DaeMon: Architectural Support for Efficient Data Movement in Fully Disaggregated Memory Systems

Christina Giannoula

Kailong Huang, Jonathan Tang, Nectarios Koziris,
Georgios Goumas, Zeshan Chishti, Nandita Vijaykumar

Executive Summary

DaeMon

Problem:

- Efficient data movement support is a major system challenge for fully Disaggregated Systems (DSs)

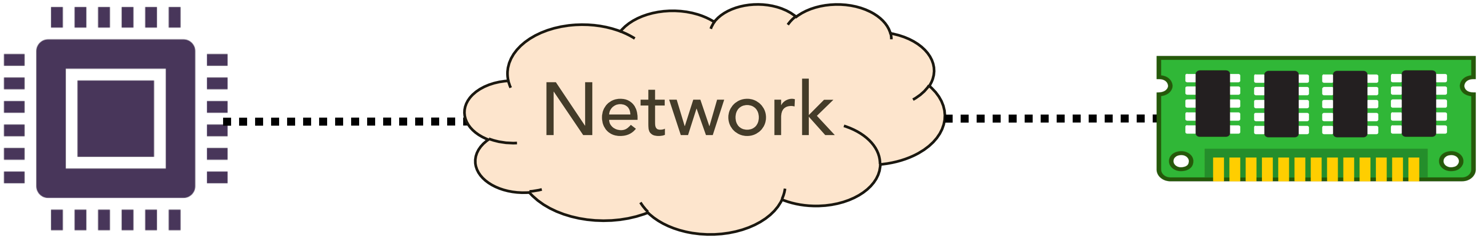
Contribution:

- DaeMon: the first adaptive data movement solution for fully DSs

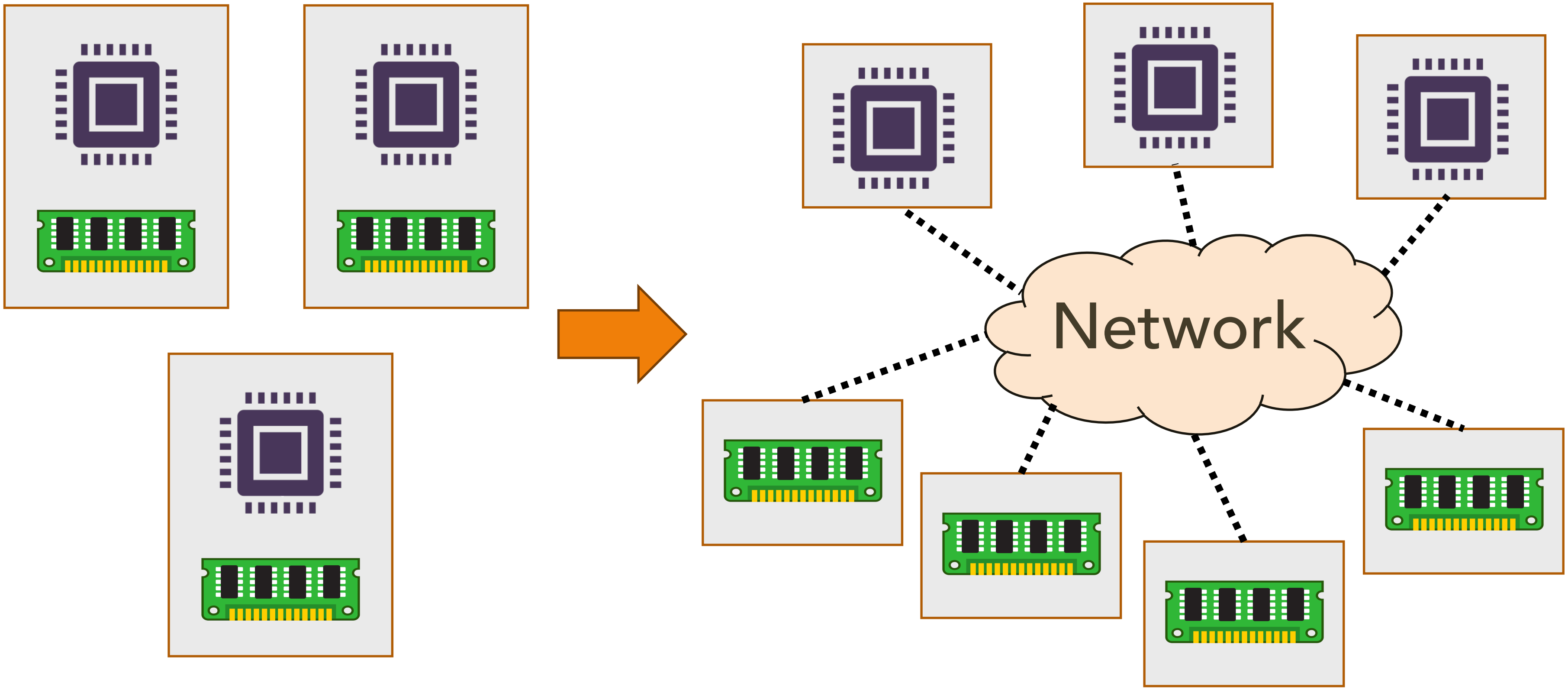
Key Results:

- DaeMon achieves 2.39x better performance and 3.06x lower data access costs over the widely-adopted scheme of moving data at page granularity

What is resource disaggregation?



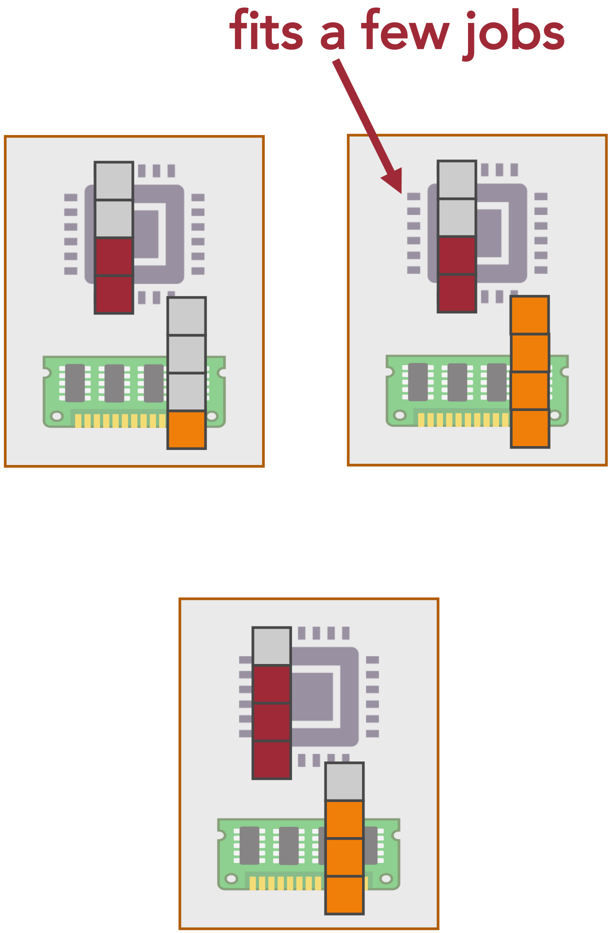
Monolithic vs Disaggregated Systems



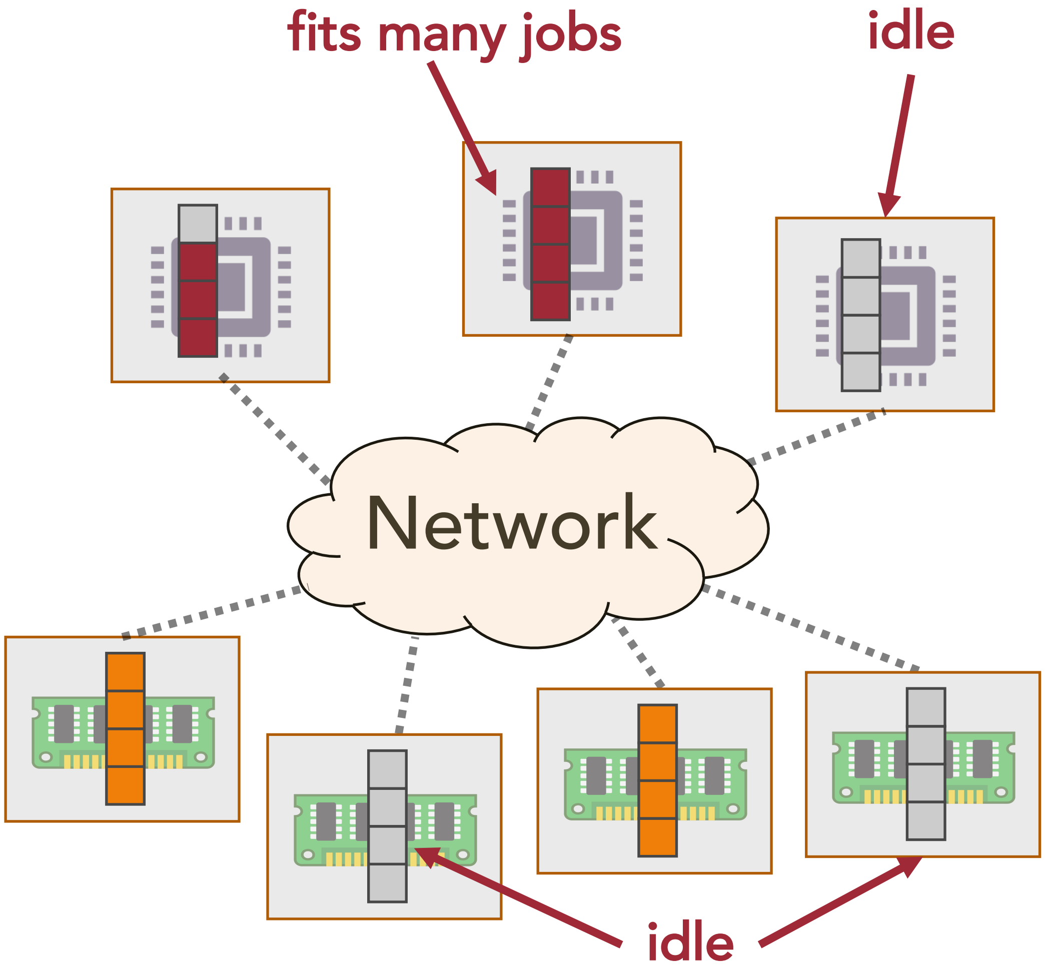
thanks to recent advances
in network technologies

Benefits of Fully Disaggregated Systems

- Resource Utilization



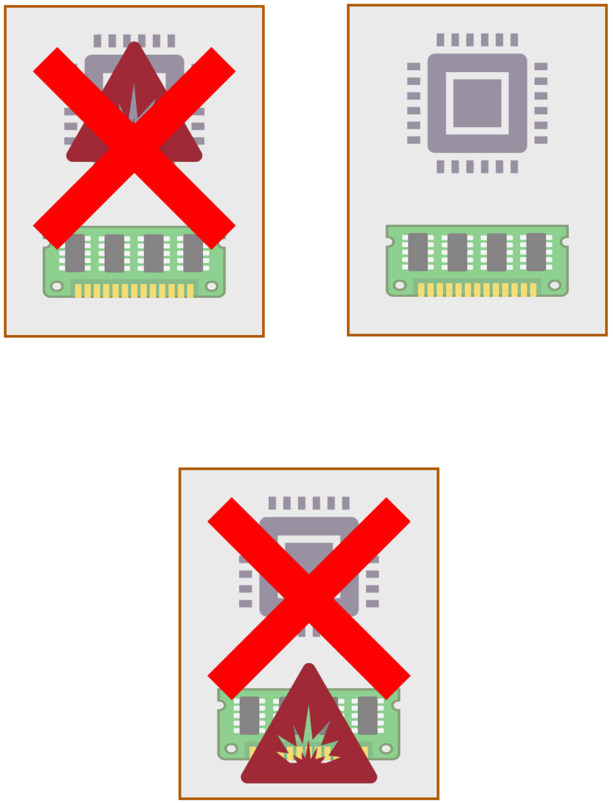
Monolithic System



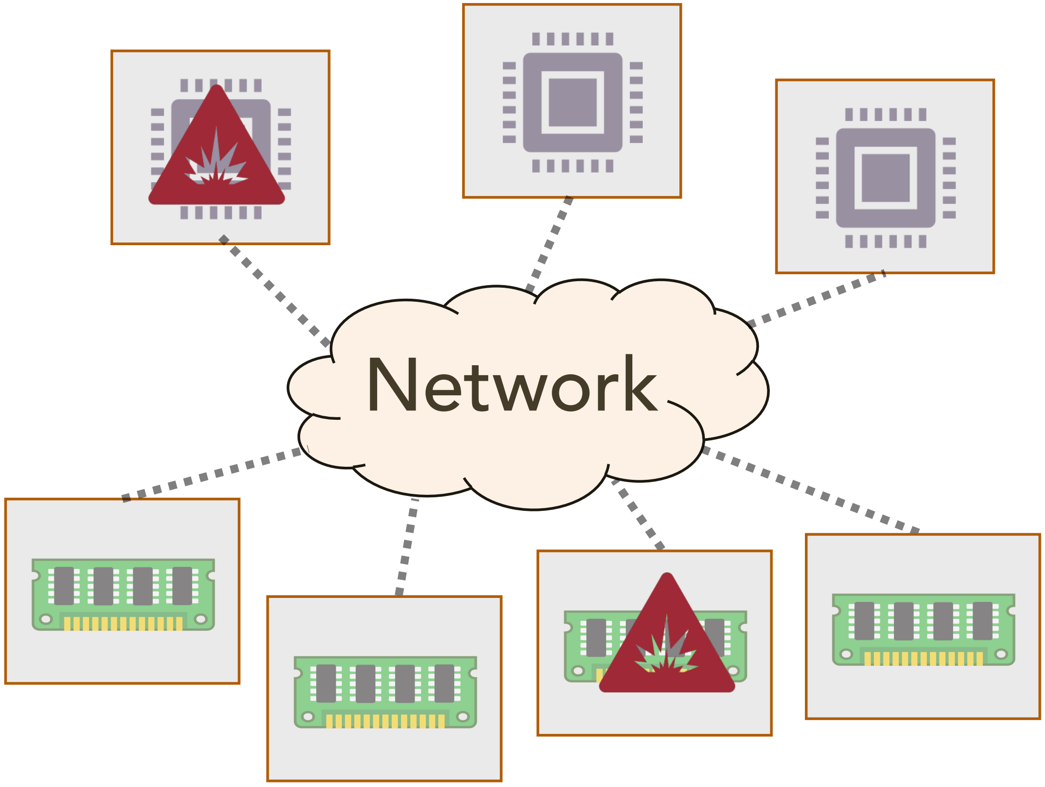
Disaggregated System

Benefits of Fully Disaggregated Systems

- Failure Handling



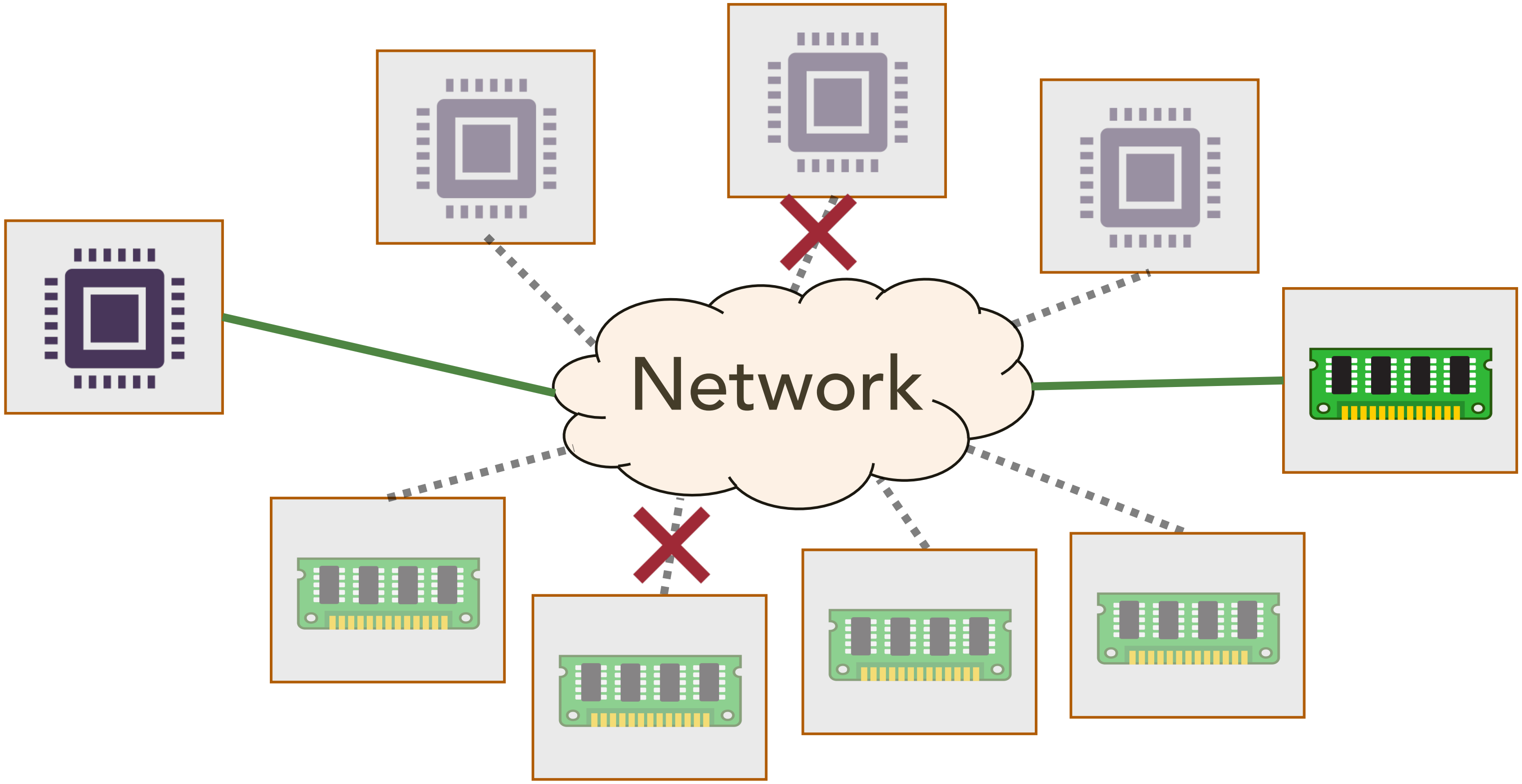
Monolithic System



Disaggregated System

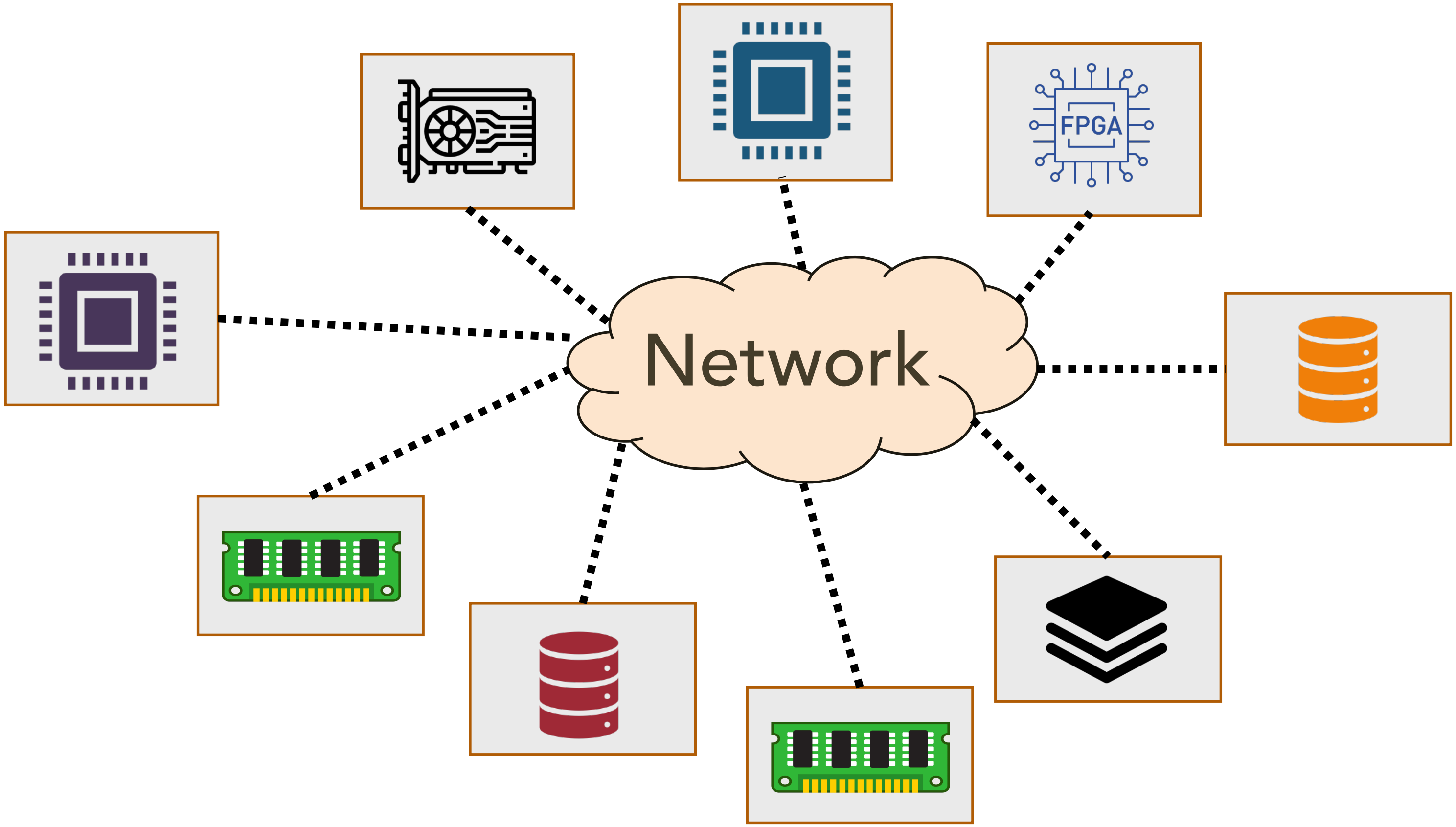
Benefits of Fully Disaggregated Systems

- Resource Scaling



Benefits of Fully Disaggregated Systems

- Heterogeneity



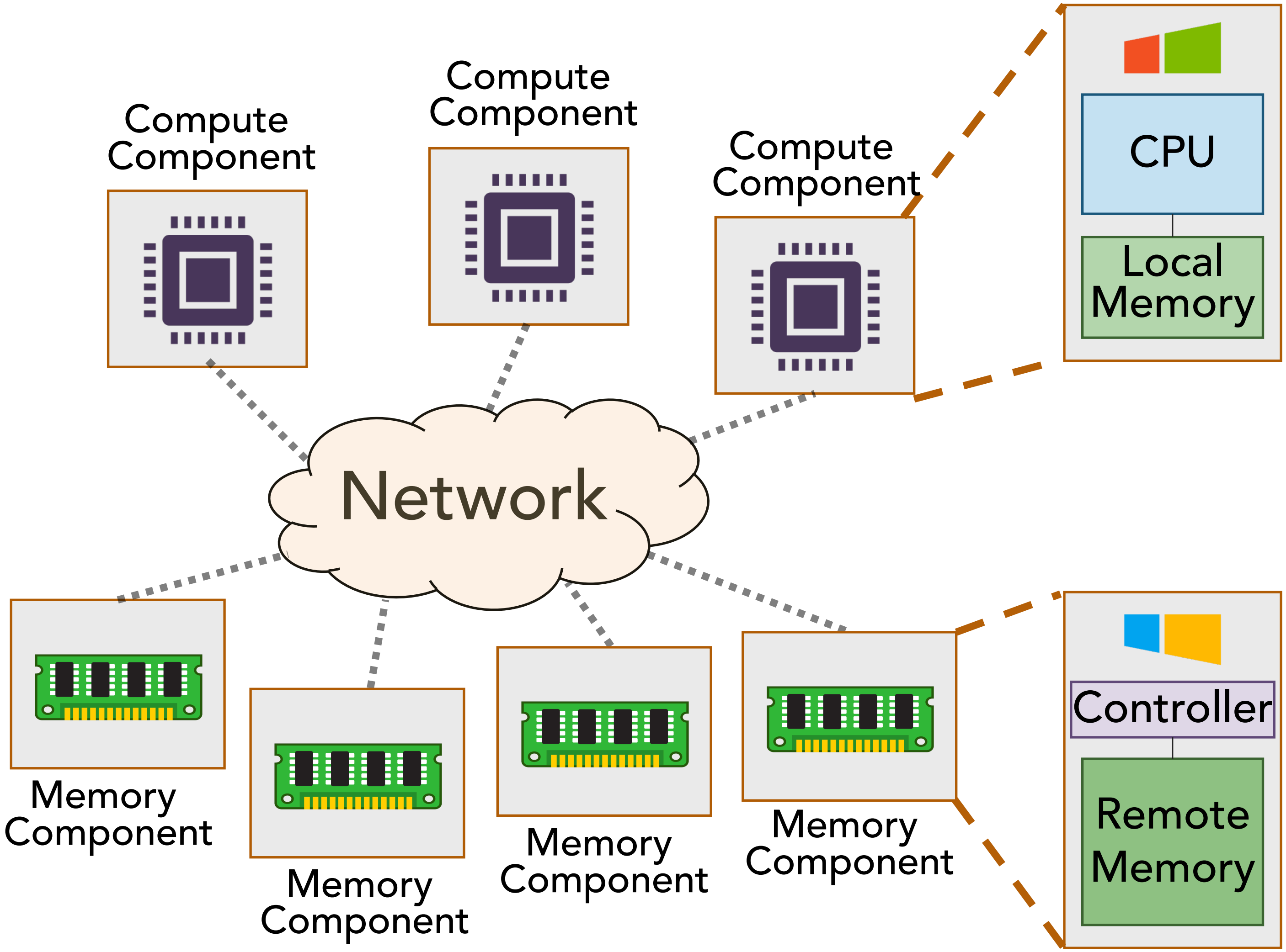
many different types of hardware devices over the network

Benefits of Fully Disaggregated Systems

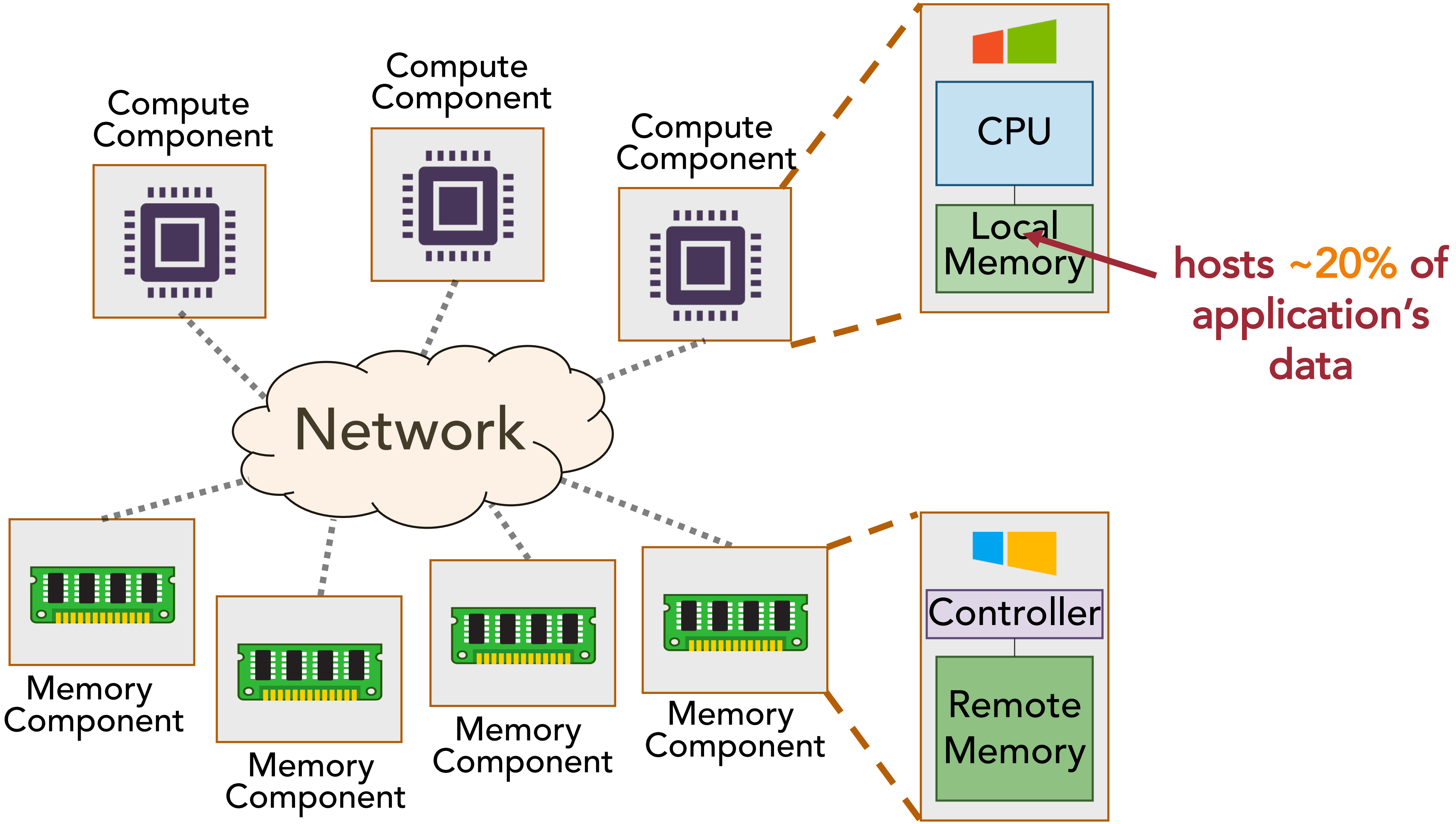
- Resource Utilization
- Failure Handling
- Resource Scaling
- Heterogeneity

Disaggregated systems can significantly decrease data center costs

Baseline Disaggregated System

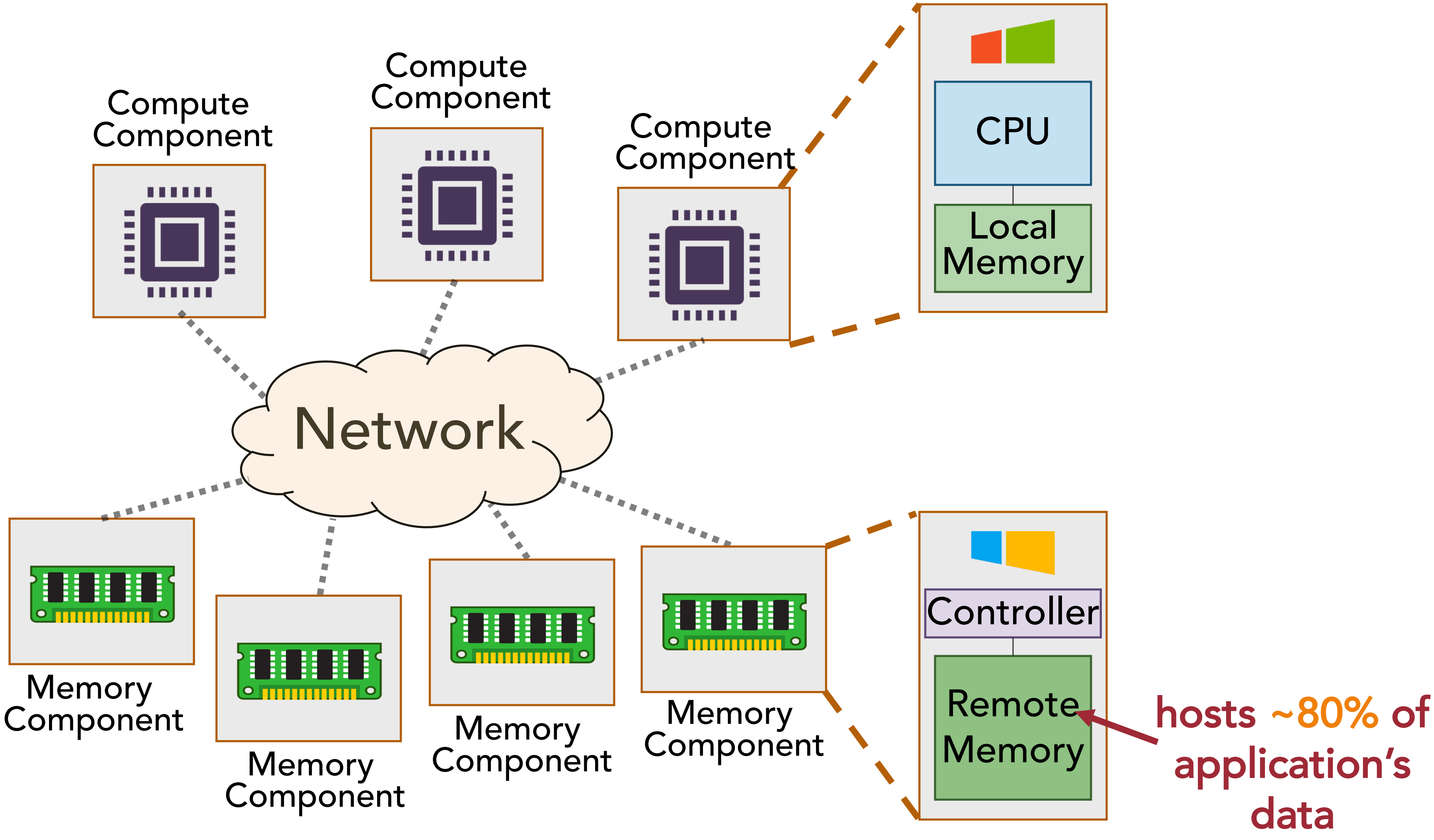


Baseline Disaggregated System

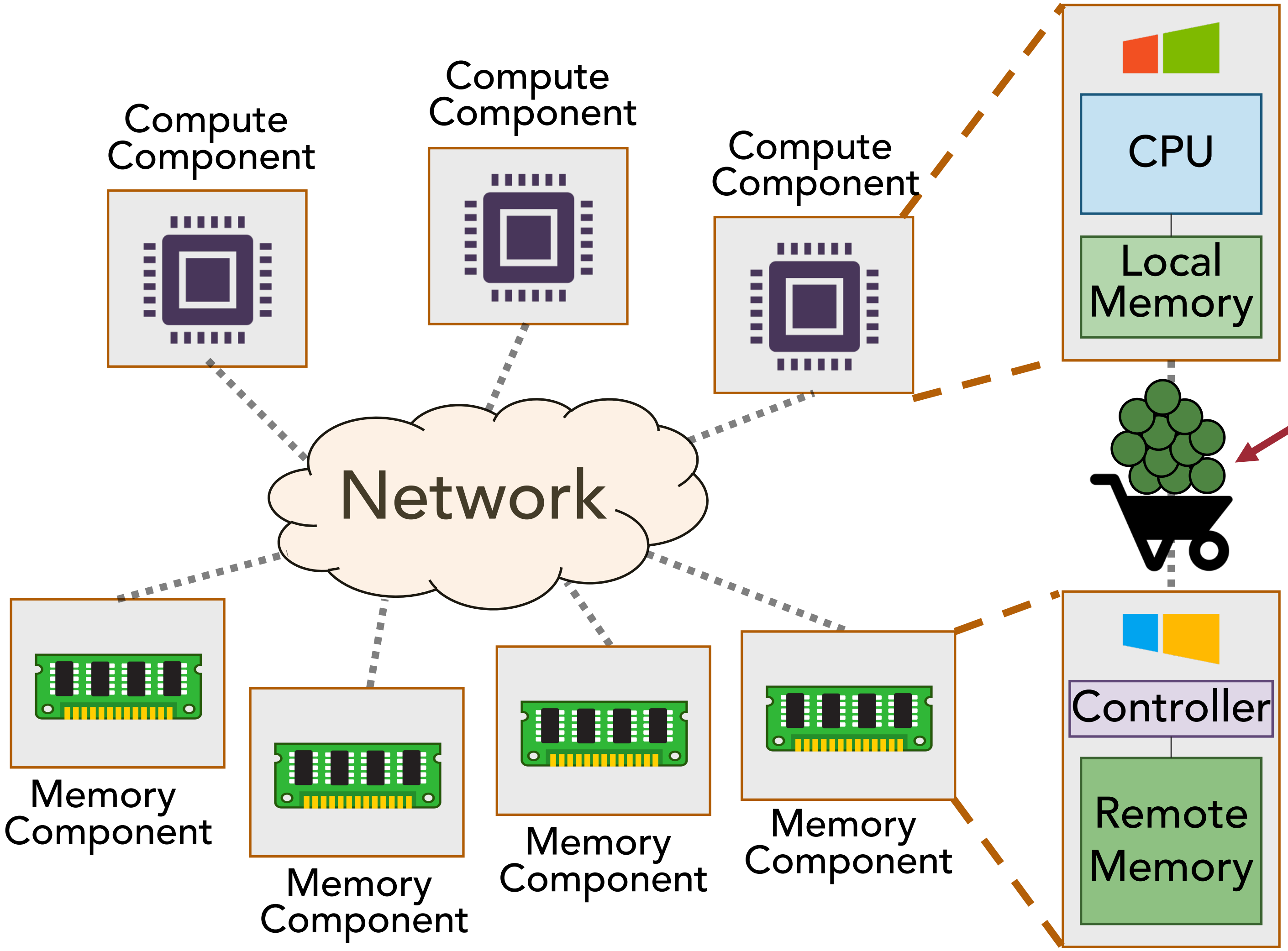


hosts ~20% of application's data

Baseline Disaggregated System

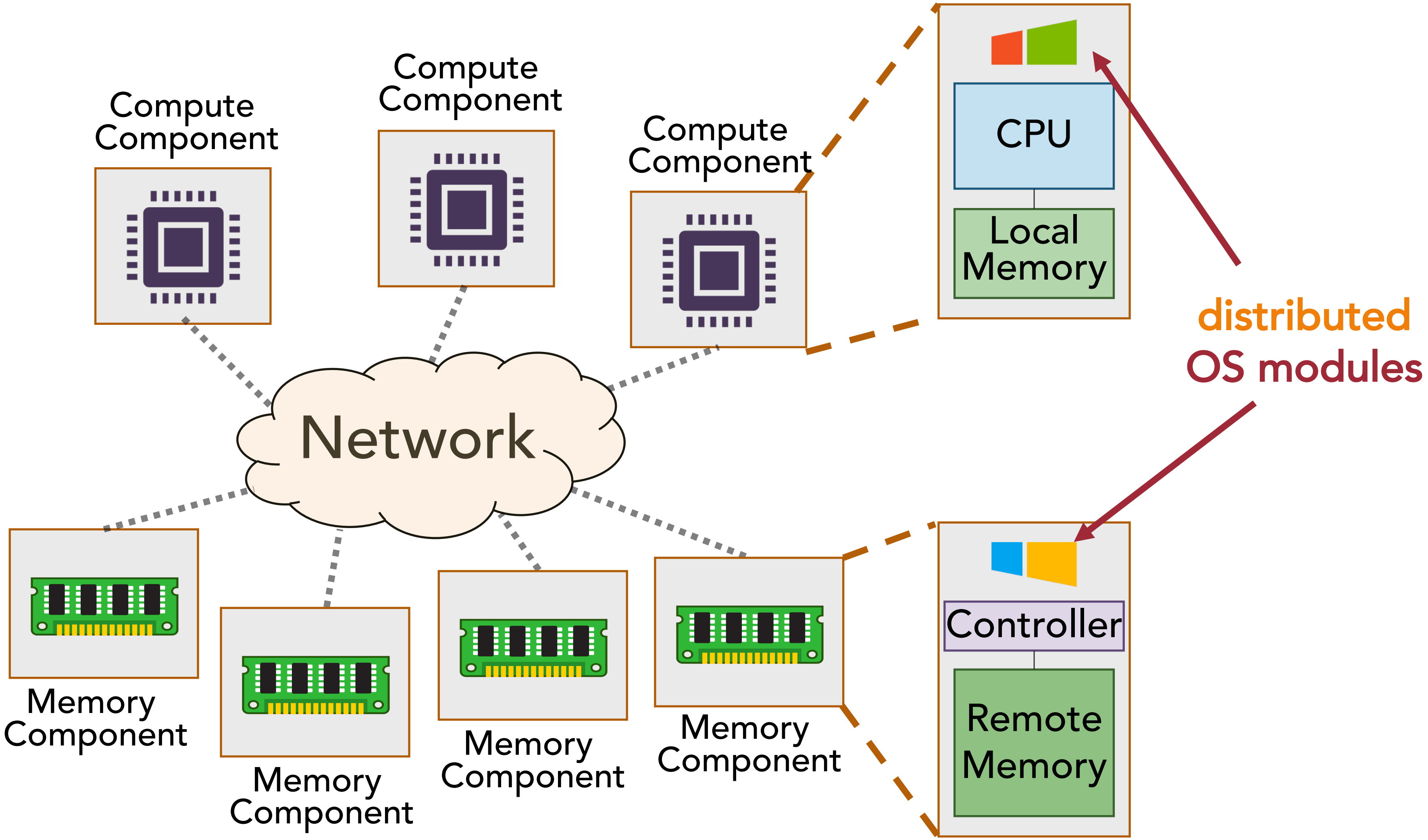


Baseline Disaggregated System

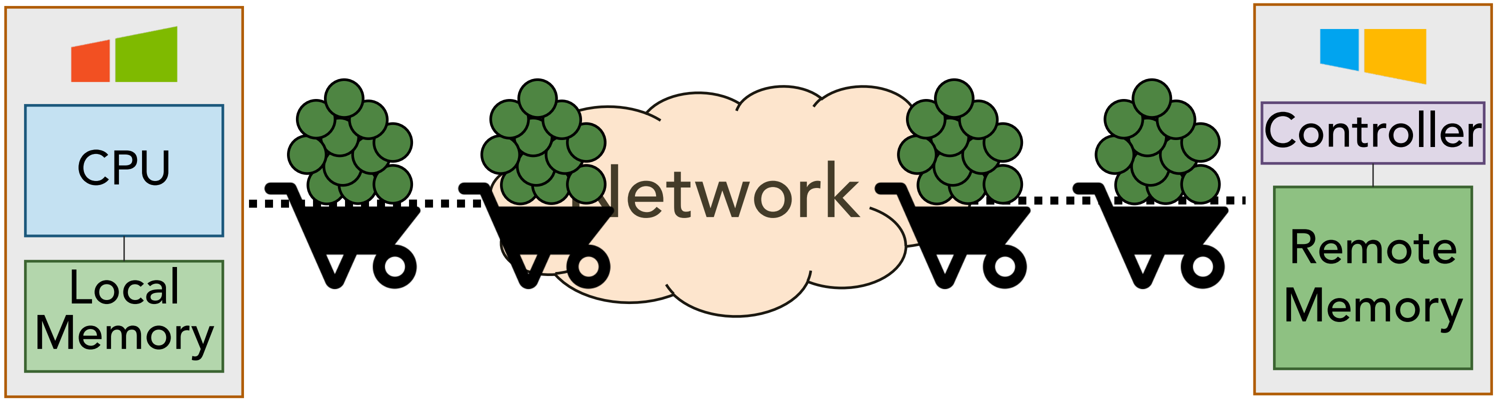


data is typically moved at page granularity

Baseline Disaggregated System

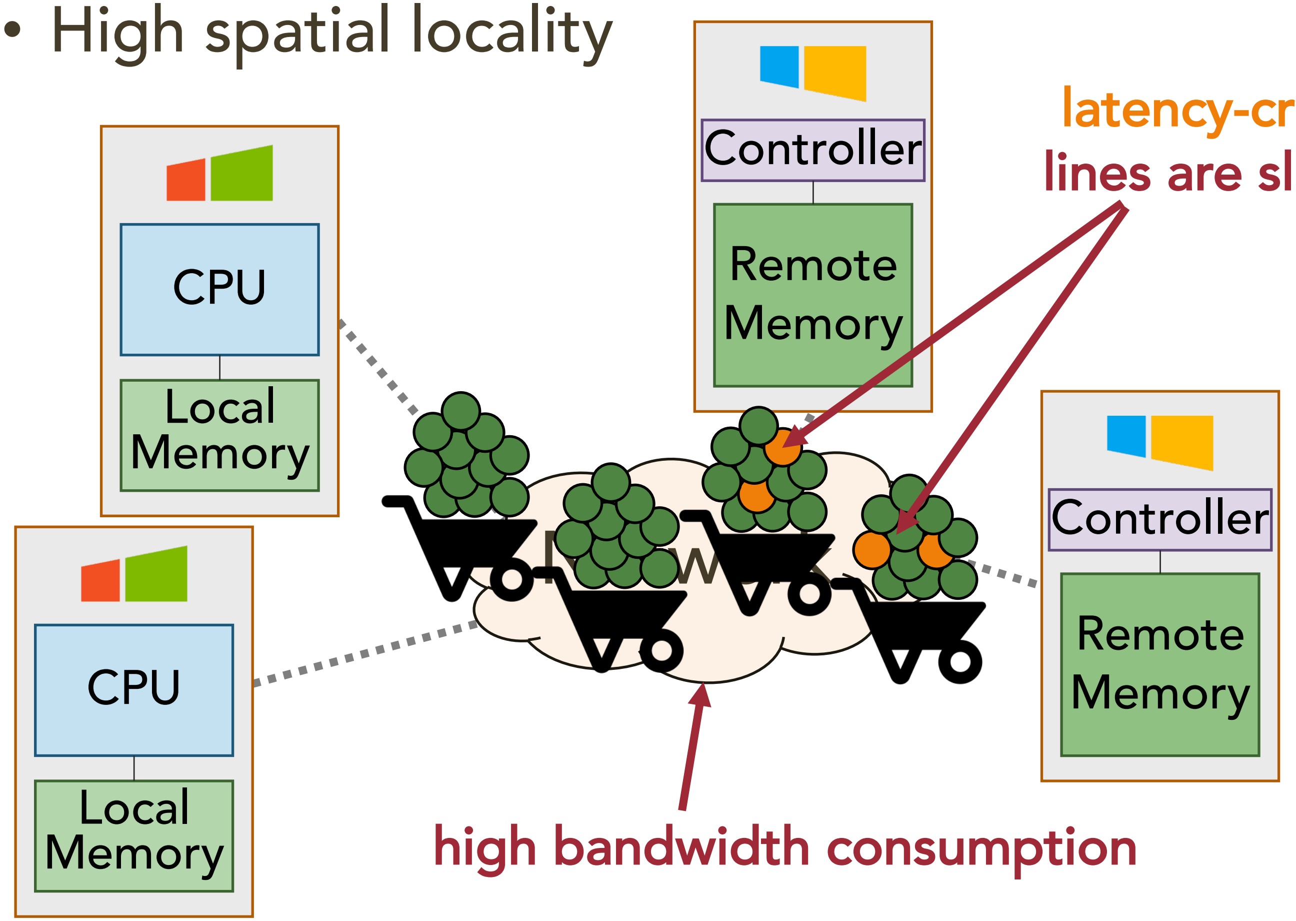


Why is data movement challenging?



#1: Coarse-Grained Data Migrations

- Page granularity (e.g., **4KB**) data migrations:
  - Software transparency
  - Low metadata overheads
  - High spatial locality

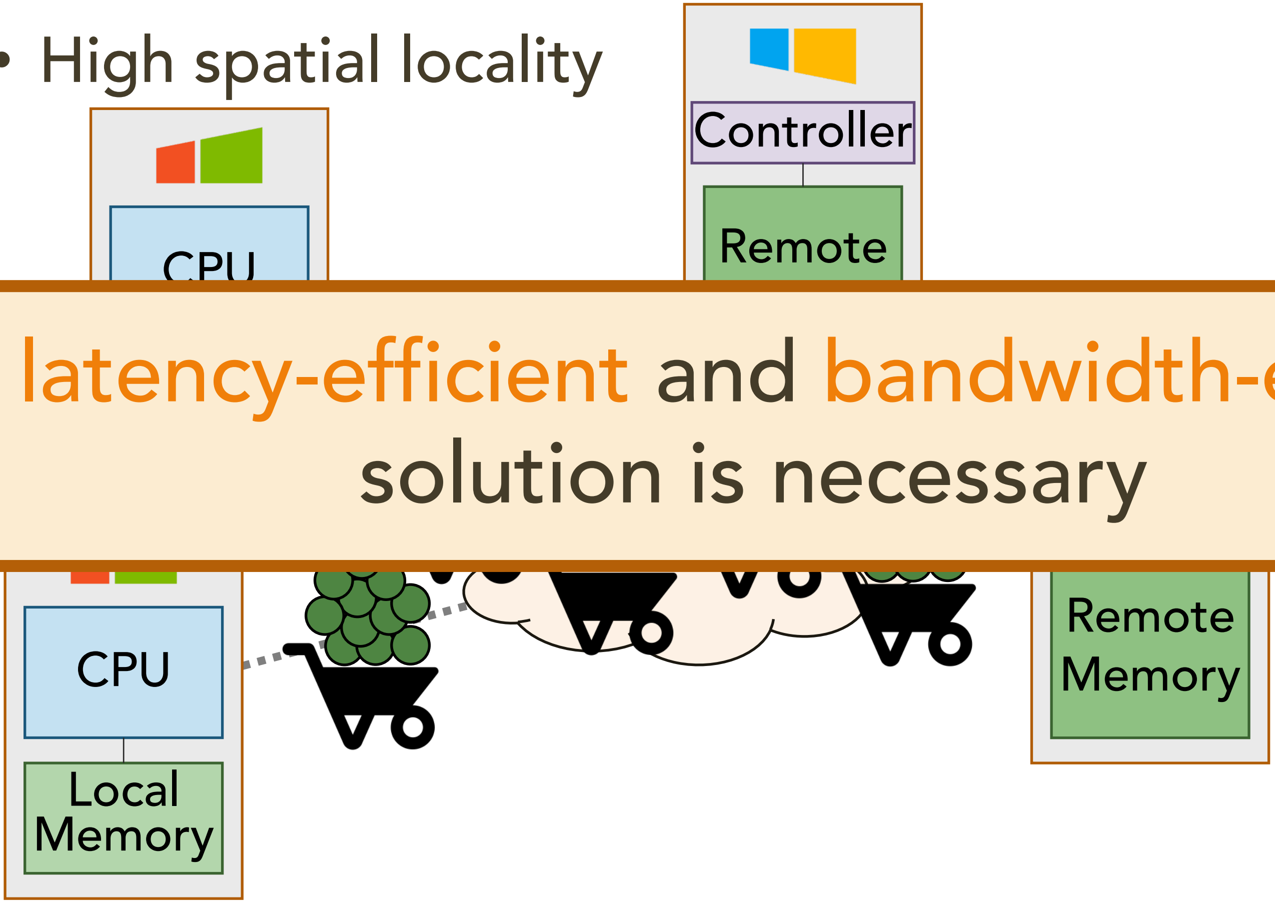


latency-critical cache lines are slowed down

high bandwidth consumption

#1: Coarse-Grained Data Migrations

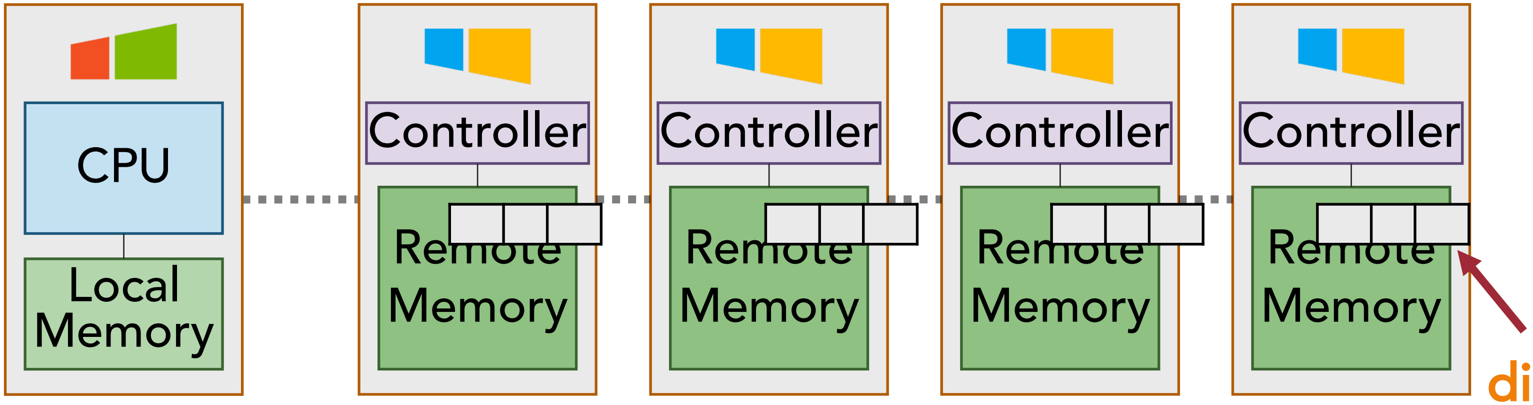
- Page granularity (e.g., **4KB**) data migrations:
  - Software transparency
  - Low metadata overheads
  - High spatial locality



A latency-efficient and bandwidth-efficient solution is necessary

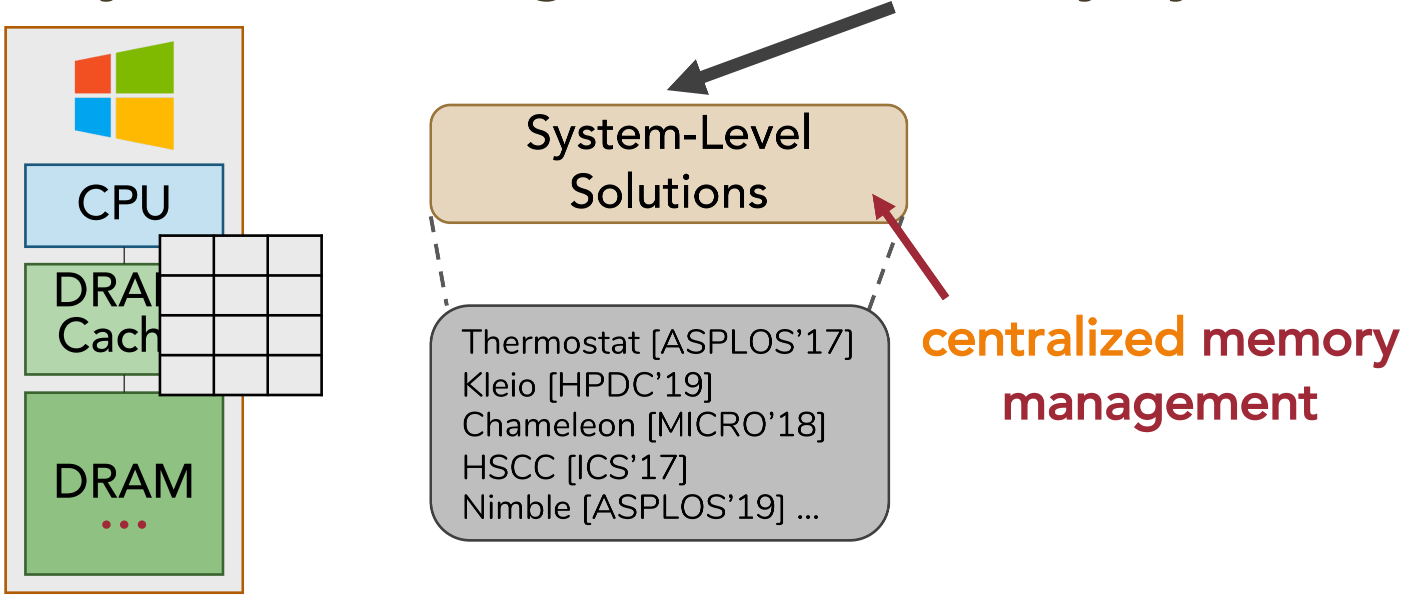
#2: Non-Conventional System Design

- Disaggregated systems are **not monolithic**



distributed memory management

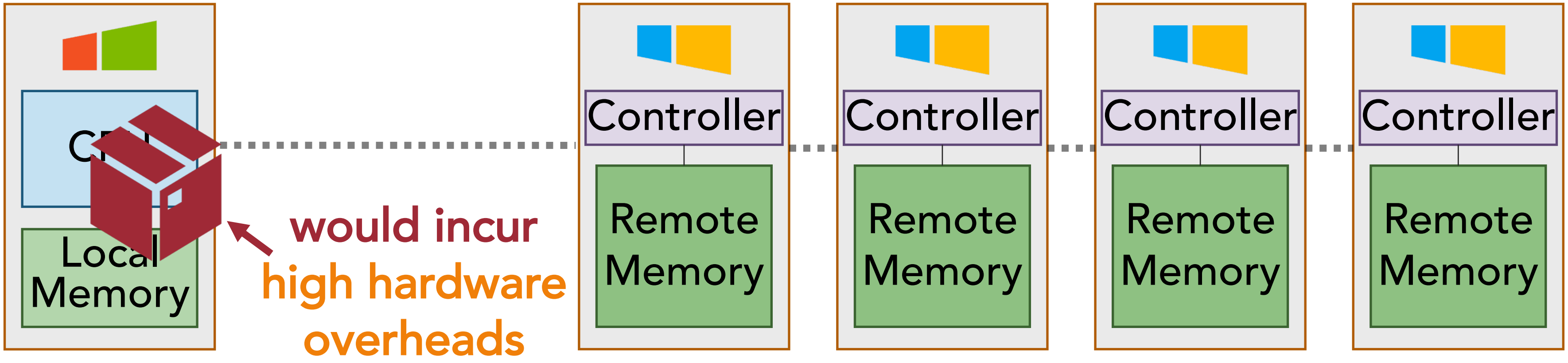
- Hybrid/heterogeneous memory systems:



Thermostat [ASPLOS'17]
Kleio [HPDC'19]
Chameleon [MICRO'18]
HSCC [ICS'17]
Nimble [ASPLOS'19] ...

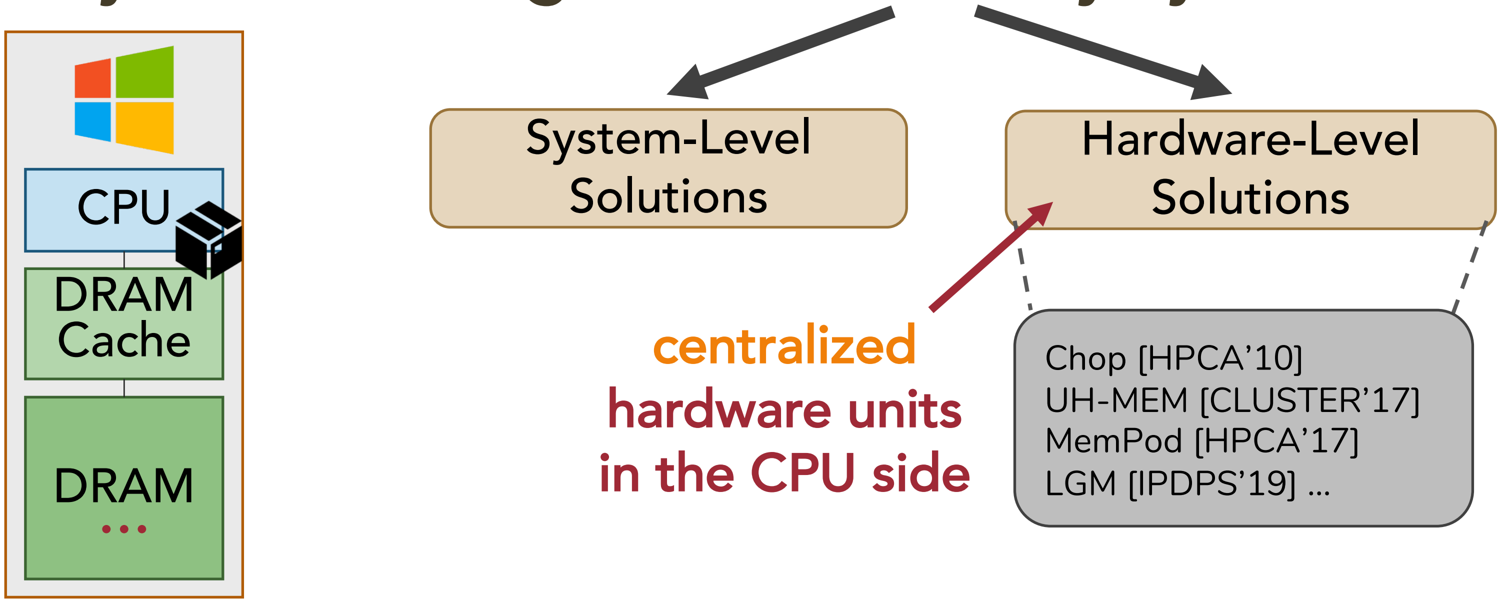
#2: Non-Conventional System Design

- Disaggregated systems are **not monolithic**



would incur **high hardware overheads**

- Hybrid/heterogeneous memory systems:

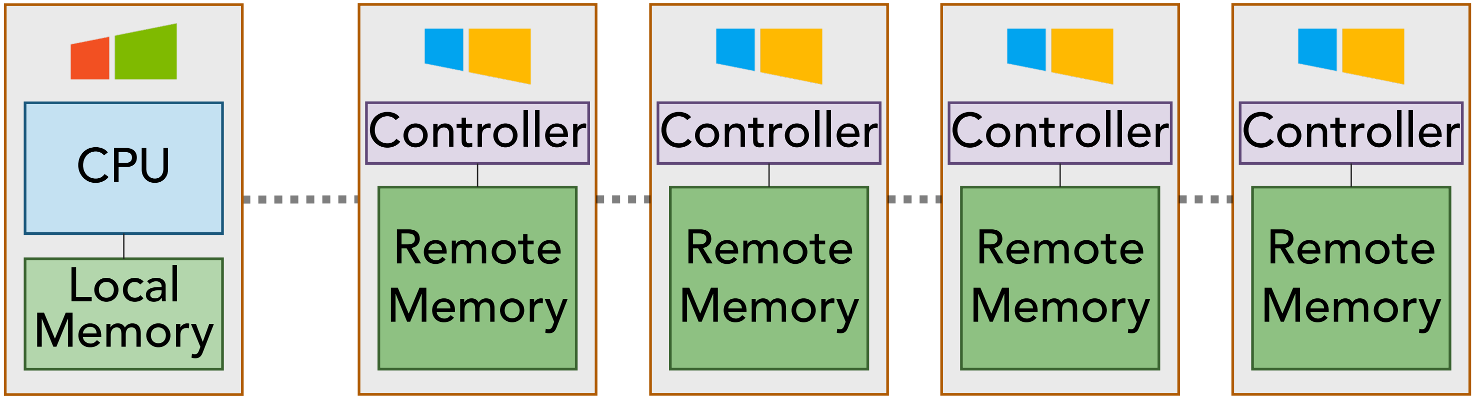


centralized hardware units in the CPU side

Chop [HPCA'10]
UH-MEM [CLUSTER'17]
MemPod [HPCA'17]
LGM [IPDPS'19] ...

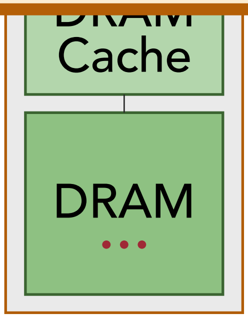
#2: Non-Conventional System Design

- Disaggregated systems are **not monolithic**



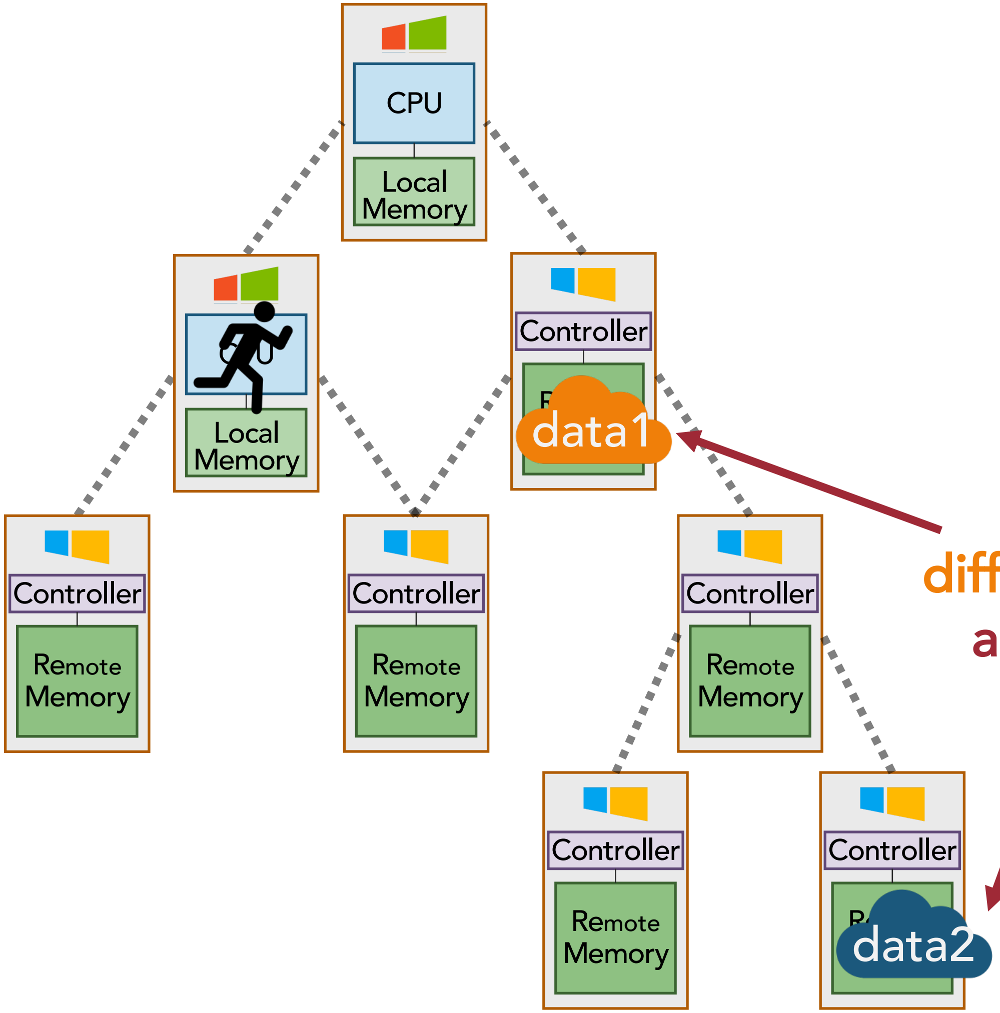
- Hybrid/heterogeneous memory systems:

Prior solutions are not suitable or efficient for disaggregated memory systems



#3: Variability in Data Access Latencies

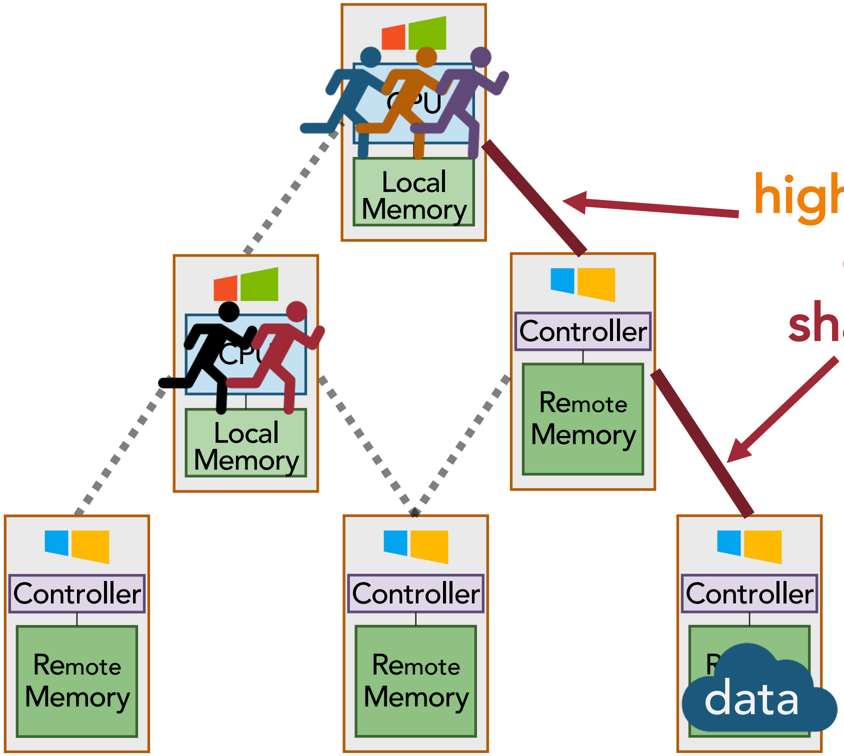
- Data access latencies depend:
  - Location of the remote memory component



different locations for application’s data

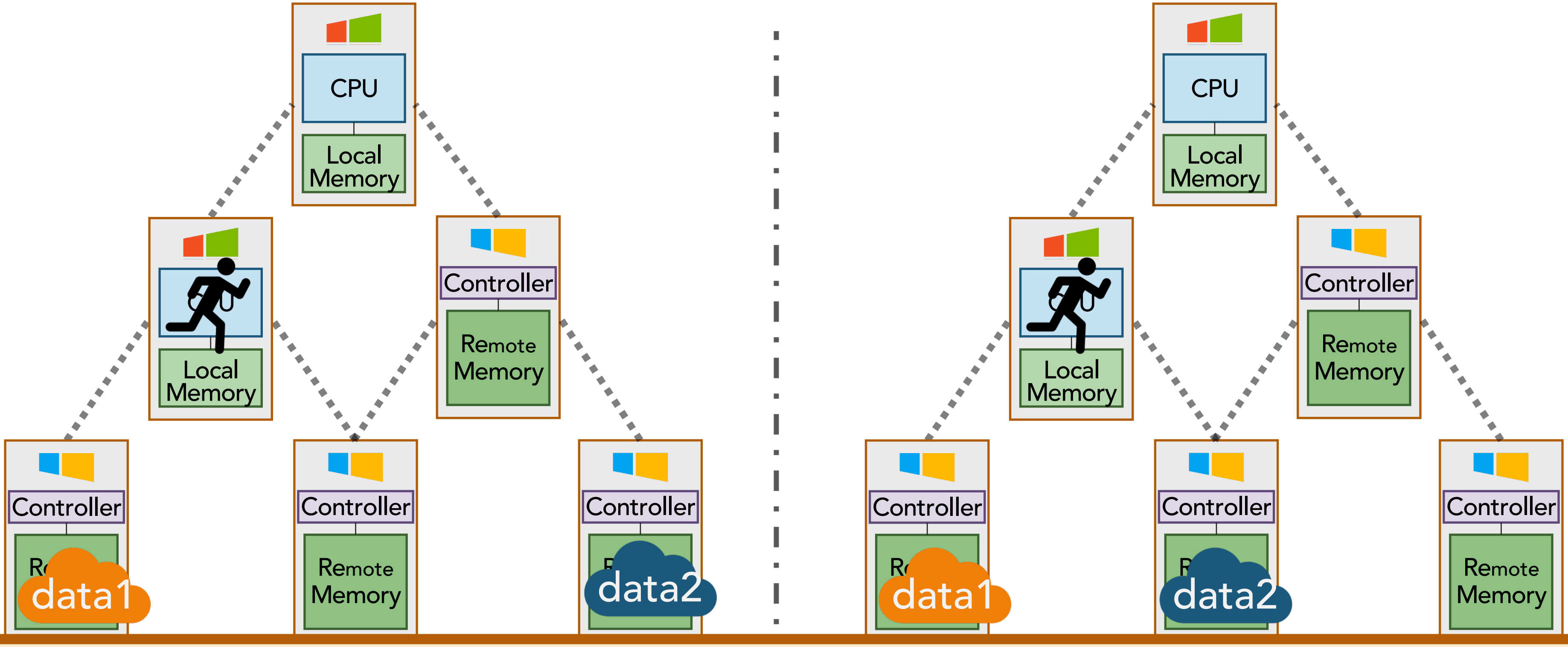
#3: Variability in Data Access Latencies

- Data access latencies depend:
  - Location of the remote memory component
  - Network contention



high contention due to concurrent jobs sharing the network

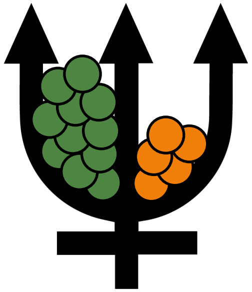
#3: Variability in Data Access Latencies



A robust solution to variability in data access latencies is necessary

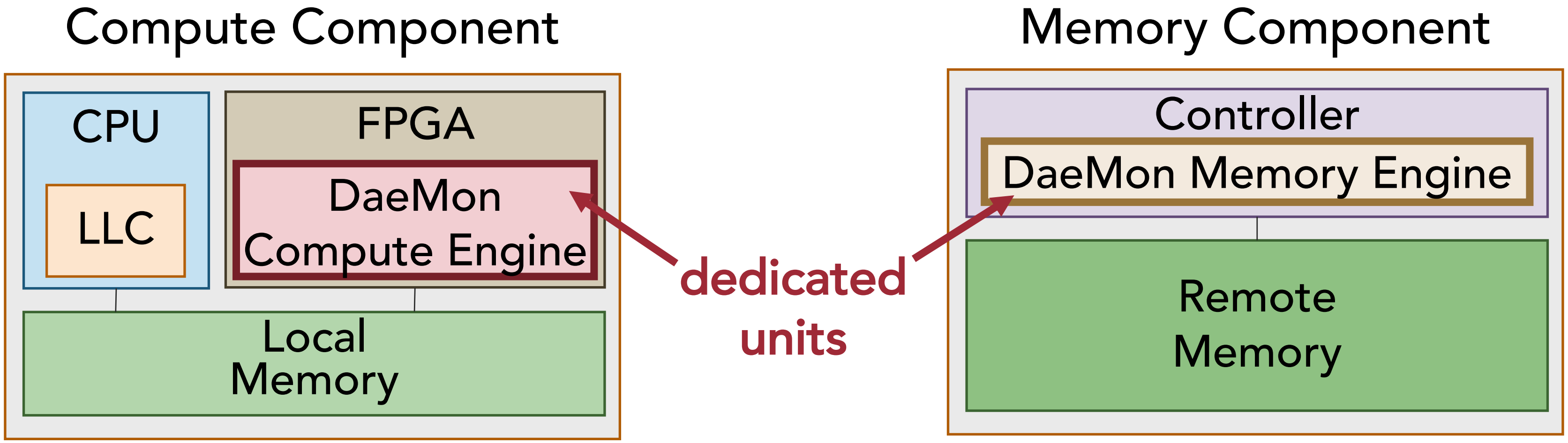
data placements can vary during runtime or between multiple executions

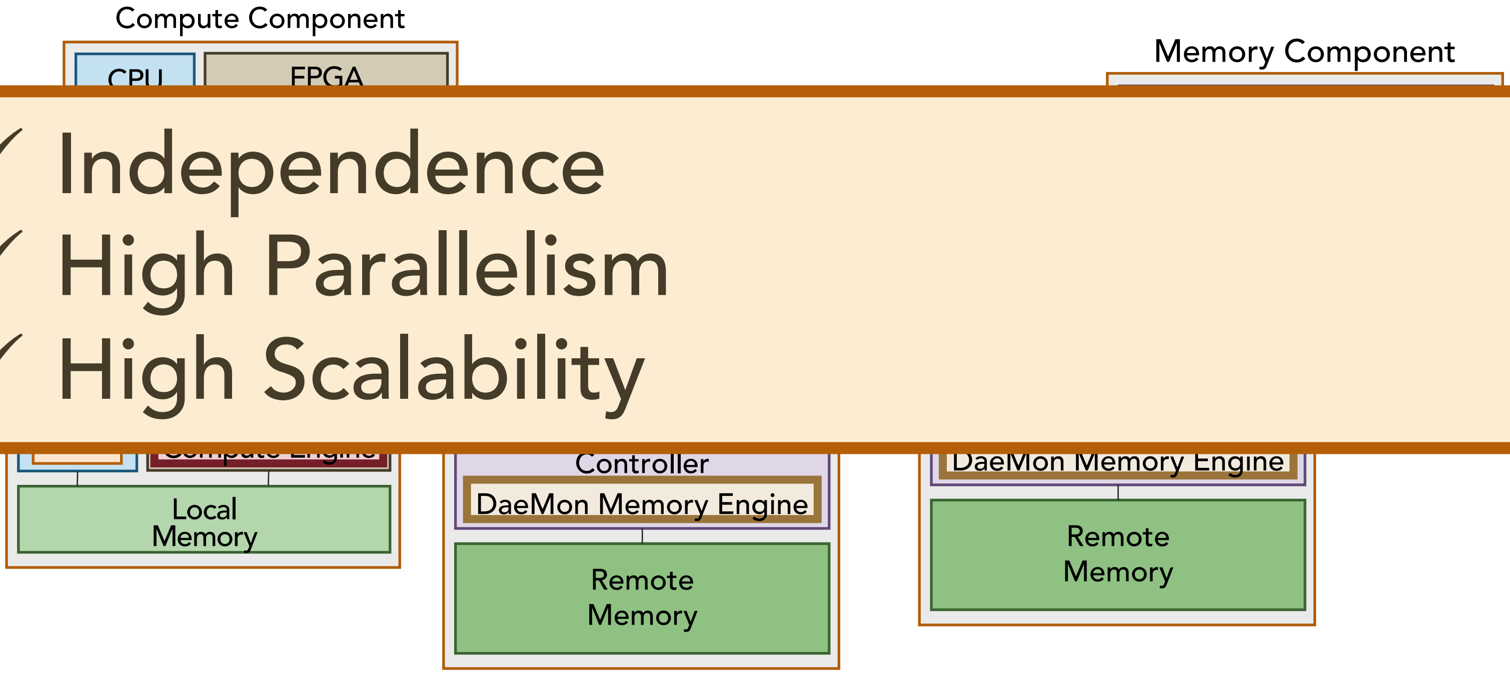
How can we build an efficient solution?



DaeMon
[Sigmetrics'23]

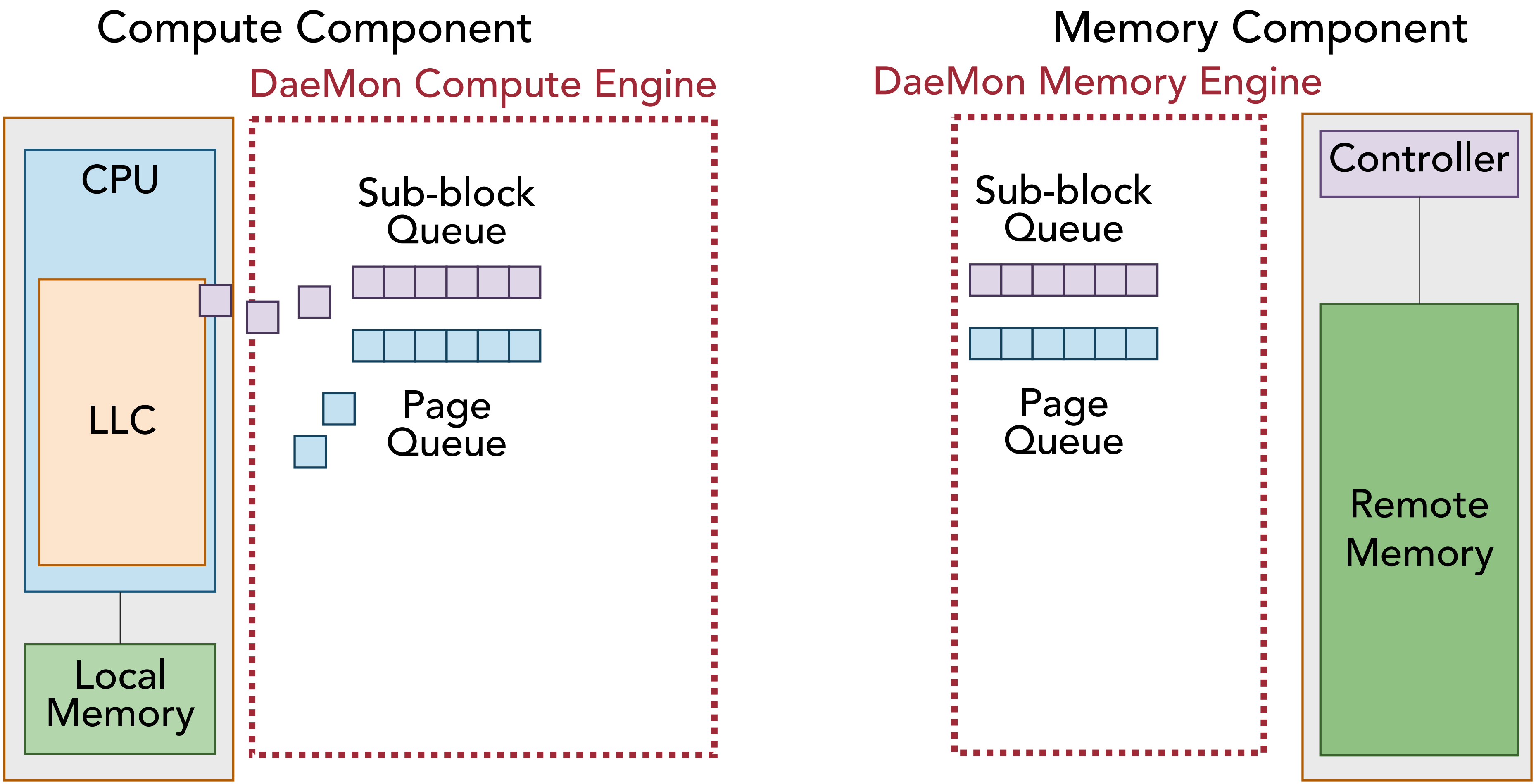
1. Disaggregated Hardware Support



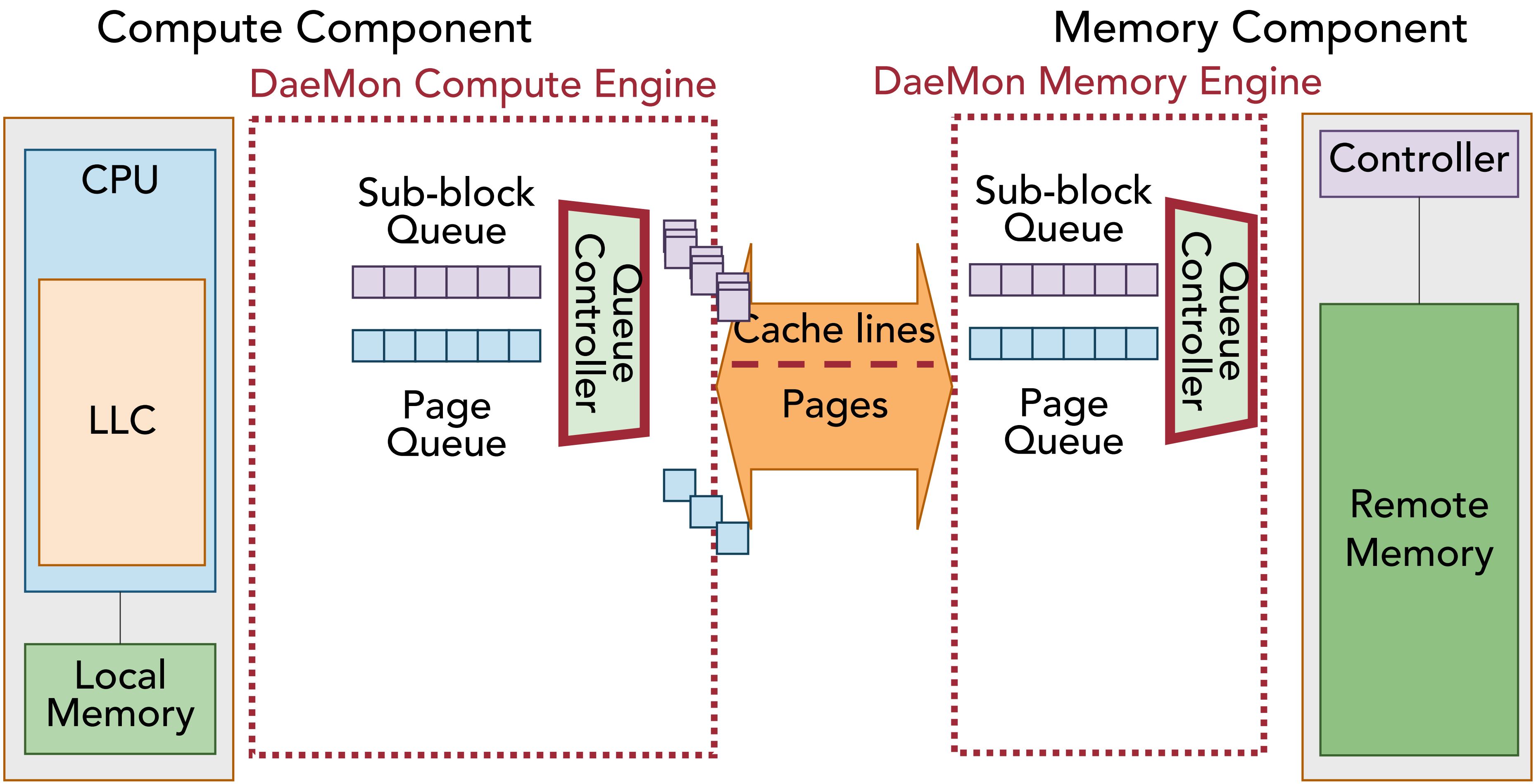


- ✓ Independence
- ✓ High Parallelism
- ✓ High Scalability

2. Multiple Granularity Data Movement

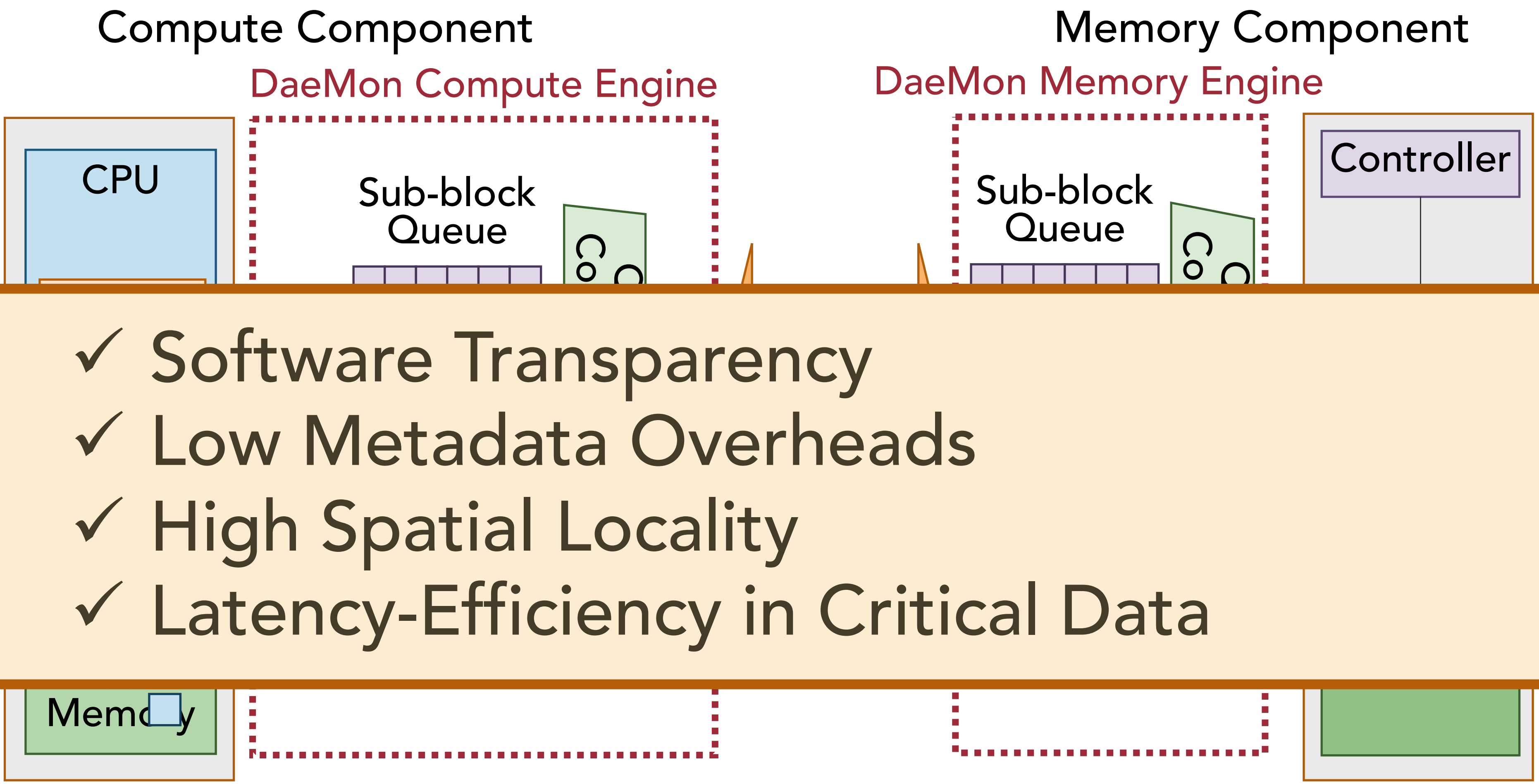


2. Multiple Granularity Data Movement



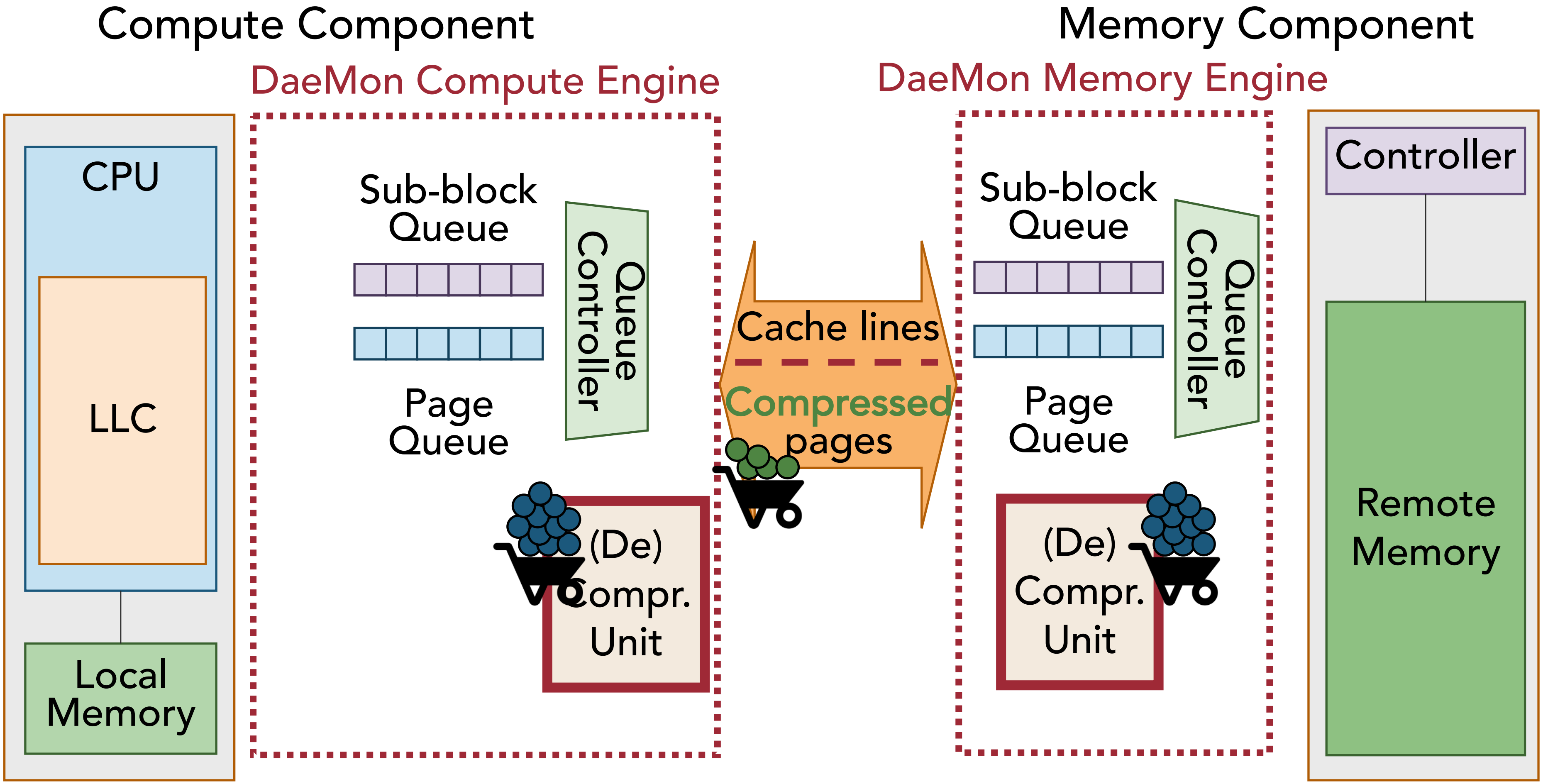
prioritization of cache line migrations

2. Multiple Granularity Data Movement



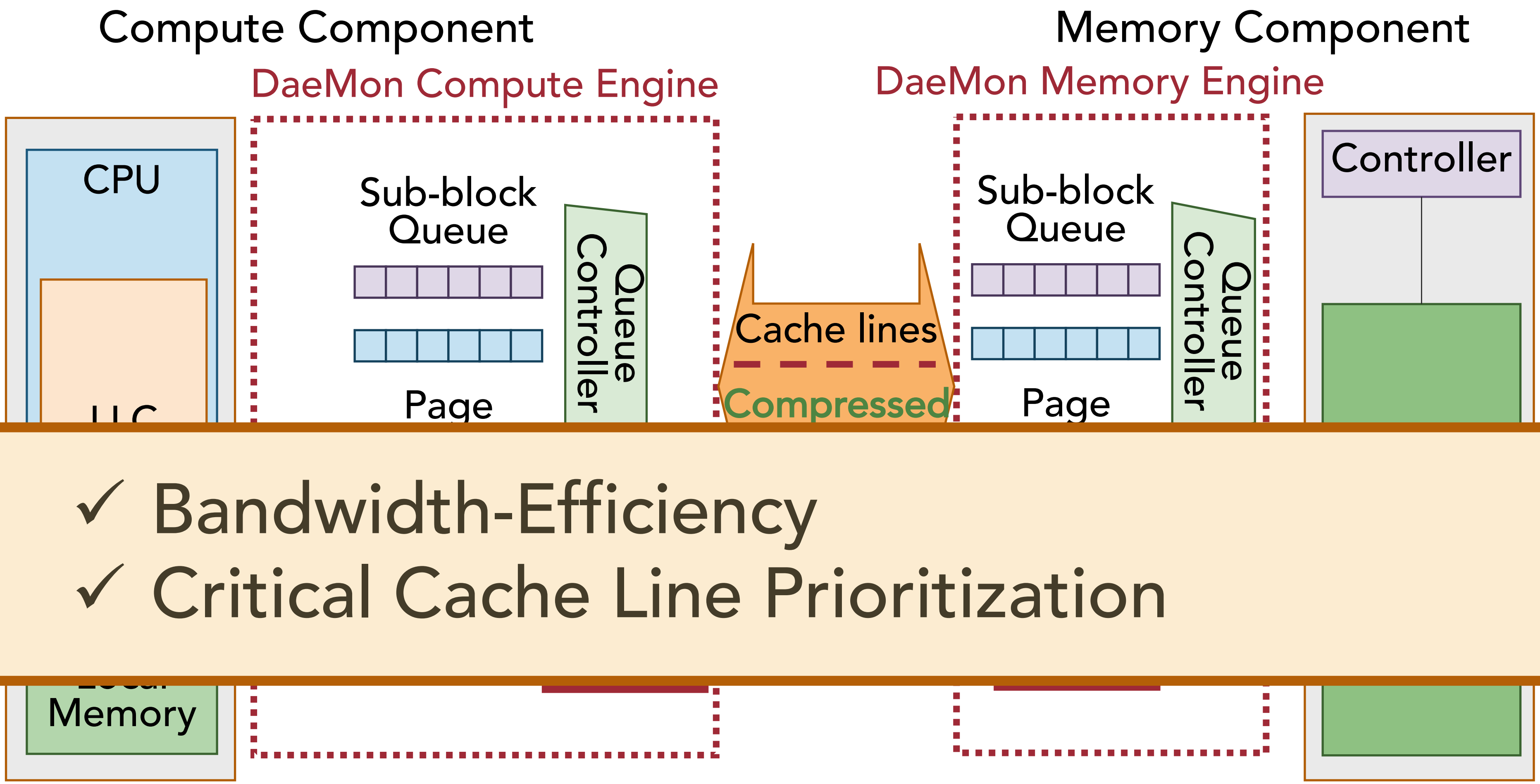
- ✓ Software Transparency
- ✓ Low Metadata Overheads
- ✓ High Spatial Locality
- ✓ Latency-Efficiency in Critical Data

3. Link Compression in Page Migrations



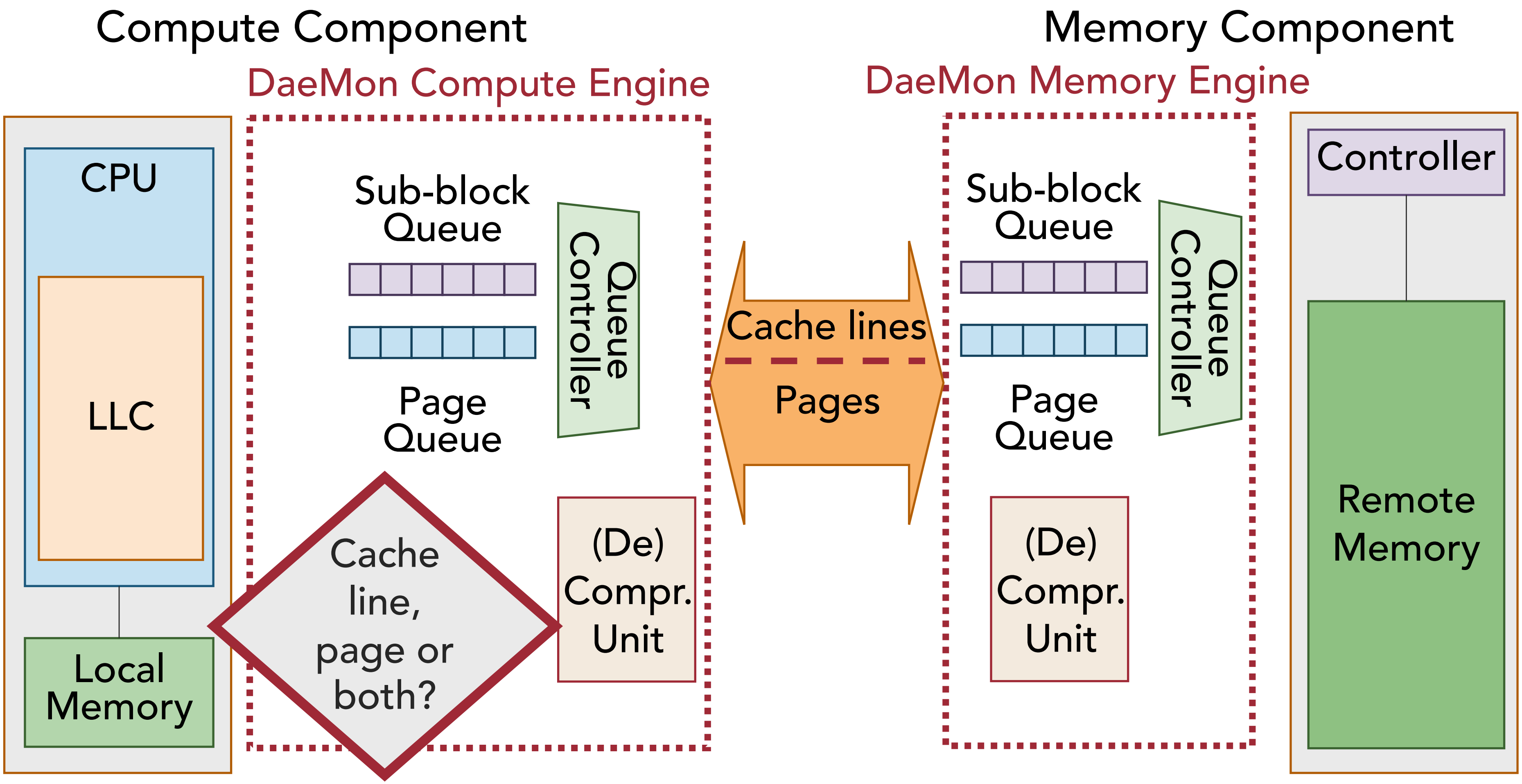
compressed pages inside the network

3. Link Compression in Page Migrations

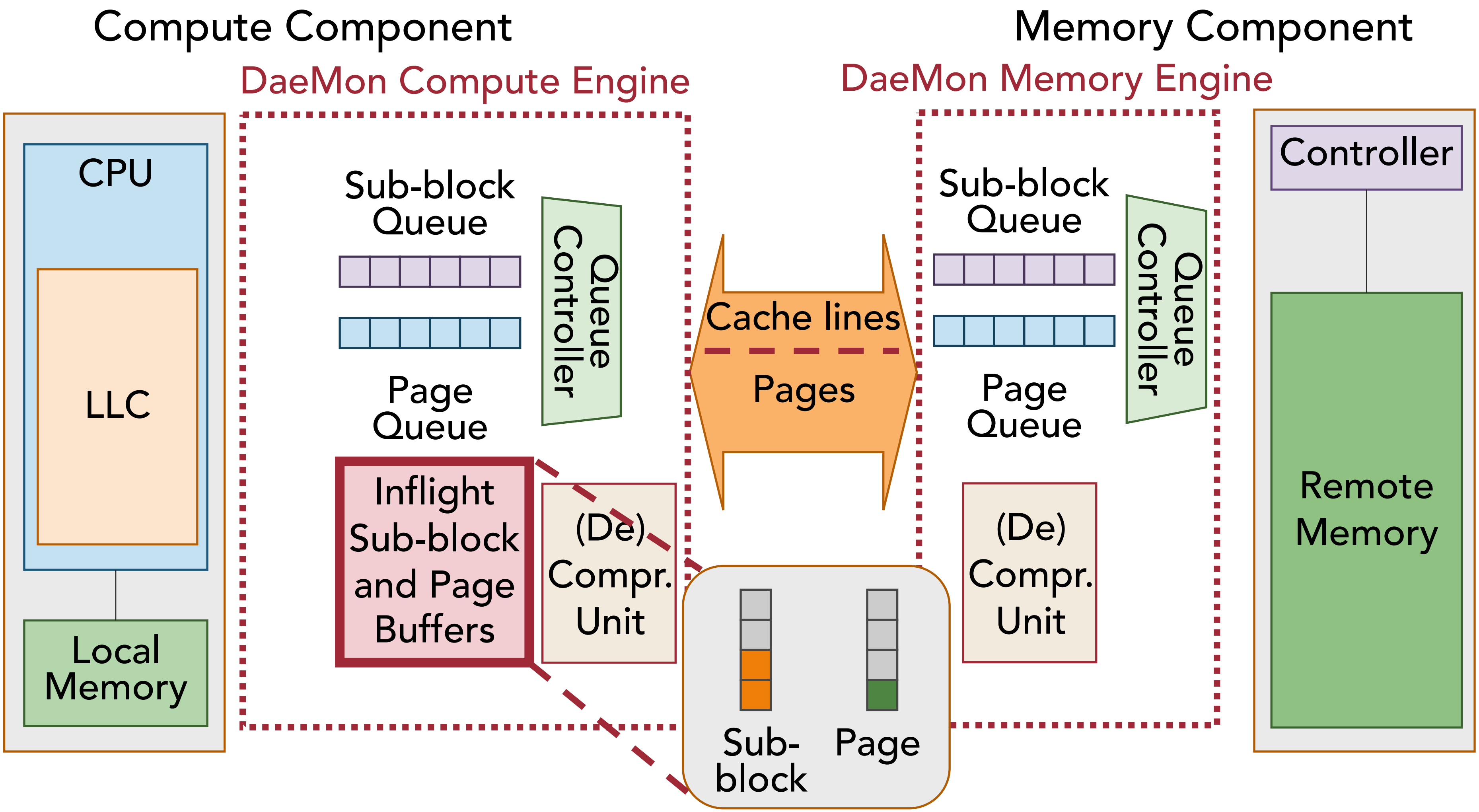


✓ Bandwidth-Efficiency
✓ Critical Cache Line Prioritization

4. Selection Granularity Data Movement

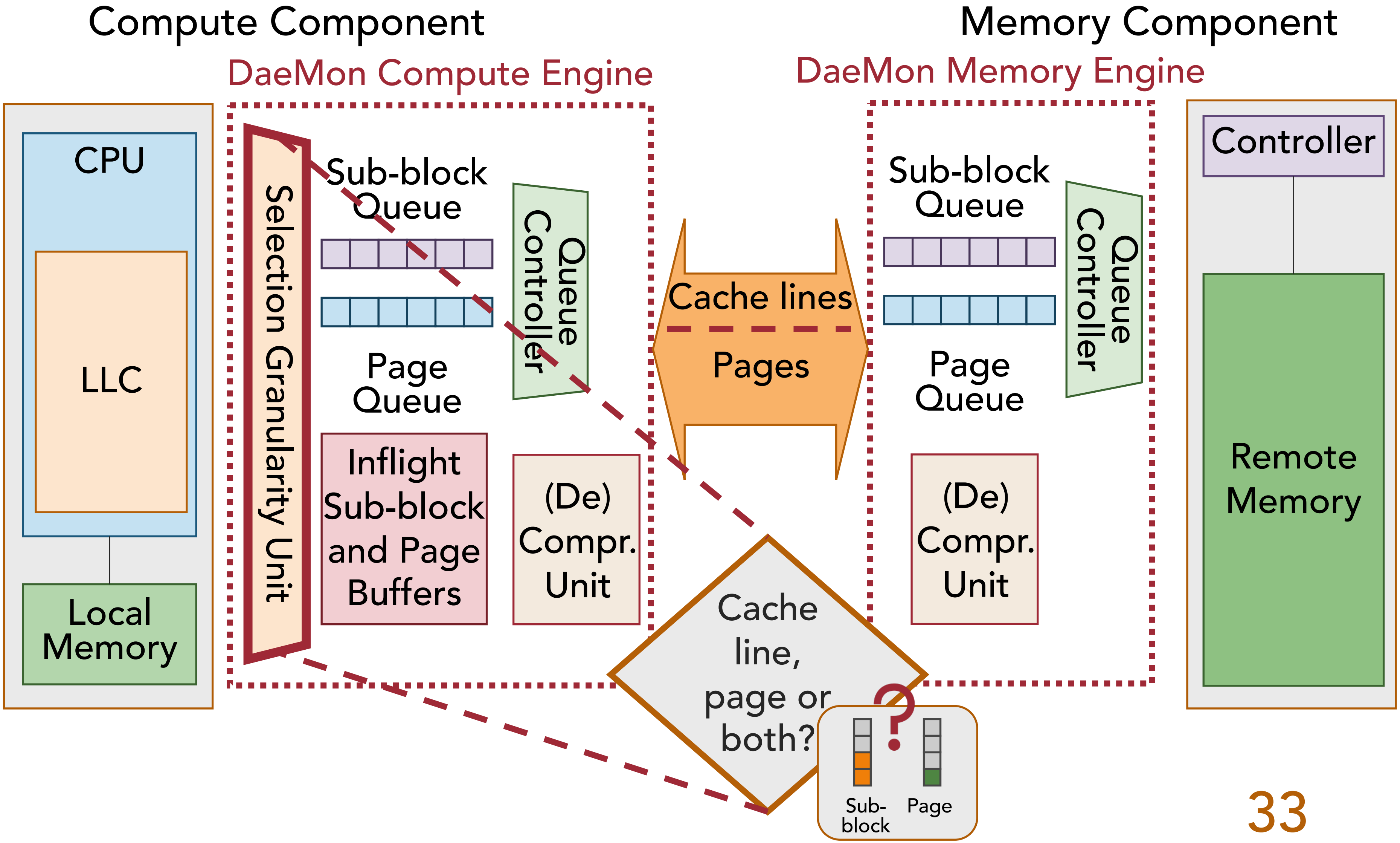


4. Selection Granularity Data Movement

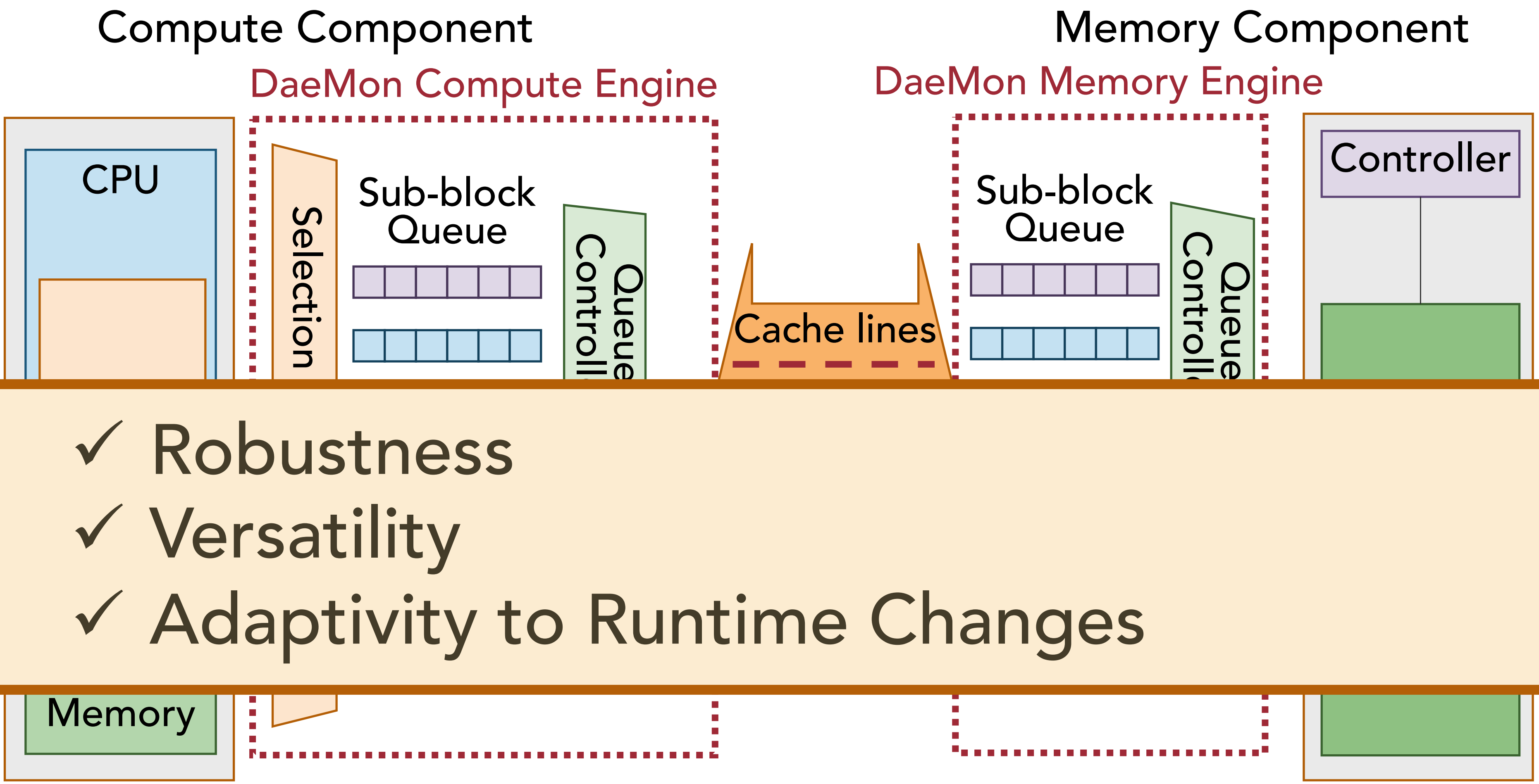


track pending data migrations

4. Selection Granularity Data Movement

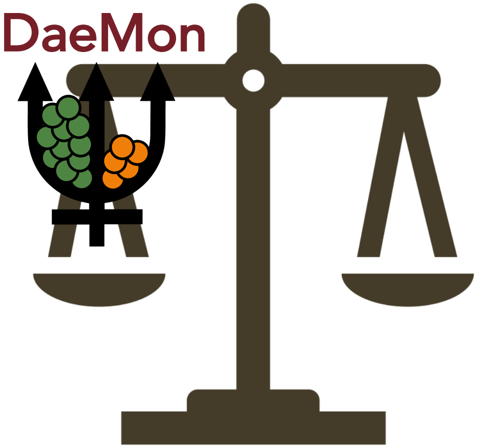


4. Selection Granularity Data Movement

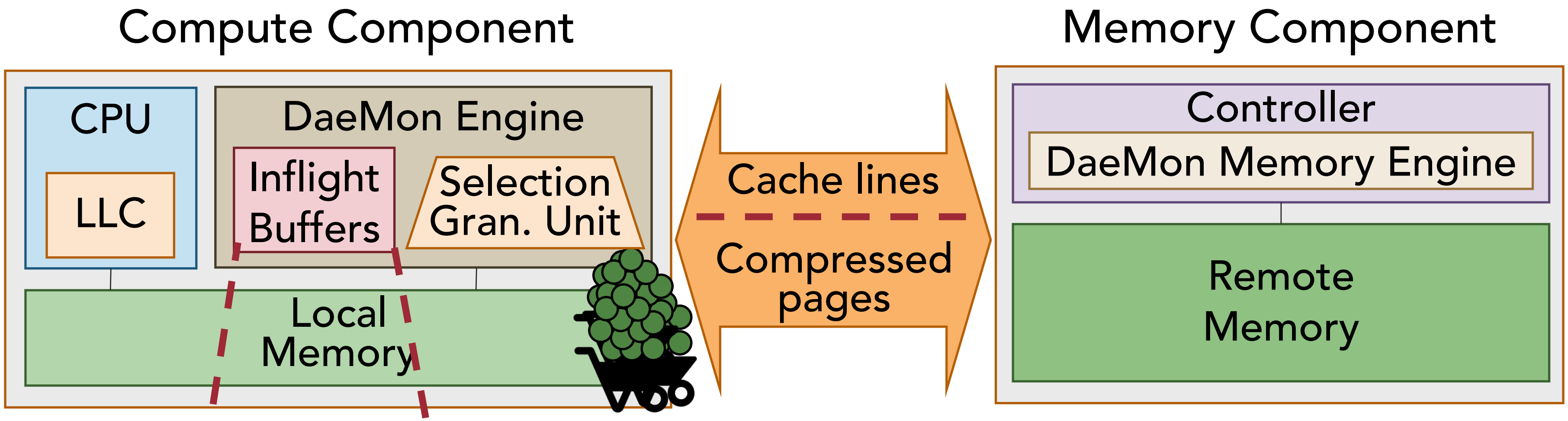


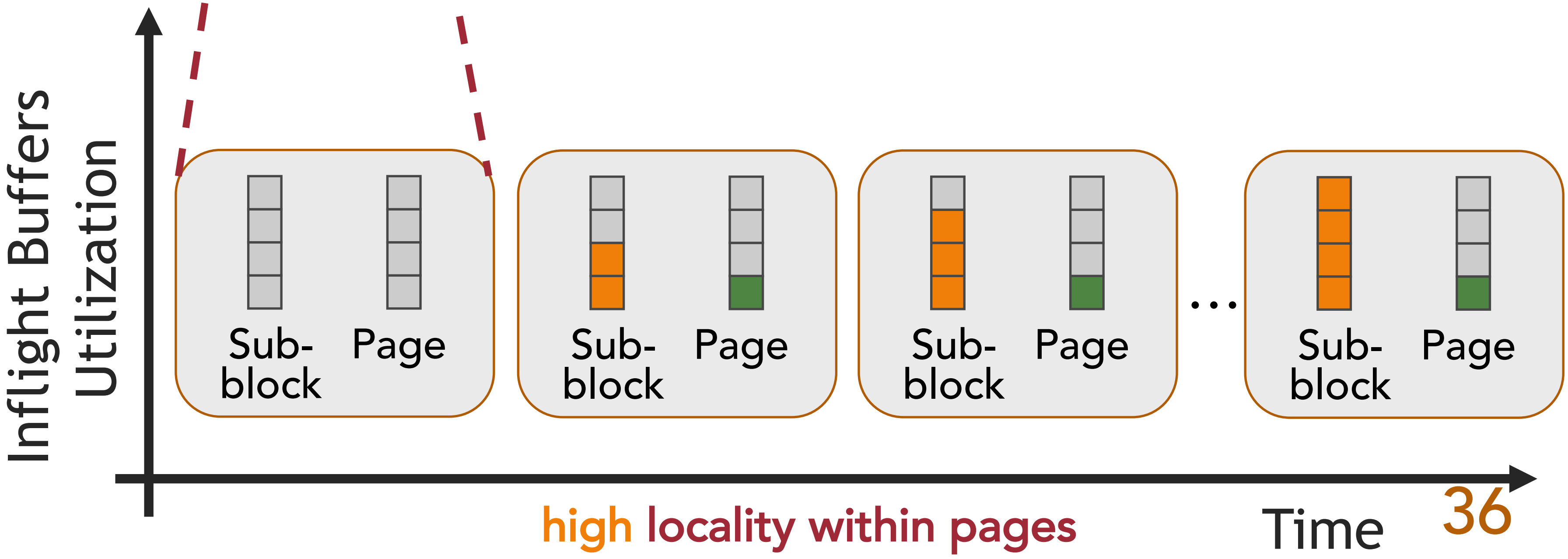
- ✓ Robustness
- ✓ Versatility
- ✓ Adaptivity to Runtime Changes

Why does this work?



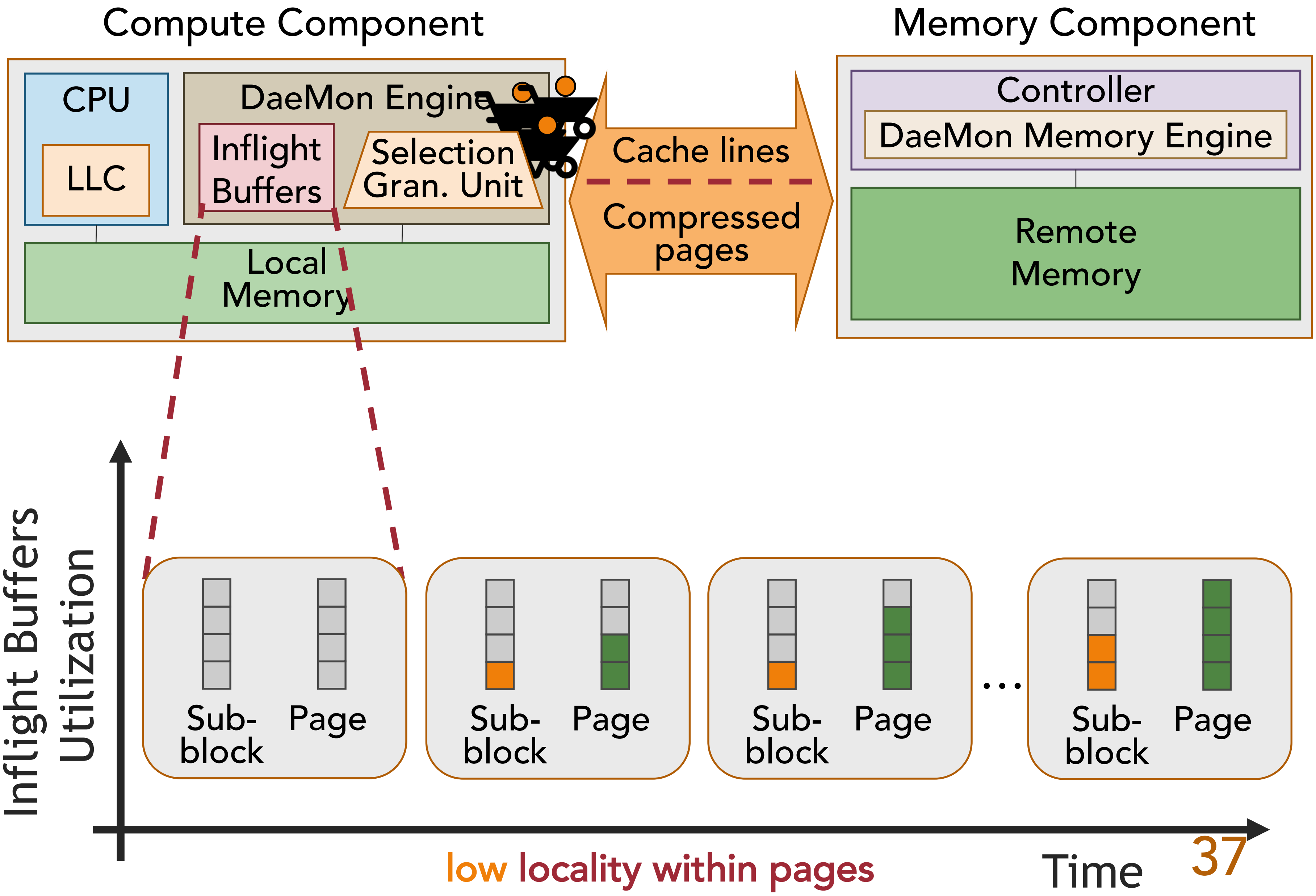
Use Case 1: Memory Access Patterns





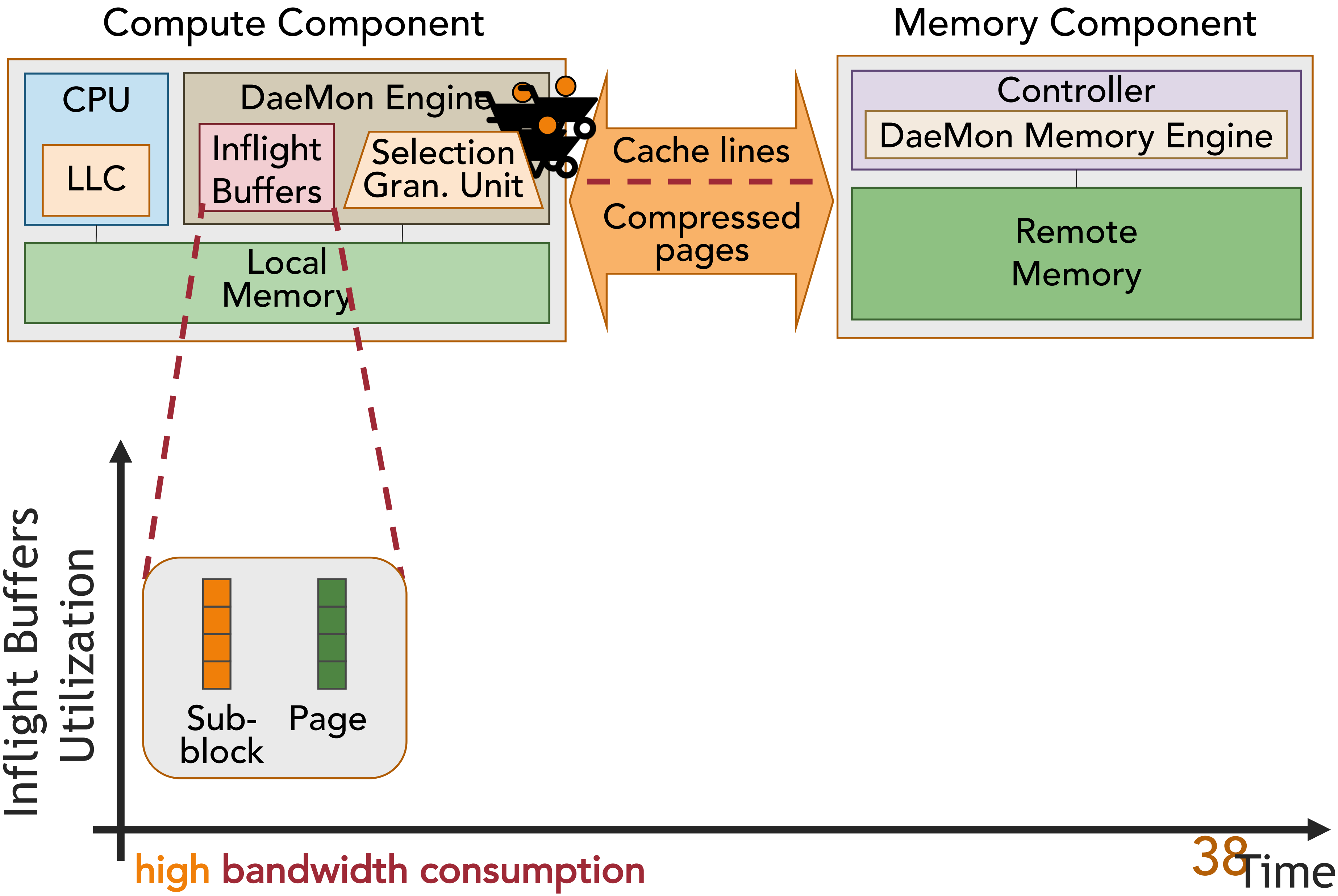
high locality within pages

Use Case 1: Memory Access Patterns



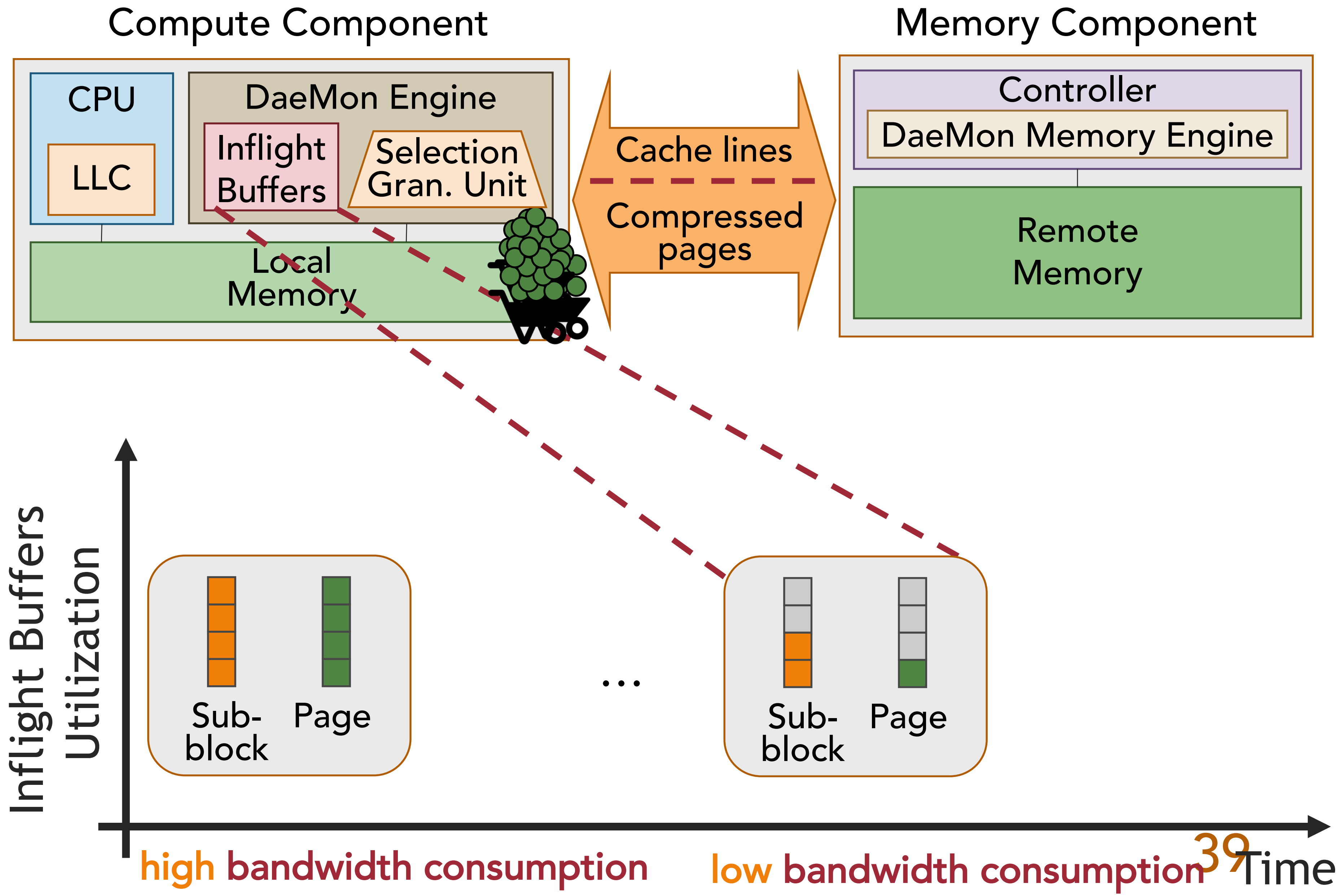
low locality within pages

Use Case 2: Network Characteristics



high bandwidth consumption

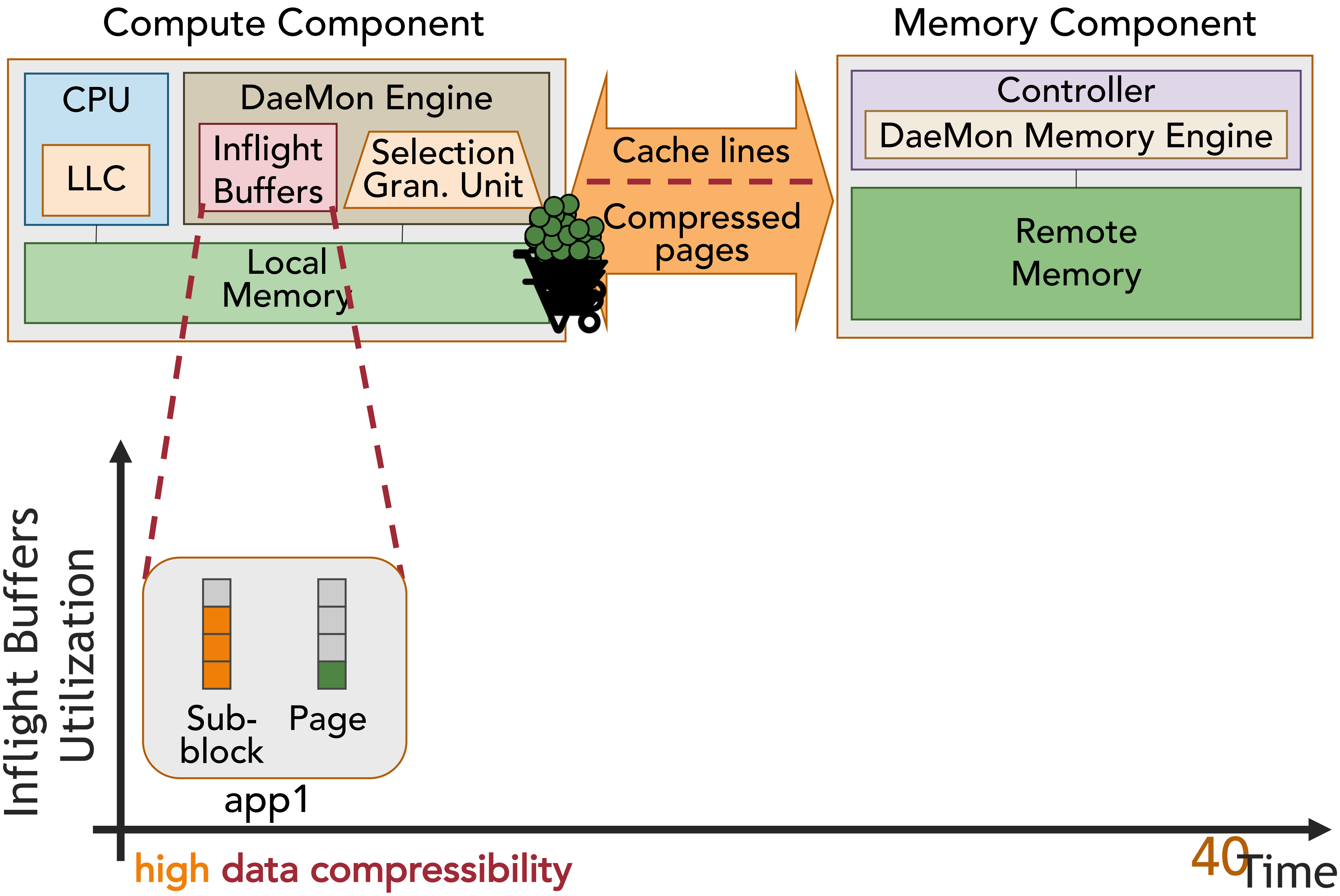
Use Case 2: Network Characteristics



high bandwidth consumption

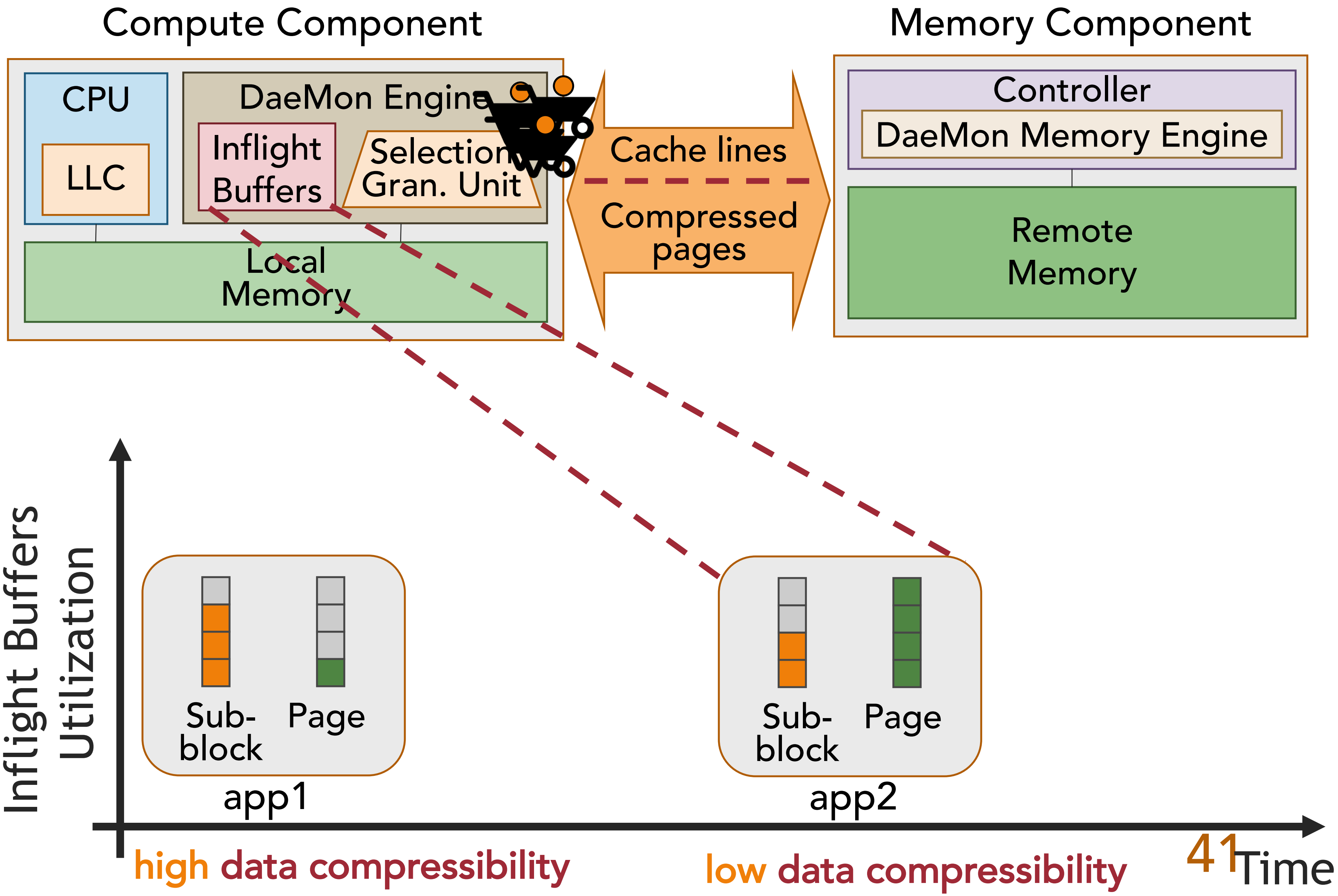
low bandwidth consumption

Use Case 3: Data Compressibility

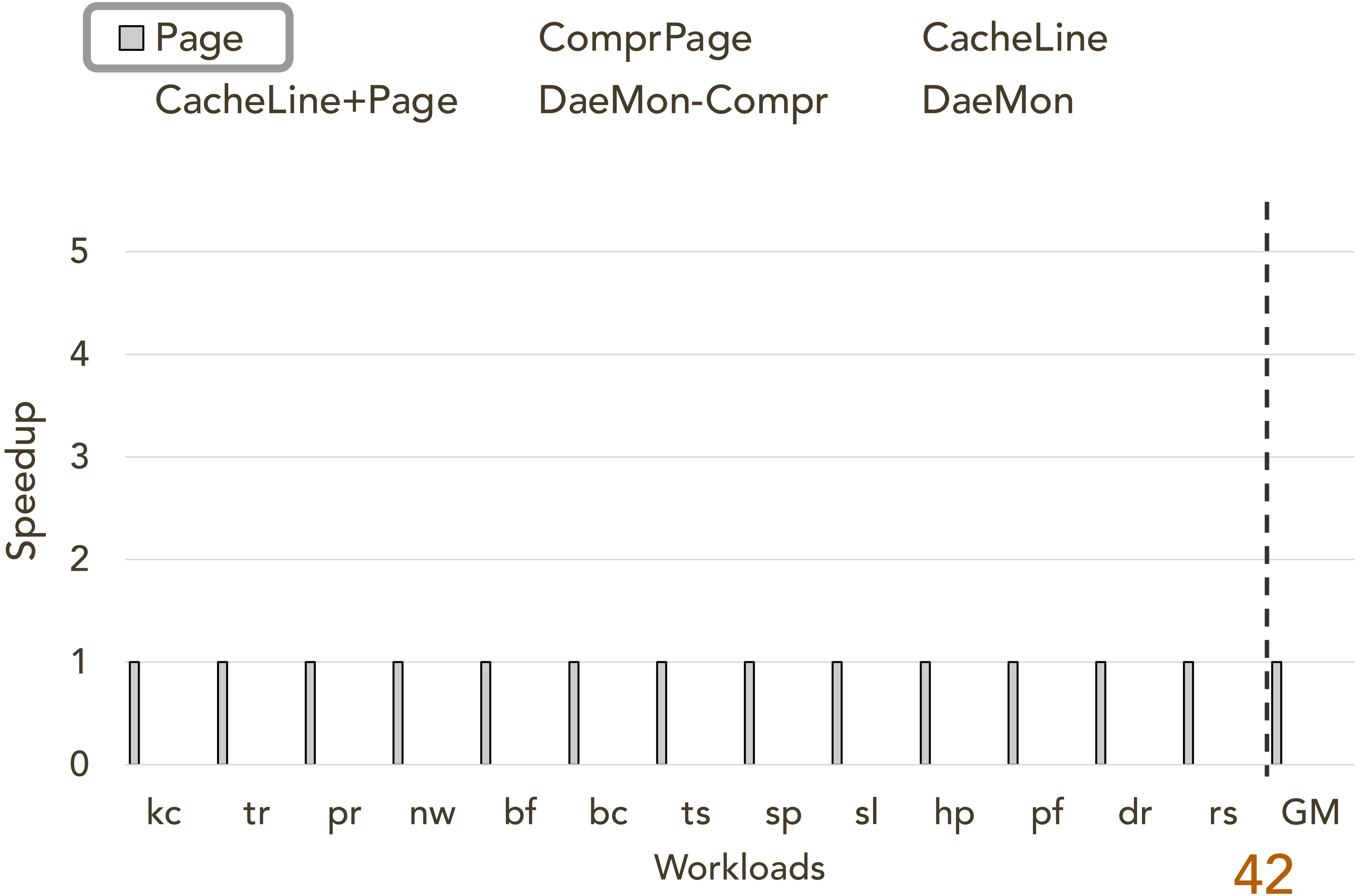


high data compressibility

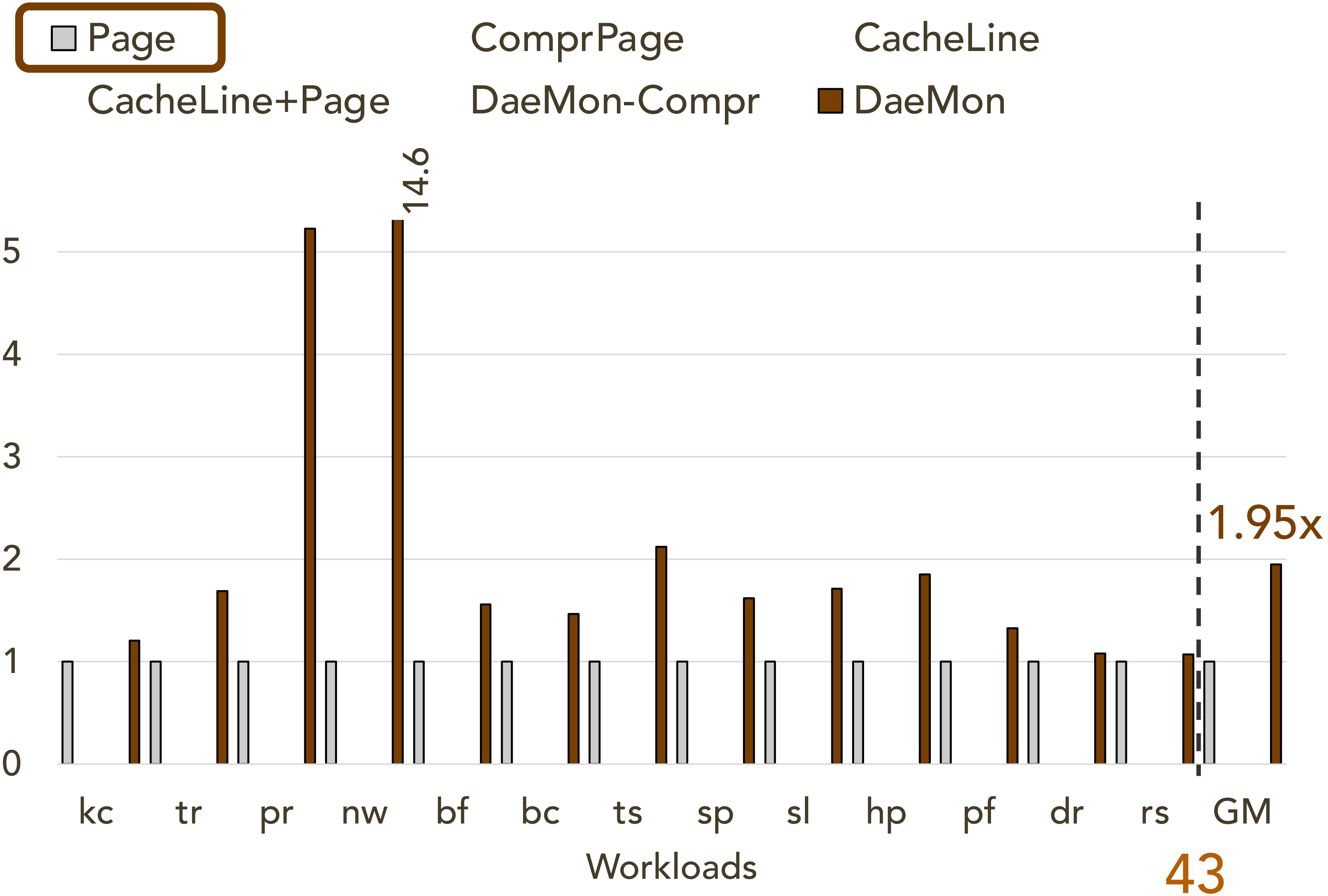
Use Case 3: Data Compressibility



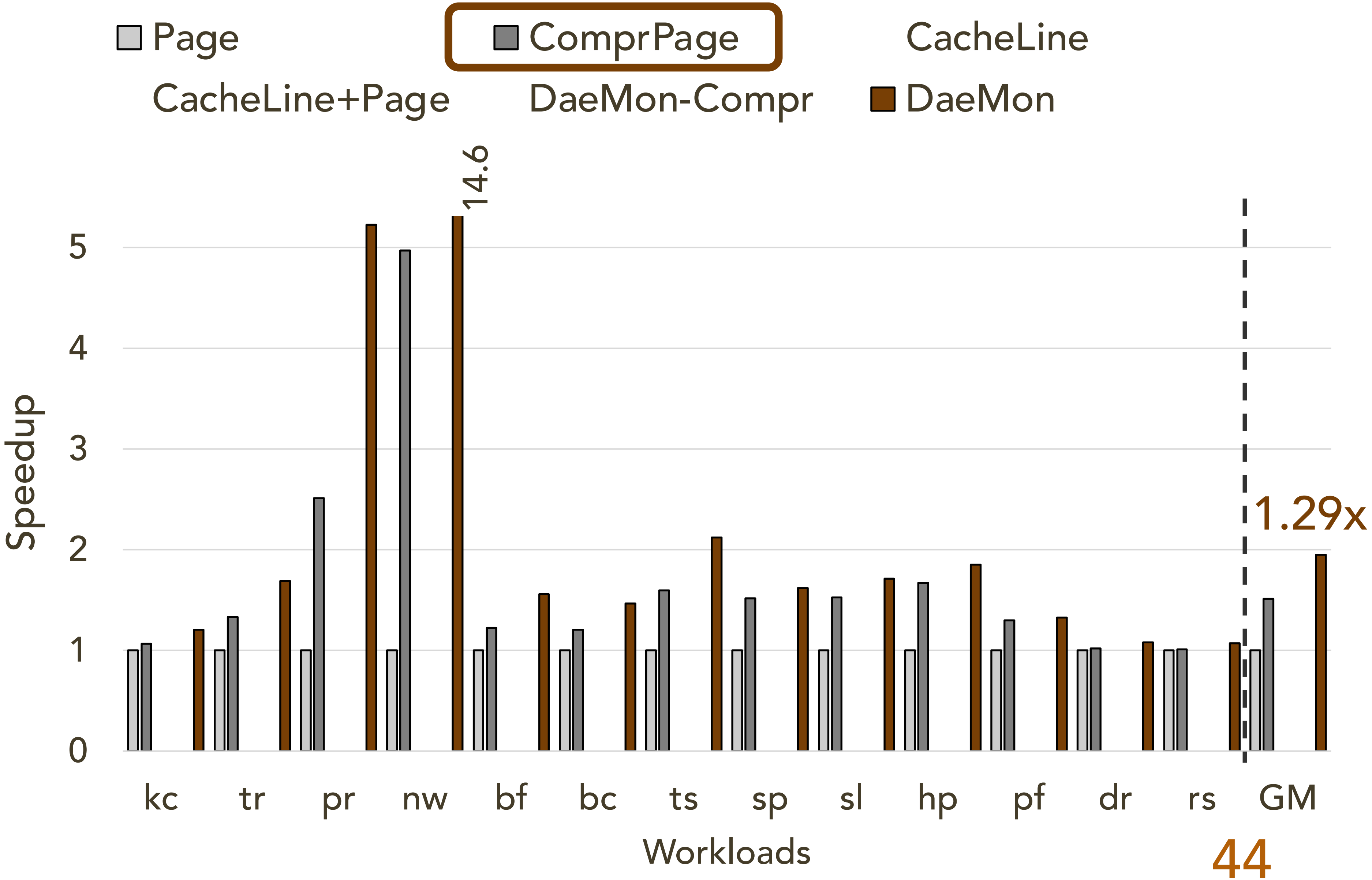
Speedup in Real Applications



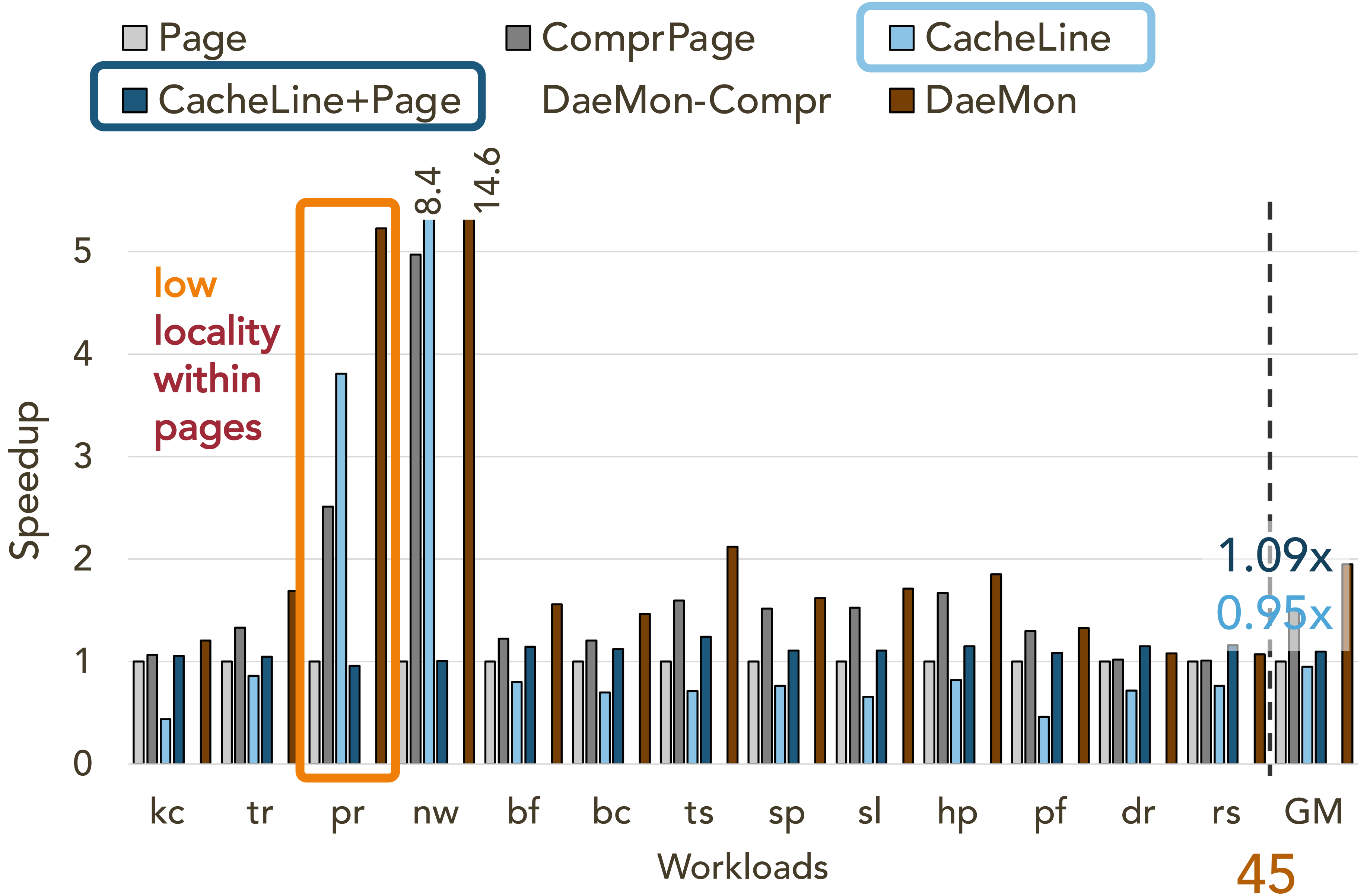
Speedup in Real Applications



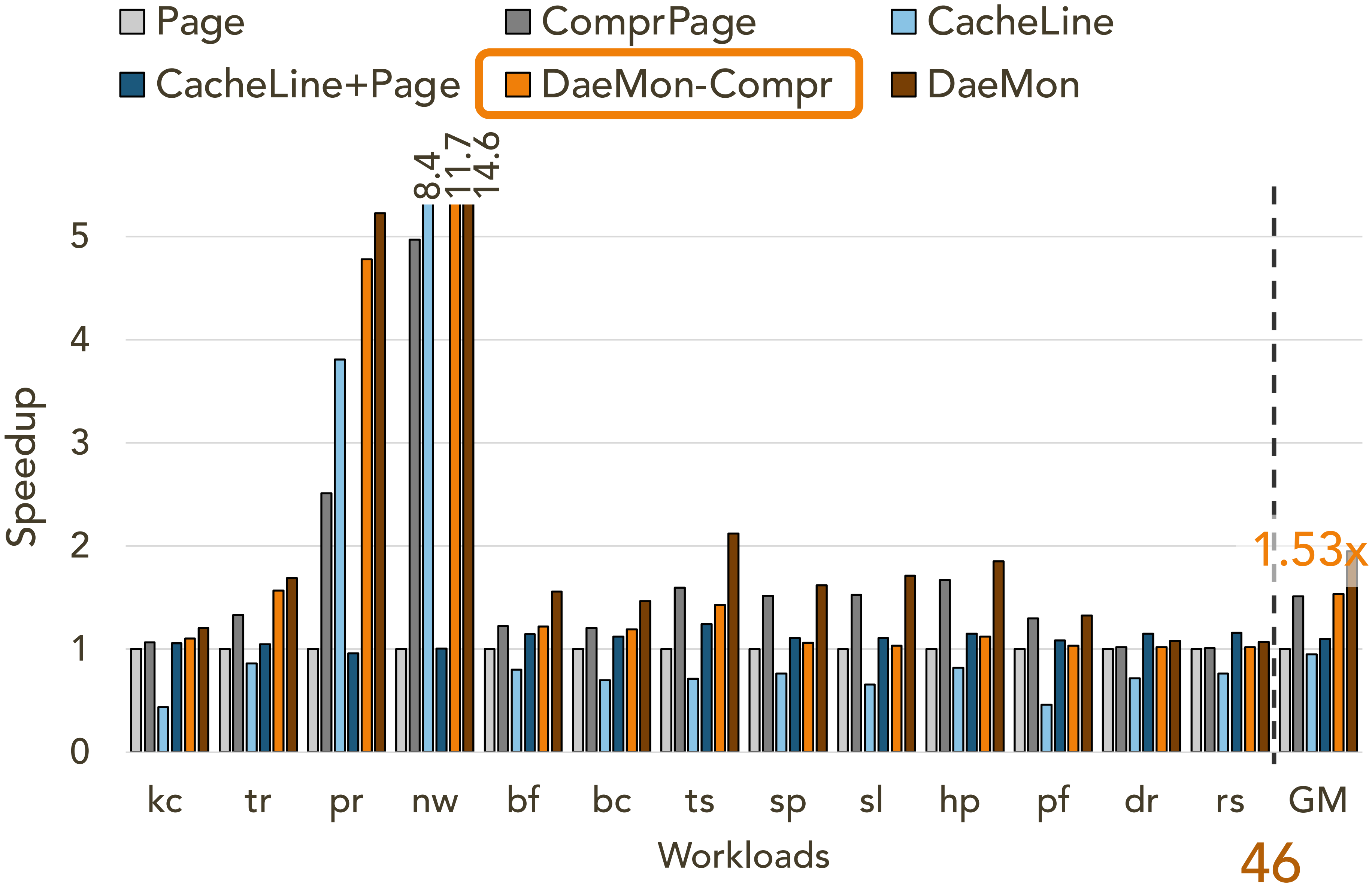
Speedup in Real Applications



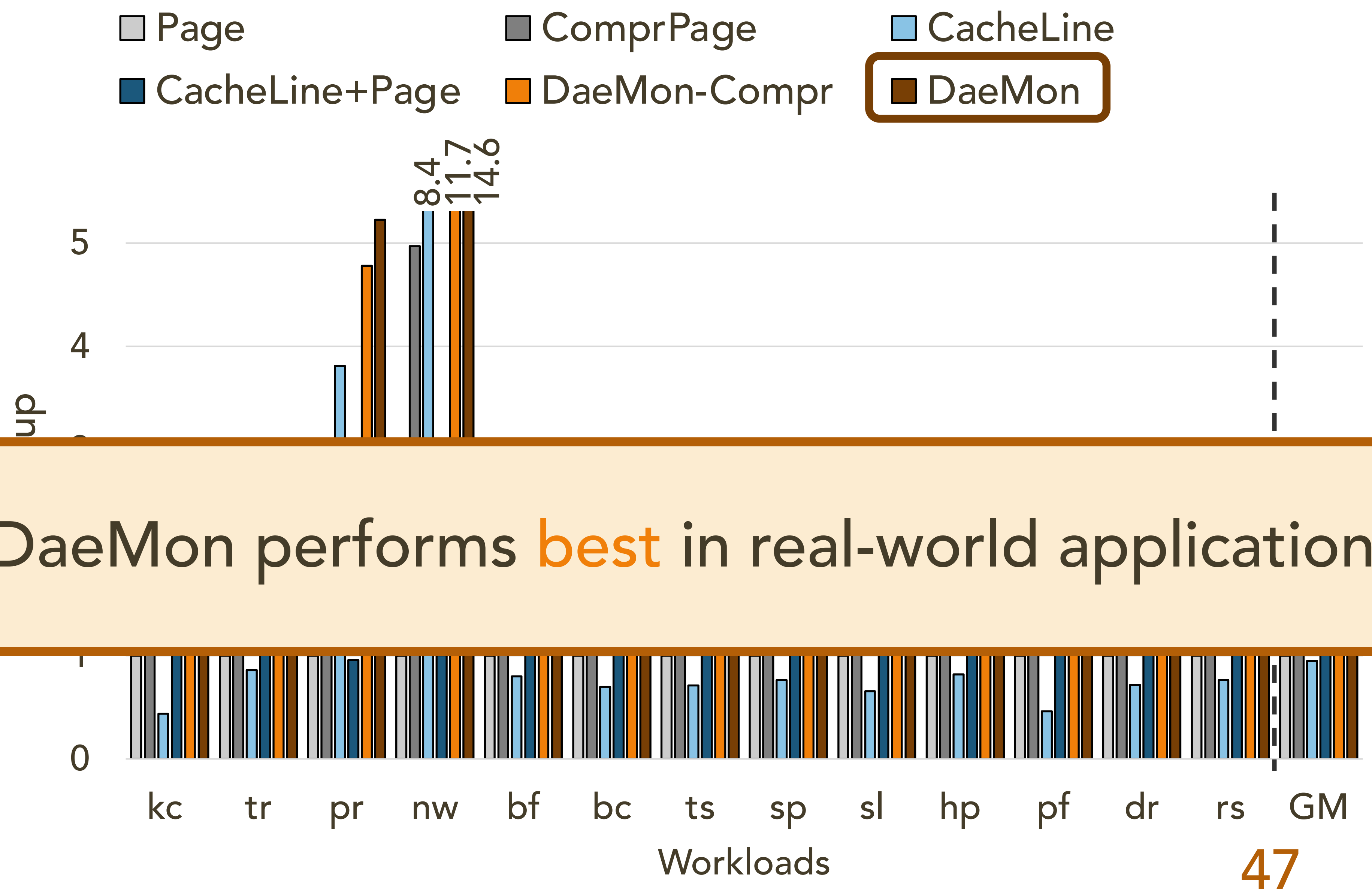
Speedup in Real Applications



Speedup in Real Applications

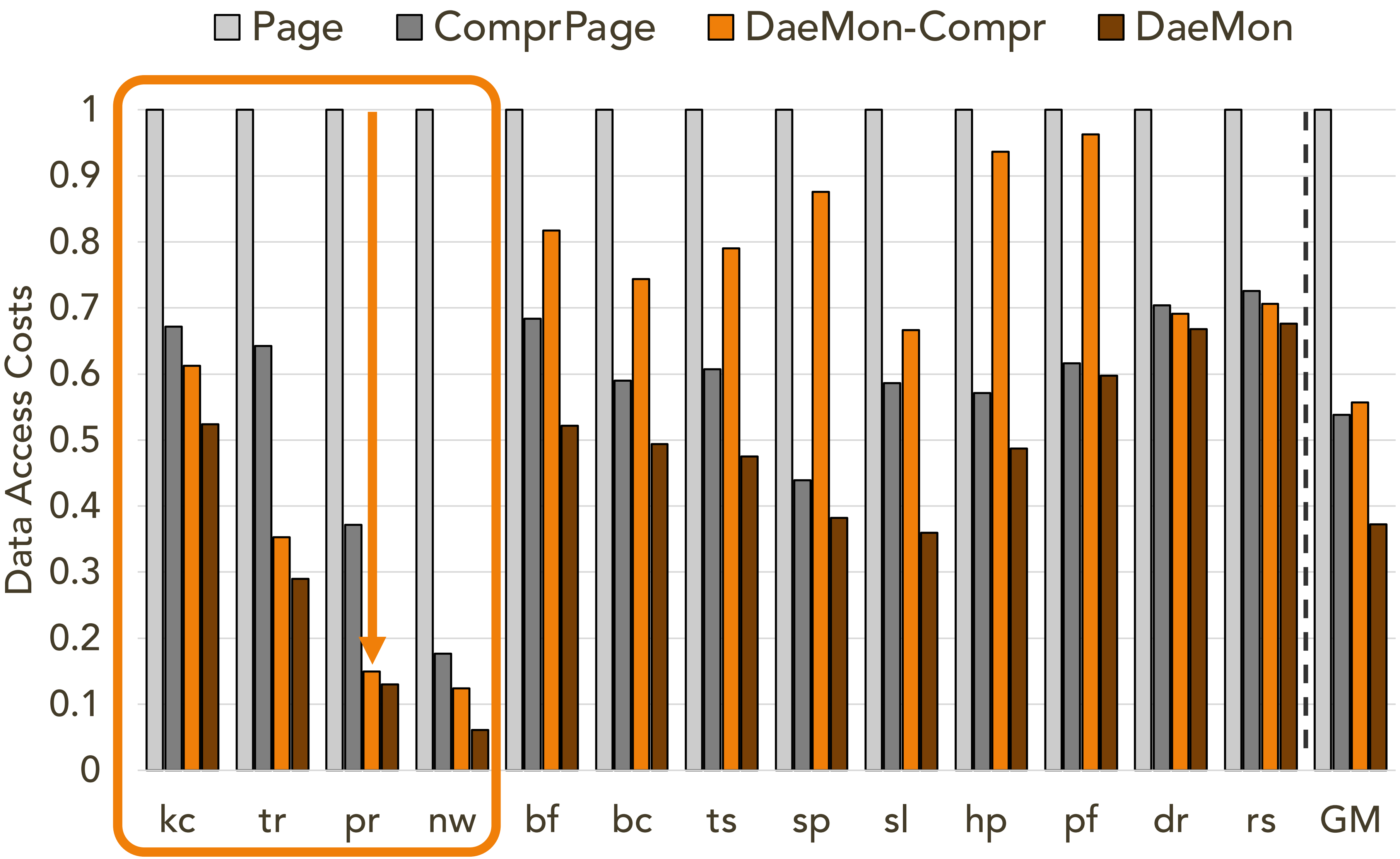


Speedup in Real Applications



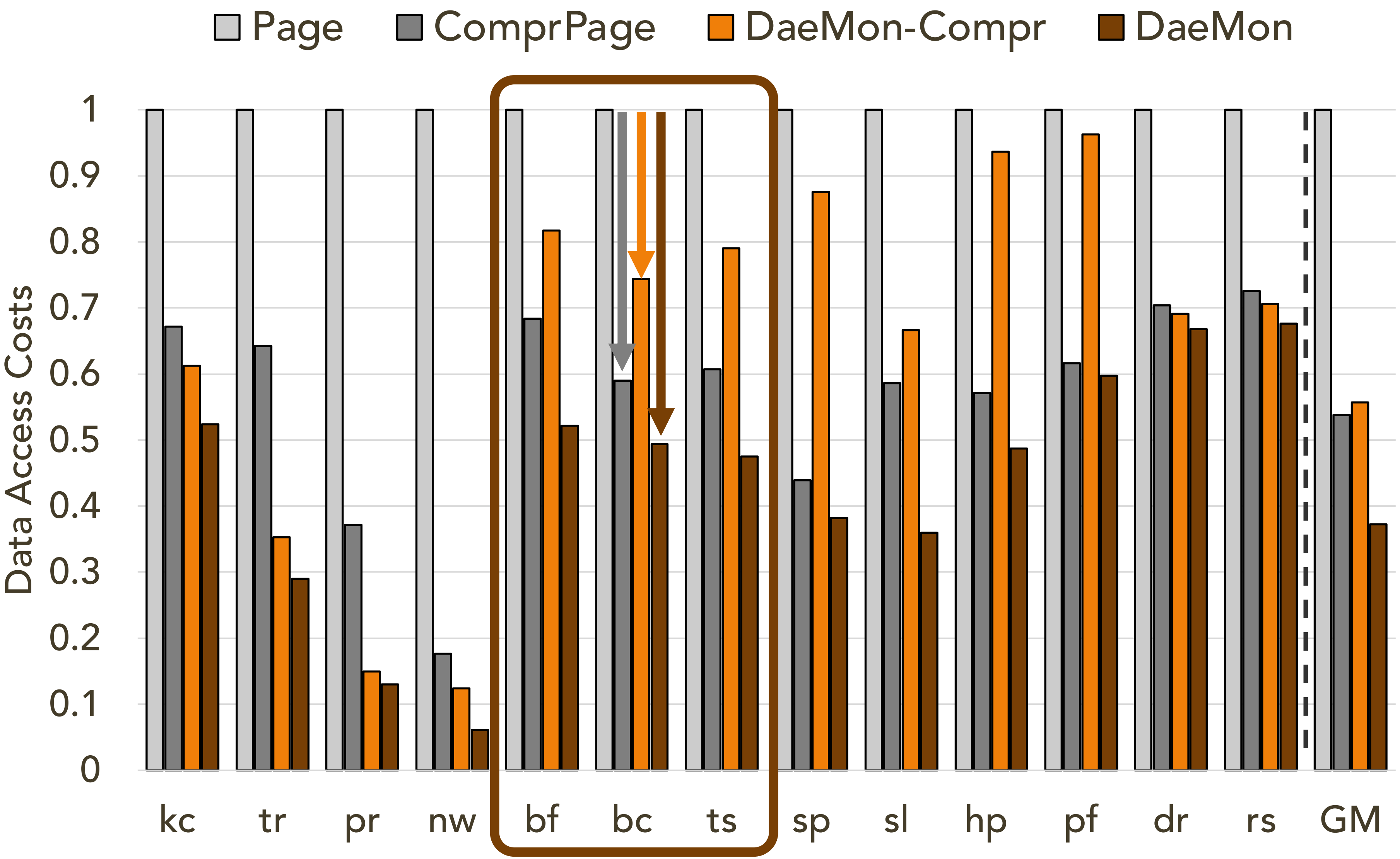
DaeMon performs best in real-world applications

Data Access Costs in Real Applications



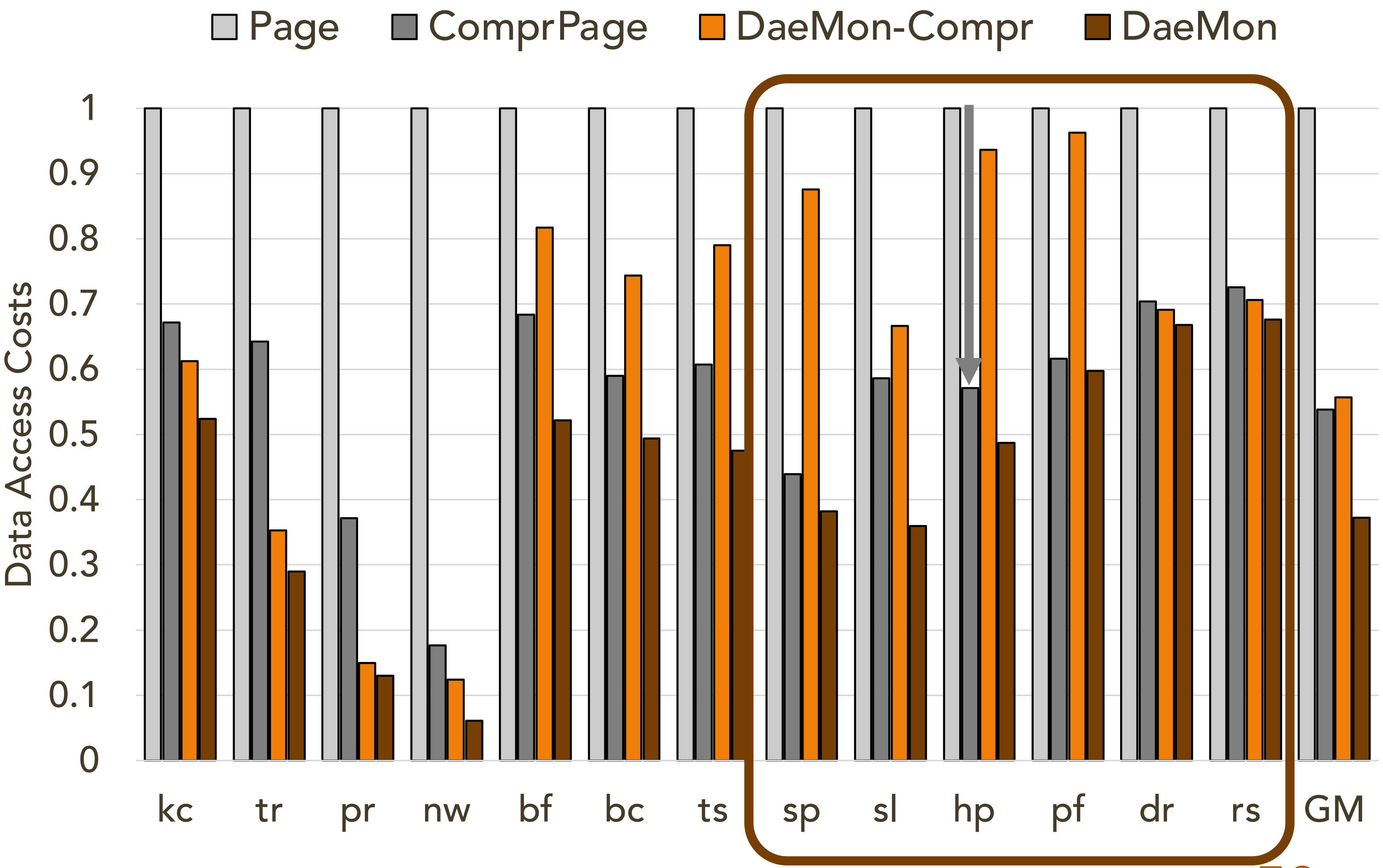
low locality within pages

Data Access Costs in Real Applications



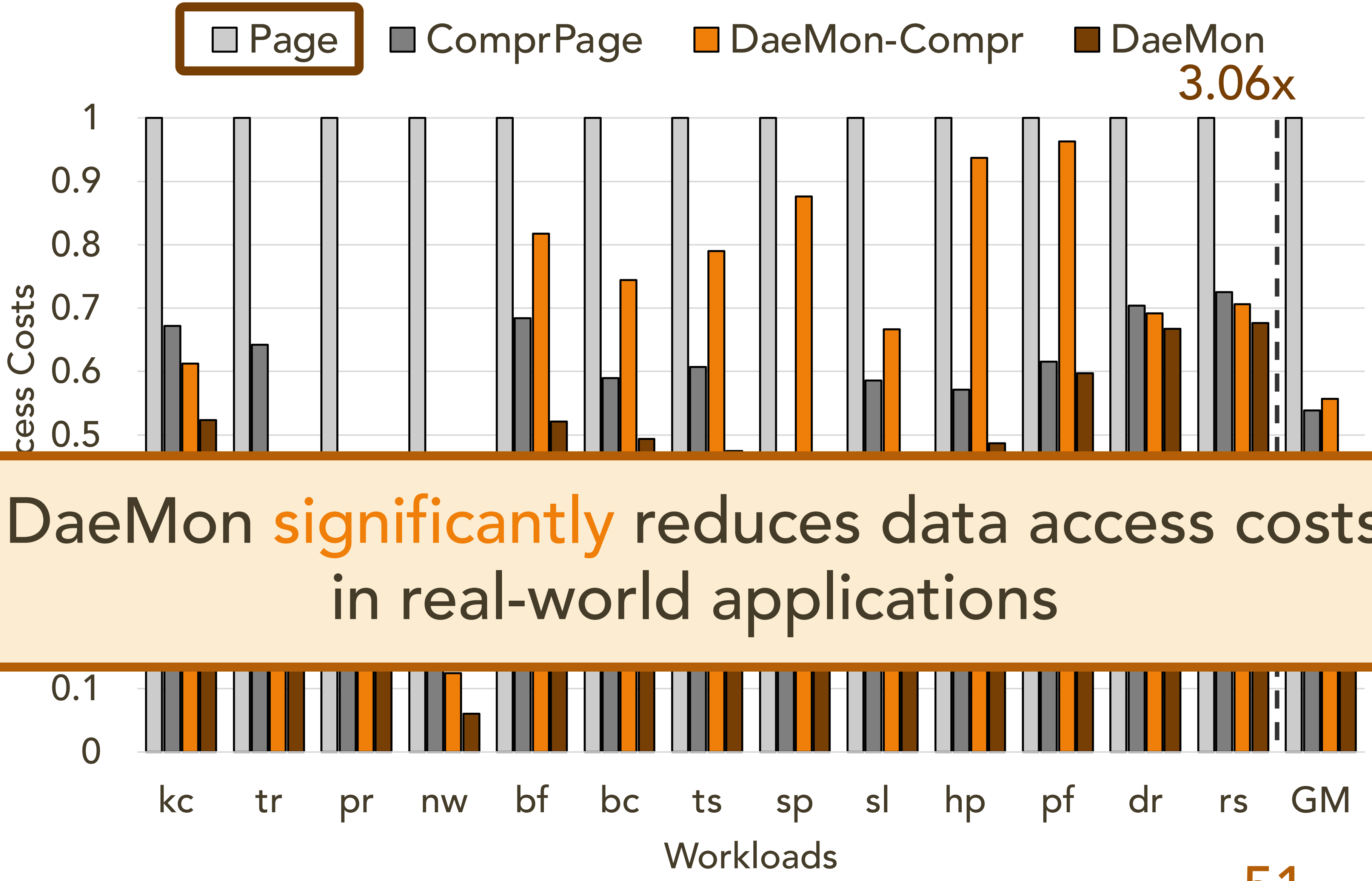
medium locality within pages

Data Access Costs in Real Applications



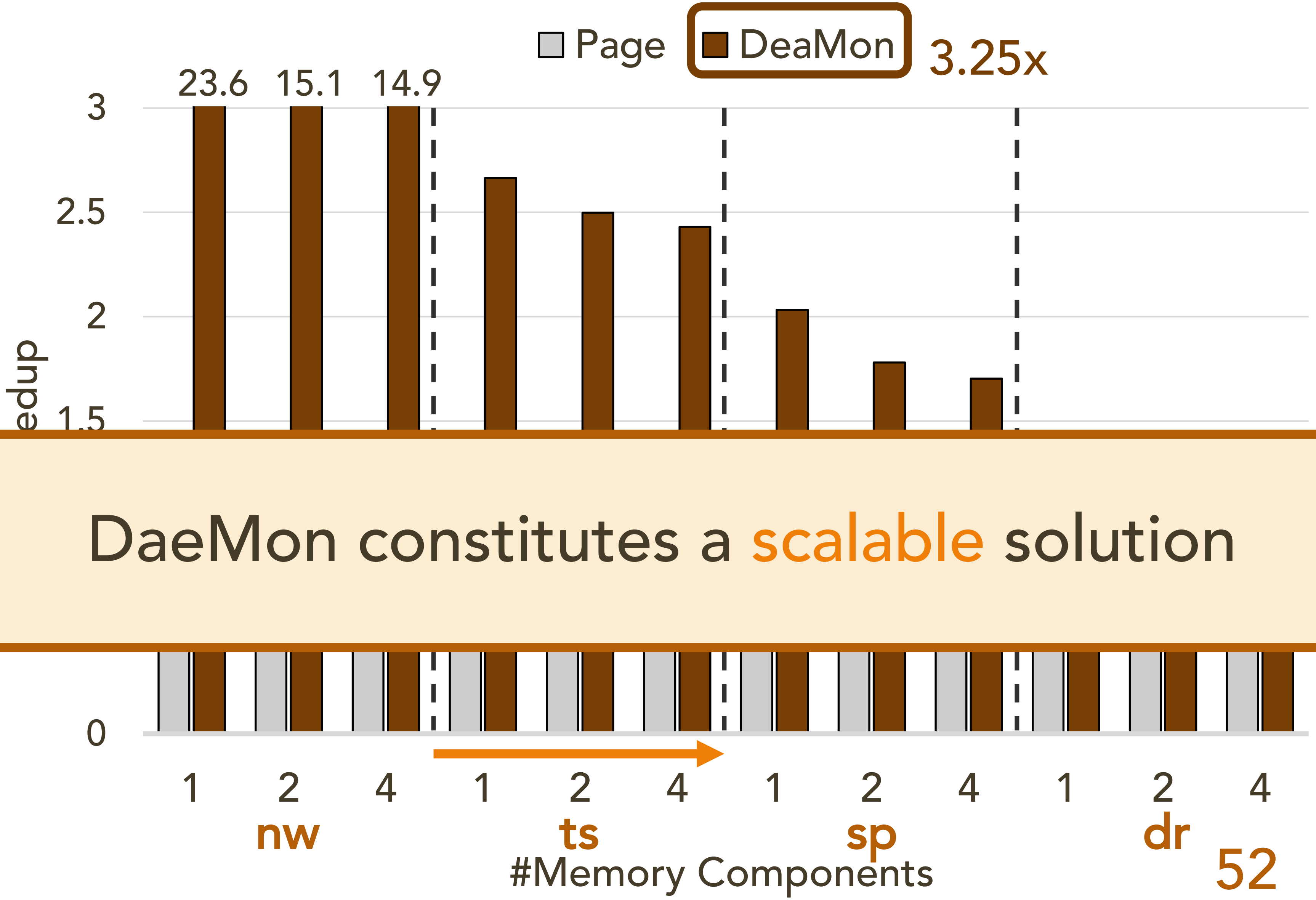
high locality within pages

Data Access Costs in Real Applications



DaeMon significantly reduces data access costs in real-world applications

Speedup in Many Memory Components

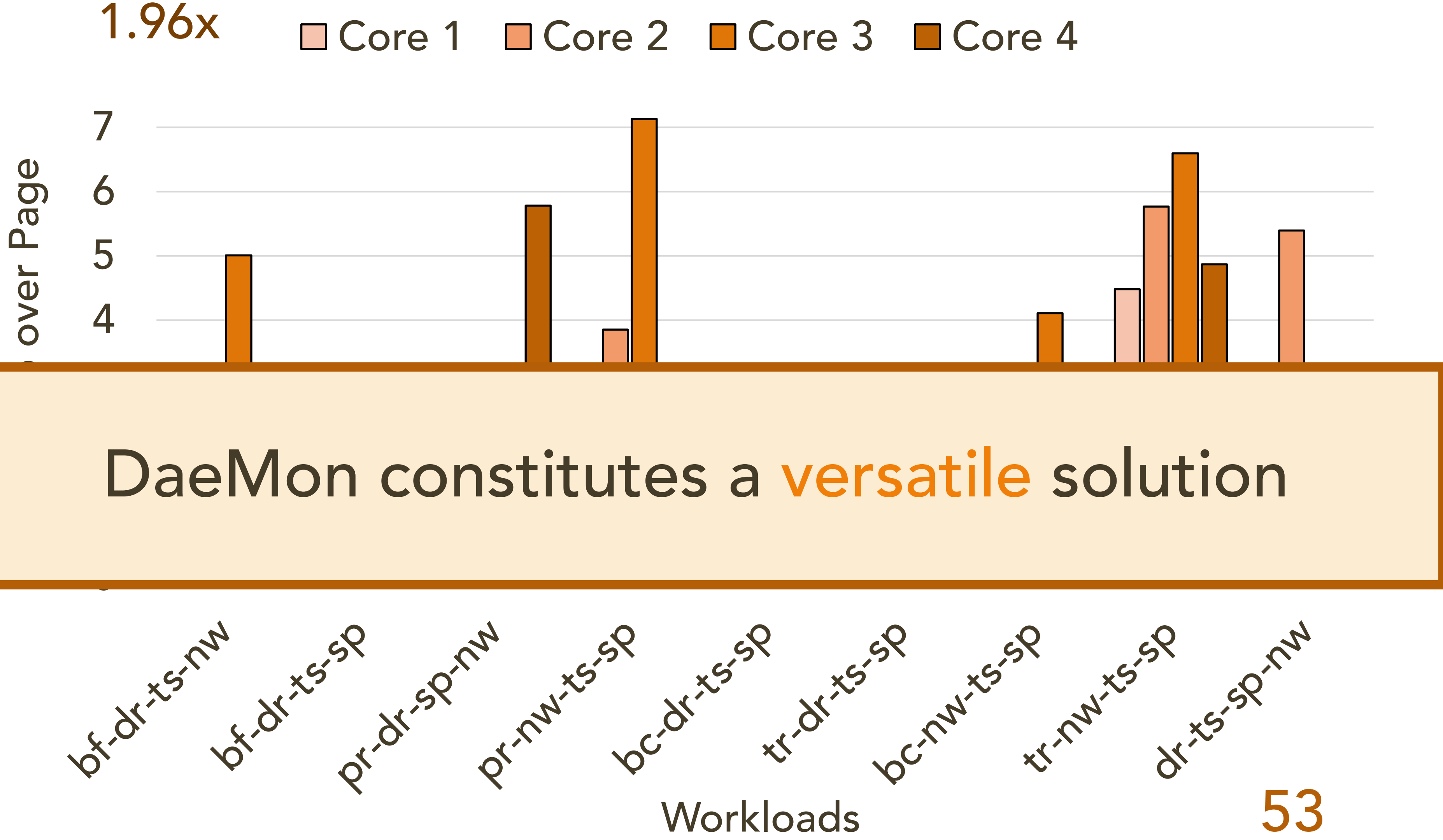


DaeMon constitutes a scalable solution

Speedup in Multiple Co-Running Jobs

- DaeMon over Page

1.96x



DaeMon constitutes a versatile solution

Conclusion

- Data movement is a major challenge for fully DSs
- Prior solutions are not suitable or efficient
- DaeMon is the first adaptive data movement solution
- DaeMon consists of four techniques:
    - Disaggregated hardware support
    - Decoupled multiple granularity data movement
    - Link compression in page movements
    - Selection granularity data movement
- DaeMon's benefits over the widely-adopted scheme:
    - 2.39x better performance
    - 3.06x lower data access
- DaeMon is highly-efficient, low-cost, scalable and robust

DaeMon: Architectural Support for Efficient Data Movement in Fully Disaggregated Memory Systems

Christina Giannoula

Kailong Huang, Jonathan Tang, Nectarios Koziris,
Georgios Goumas, Zeshan Chishti, Nandita Vijaykumar

Thank you!