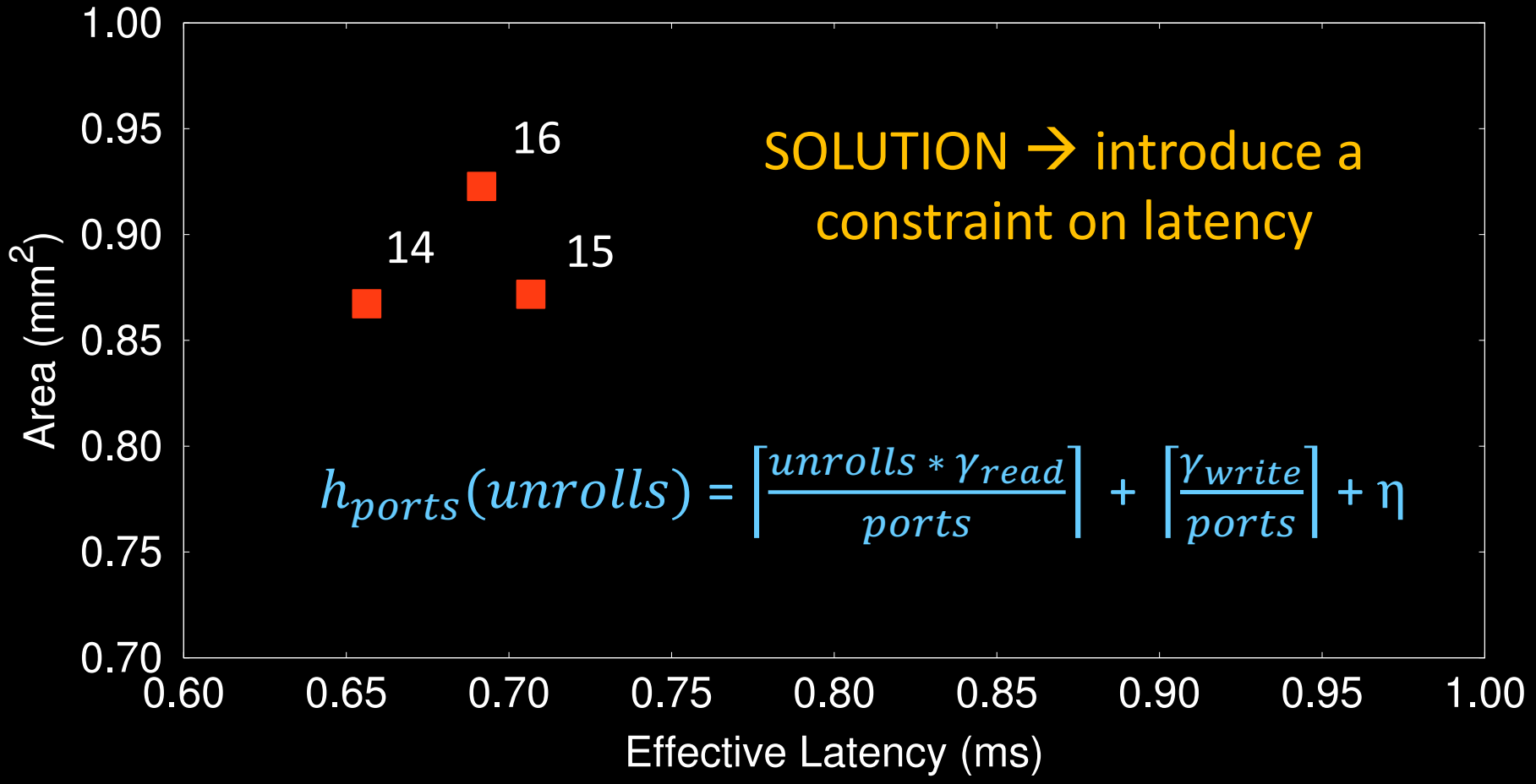
Component Characterization

How to identify the upper-left point



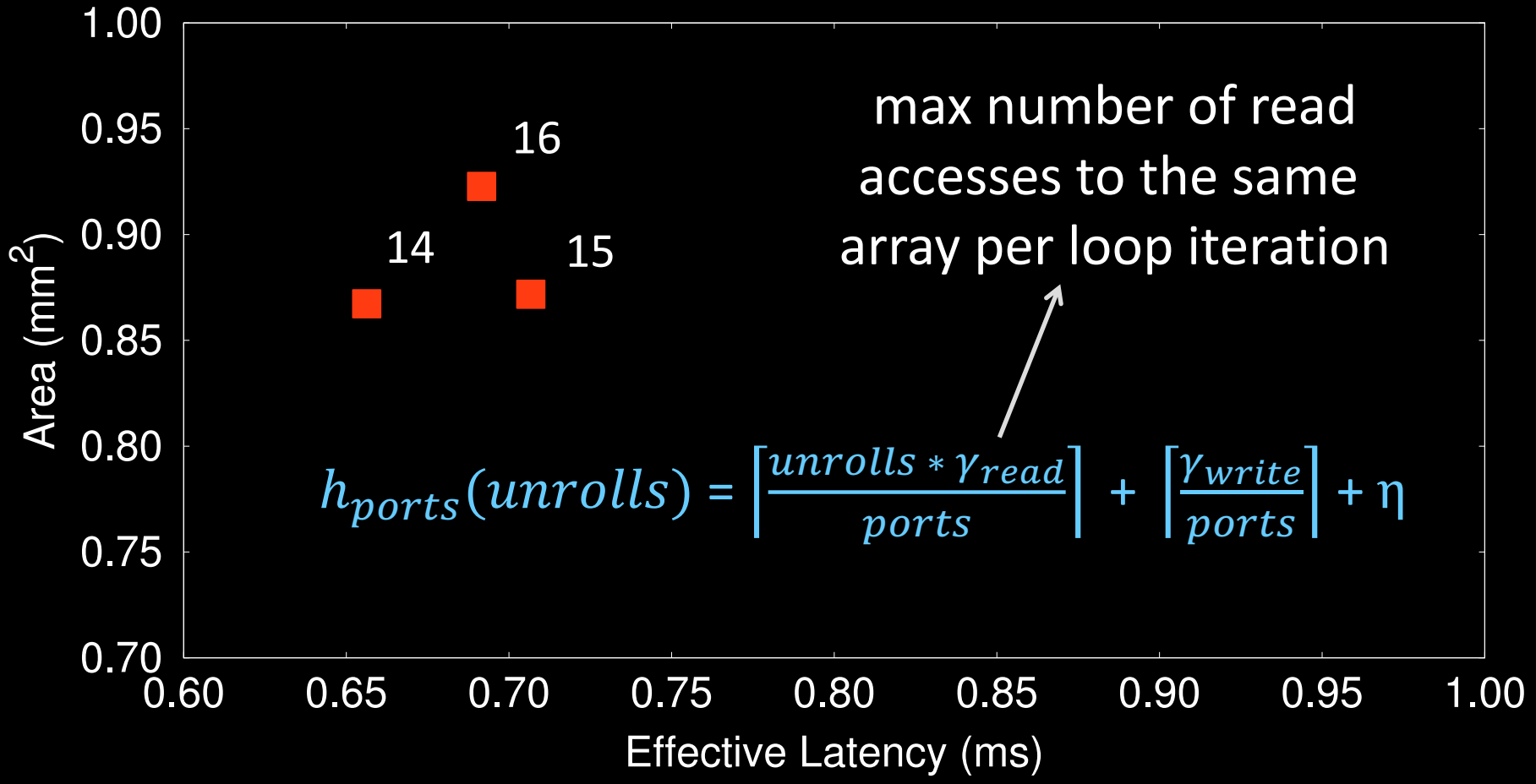
Component Characterization

How to identify the upper-left point



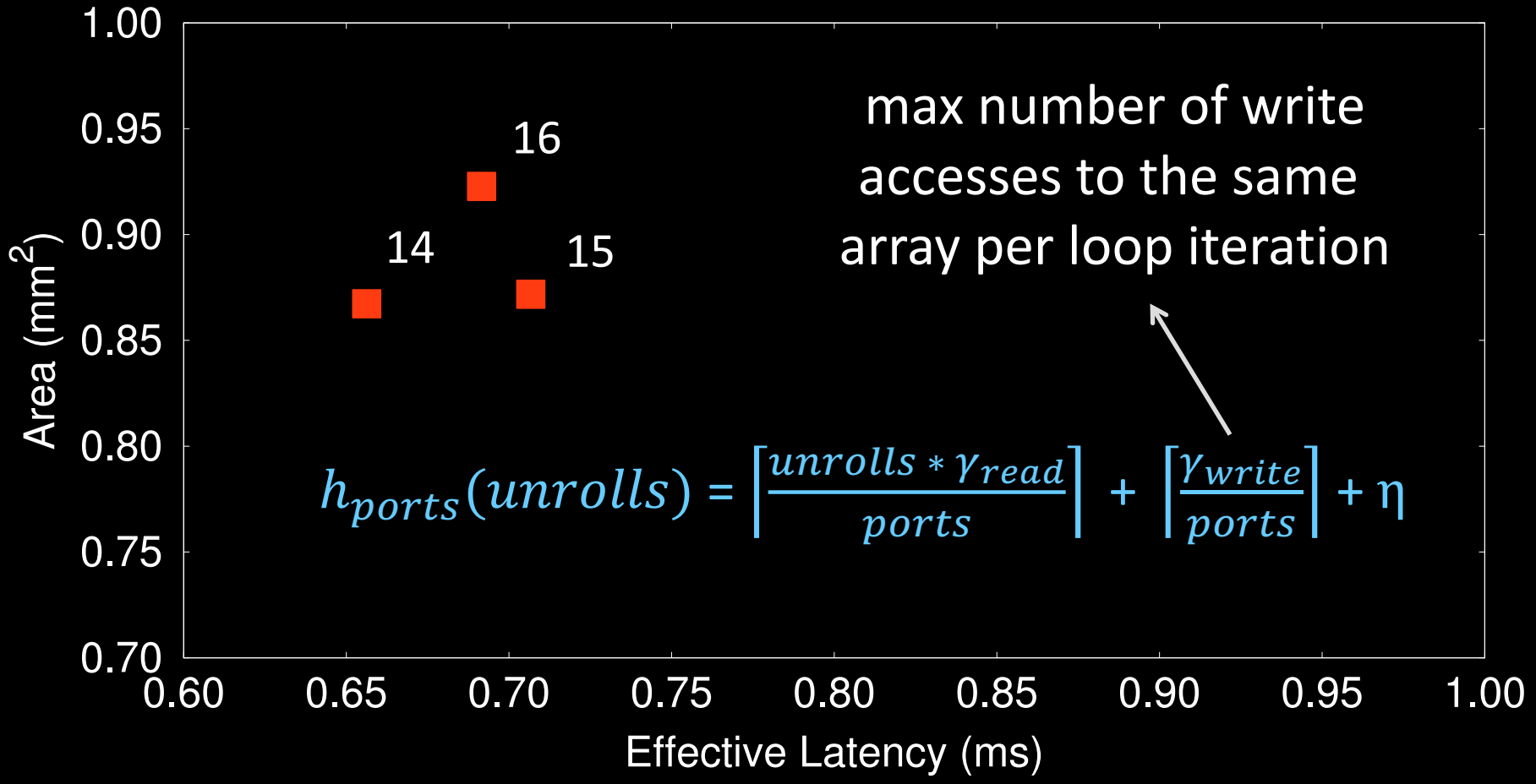
Component Characterization

How to identify the upper-left point



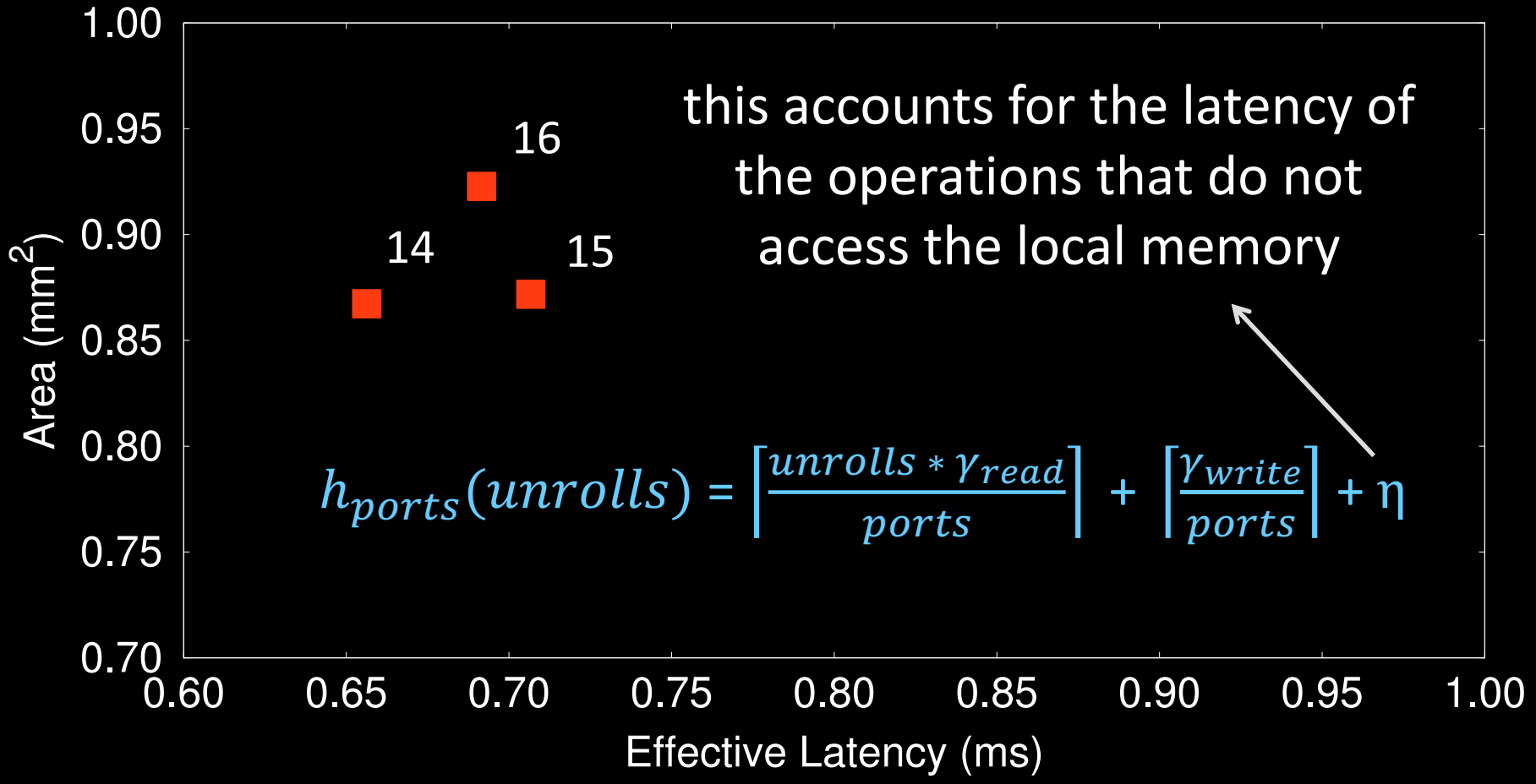
Component Characterization

How to identify the upper-left point



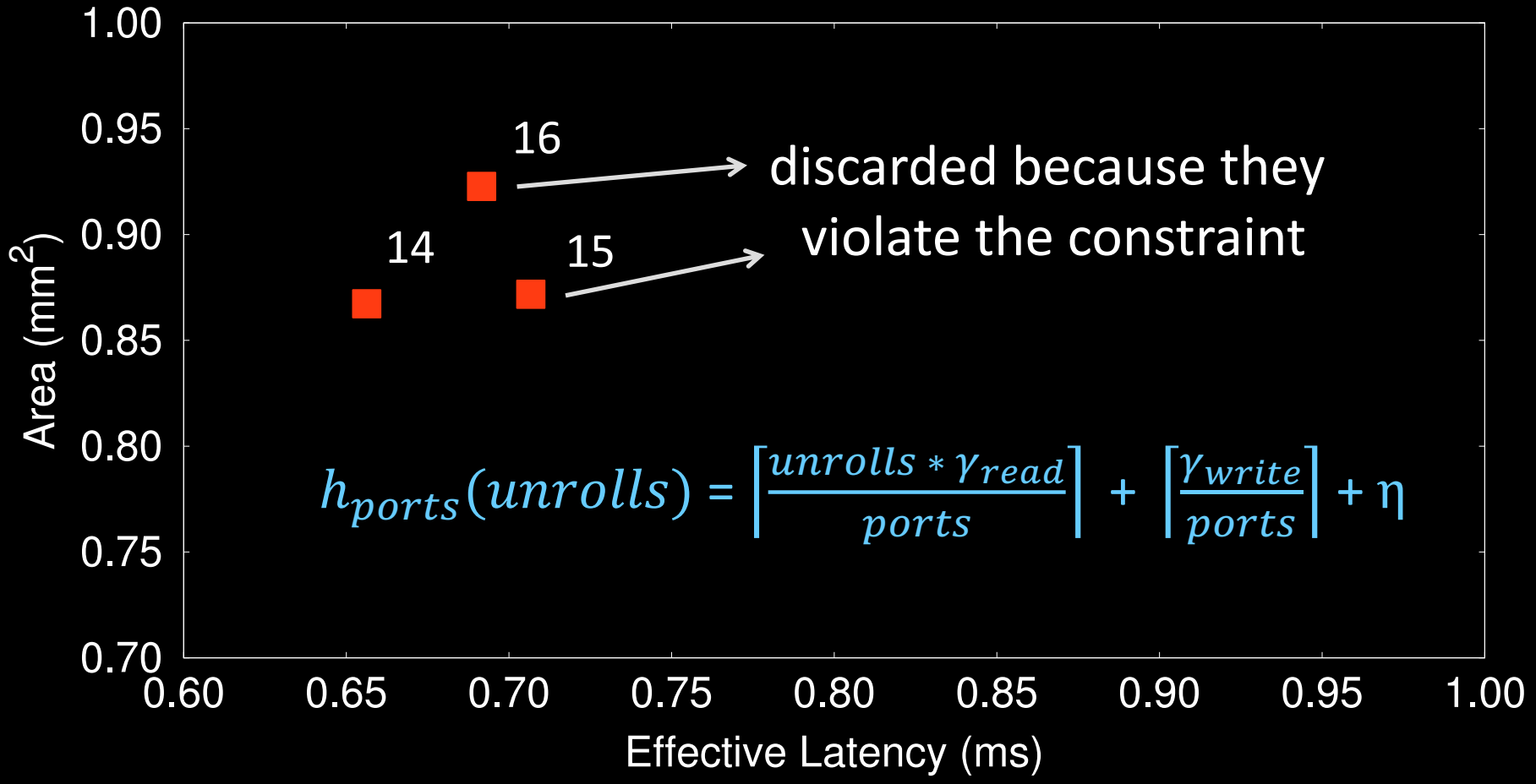
Component Characterization

How to identify the upper-left point

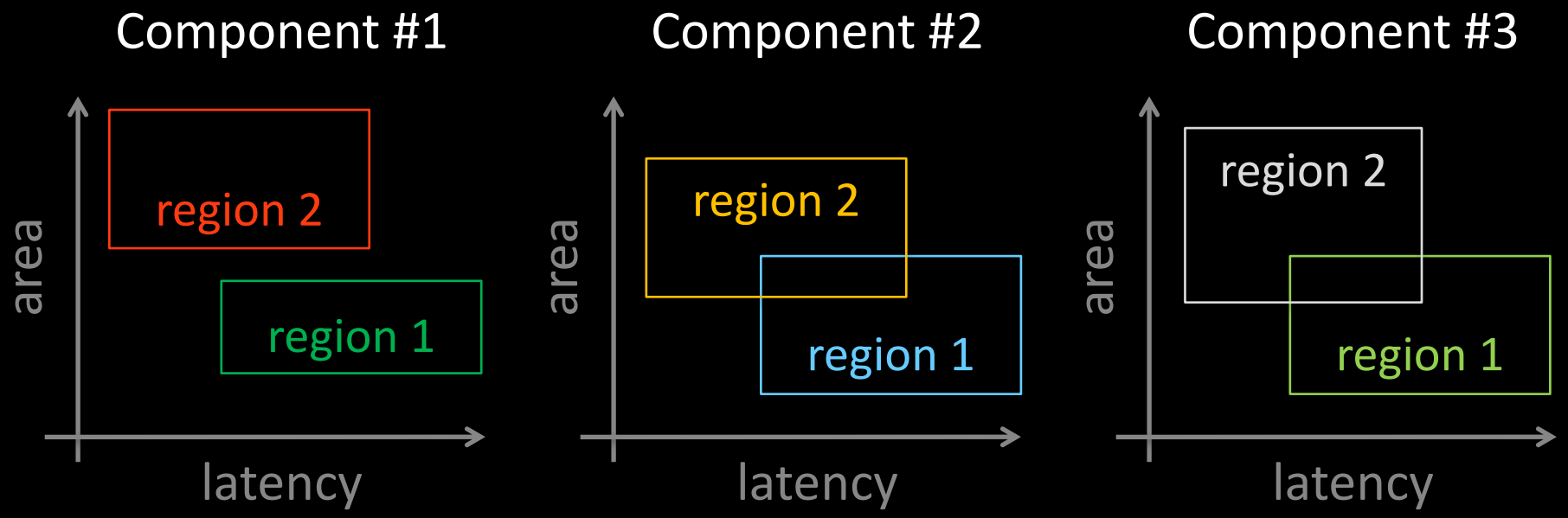


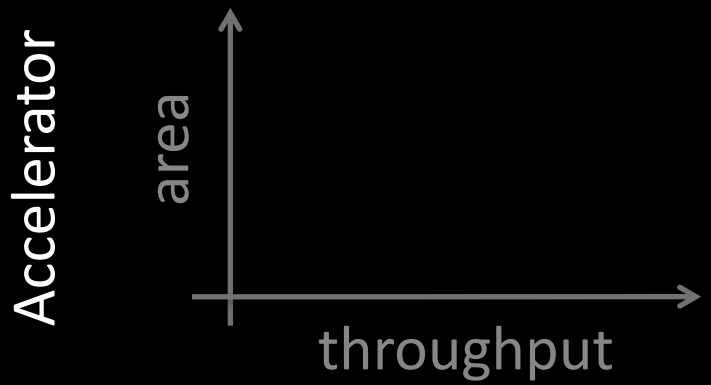
Component Characterization

Identifying the upper-left point



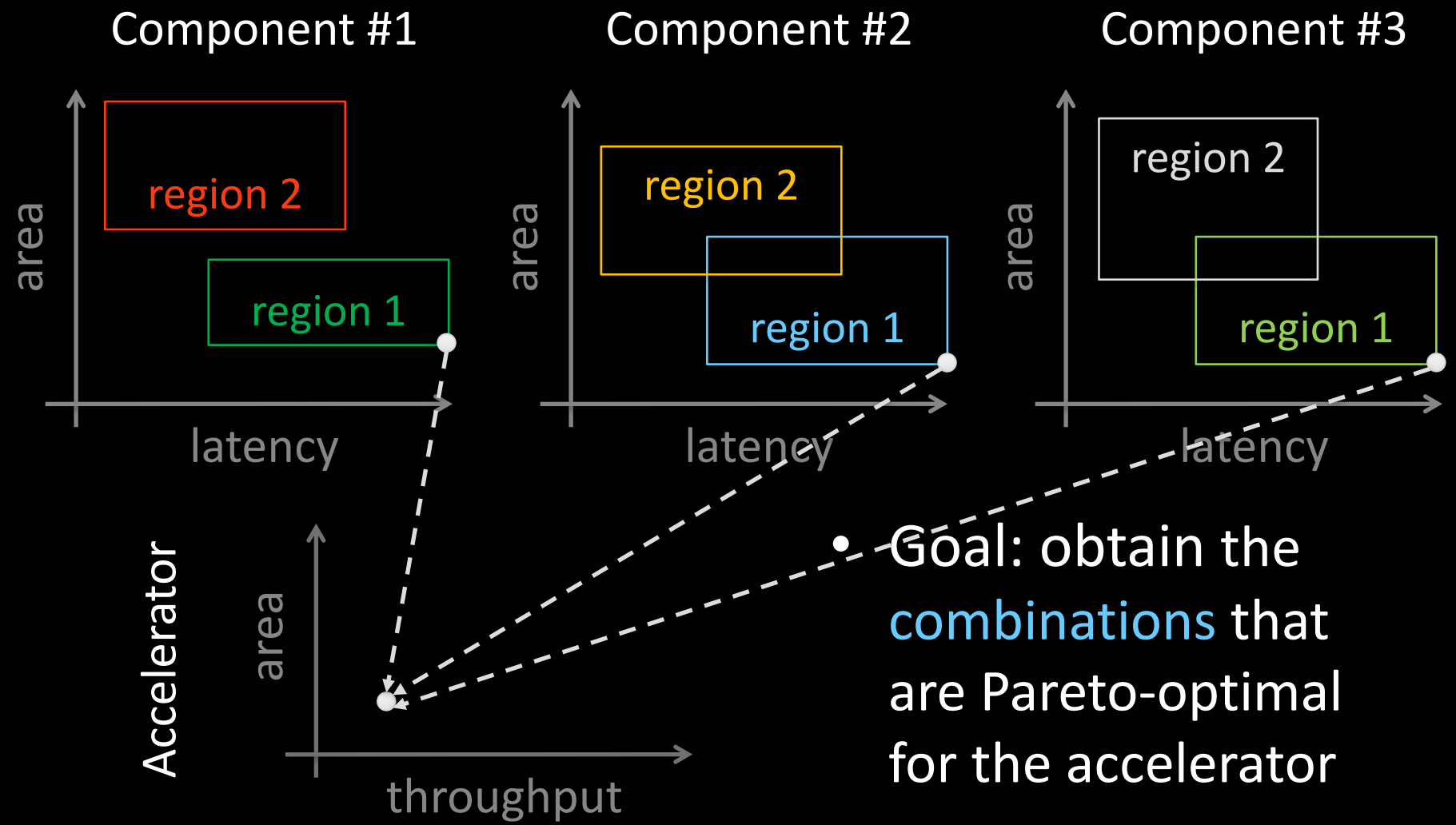
Design-Space Exploration





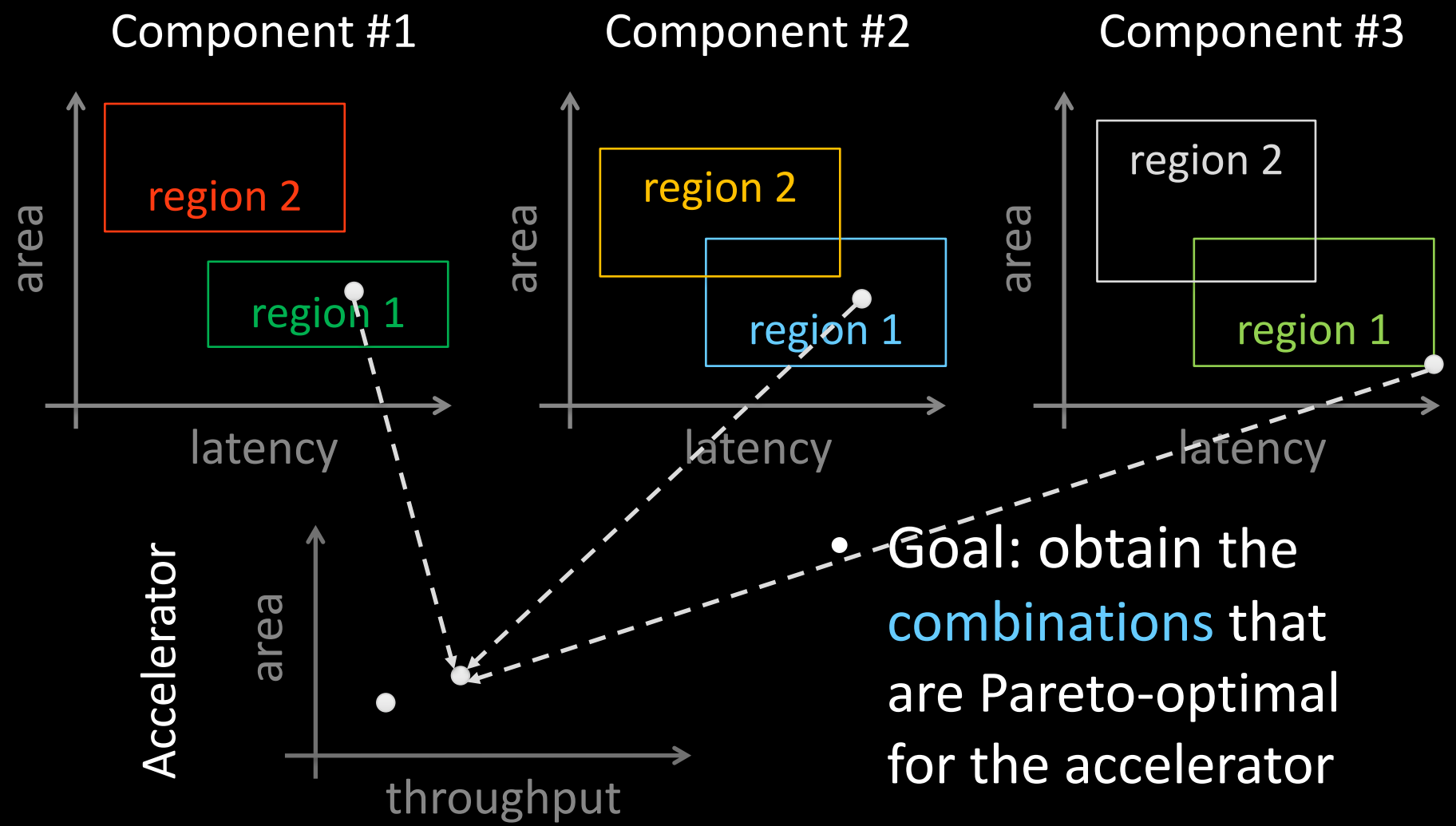
- Goal: obtain the combinations that are Pareto-optimal for the accelerator

Design-Space Exploration

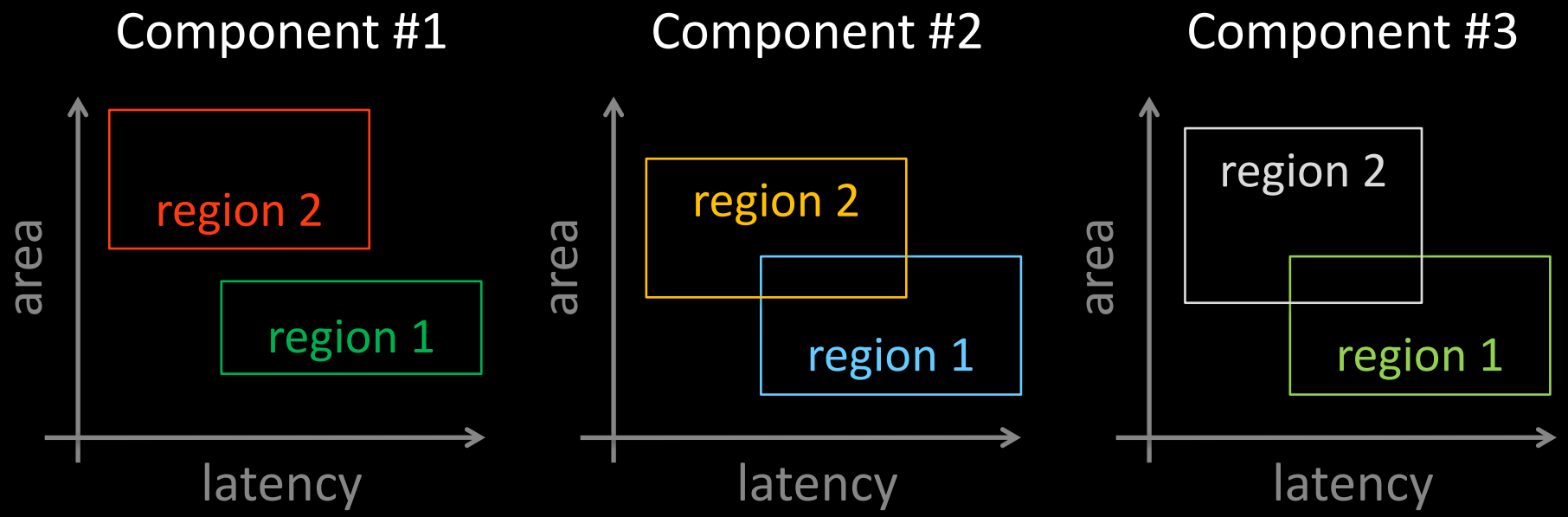


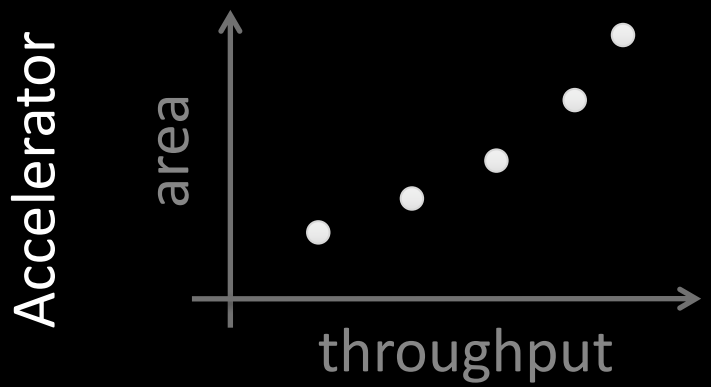
- Goal: obtain the combinations that are Pareto-optimal for the accelerator

Design-Space Exploration



Design-Space Exploration

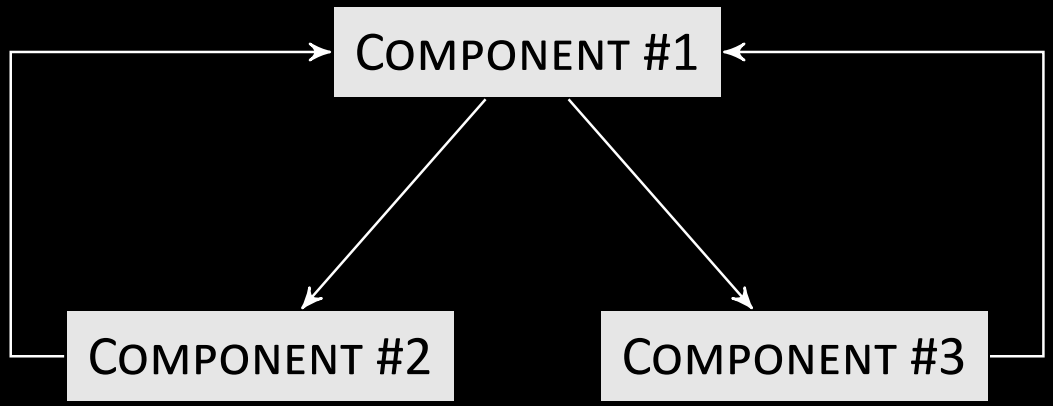




- Goal: obtain the combinations that are Pareto-optimal for the accelerator

Design-Space Exploration

Step 1: Synthesis Planning



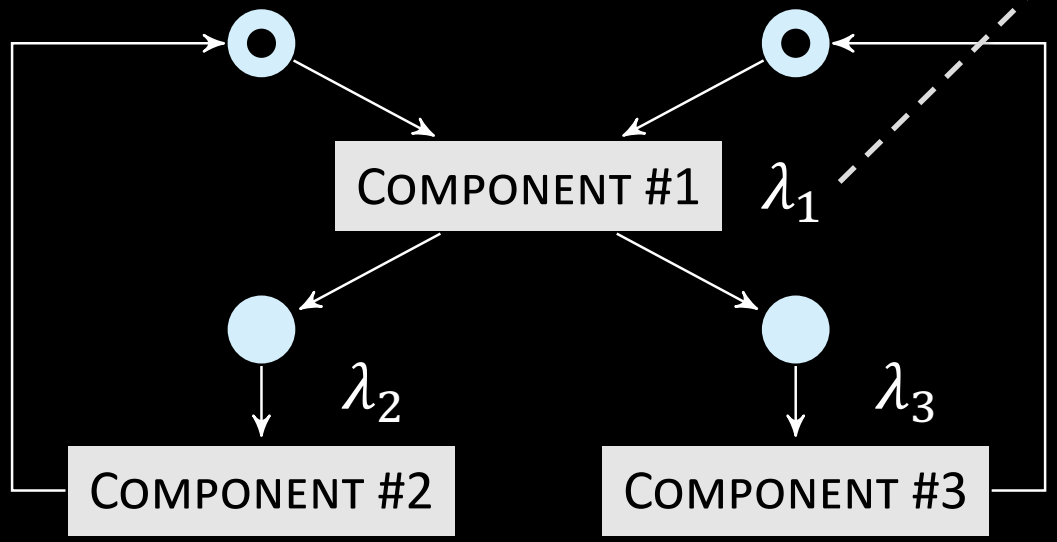
Computational dependencies among the components of the accelerator

Design-Space Exploration

Step 1: Synthesis Planning

$\lambda_{min1} < \lambda_1 < \lambda_{max1}$

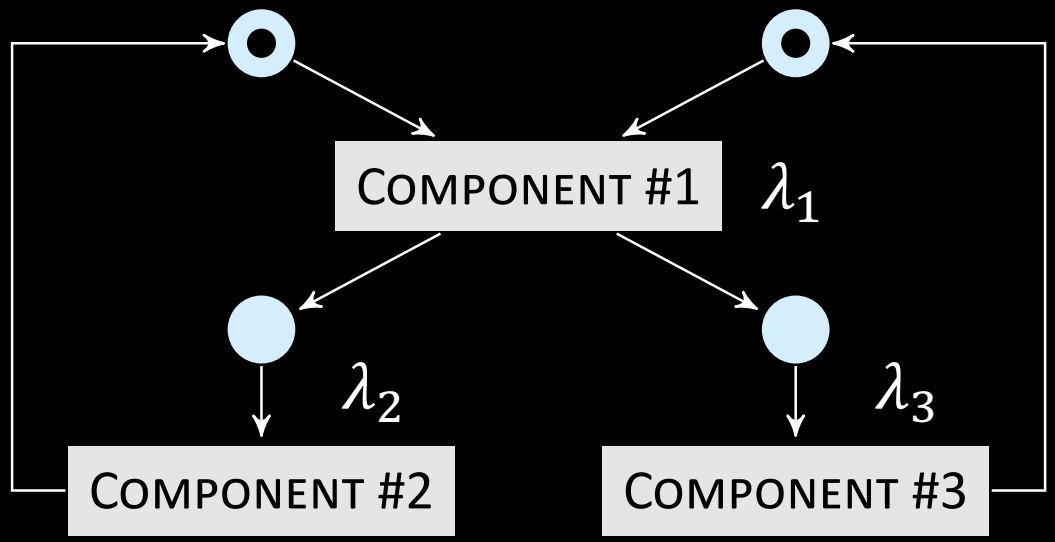
from component characterization



Timed Marked Graph (TMG)

Design-Space Exploration

Step 1: Synthesis Planning



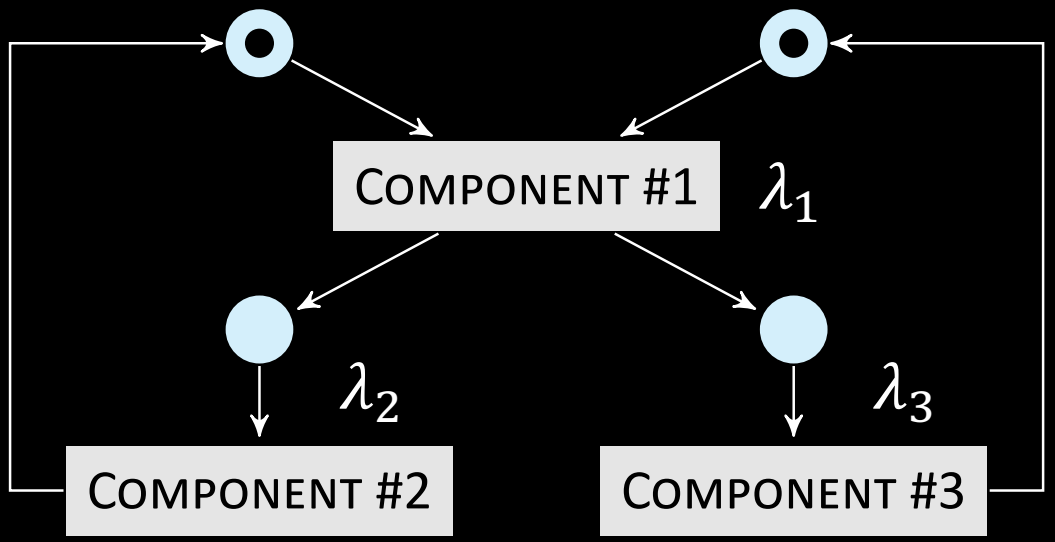
Timed Marked Graph (TMG)

throughput of the accelerator:

$$\vartheta = \frac{1}{min(\frac{1}{\lambda_1 + \lambda_2}, \frac{1}{\lambda_1 + \lambda_3})}$$

Design-Space Exploration

Step 1: Synthesis Planning



Timed Marked Graph (TMG)

throughput of the accelerator:

$$\vartheta = \frac{1}{min(\frac{1}{\lambda_1+\lambda_2}, \frac{1}{\lambda_1+\lambda_3})}$$

throughput ϑ

Linear Programming

- minimize area

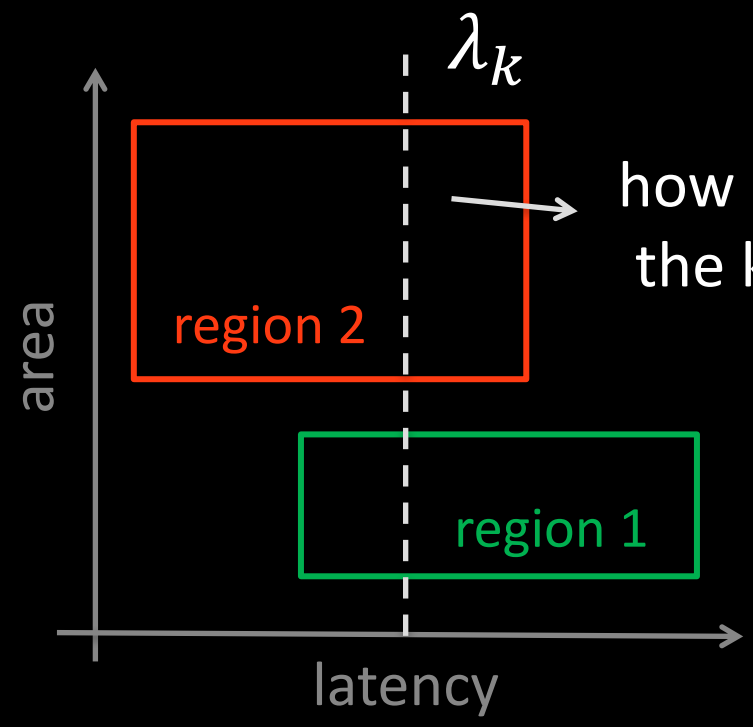
[Liu et al., DATE '12]

$\lambda_1, \lambda_2, \lambda_3$

Design-Space Exploration

Step 2: Synthesis Mapping

λ_k is a theoretical solution, thus we need to map λ_k to knob setting



how do we choose the knob setting?

throughput ϑ

Linear Programming

- minimize area

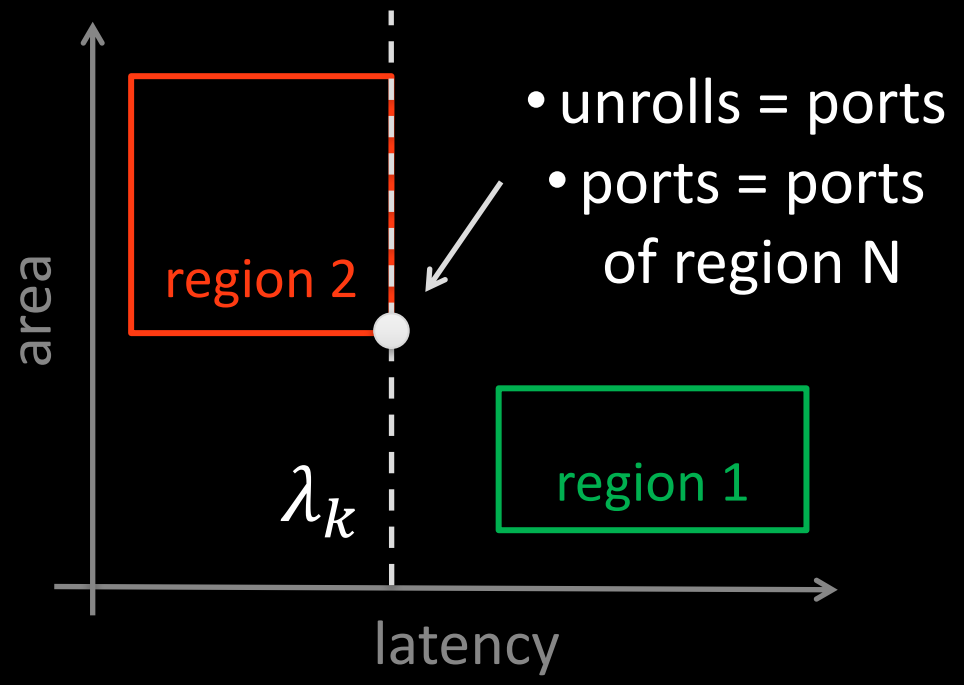
[Liu et al., DATE '12]

λ_k

Design-Space Exploration

Step 2: Synthesis Mapping

CASE 1: λ_k corresponds to one of the extreme point of region N
→ no synthesis required!



throughput ϑ

Linear Programming
• minimize area

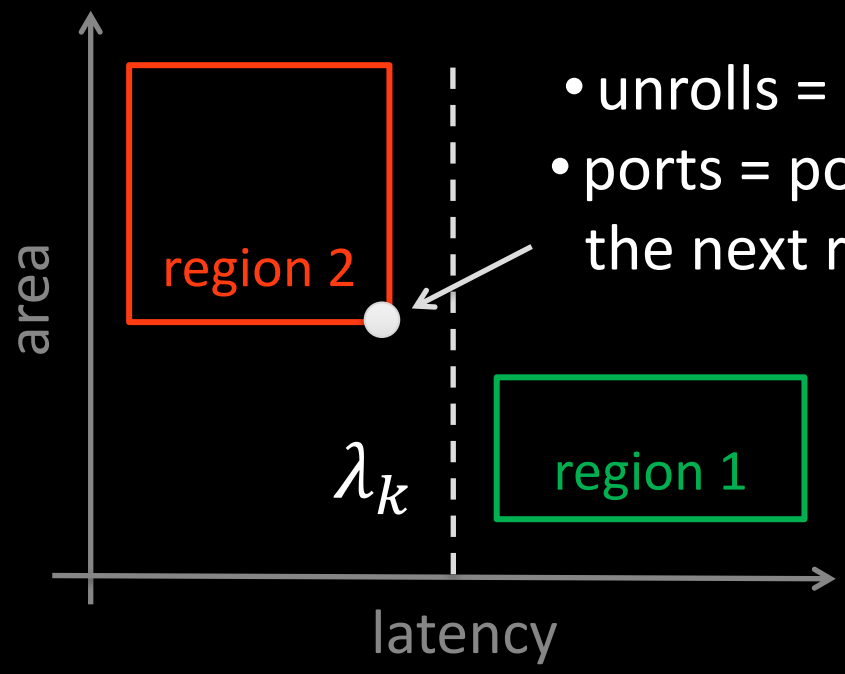
[Liu et al., DATE '12]

λ_k

Design-Space Exploration

Step 2: Synthesis Mapping

CASE 2: λ_k falls outside the regions
→ no synthesis and preserving throughput is our objective



- unrolls = ports
- ports = ports of the next region

throughput ϑ

Linear Programming
- minimize area

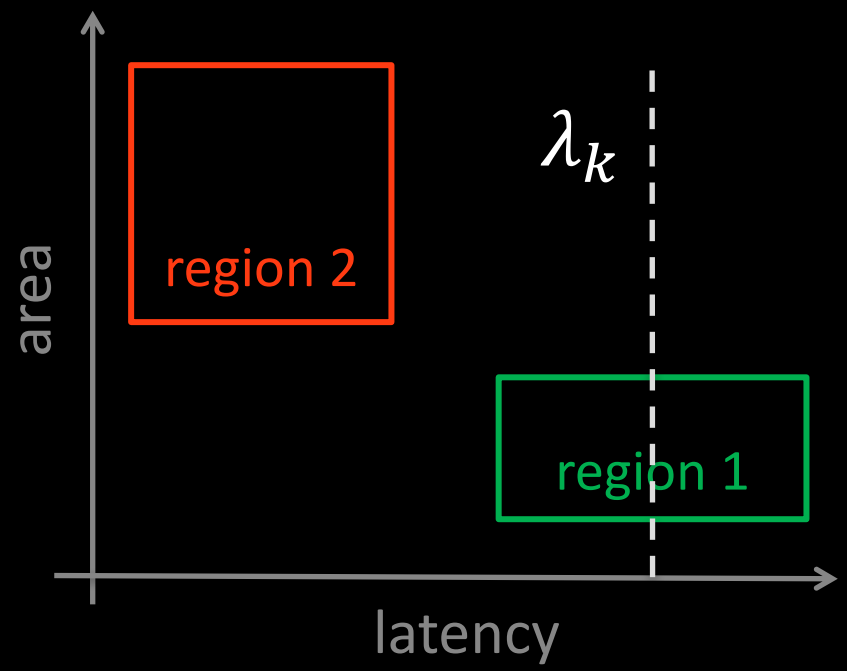
[Liu et al., DATE '12]

λ_k

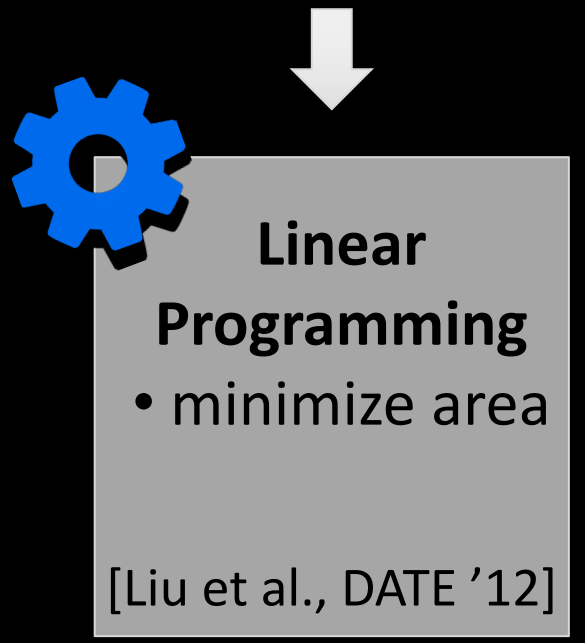
Design-Space Exploration

Step 2: Synthesis Mapping

CASE 3: λ_k falls inside a region
→ synthesis required to get the actual implementation



throughput ϑ

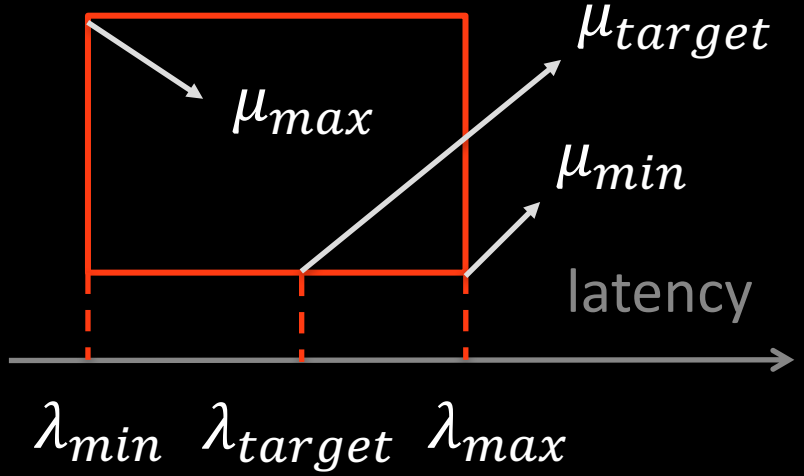


λ_k

Design-Space Exploration

Step 2: Synthesis Mapping

CASE 3: λ_k falls inside a region



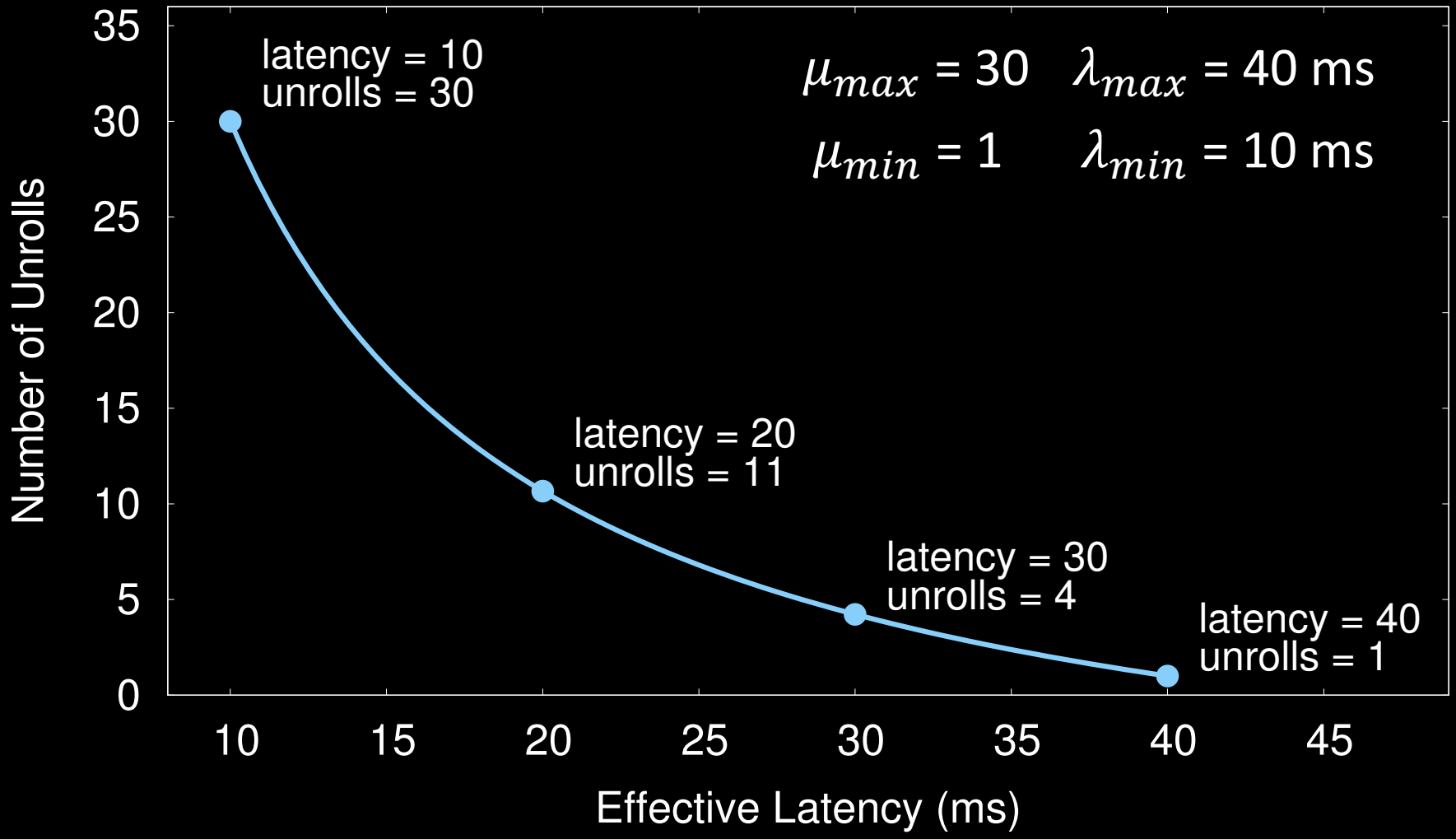
$$S = \frac{\lambda_{target}}{\lambda_{max}} \quad F = \frac{\lambda_{min}}{\lambda_{max}}$$

$$P = \frac{\mu_{target} - \mu_{min}}{\mu_{max} - \mu_{min}}$$

Amdahl's Law $$S = \frac{1}{(1 - P) + \frac{P}{F}}$$

Design-Space Exploration

Step 2: Synthesis Mapping



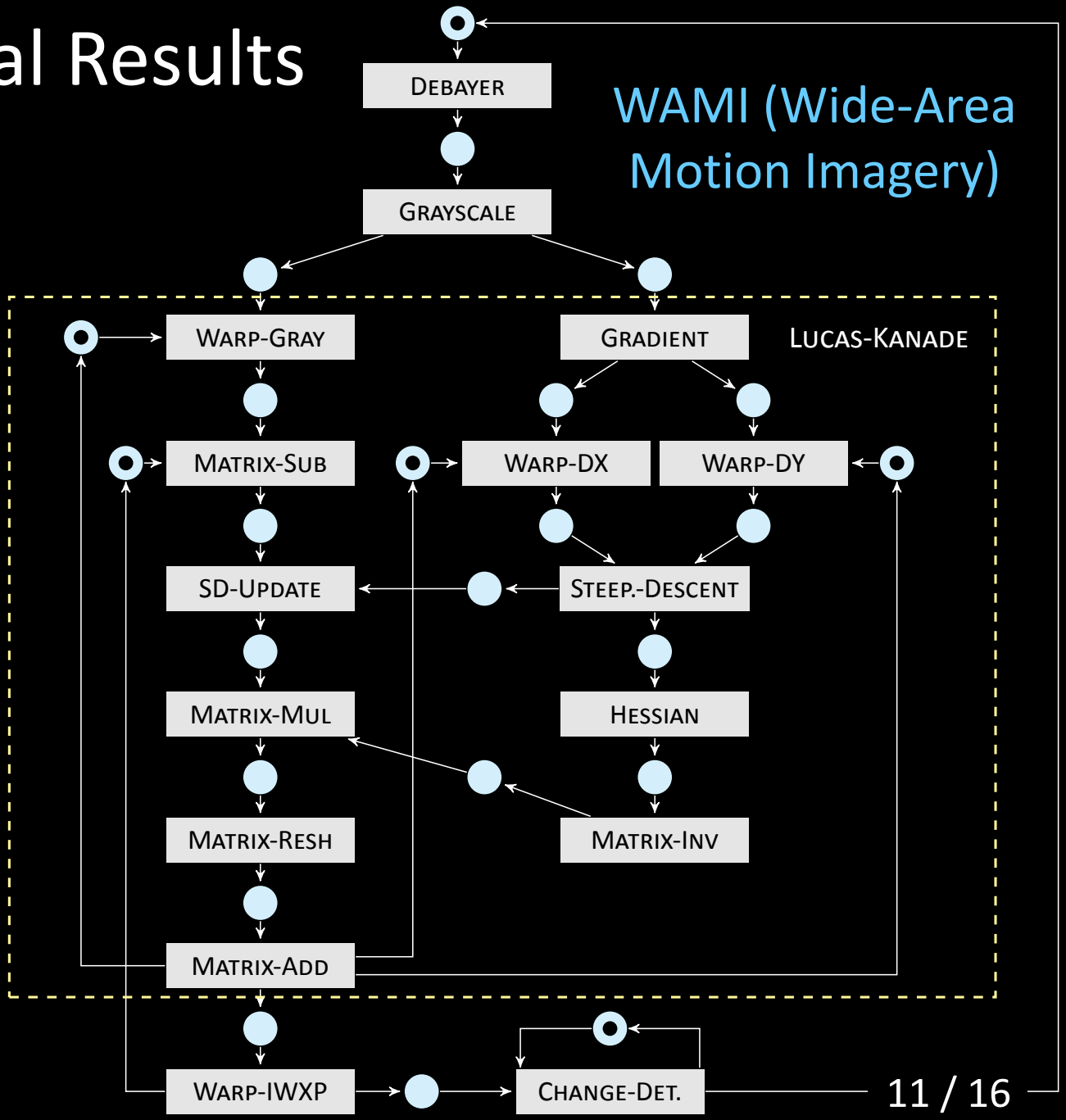
Experimental Results

Case Study

C Specification

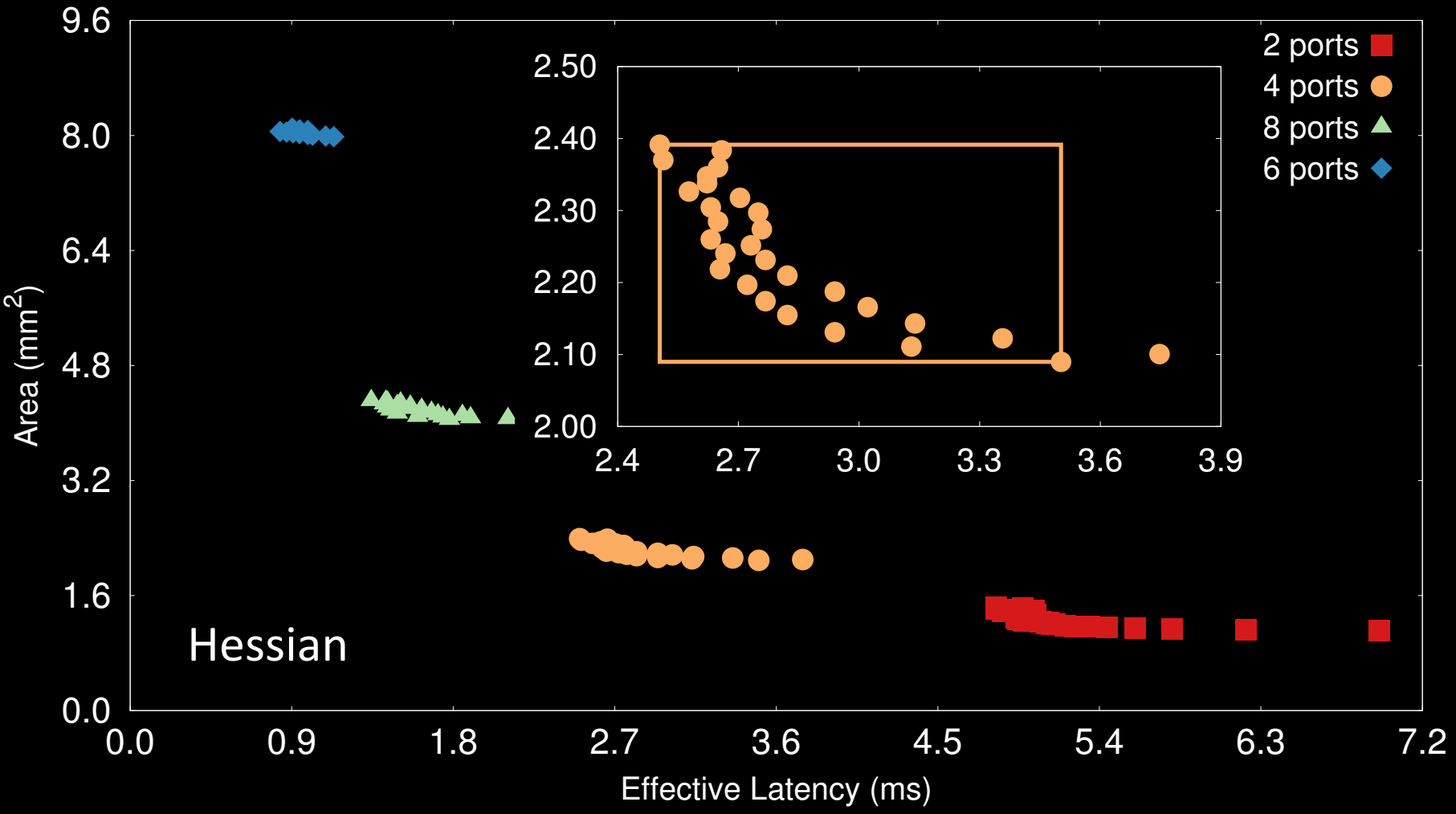
SystemC HLS-ready Specif.

RTL code for 32nm ASIC tech.



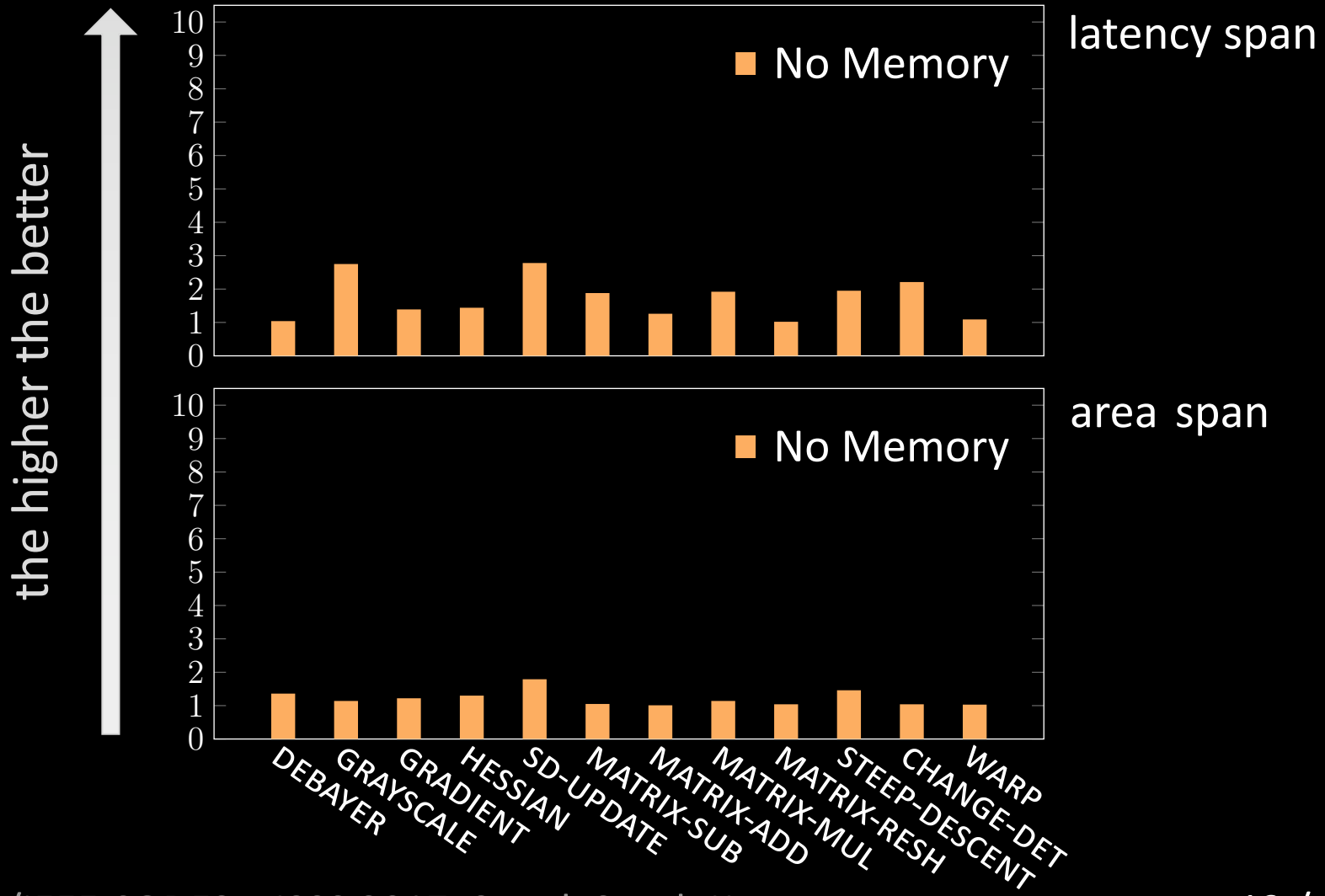
Experimental Results

Component Characterization



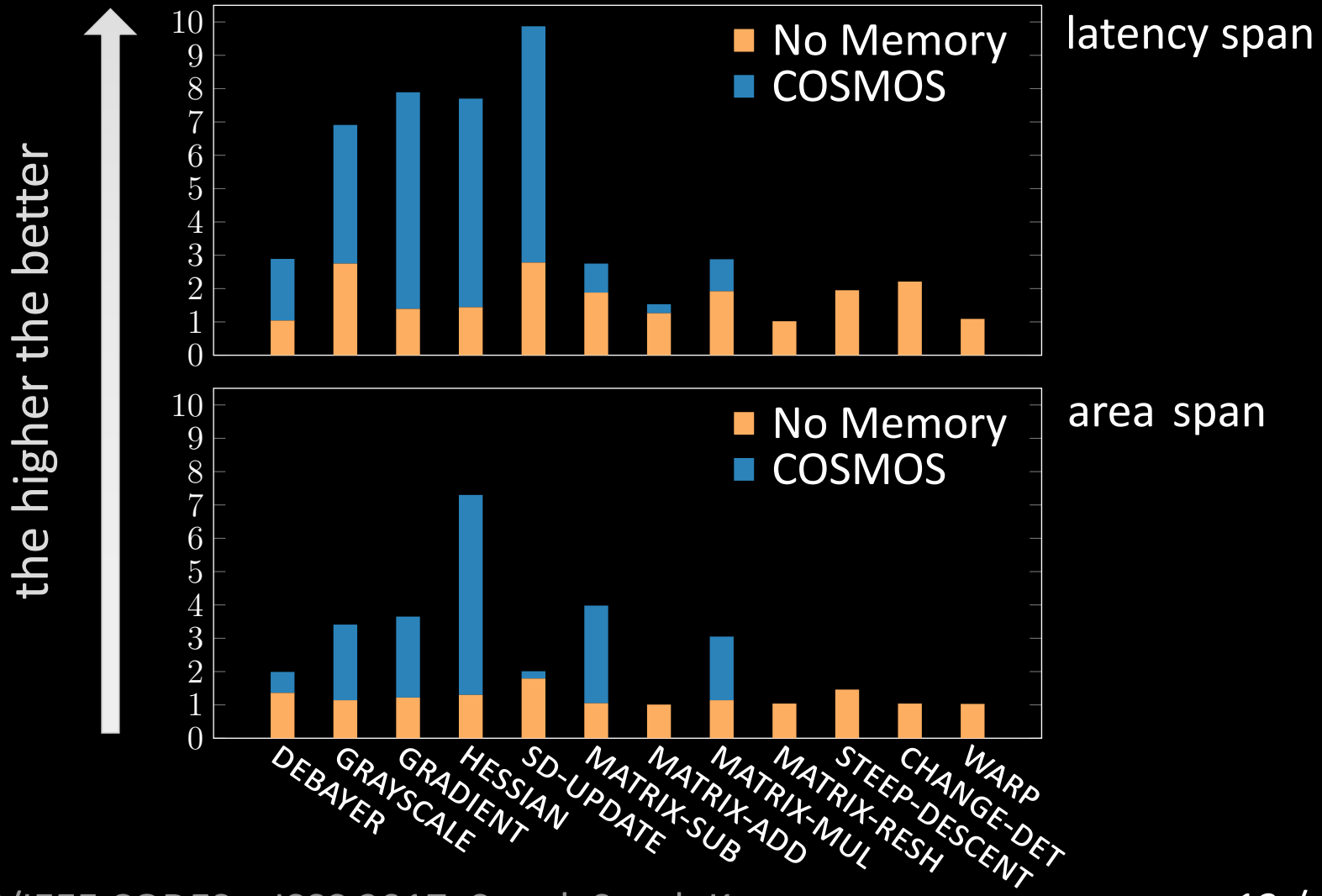
Experimental Results

Component Characterization



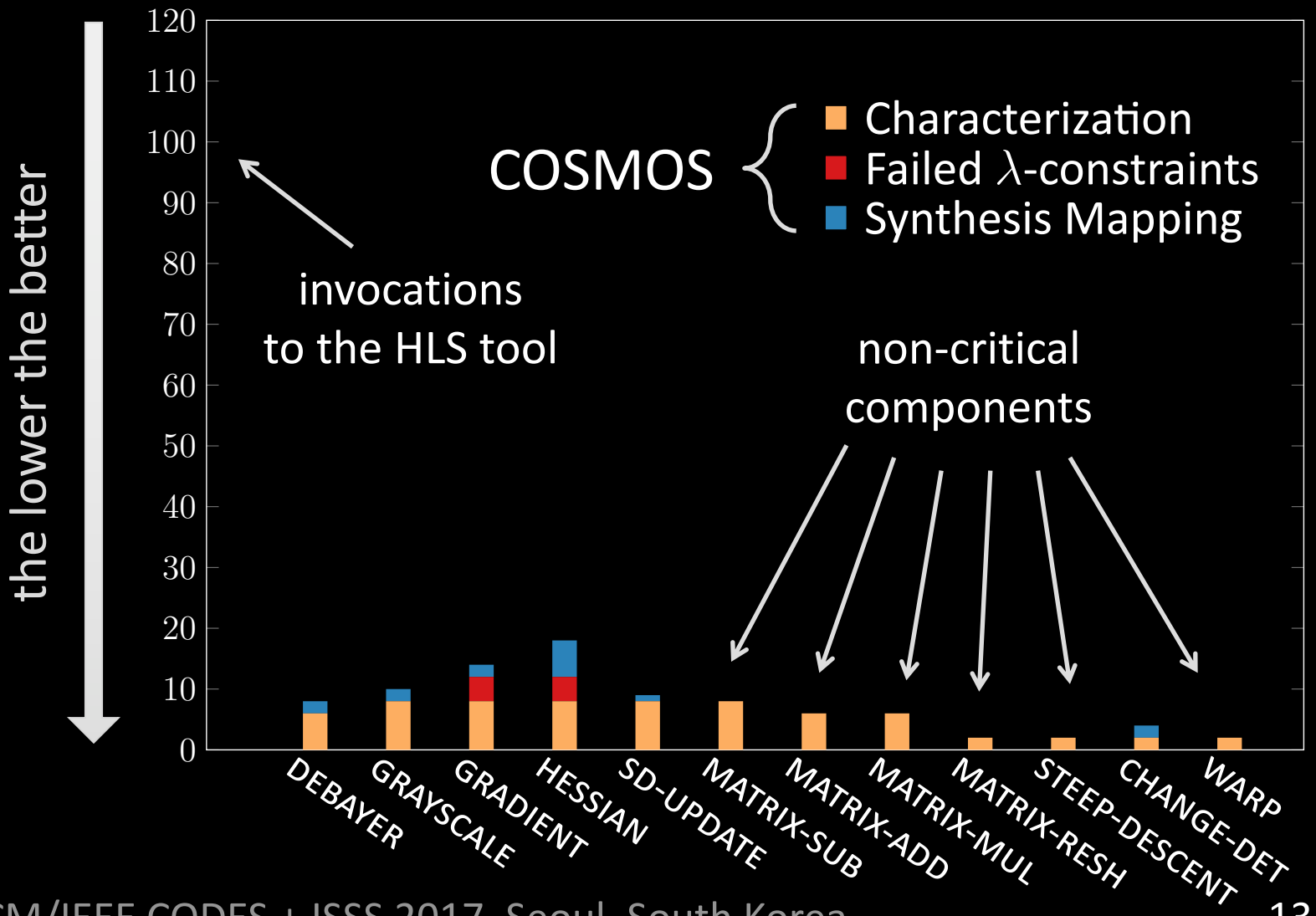
Experimental Results

Component Characterization



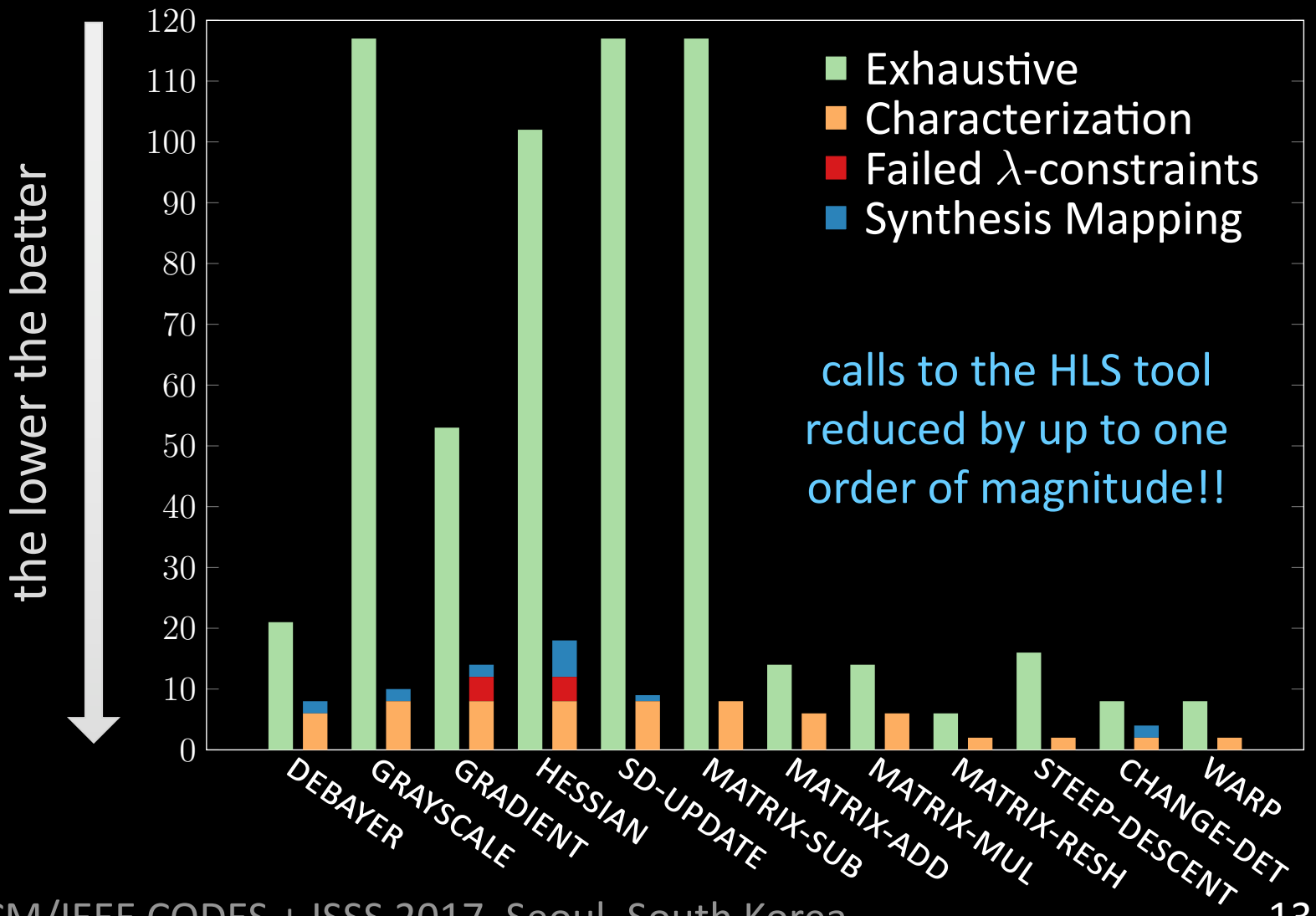
Experimental Results

Design-Space Exploration (Efficiency)



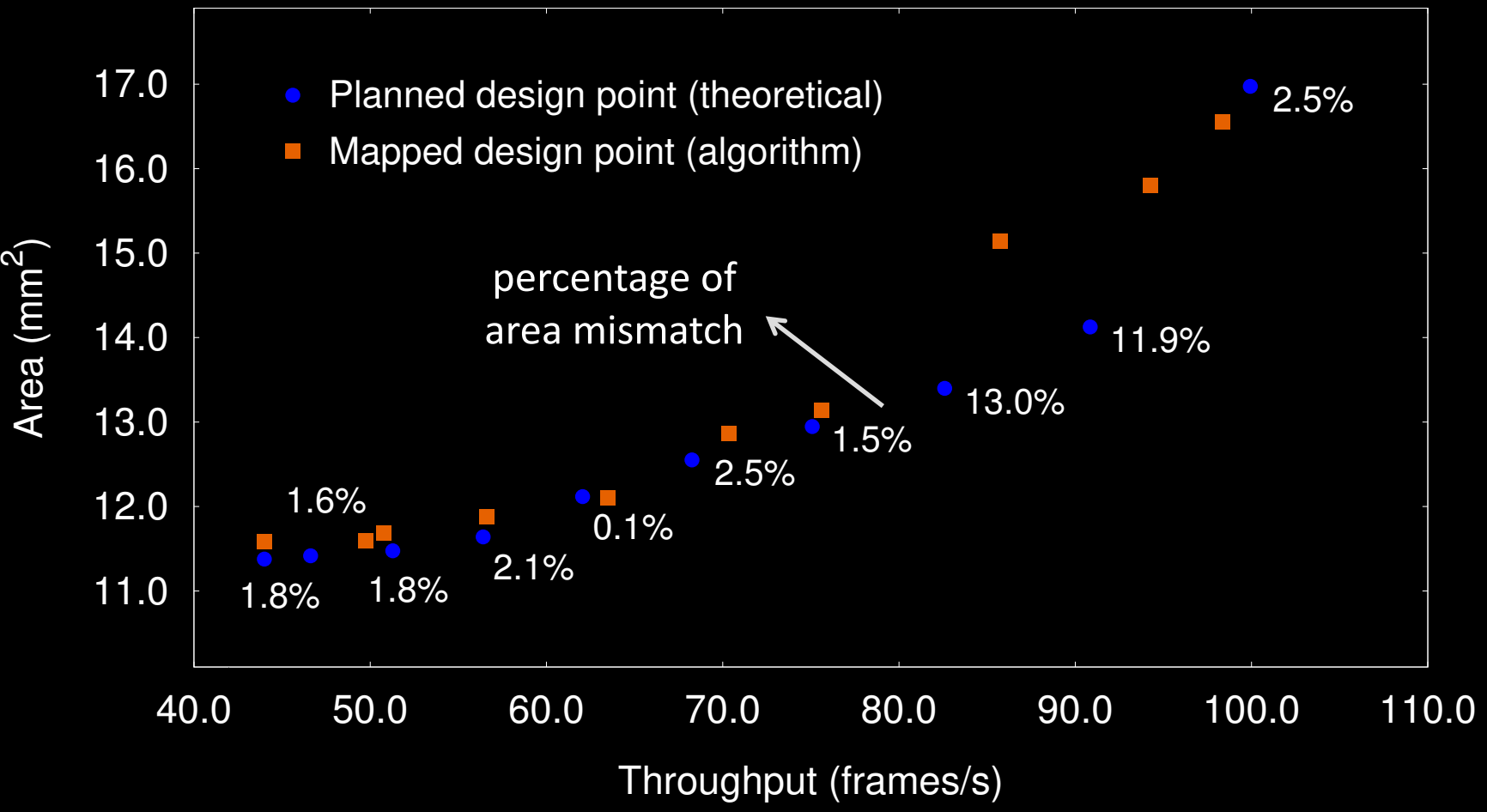
Experimental Results

Design-Space Exploration (Efficiency)



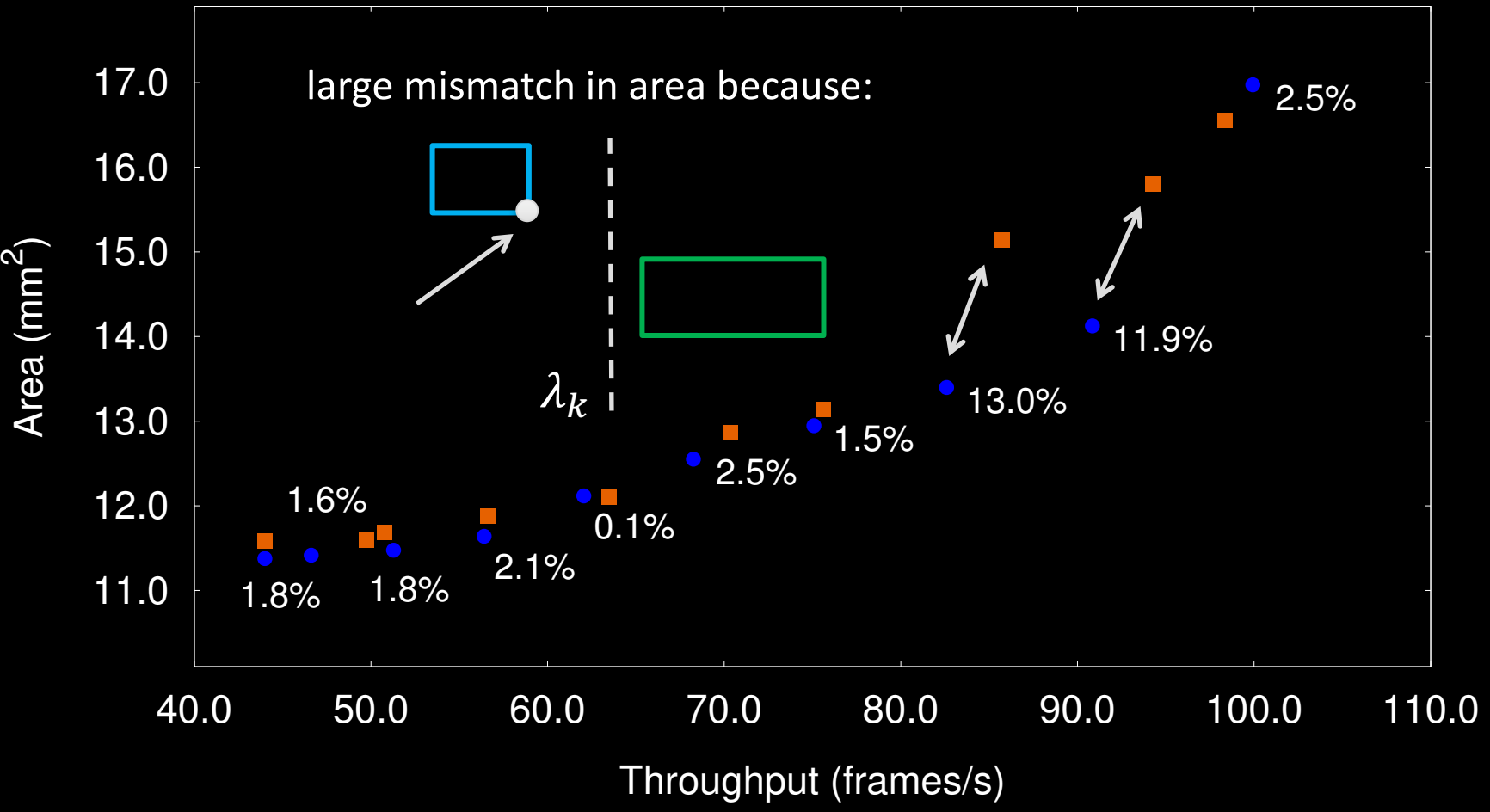
Experimental Results

Design-Space Exploration (Accuracy)



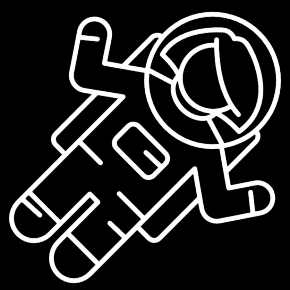
Experimental Results

Design-Space Exploration (Accuracy)



Concluding Remarks

- We presented **COSMOS**, an automatic methodology for design-space exploration (DSE) of accelerators that coordinates HLS and memory generator tools



Concluding Remarks

- We presented **COSMOS**, an automatic methodology for design-space exploration (DSE) of accelerators that coordinates HLS and memory generator tools

1. COSMOS guarantees a richer DSE compared to the methods that do not consider the accelerator PLMs

Concluding Remarks

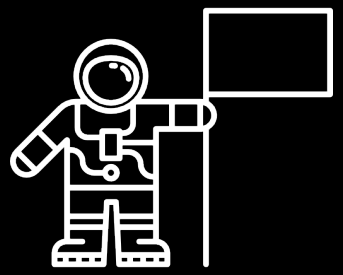
- We presented **COSMOS**, an automatic methodology for design-space exploration (DSE) of accelerators that coordinates HLS and memory generator tools

1. COSMOS guarantees a richer DSE compared to the methods that do not consider the accelerator PLMs
2. COSMOS guarantees a much faster DSE compared to exhaustive methods in case of complex accelerators

Concluding Remarks

- We presented **COSMOS**, an automatic methodology for design-space exploration (DSE) of accelerators that coordinates HLS and memory generator tools

1. COSMOS guarantees a richer DSE compared to the methods that do not consider the accelerator PLMs
2. COSMOS guarantees a much faster DSE compared to exhaustive methods in case of complex accelerators
3. COSMOS is a scalable methodology for DSE



COSMOS: Coordination of High-Level Synthesis and Memory Optimization for Hardware Accelerators

Questions?

Speaker: Luca Piccolboni
Columbia University, NY