TraceTracker: Hardware/Software Co-Evaluation for Large-Scale I/O Workload Reconstruction

Miryeong Kwon, Jie Zhang, Gyuyoung Park, Wonil Choi, David Donofrio, John Shalf, Mahmut Kandemir, and Myoungsoo Jung

YONSEI UNIVERSITY

CAMELab.org
Computer Architecture and Memory Systems Laboratory

Summary

- **Motivation:** Block traces collected on old systems (a decade ago HDDs) can make system analysis significantly different from the modern systems
- **Challenges**: Trace reconstruction is high-cost work and inaccurate
  - Application execution requires prohibitive resources; thousands of users, large-scale computing systems (**impractical**)
  - Overly-simplified methods are **excessively imprecise** due to lack of runtime contexts
- **Goal**: **Accurately** reconstruct workloads **without** application execution
- **Solutions**: Hardware/Software co-evaluation method
  1. Software method: Infer runtime contexts from existing block traces
  2. Hardware method: Remaster storage traces; aware of target system
- **Evaluation:**
  - TraceTracker infer runtime contexts 99%, 96% for number of occurrences and total periods, respectively
  - Apply TraceTracker to 577 open-license block traces

TraceTracker

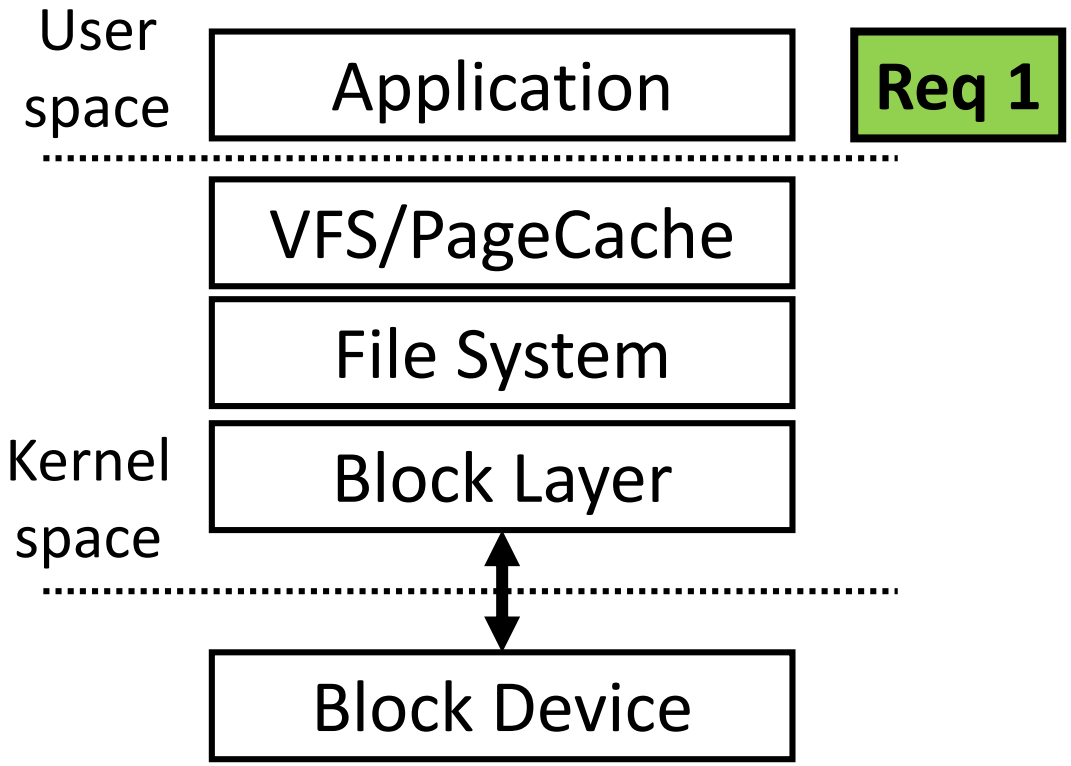
1. Background of Block Trace
2. Trace Reconstruction Methods
3. Insights on Trace Reconstruction
4. TraceTracker Method
5. Evaluation

Block Trace

: Consists of detailed **block device access information**

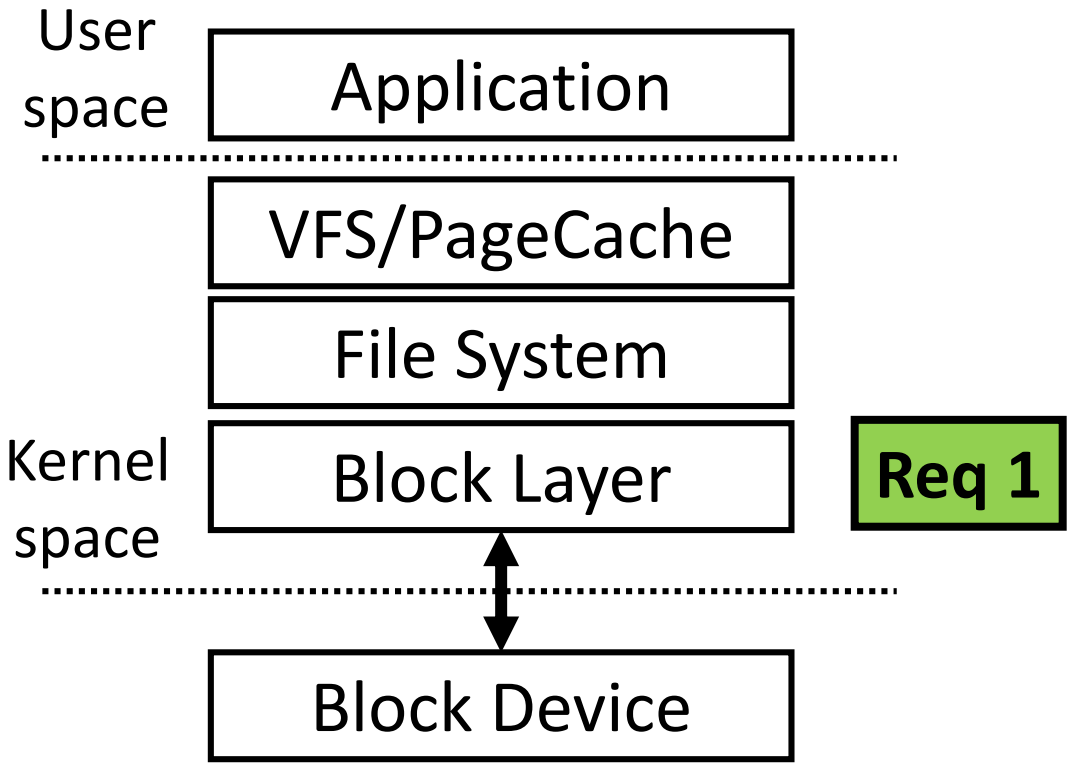
Block Trace

: Consists of detailed **block device access information**



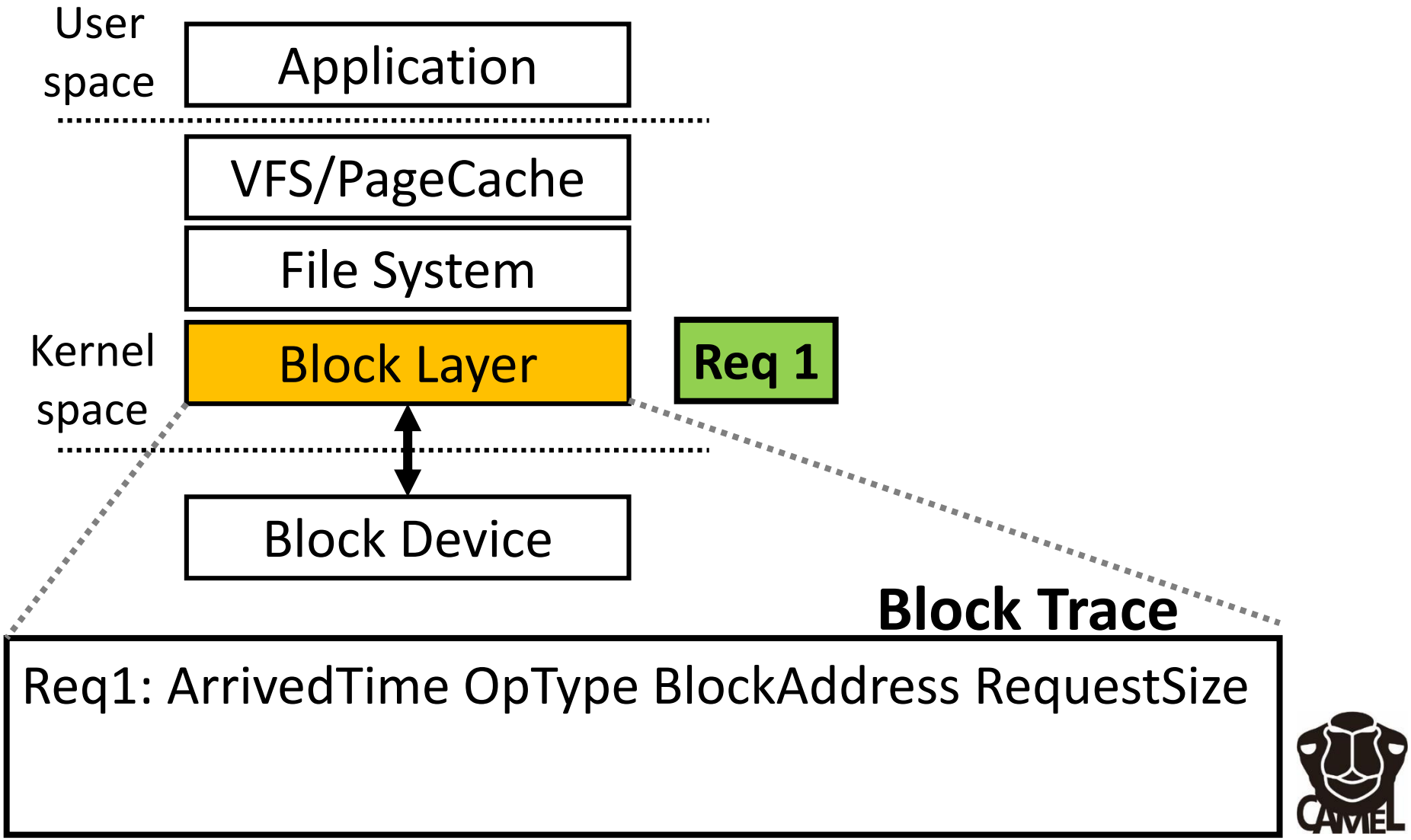
Block Trace

: Consists of detailed **block device access information**



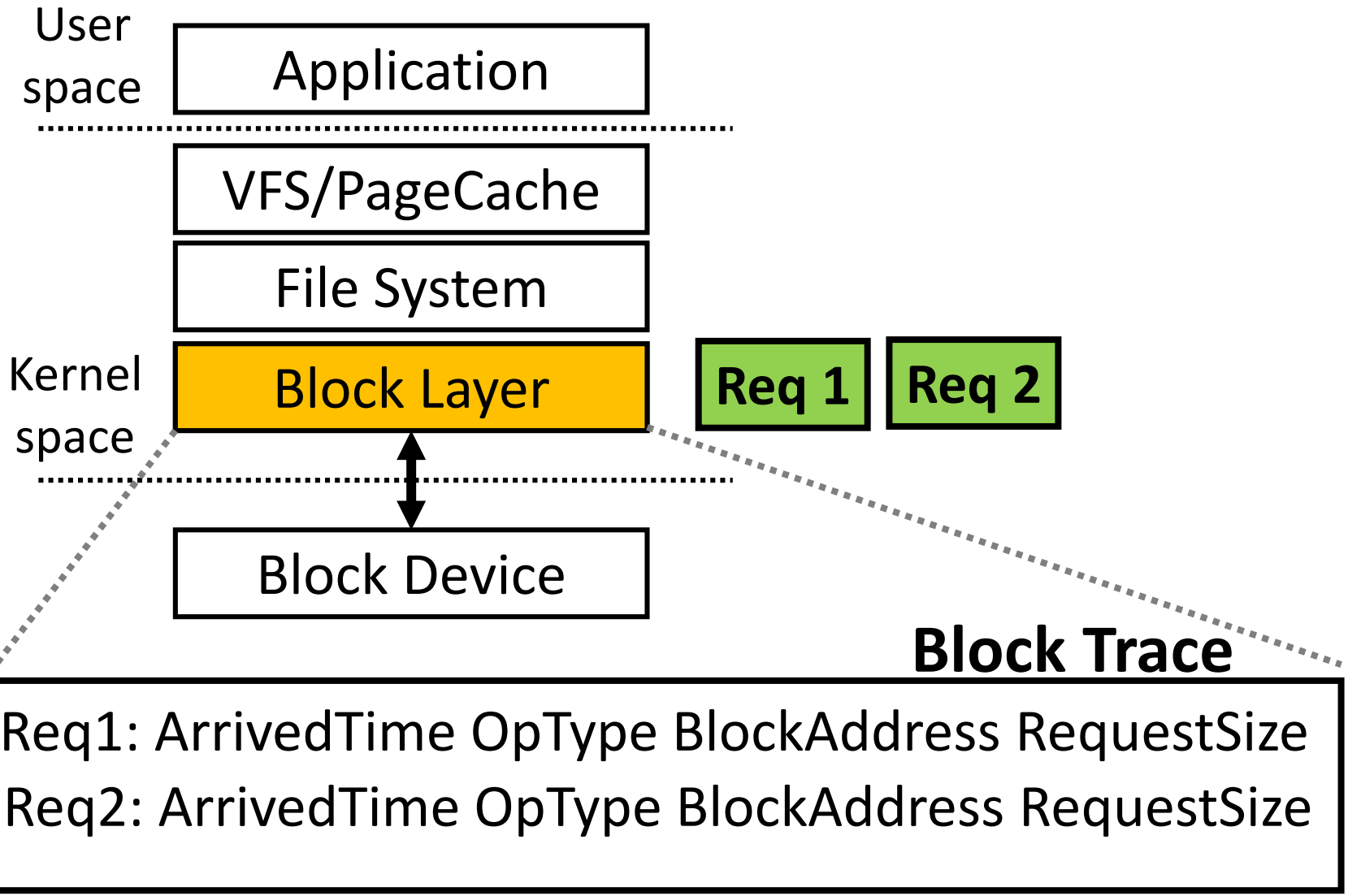
Block Trace

: Consists of detailed **block device access information**



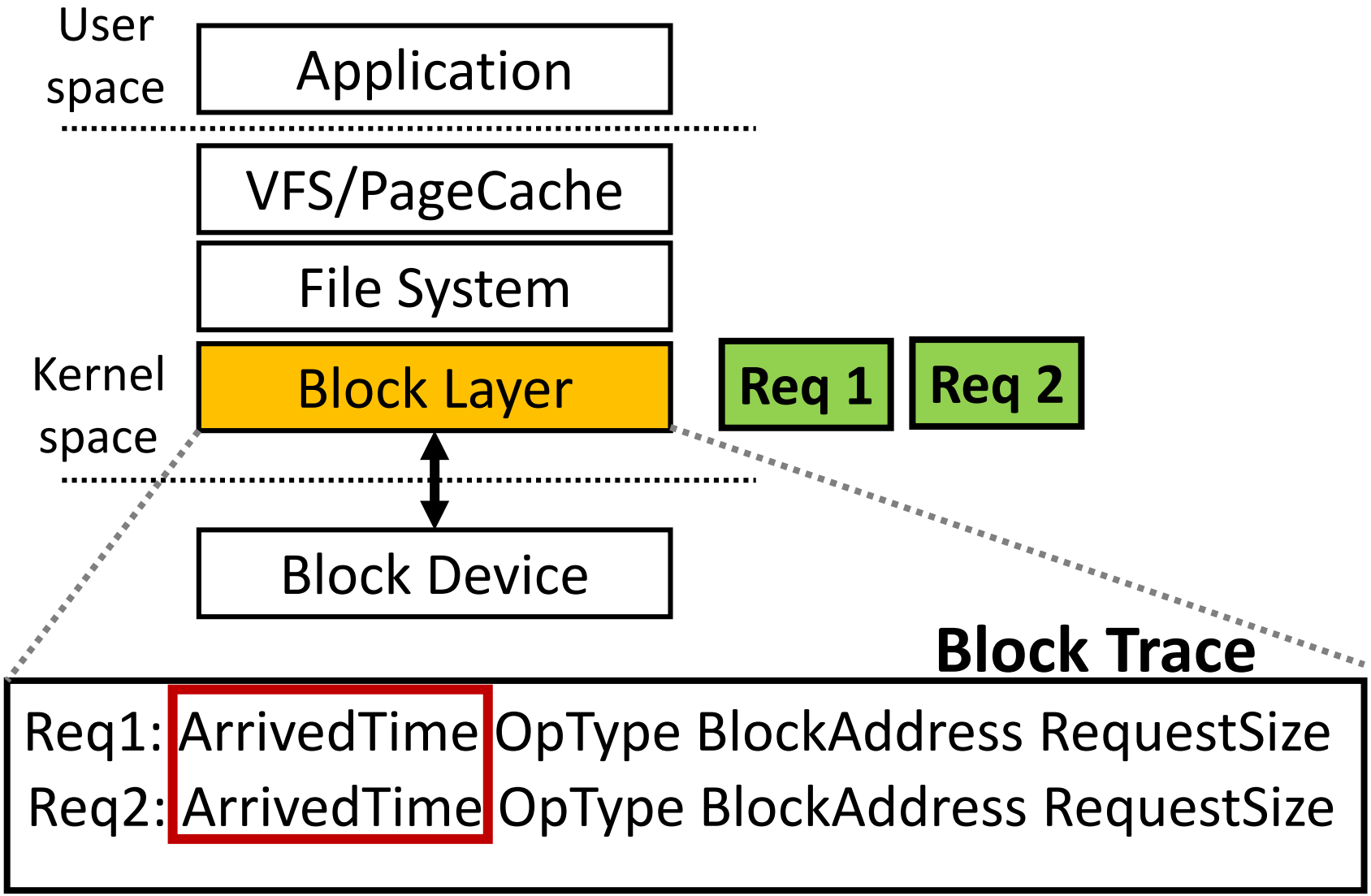
Block Trace

: Consists of detailed **block device access information**



Block Trace

: Consists of detailed **block device access information**

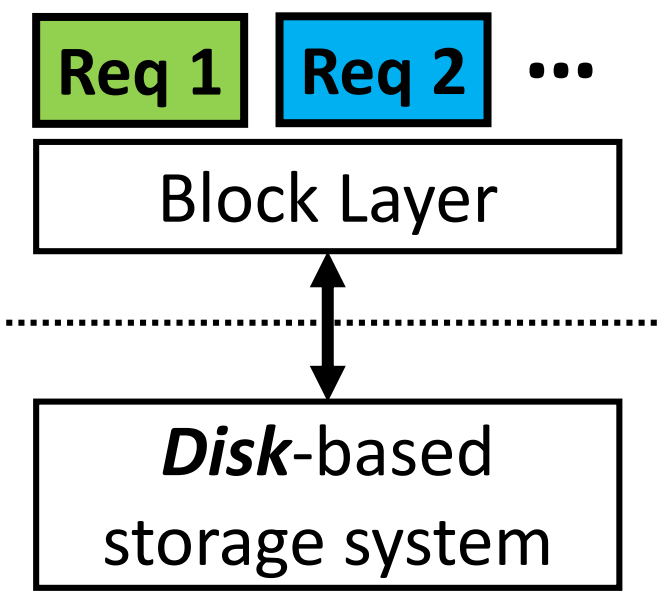


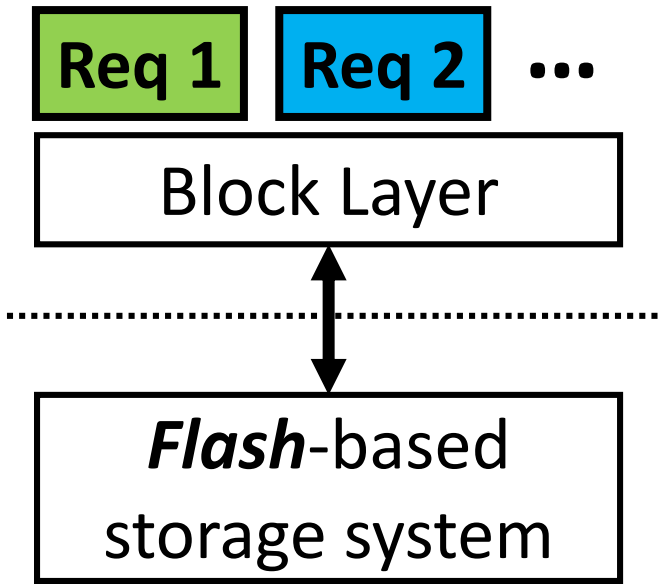
Inaccuracy of existing block traces

Existing traces: collected on the ***disk-based*** storage systems and published around ***2007***

Inaccuracy of existing block traces

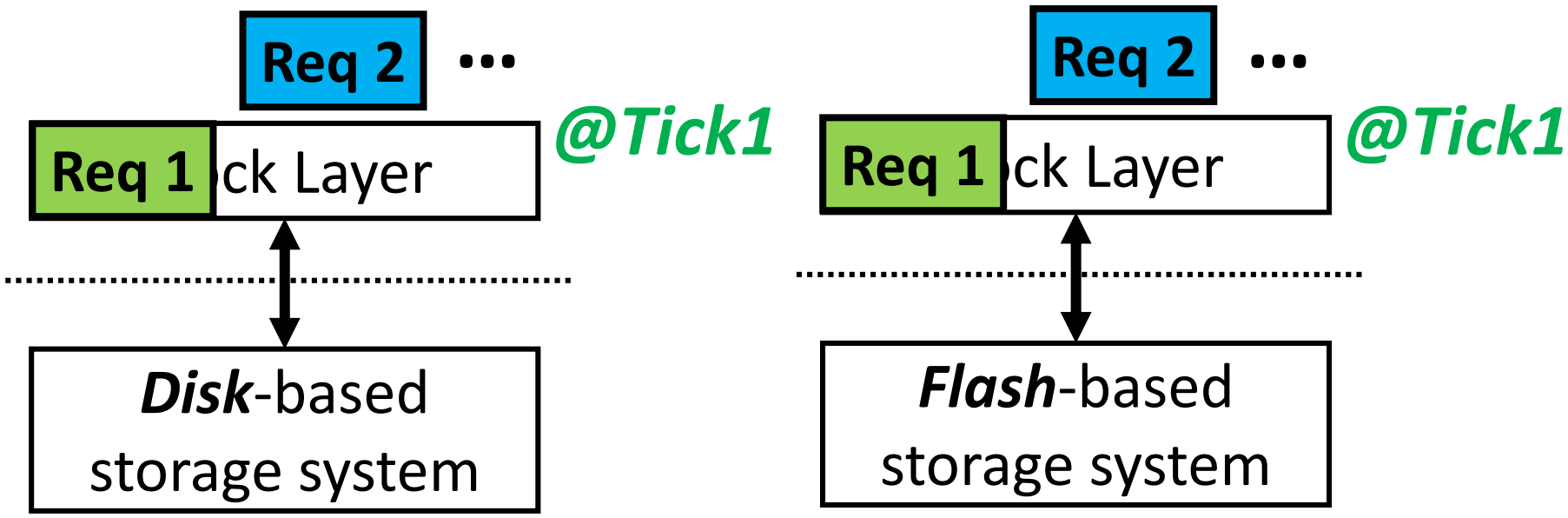
Existing traces: collected on the ***disk-based*** storage systems and published around ***2007***





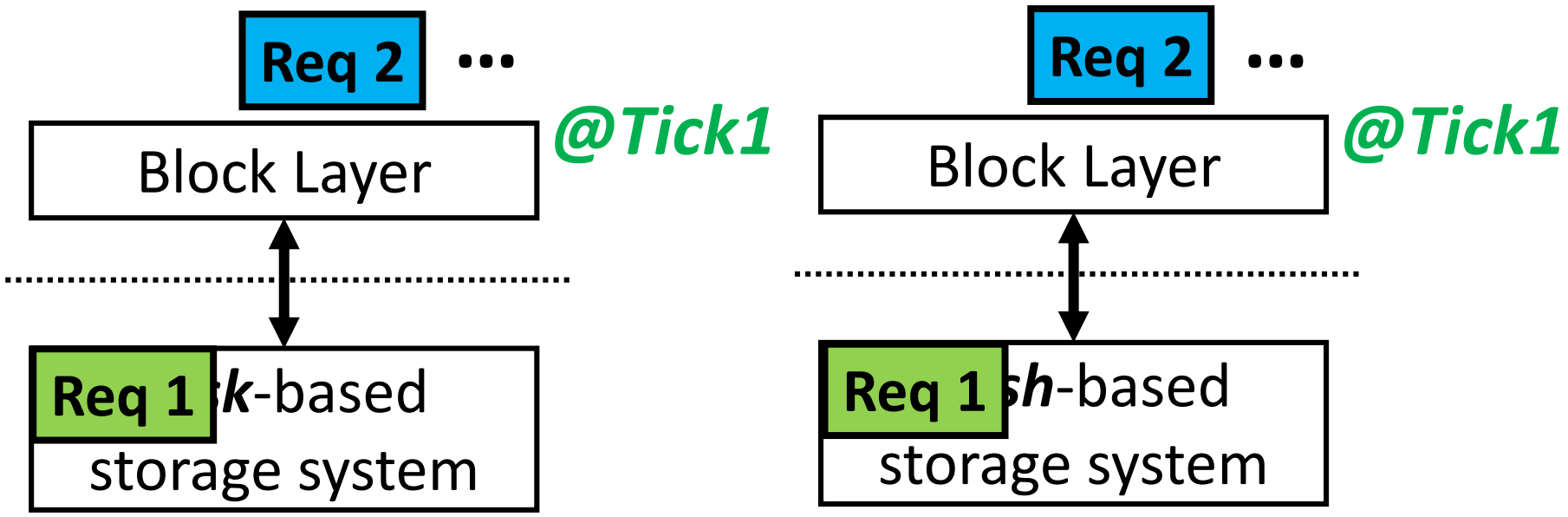
Inaccuracy of existing block traces

Existing traces: collected on the ***disk-based*** storage systems and published around ***2007***



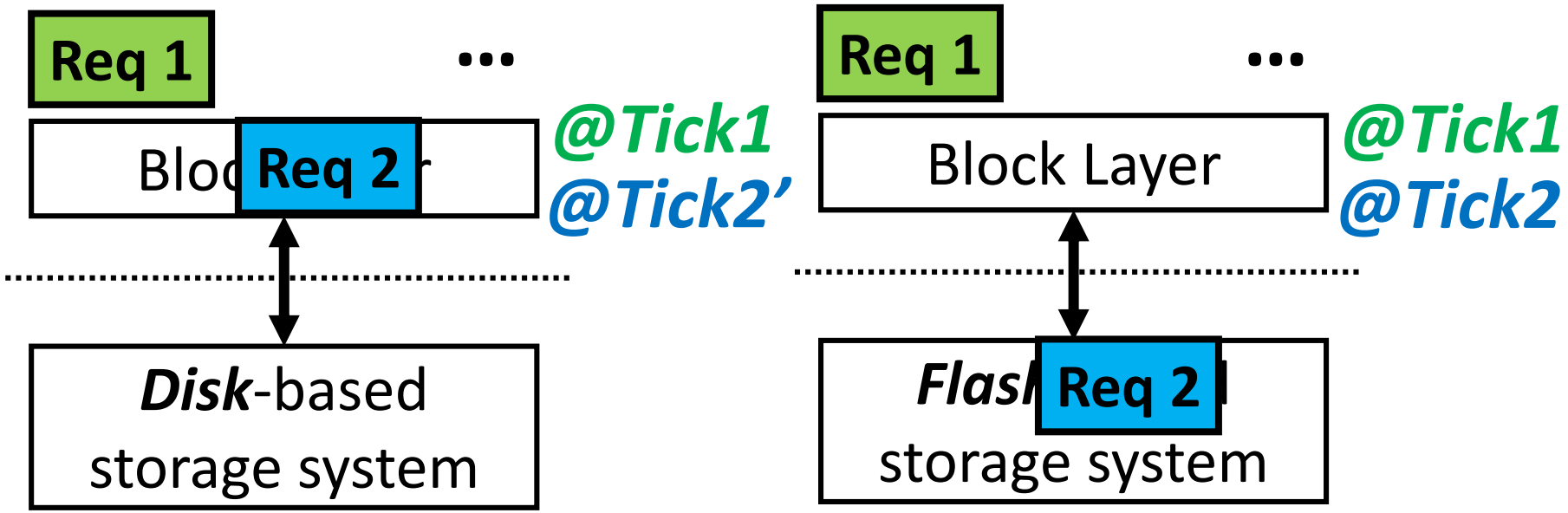
Inaccuracy of existing block traces

Existing traces: collected on the ***disk-based*** storage systems and published around ***2007***



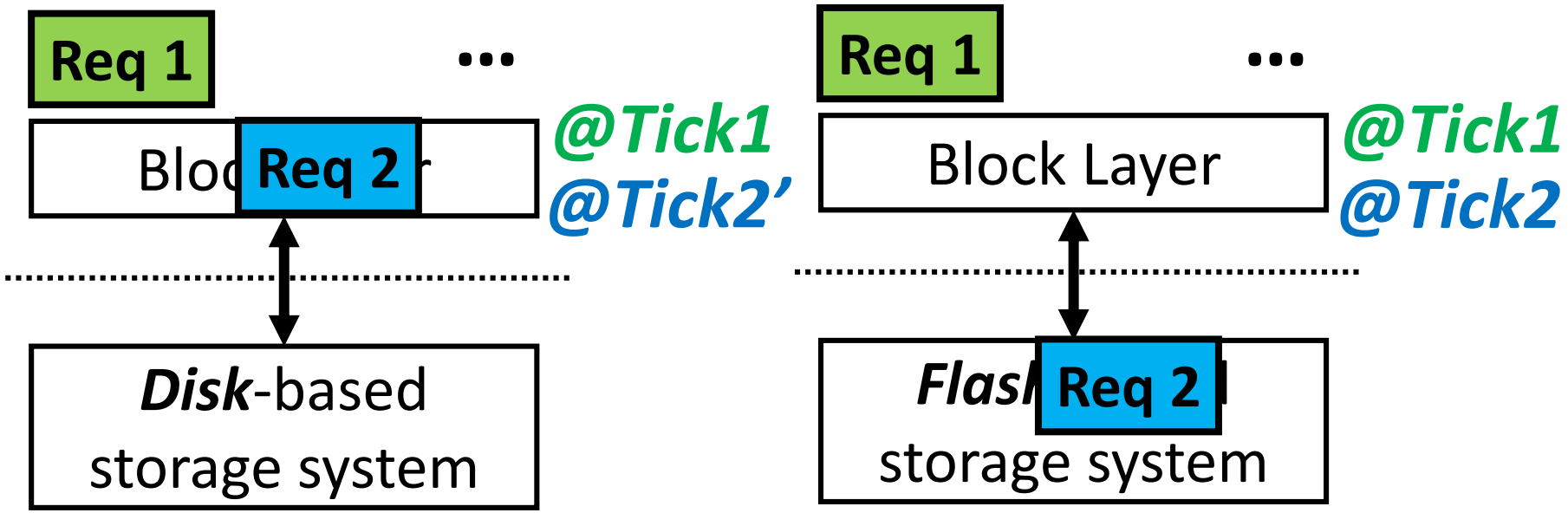
Inaccuracy of existing block traces

Existing traces: collected on the ***disk-based*** storage systems and published around ***2007***



Inaccuracy of existing block traces

Existing traces: collected on the ***disk-based*** storage systems and published around ***2007***



Timing information of block traces should be revised correspond to target storage system

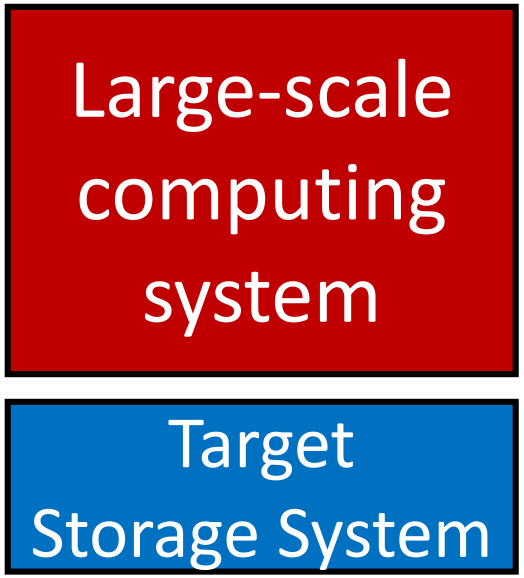
TraceTracker

1. Background of Block Trace
2. Trace Reconstruction Methods
3. Insights on Trace Reconstruction
4. TraceTracker Method
5. Evaluation

Trace Reconstruction Methods

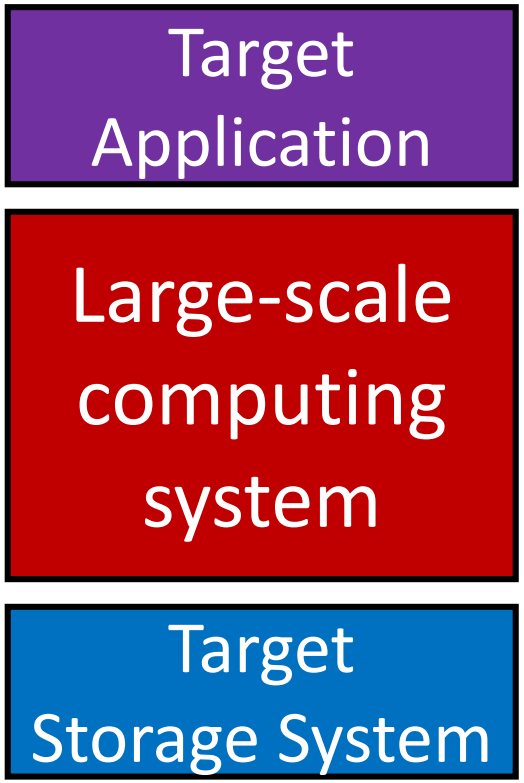
Trace Reconstruction Methods

- ***APPLICATION***



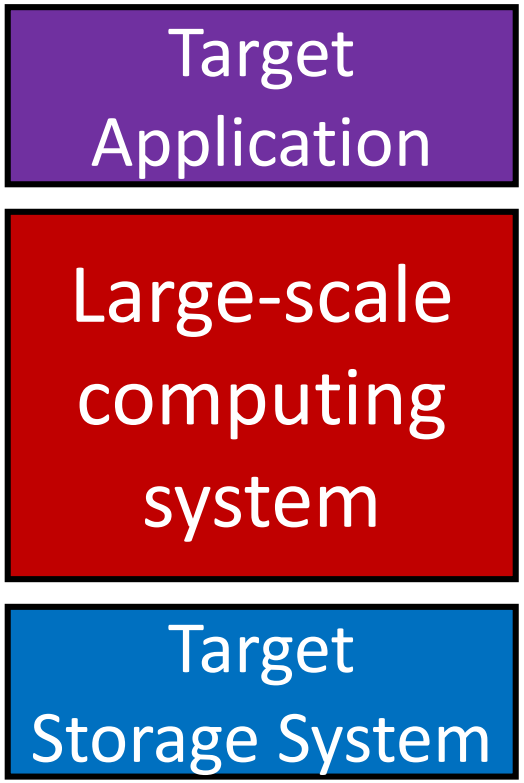
Trace Reconstruction Methods

- ***APPLICATION***

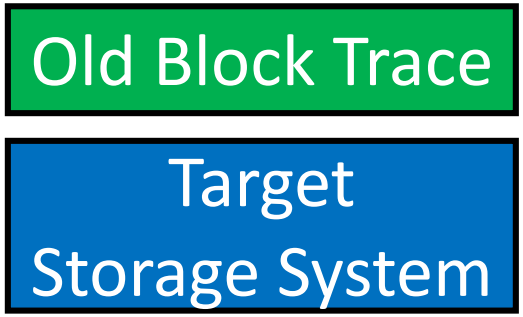


Trace Reconstruction Methods

- ***APPLICATION***

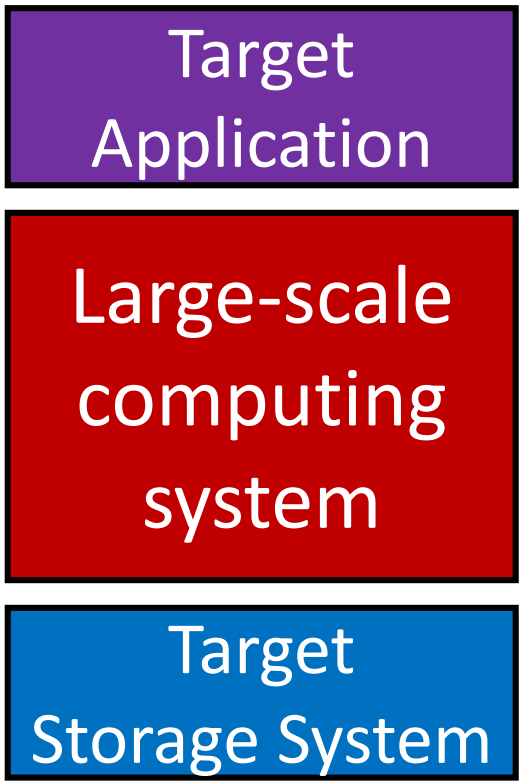


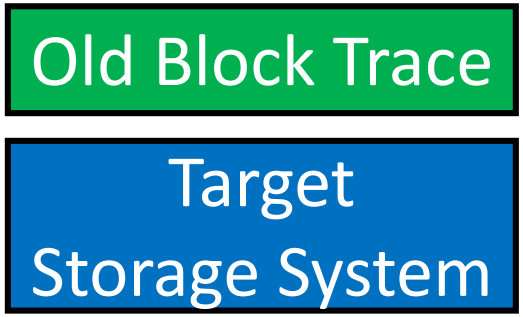
- ***REVISION***

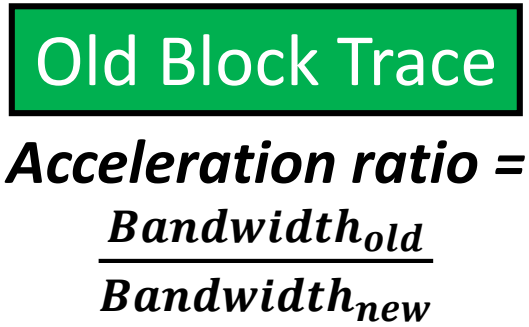


Trace Reconstruction Methods

- ***APPLICATION***
- ***REVISION***
- ***ACCELERATION***







Trace Reconstruction Methods

- ***APPLICATION***
- ***REVISION***
- ***ACCELERATION***

Target Application

Large-scale computing system

Target Storage System

Old Block Trace

Target Storage System

Old Block Trace

Acceleration ratio = $\frac{Bandwidth_{old}}{Bandwidth_{new}}$

Accuracy:	**>>**	**>>**
Cost:	**>>>**	**>>**

Trace Reconstruction Methods

- ***APPLICATION***
- ***REVISION***
- ***ACCELERATION***

Target Application

Large-scale computing system

Target Storage System

Old Block Trace

Target Storage System

Old Block Trace

Do not include runtime context

$$\frac{Bandwidth_{old}}{Bandwidth_{new}}$$

Accuracy:	**>>**	**>>**
Cost:	**>>>**	**>>**

Trace Reconstruction Methods

- ***APPLICATION***
- ***REVISION***
- ***ACCELERATION***

Target Application

User idle times

Large-scale computing system

Target Storage System

Old Block Trace

Target Storage System

Old Block Trace

Do not include runtime context

$$\frac{Bandwidth_{old}}{Bandwidth_{new}}$$

Accuracy:	**>>**	**>>**
Cost:	**>>>**	**>>**

Trace Reconstruction Methods

- ***APPLICATION***
- ***REVISION***
- ***ACCELERATION***

Target Application

Large-scale computing system

Target Storage System

User idle times

System delay times

Old Block Trace

Old Block Trace

Target Storage System

Do not include runtime context

$$\frac{Bandwidth_{old}}{Bandwidth_{new}}$$

Accuracy:	**>>**	**>>**
Cost:	**>>>**	**>>**

CAMEL

Trace Reconstruction Methods

- ***APPLICATION***
- ***REVISION***
- ***ACCELERATION***

Target Application

Large-scale computing system

Target Storage System

Old Block Trace

Target Storage System

Old Block Trace

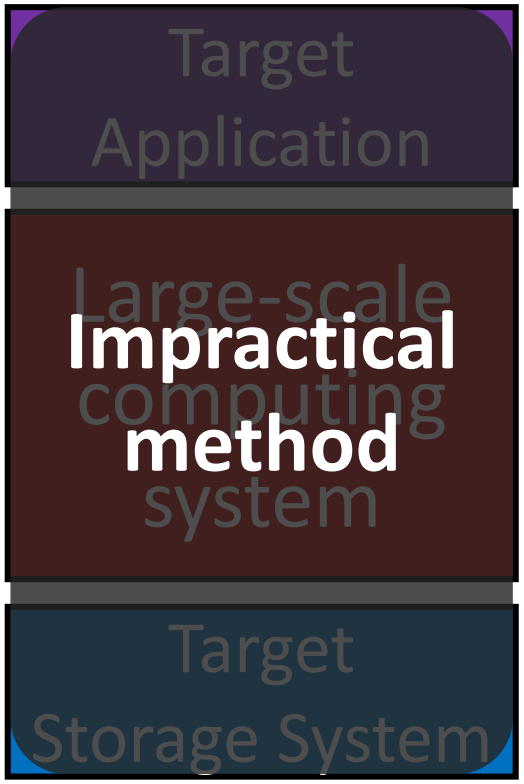
Lack of detailed timing information

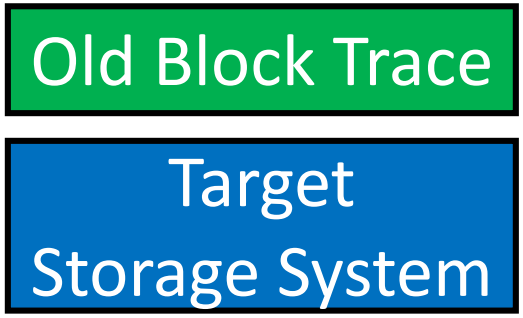
$$Acceleration\ ratio = \frac{Bandwidth_{old}}{Bandwidth_{new}}$$

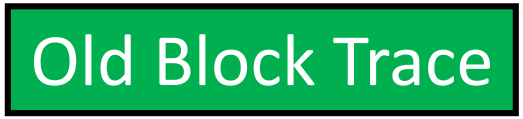
Accuracy:	**>>**	**>>**
Cost:	**>>>**	**>>**

Trace Reconstruction Methods

- ***APPLICATION***
- ***REVISION***
- ***ACCELERATION***



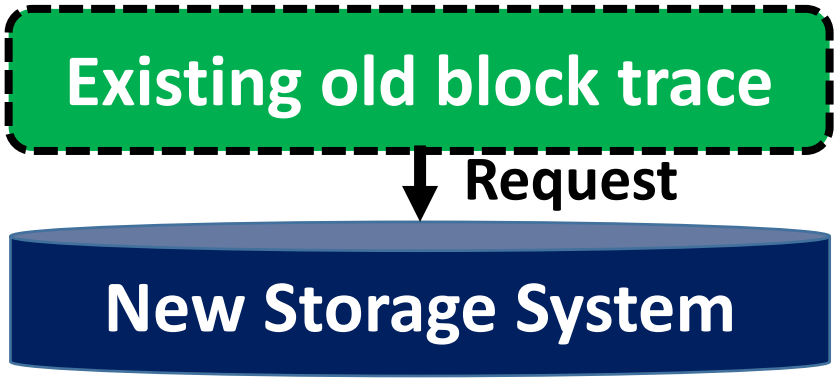




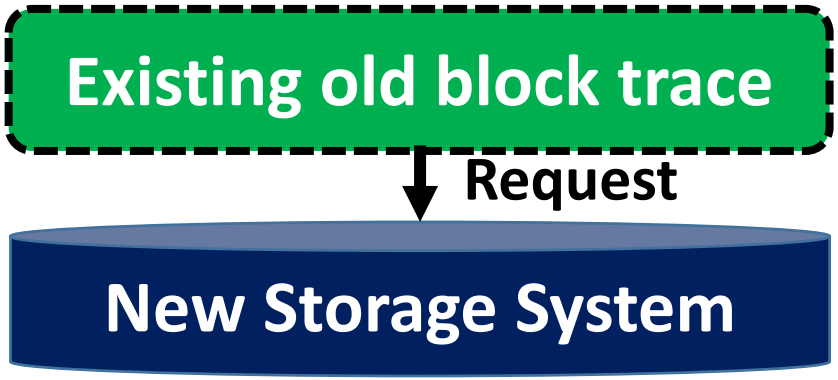
Acceleration ratio = $\frac{Bandwidth_{old}}{Bandwidth_{new}}$

Accuracy:	**>>**	**>>**
Cost:	**>>>**	**>>**

Goal of TraceTracker

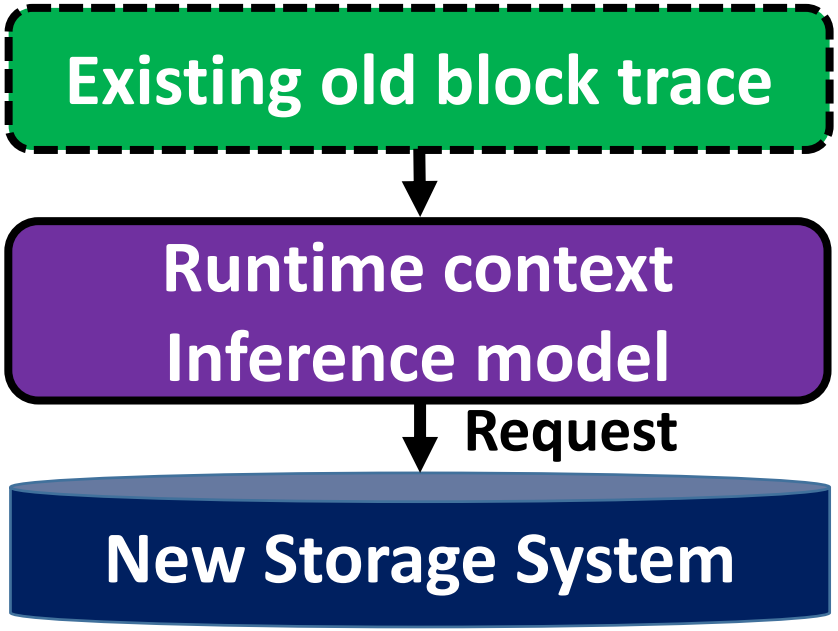


Goal of TraceTracker



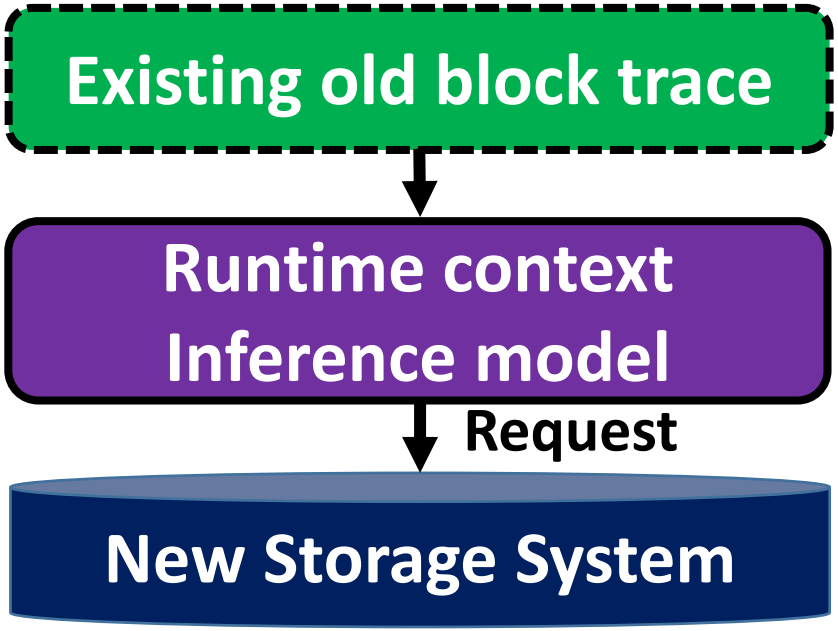
Goal: Accurately reconstruct workloads without application execution (at low cost)

Goal of TraceTracker



Goal: Accurately reconstruct workloads without application execution (at low cost)

Goal of TraceTracker



However, how runtime context can be inferred from block trace?

TraceTracker

1. Background of Block Trace
2. Trace Reconstruction Methods
3. Insights on Trace Reconstruction
4. TraceTracker Method
5. Evaluation

Insights of TraceTracker method

Goal: Infer runtime contexts from timing information (Inter-arrival times) of old block traces

Insights of TraceTracker method

Goal: Infer runtime contexts from timing information (Inter-arrival times) of old block traces

Insight1: Inter-arrival times decomposition

Insights of TraceTracker method

Goal: Infer runtime contexts from timing information (Inter-arrival times) of old block traces

Insight1: Inter-arrival times decomposition

Insight2: Three inter-arrival time types

Insights of TraceTracker method

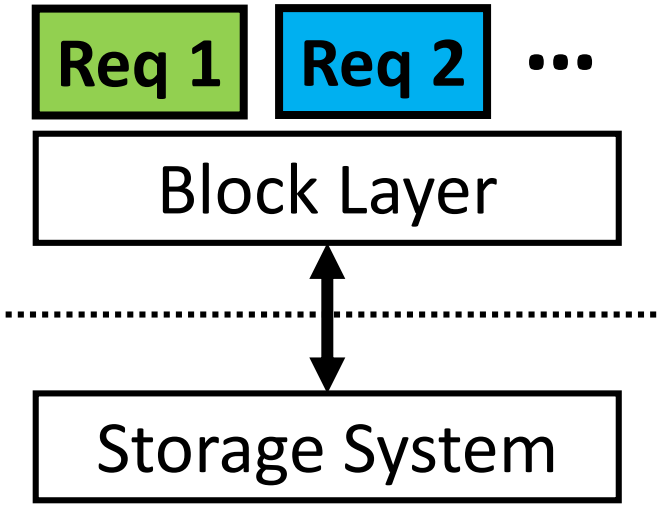
Goal: Infer runtime contexts from timing information (Inter-arrival times) of old block traces

Insight1: Inter-arrival times decomposition

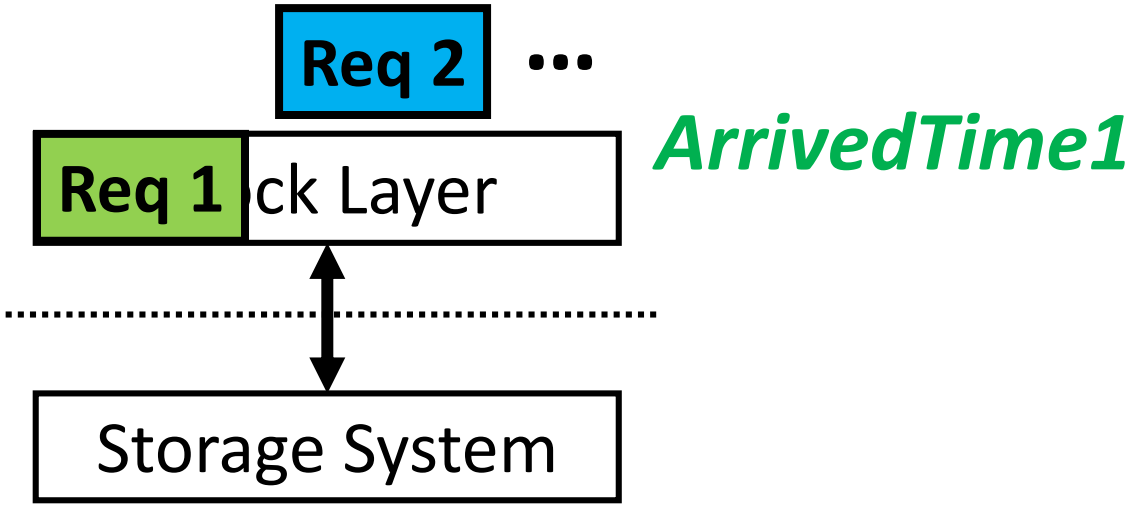
Insight2: Three inter-arrival time types

Insight3: Inter-arrival time components' characteristics

Insight1: Inter-arrival times decomposition

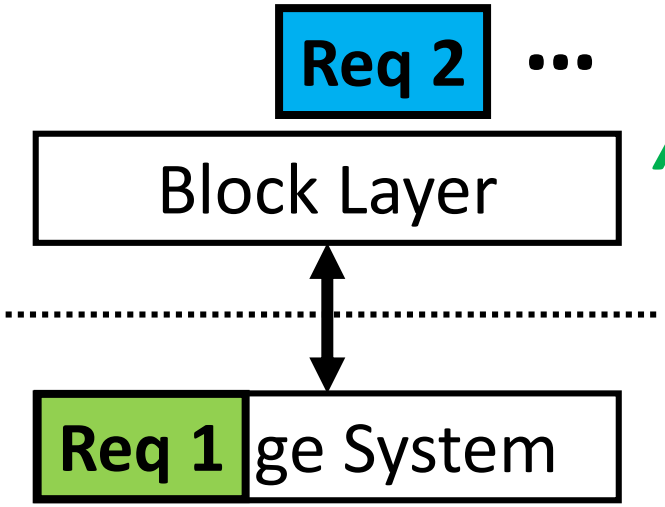


Insight1: Inter-arrival times decomposition



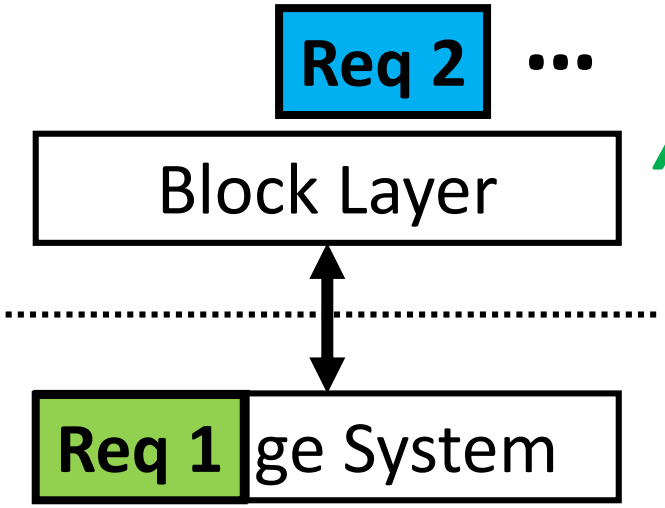
Insight1: Inter-arrival times decomposition

1) Data movement



ArrivedTime1

Insight1: Inter-arrival times decomposition

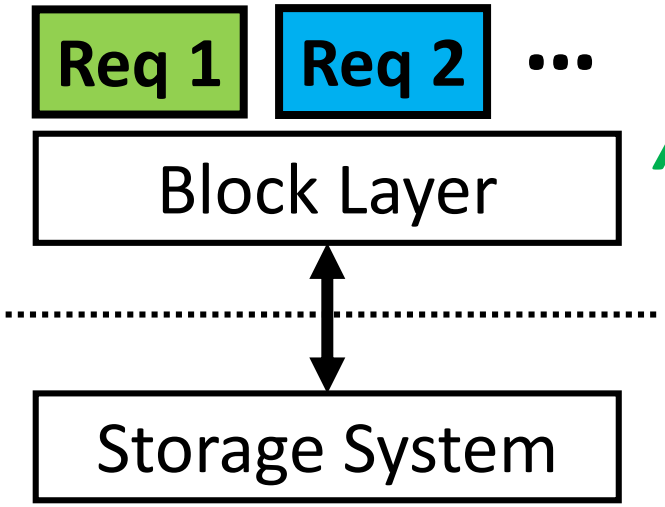


ArrivedTime1

1) Data movement

2) Device time (READ, WRITE)

Insight1: Inter-arrival times decomposition



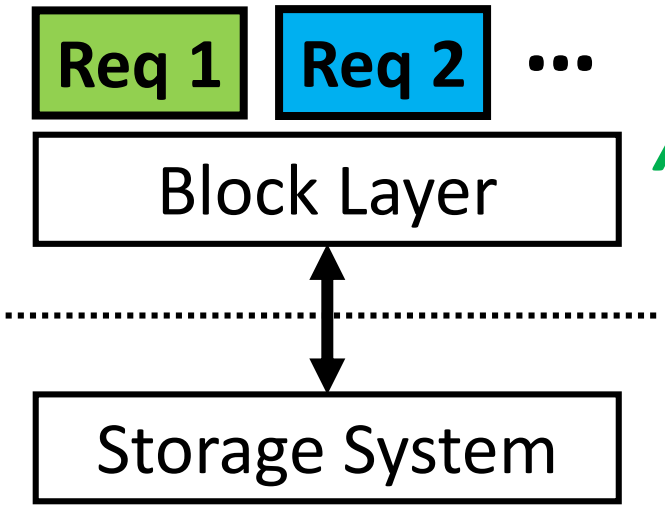
ArrivedTime1

1) Data movement

2) Device time (READ, WRITE)

Insight1: Inter-arrival times decomposition

3) User idle times



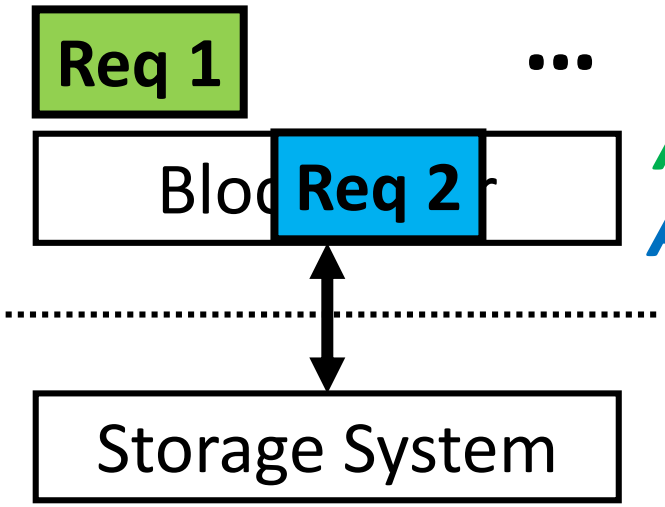
ArrivedTime1

1) Data movement

2) Device time (READ, WRITE)

Insight1: Inter-arrival times decomposition

3) User idle times

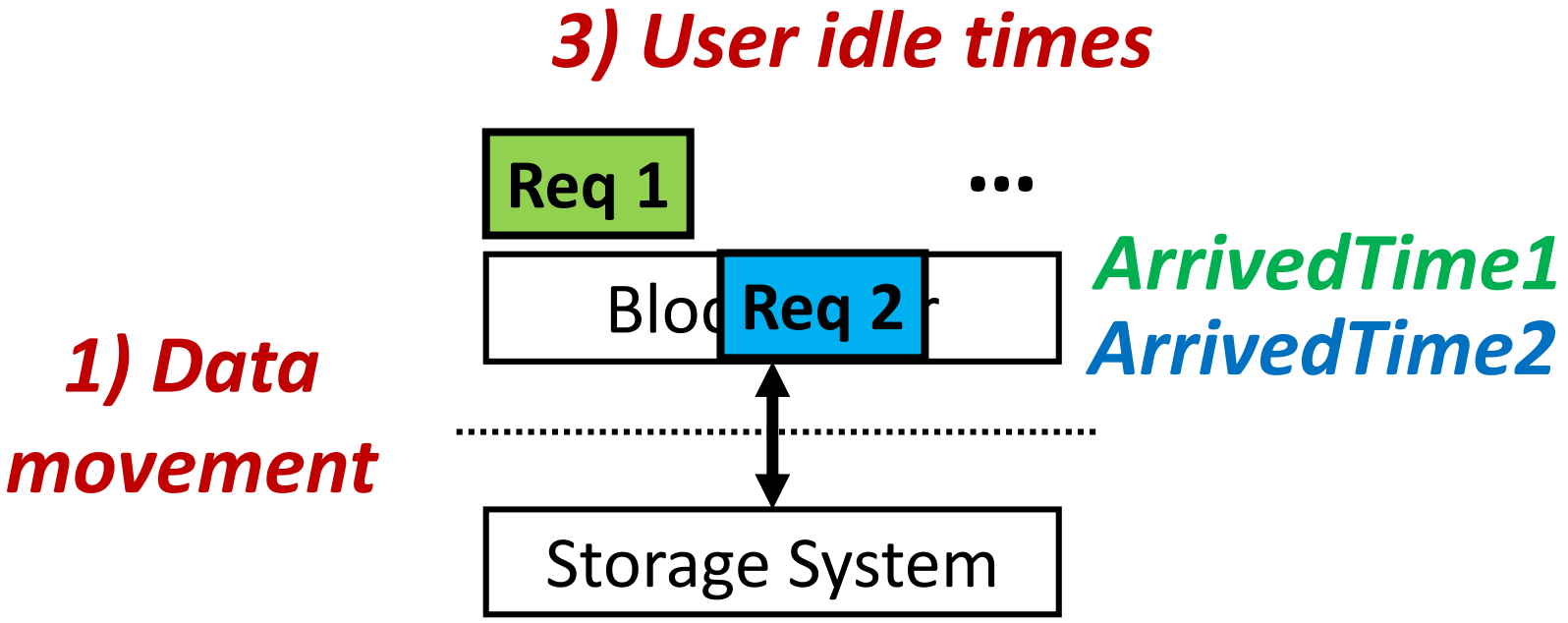


ArrivedTime1
ArrivedTime2

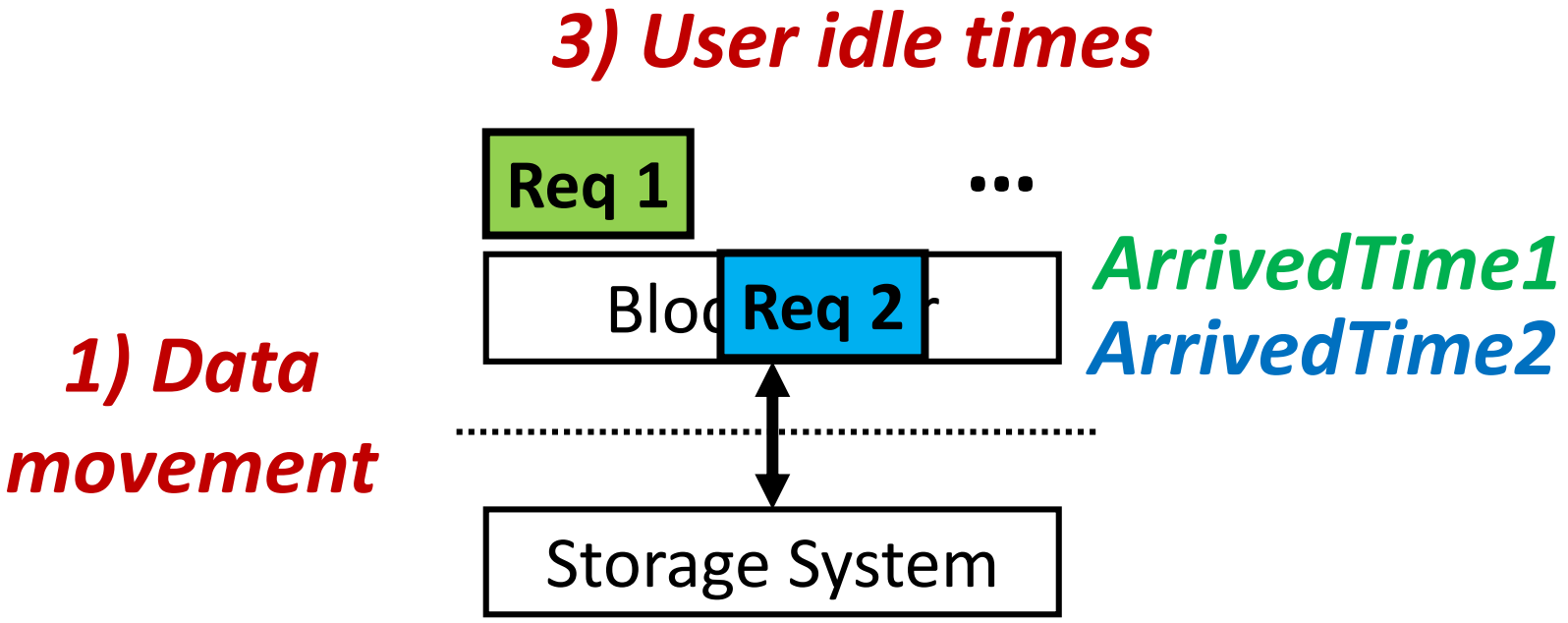
1) Data movement

2) Device time (READ, WRITE)

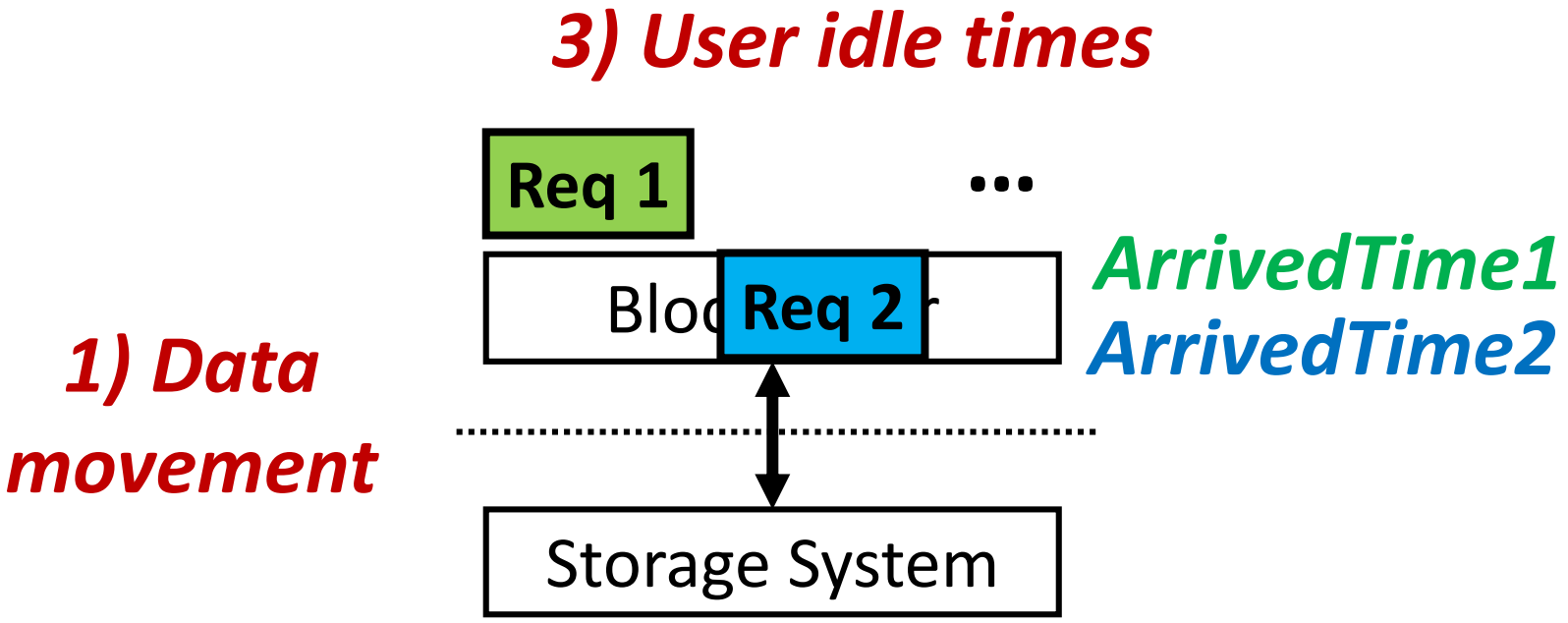
Insight1: Inter-arrival times decomposition

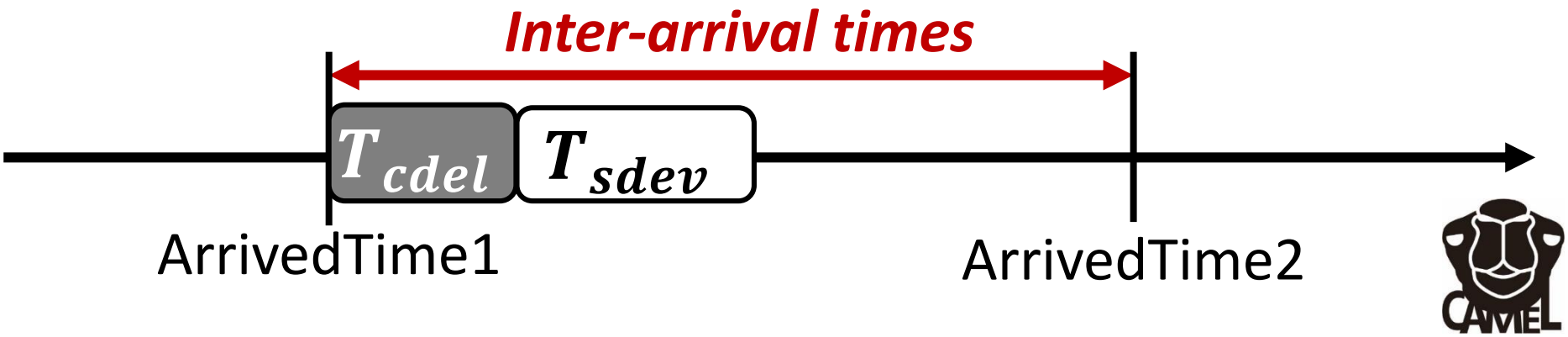


Insight1: Inter-arrival times decomposition

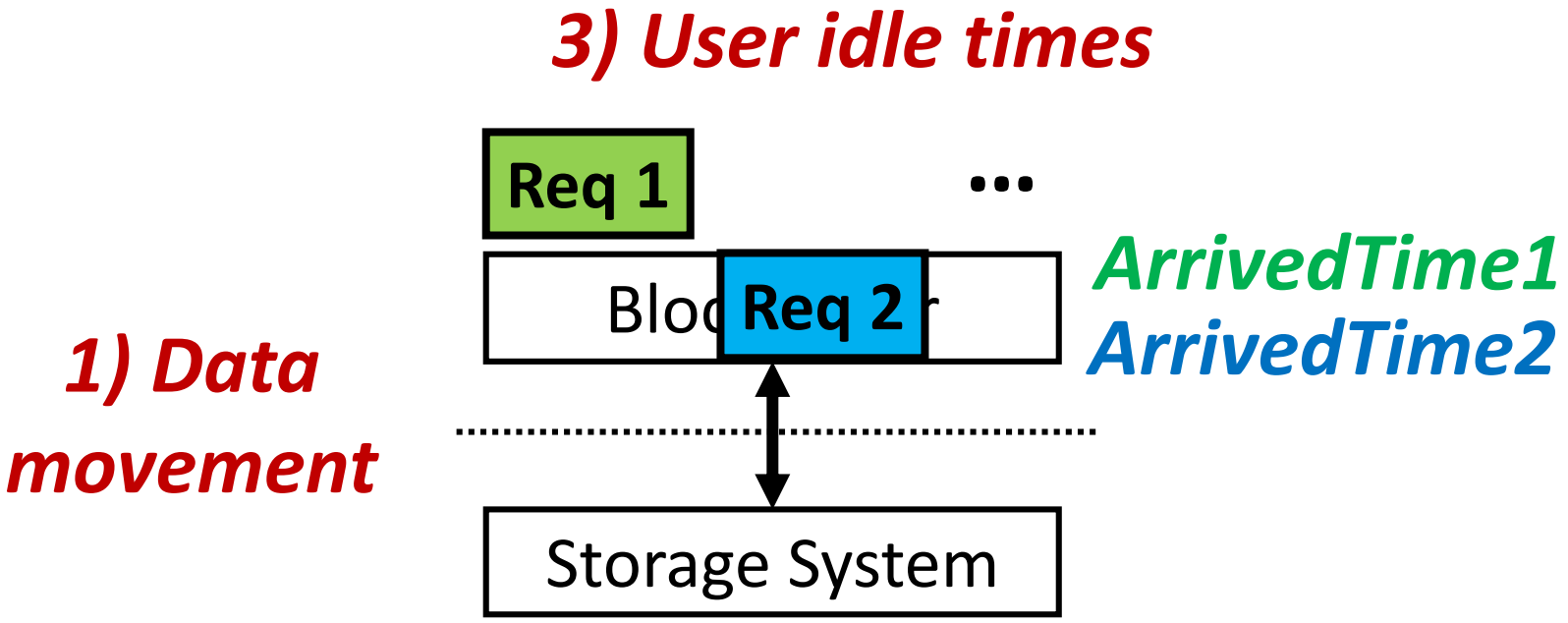


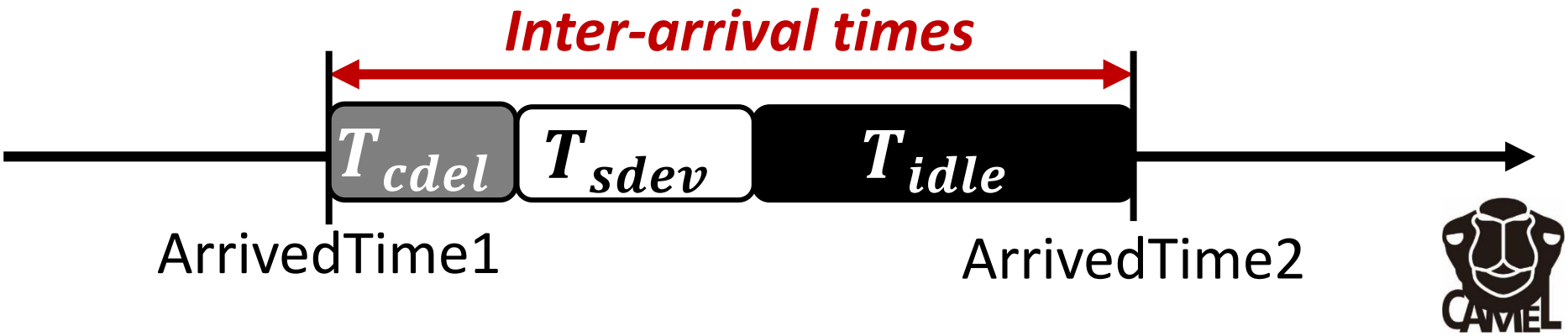
Insight1: Inter-arrival times decomposition





Insight1: Inter-arrival times decomposition





Insight2: Three inter-arrival times types
(I/O execution mode, application busy/idle status)

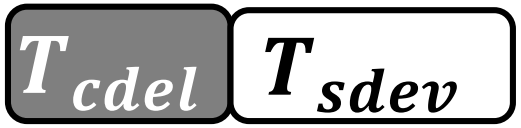
Insight2: Three inter-arrival times types
(I/O execution mode, application busy/idle status)

Sync I/O,
Non-busy status

Insight2: Three inter-arrival times types
(I/O execution mode, application busy/idle status)

Sync I/O,
Non-busy status

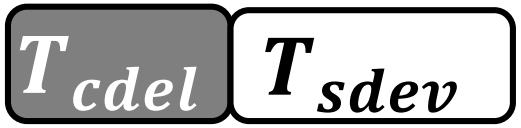
Sync I/O,
Busy status



Insight2: Three inter-arrival times types
(I/O execution mode, application busy/idle status)

Sync I/O,
Non-busy status

Sync I/O,
Busy status



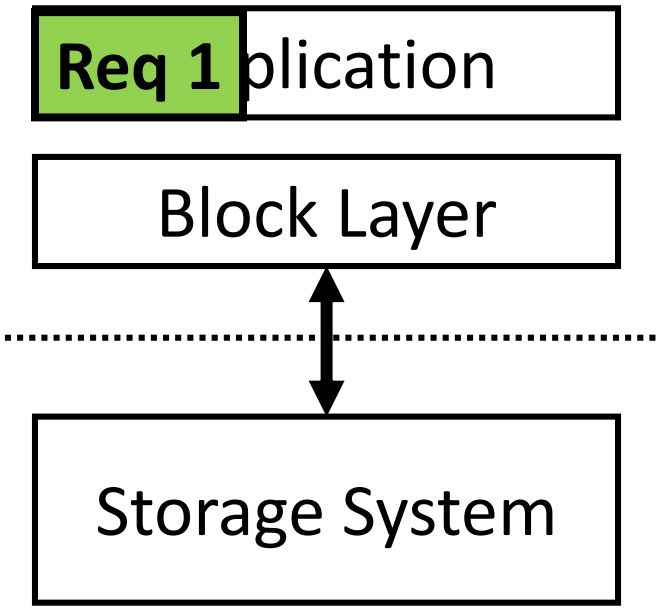
Async I/O

Insight3: Different characteristics of three inter-arrival times components ($T_{cdel}, T_{sdev}, T_{idle}$)

	T_{cdel}	T_{sdev}	T_{idle}
Time length			
Predictable	O	O	X (rare)

Insight3: Different characteristics of three inter-arrival times components ($T_{cdel}, T_{sdev}, T_{idle}$)

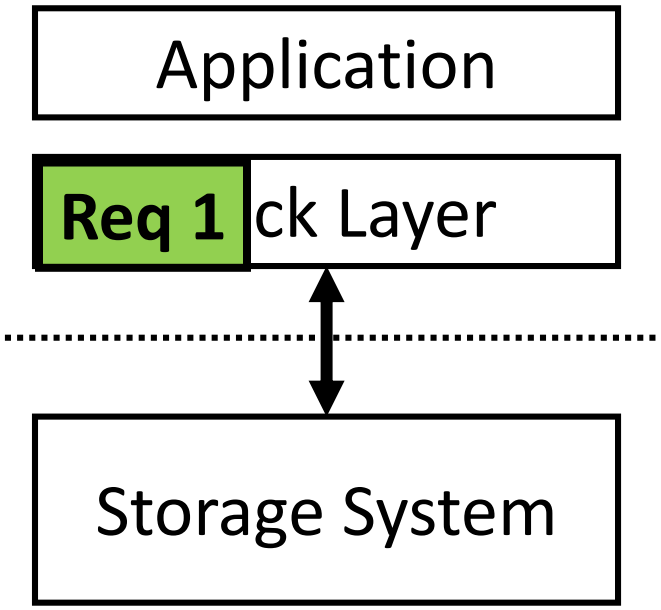
	T_{cdel}	T_{sdev}	T_{idle}
Time length	*short* ← → *long*		
Predictable	O	O	X (rare)



Insight3: Different characteristics of three inter-arrival times components ($T_{cdel}, T_{sdev}, T_{idle}$)

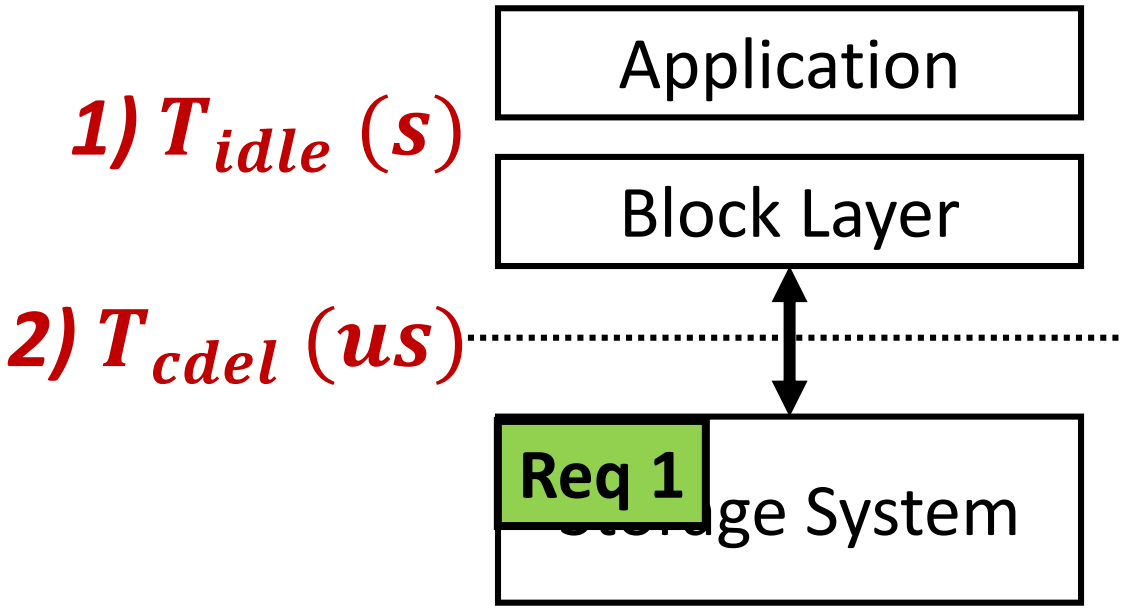
	T_{cdel}	T_{sdev}	T_{idle}
Time length	*short* ← → *long*		
Predictable	O	O	X (rare)

1) T_{idle} (s)



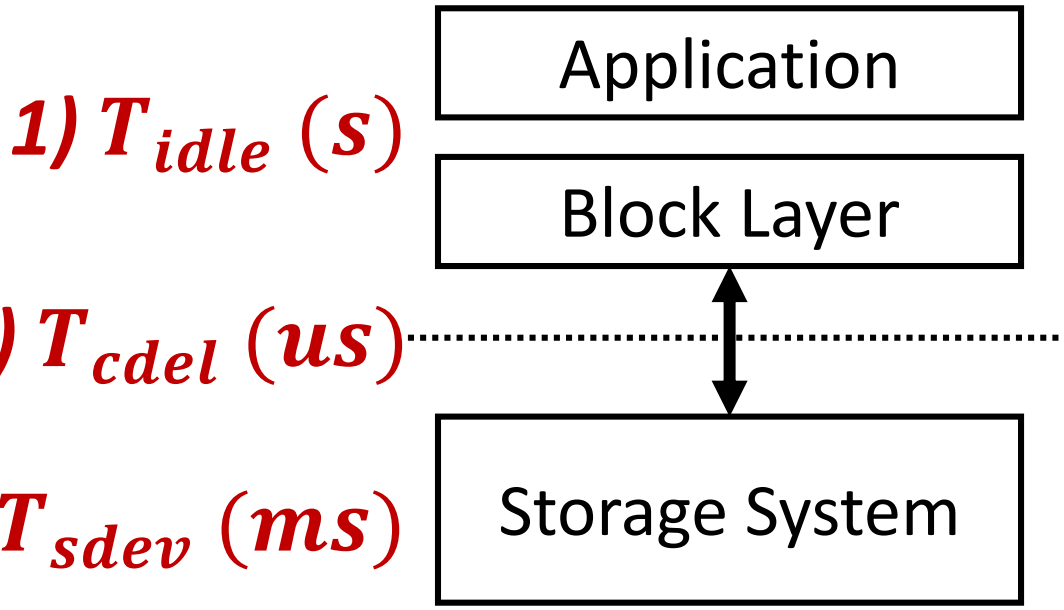
Insight3: Different characteristics of three inter-arrival times components ($T_{cdel}, T_{sdev}, T_{idle}$)

	T_{cdel}	T_{sdev}	T_{idle}
Time length	*short* ← → *long*		
Predictable	O	O	X (rare)



Insight3: Different characteristics of three inter-arrival times components ($T_{cdel}, T_{sdev}, T_{idle}$)

	T_{cdel}	T_{sdev}	T_{idle}
Time length	*short* ← → *long*		
Predictable	O	O	X (rare)



TraceTracker

1. Background of Block Trace
2. Trace Reconstruction Methods
3. Insights on Trace Reconstruction
4. **TraceTracker Method**
5. Evaluation

Overview of TraceTracker

Overview of TraceTracker

Pre-software evaluation

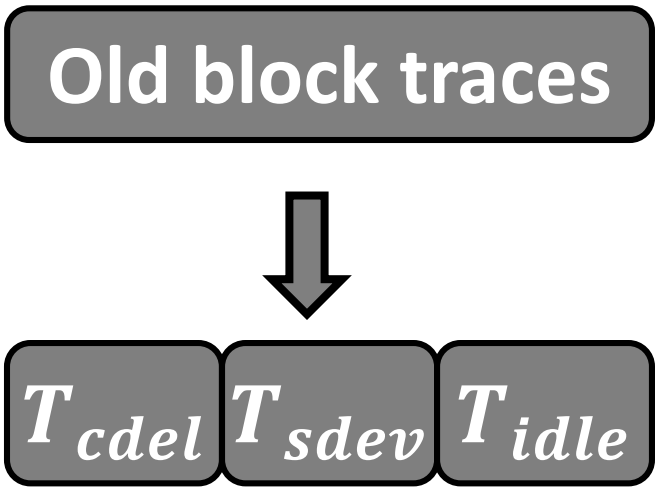
Overview of TraceTracker

Pre-software evaluation  ***Hardware evaluation***

Overview of TraceTracker

Overview of TraceTracker

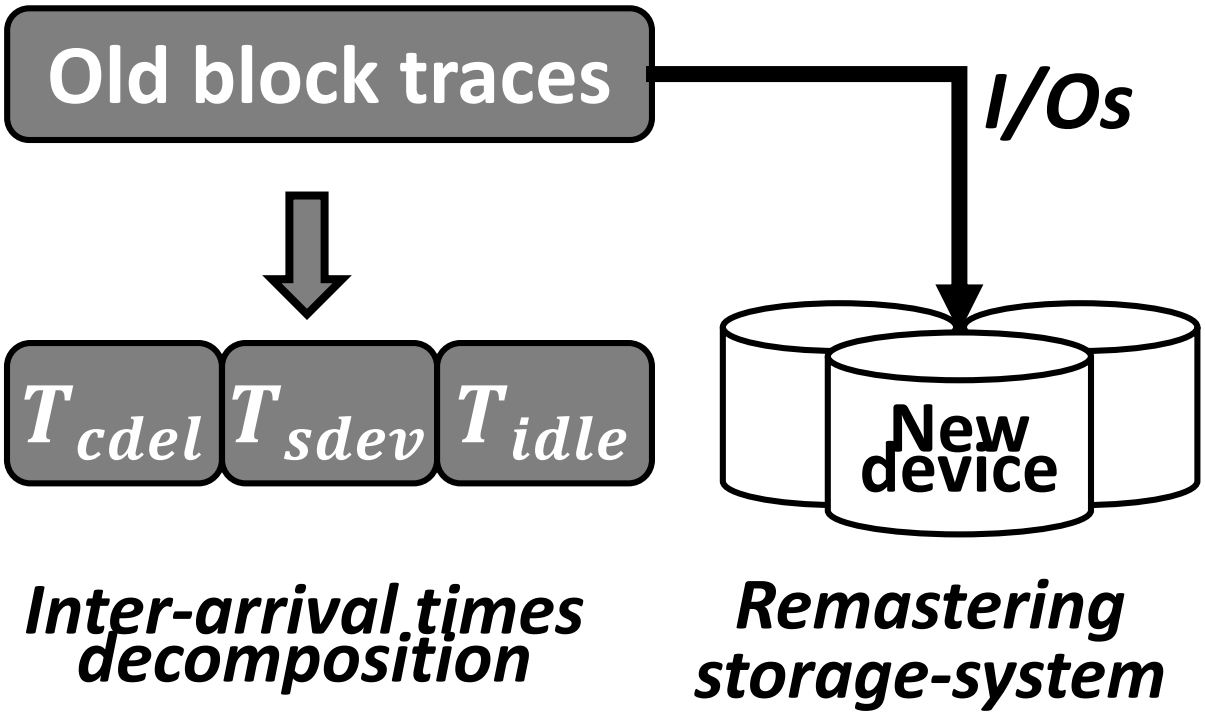
Pre-software evaluation → *Hardware evaluation* → *Post-software evaluation*



Inter-arrival times decomposition

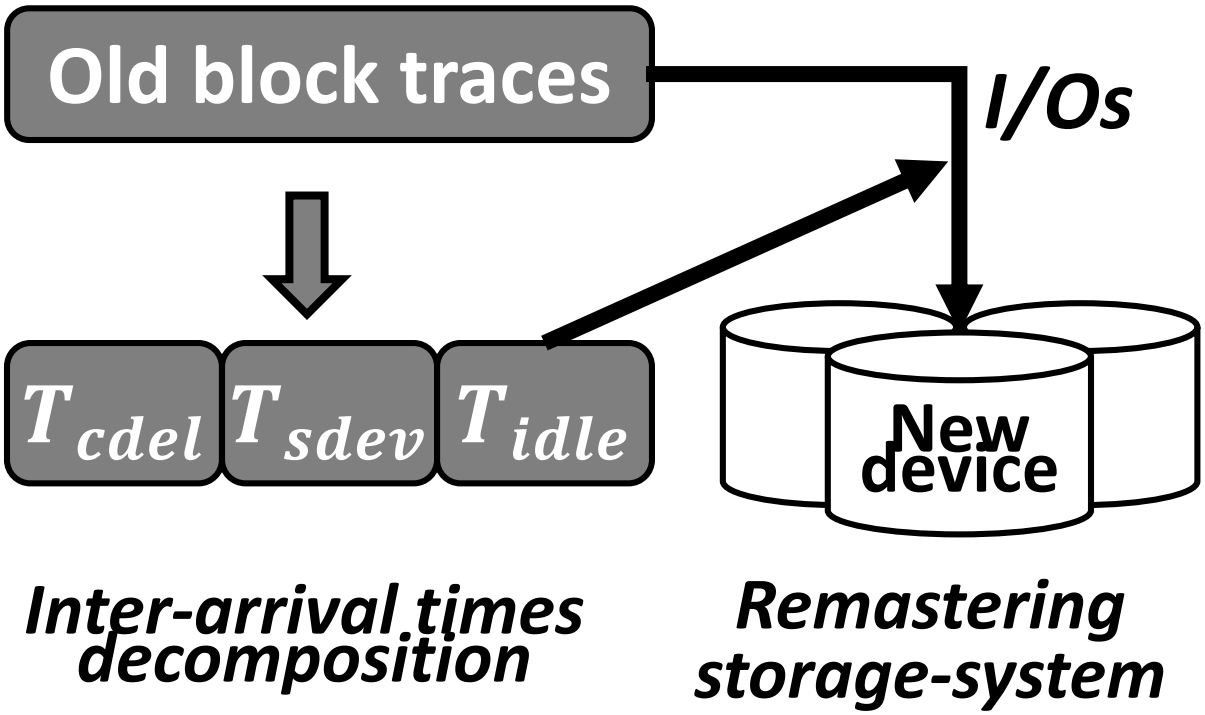
Overview of TraceTracker

Pre-software evaluation → *Hardware evaluation* → *Post-software evaluation*

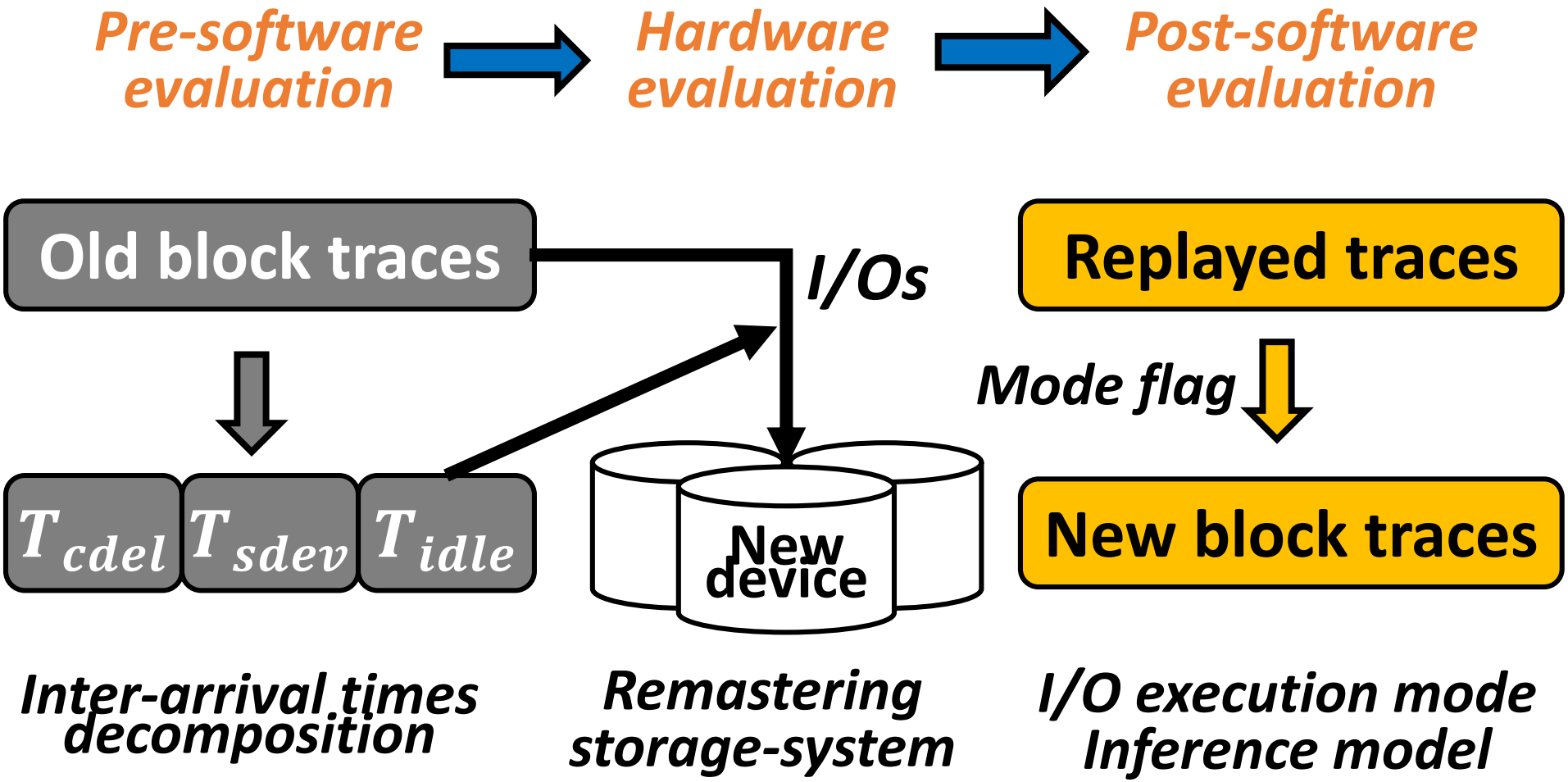


Overview of TraceTracker

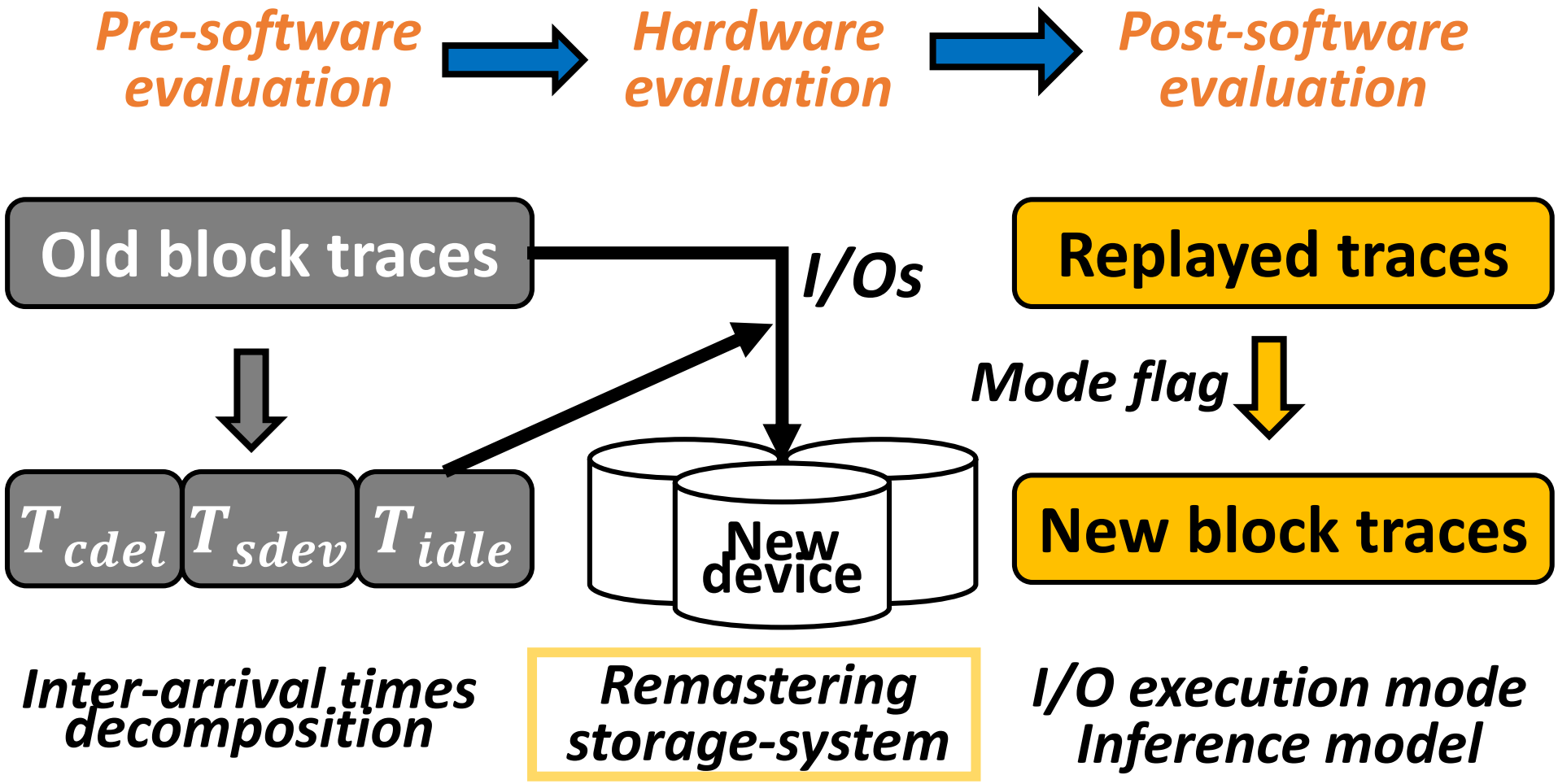
Pre-software evaluation → *Hardware evaluation* → *Post-software evaluation*



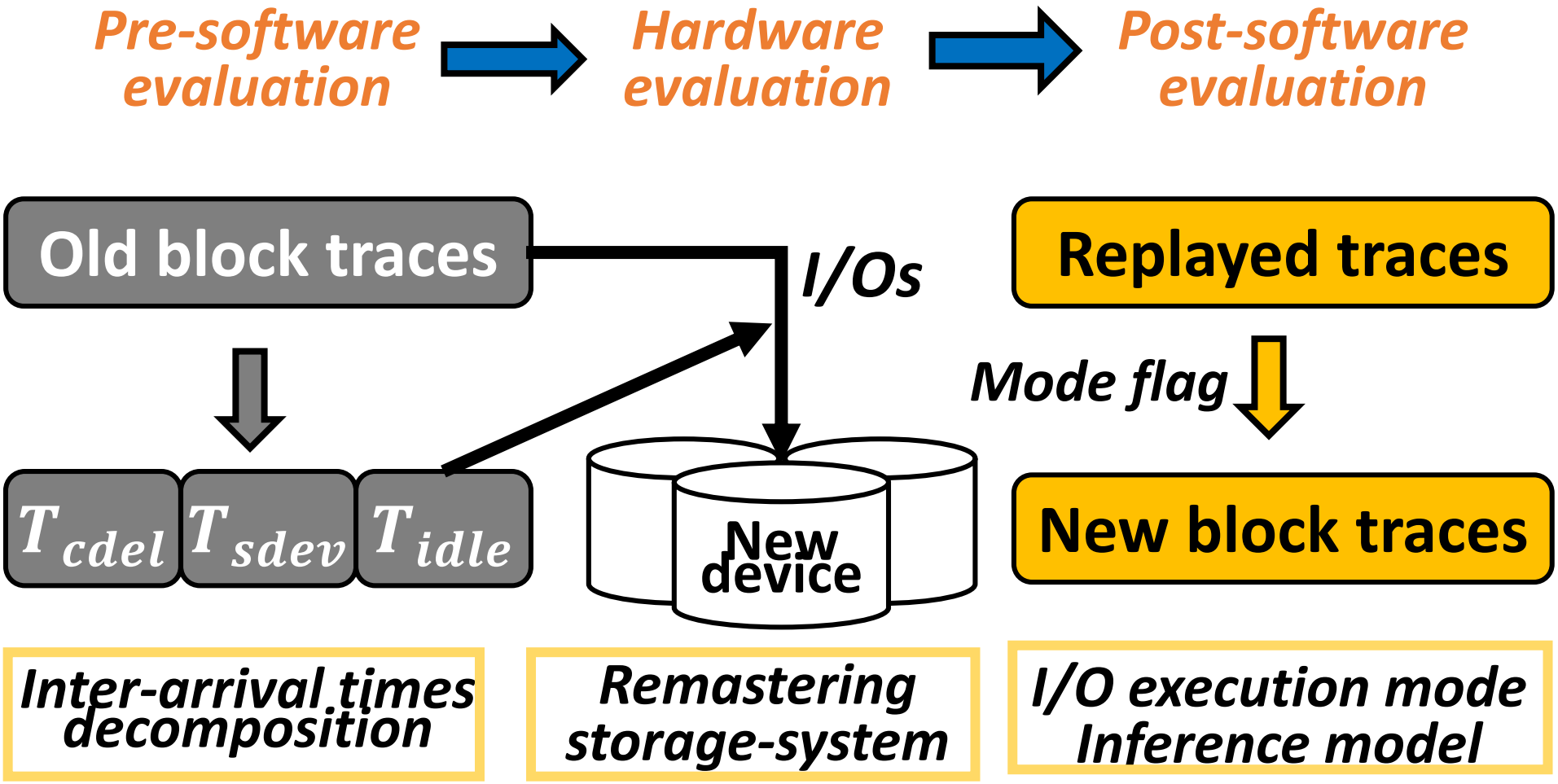
Overview of TraceTracker



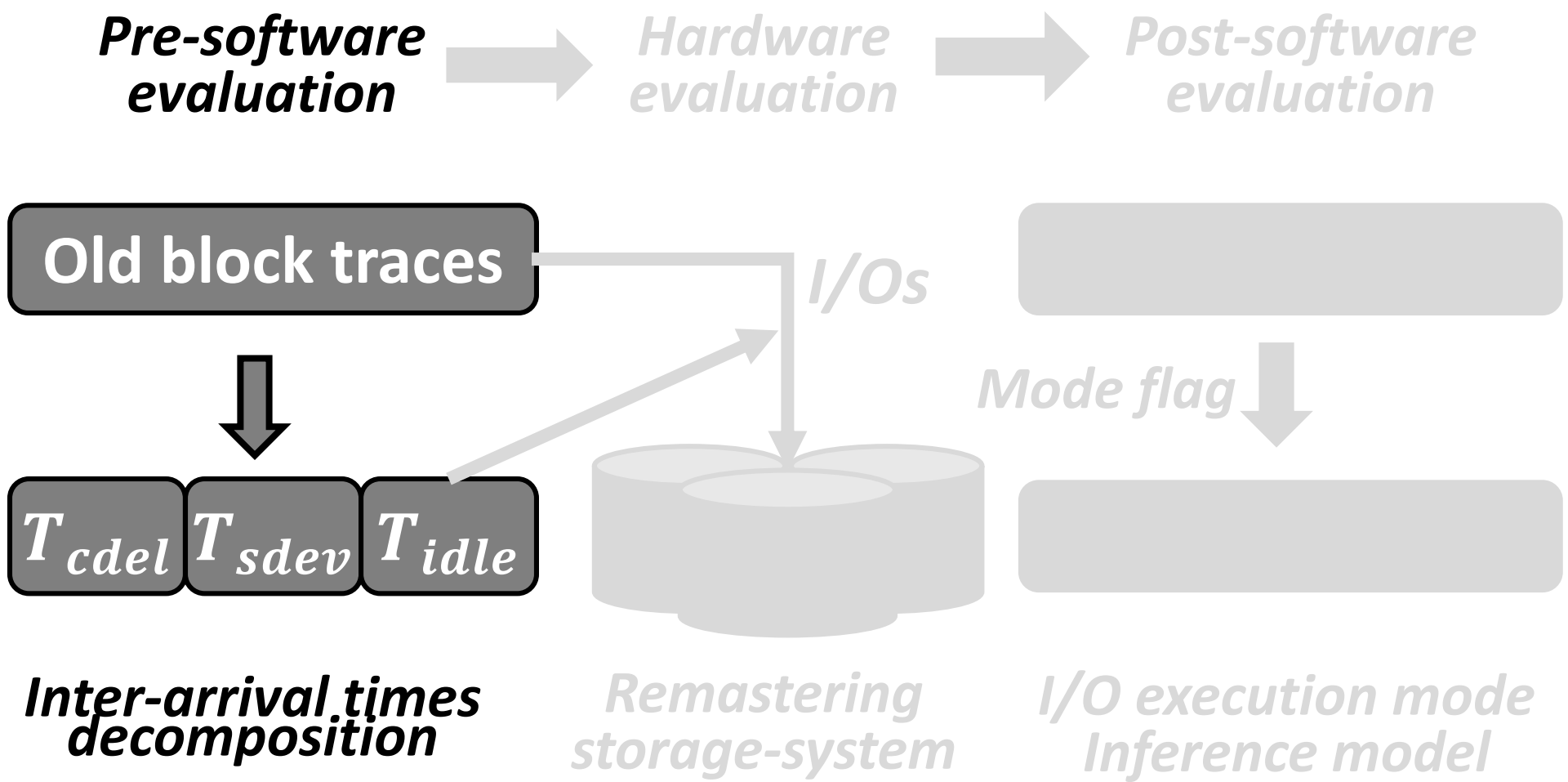
Overview of TraceTracker



Overview of TraceTracker



Overview of TraceTracker



Request classification

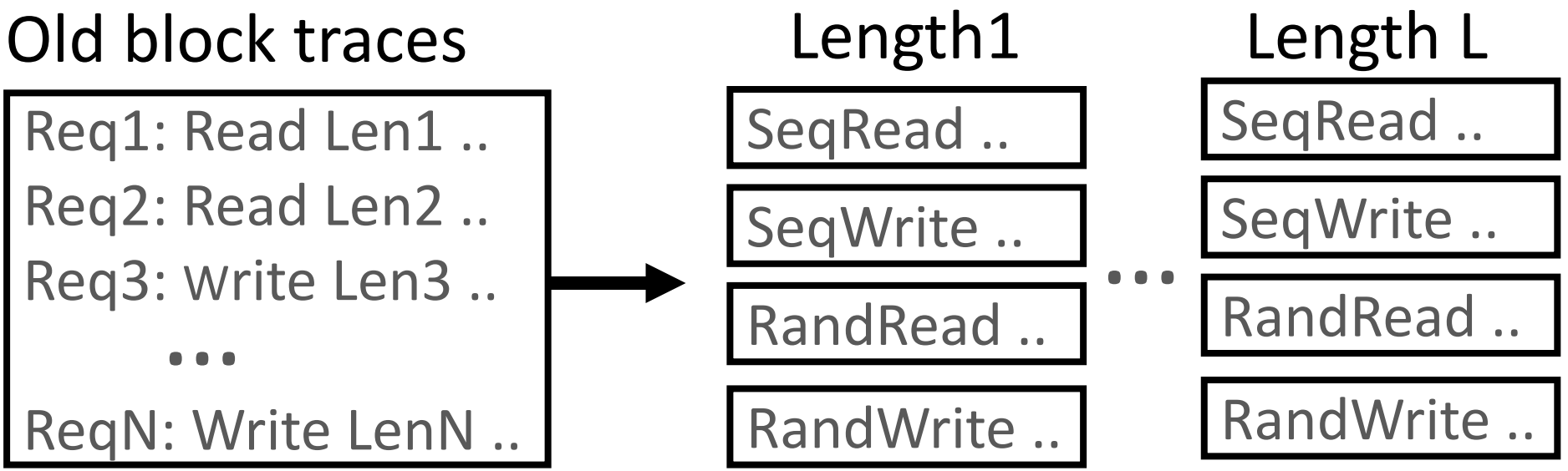
Key idea: Classify all requests in trace with access pattern, operation type, request length

Old block traces

Req1: Read Len1 ..
Req2: Read Len2 ..
Req3: Write Len3 ..
...
ReqN: Write LenN ..

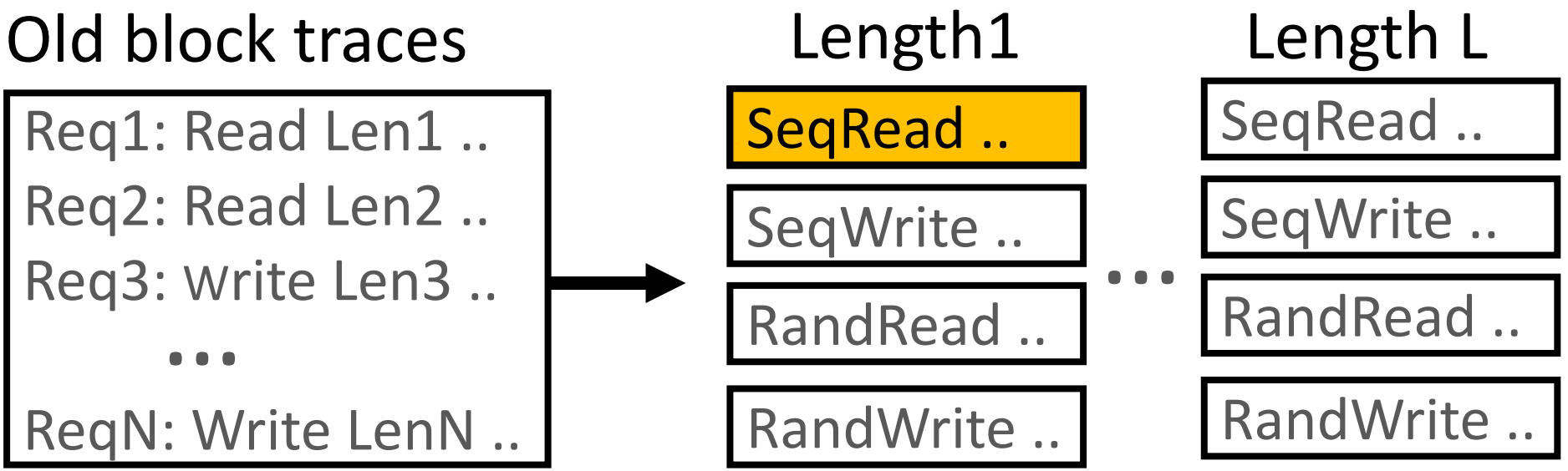
Request classification

Key idea: Classify all requests in trace with access pattern, operation type, request length



Request classification

Key idea: Classify all requests in trace with access pattern, operation type, request length



Inter-arrival times distribution

Key idea: Easily decompose T_{intt} from same category requests' T_{intt} distribution

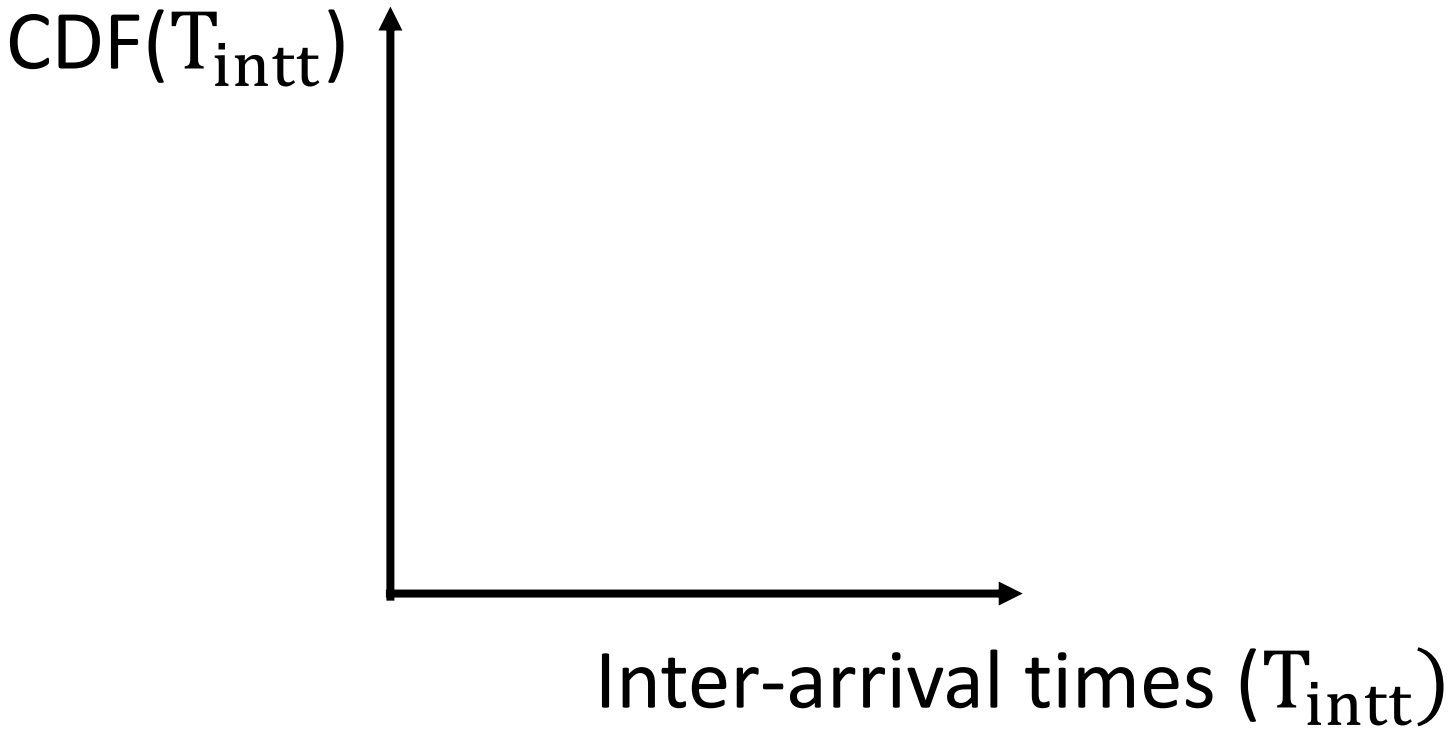
Inter-arrival times distribution

Key idea: Easily decompose T_{intt} from same category requests' T_{intt} distribution

Length1 sequential read

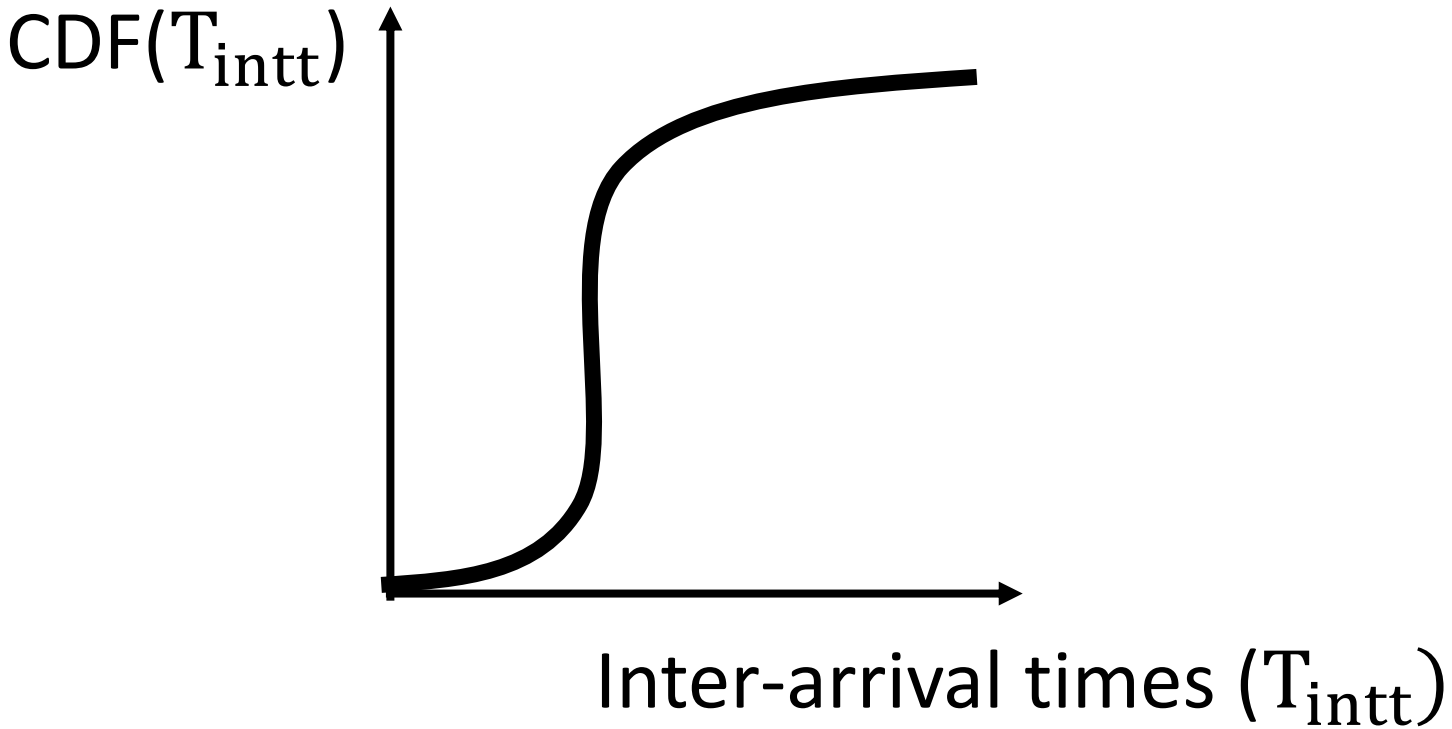
Inter-arrival times distribution

Key idea: Easily decompose T_{intt} from same category requests' T_{intt} distribution



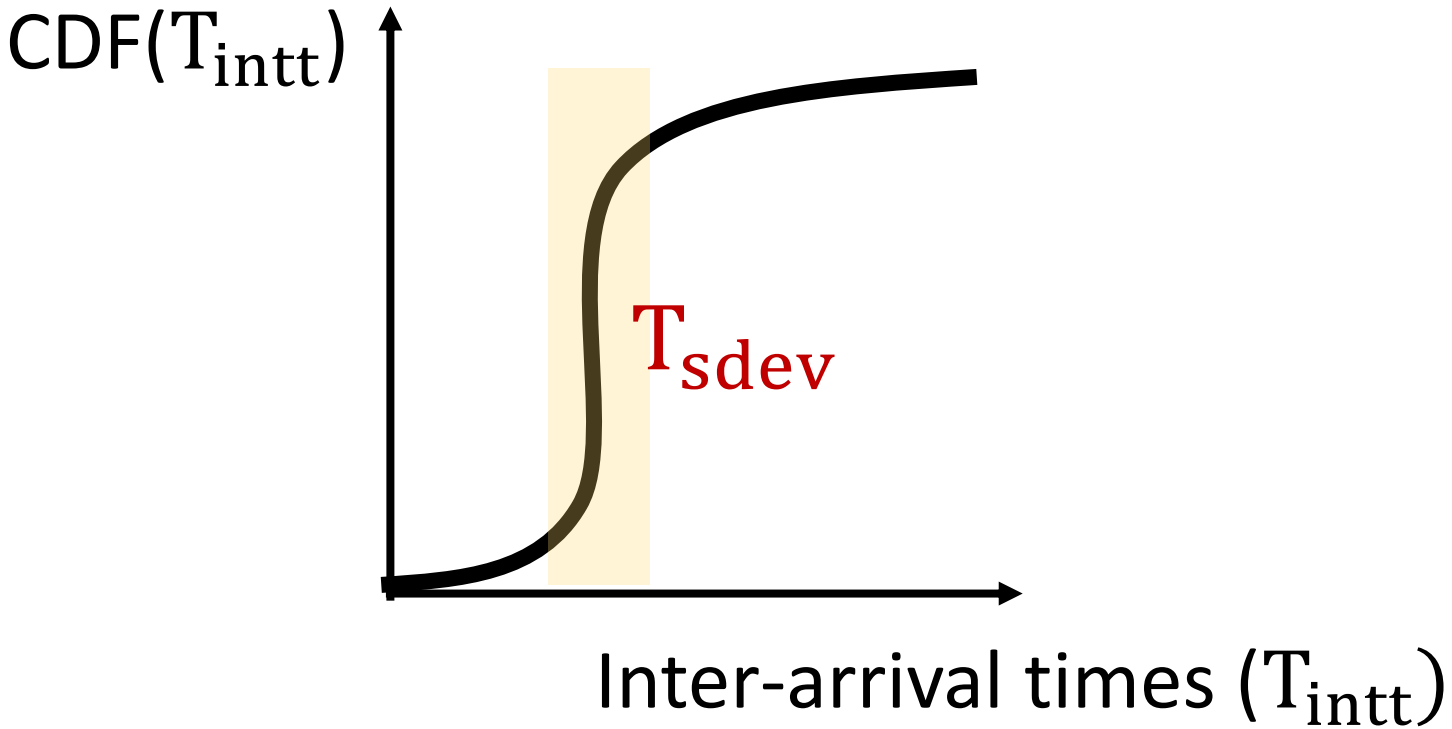
Inter-arrival times distribution

Key idea: Easily decompose T_{intt} from same category requests' T_{intt} distribution



Inter-arrival times distribution

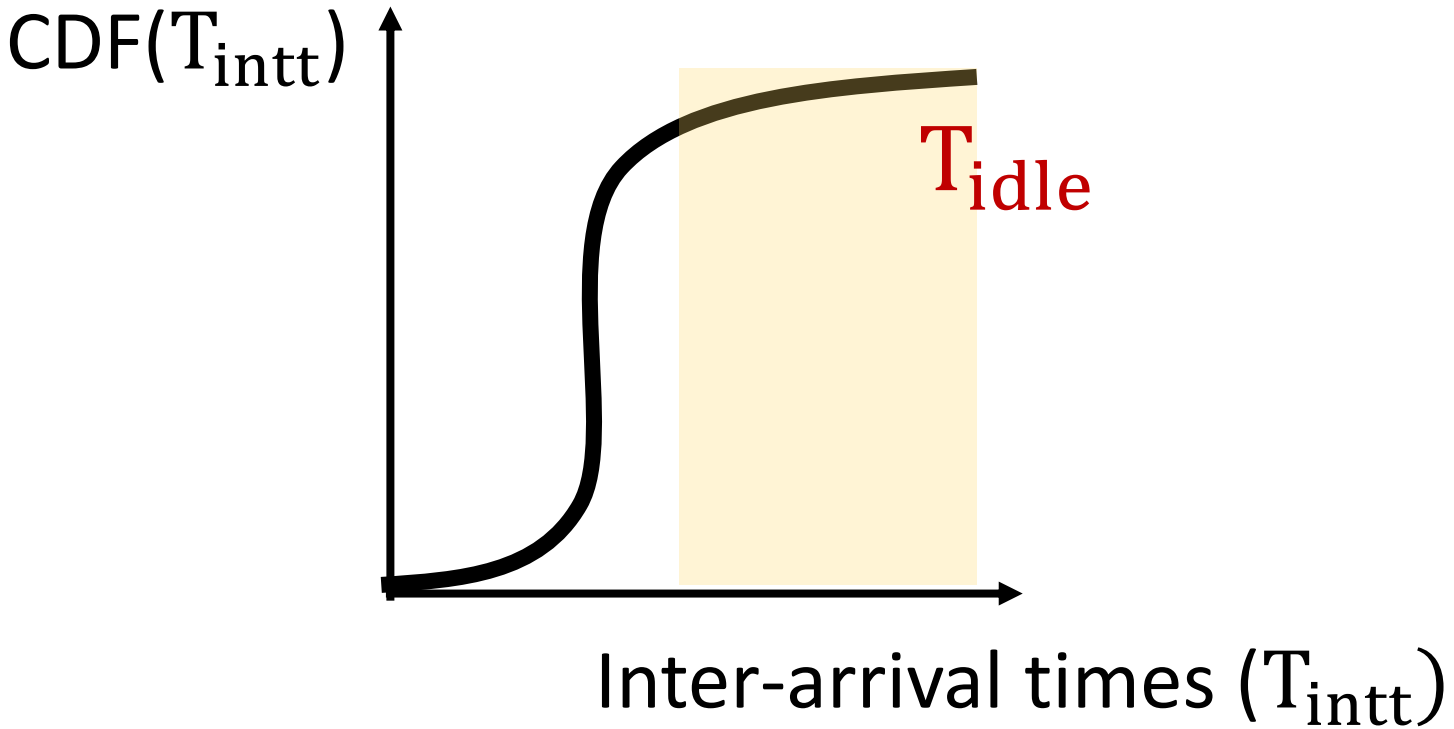
Key idea: Easily decompose T_{intt} from same category requests' T_{intt} distribution



Length1 sequential read

Inter-arrival times distribution

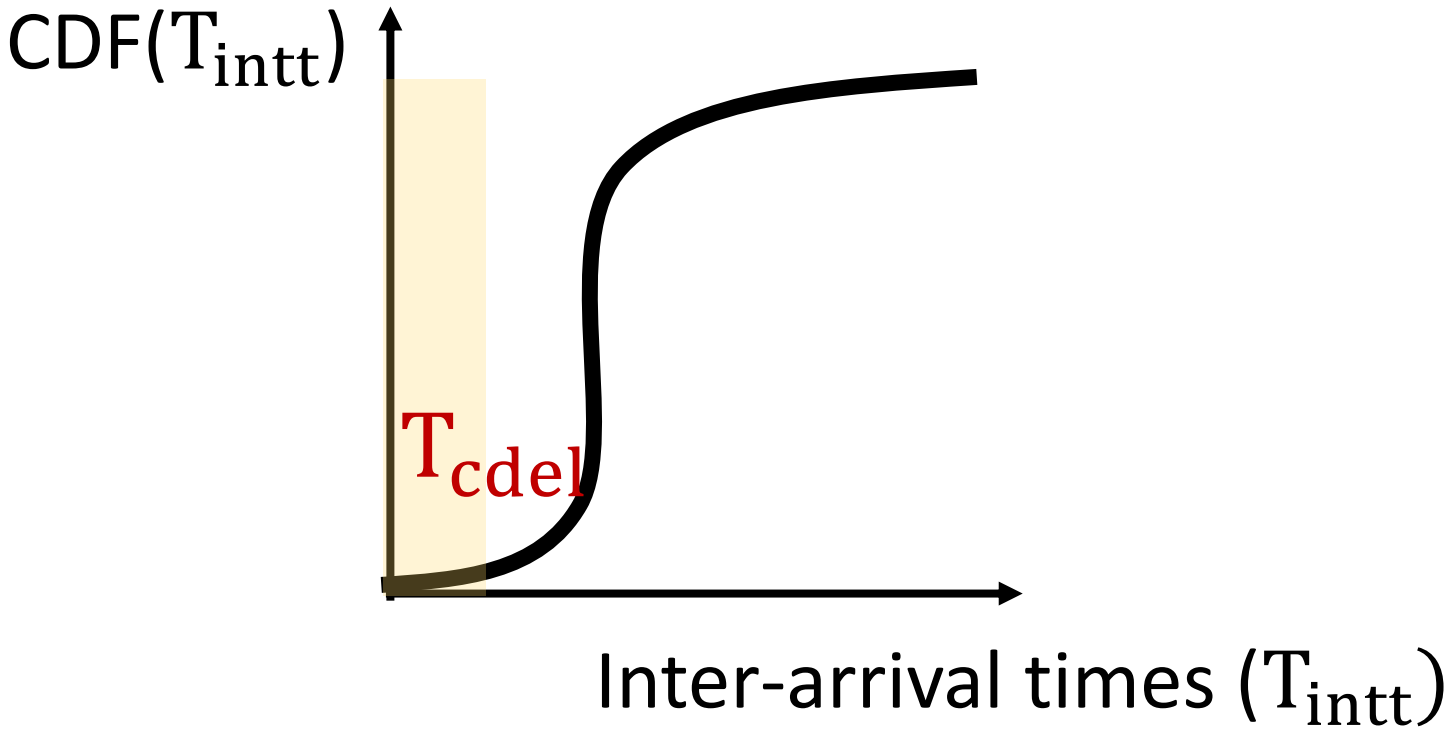
Key idea: Easily decompose T_{intt} from same category requests' T_{intt} distribution



Length1 sequential read

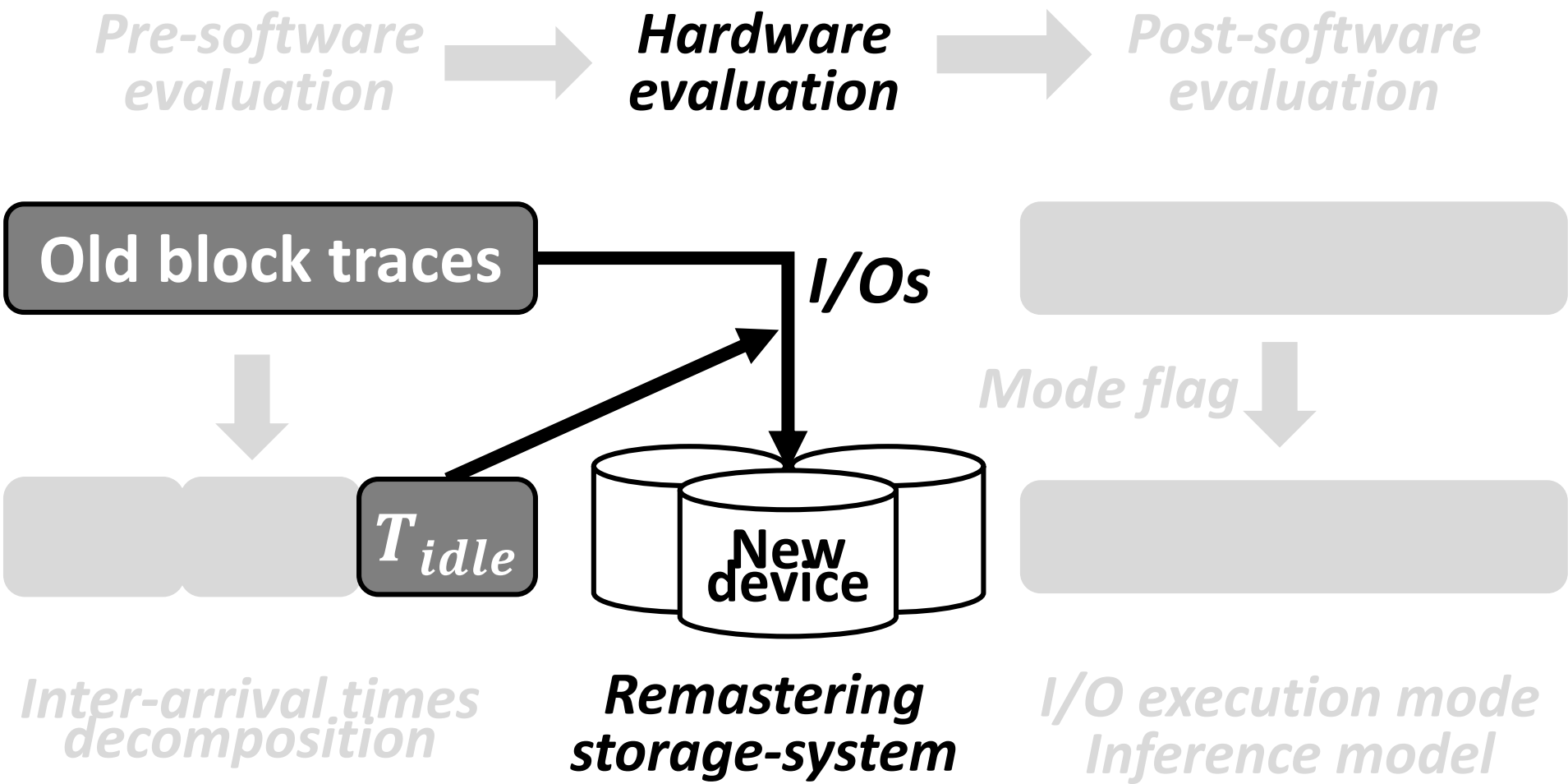
Inter-arrival times distribution

Key idea: Easily decompose T_{intt} from same category requests' T_{intt} distribution



Length1 sequential read

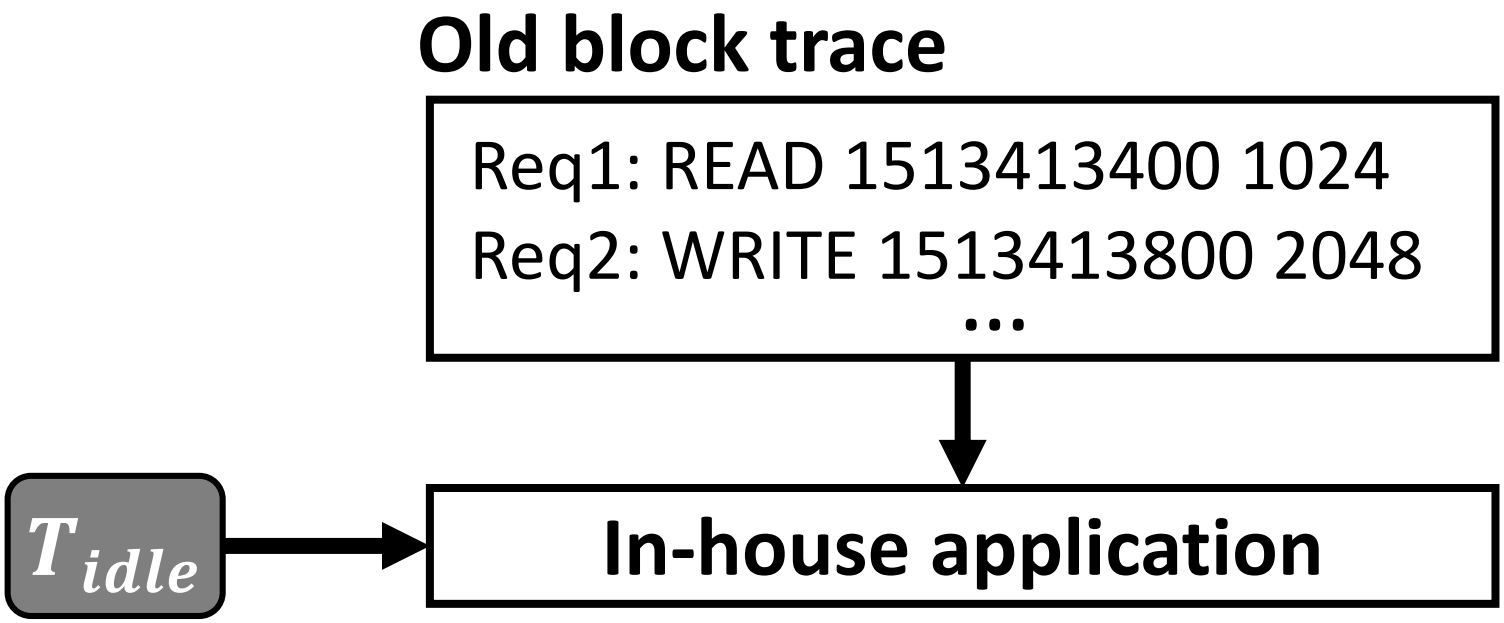
Overview of TraceTracker



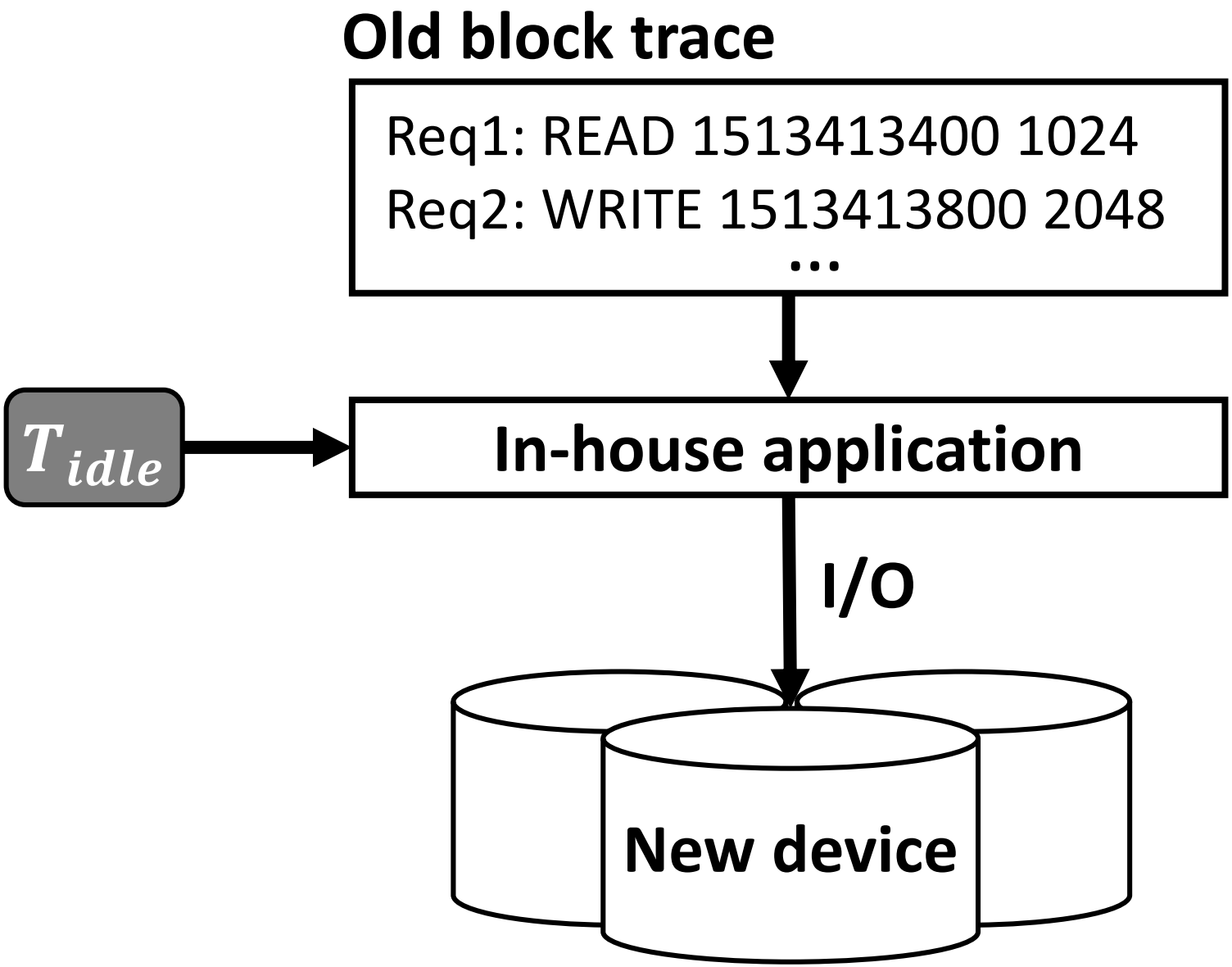
Trace replay

In-house application

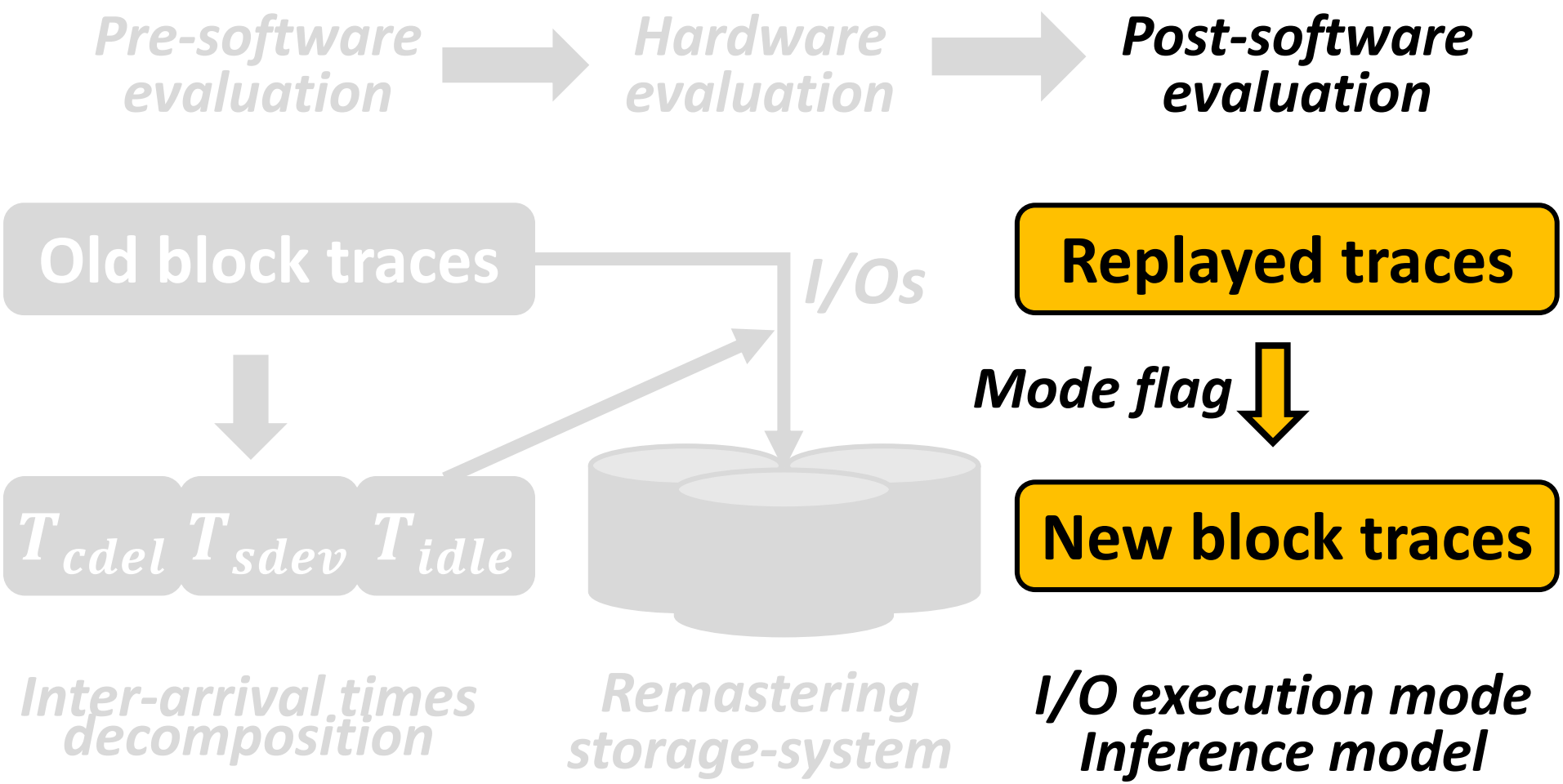
Trace replay



Trace replay

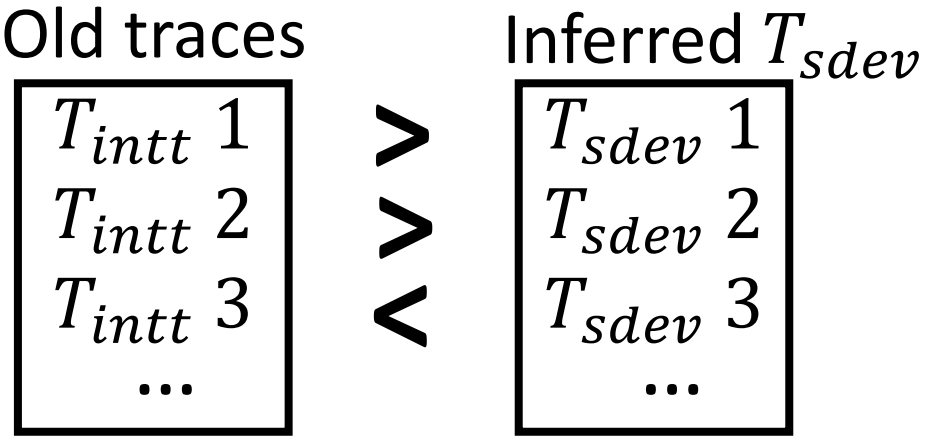


Overview of TraceTracker



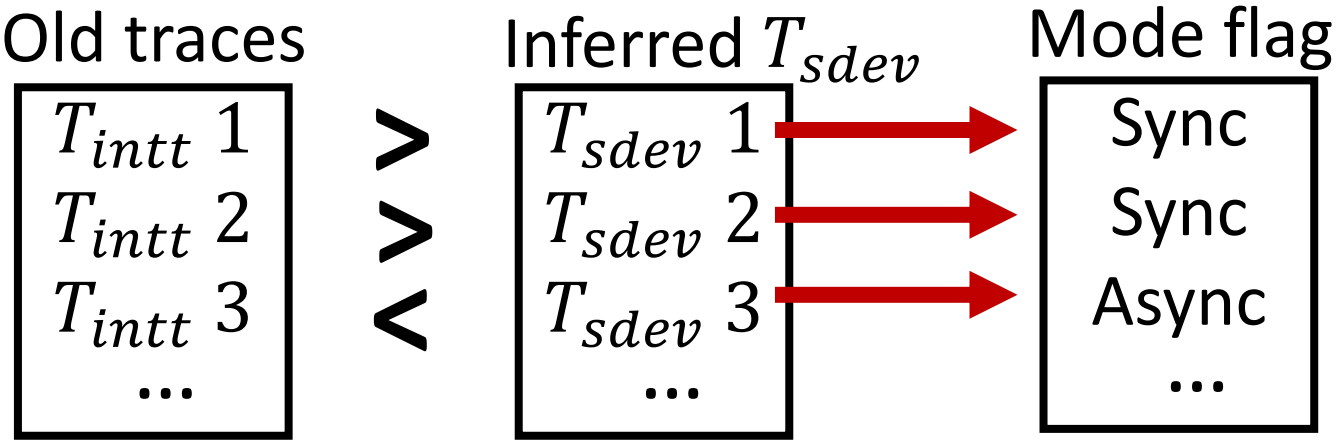
I/O execution mode inference

Key idea: Asynchronous I/O has short period which is less than storage I/O latency (T_{sdev})



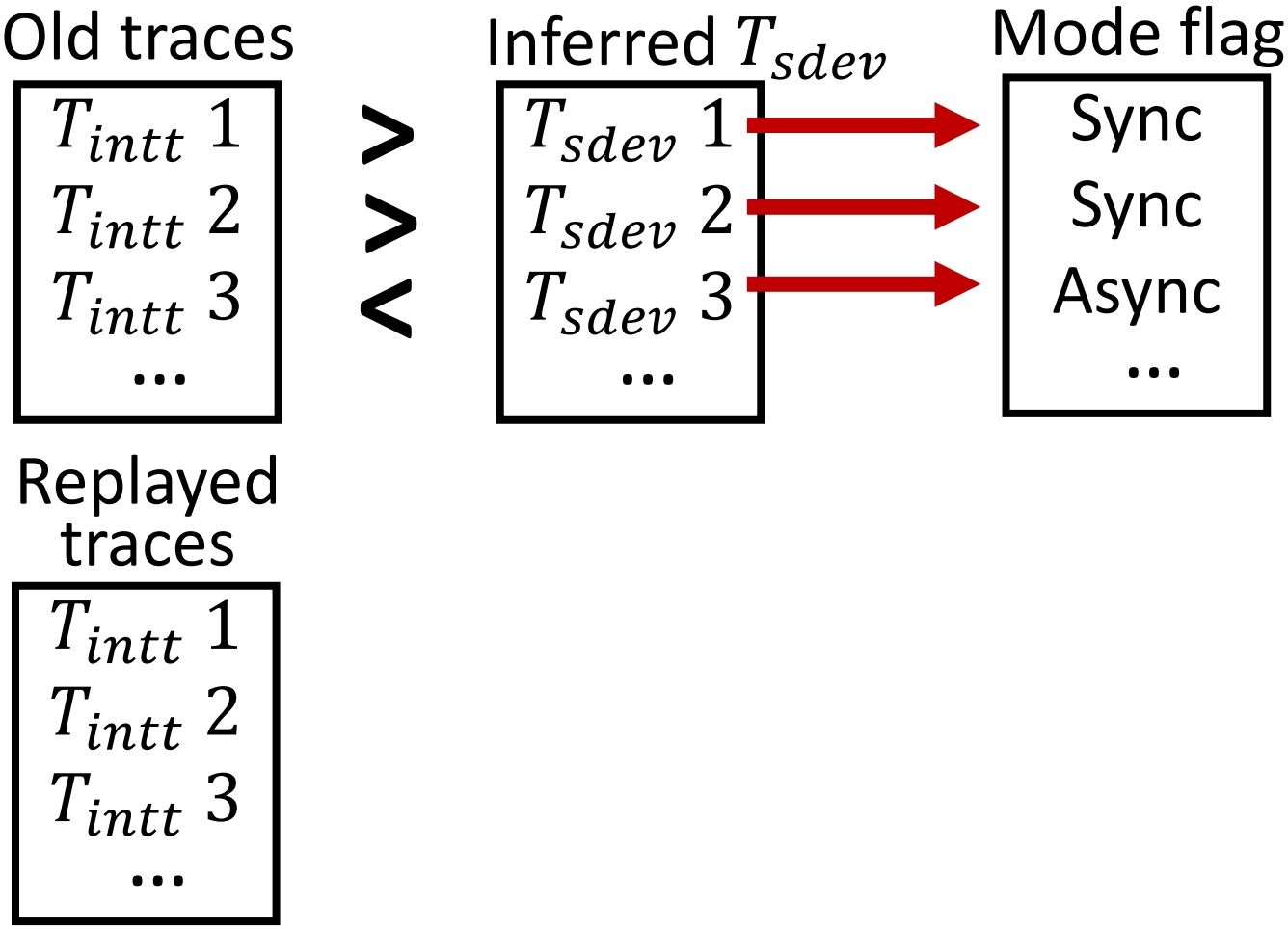
I/O execution mode inference

Key idea: Asynchronous I/O has short period which is less than storage I/O latency (T_{sdev})



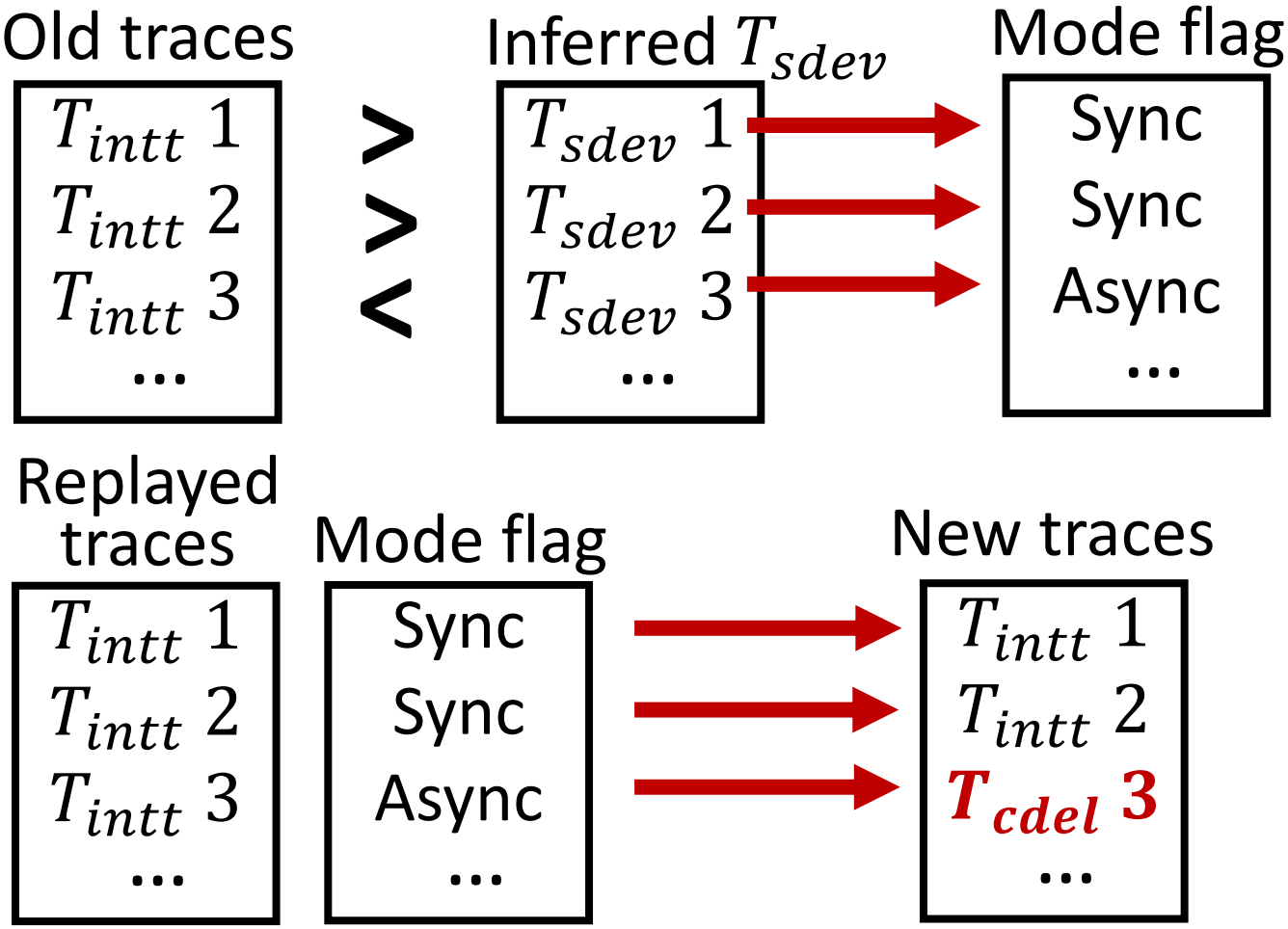
I/O execution mode inference

Key idea: Asynchronous I/O has short period which is less than storage I/O latency (T_{sdev})



I/O execution mode inference

Key idea: Asynchronous I/O has short period which is less than storage I/O latency (T_{sdev})



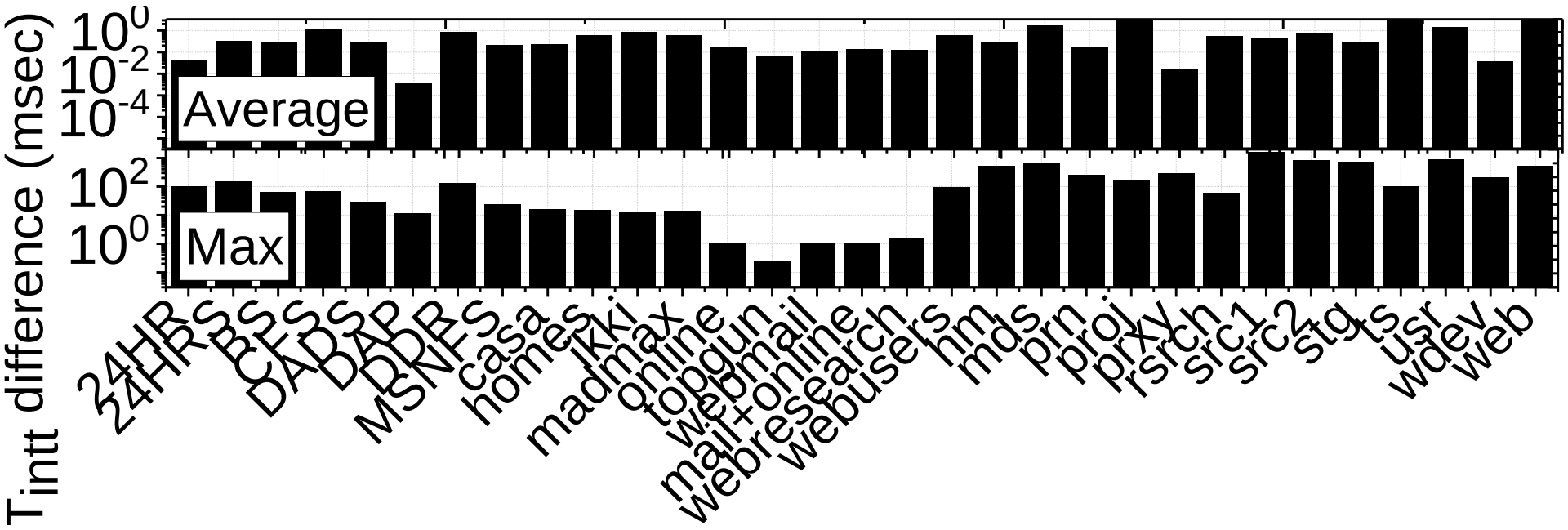
TraceTracker

1. Background of Block Trace
2. Trace Reconstruction Methods
3. Insights on Trace Reconstruction
4. TraceTracker Method
5. Evaluation

Evaluation Methodology

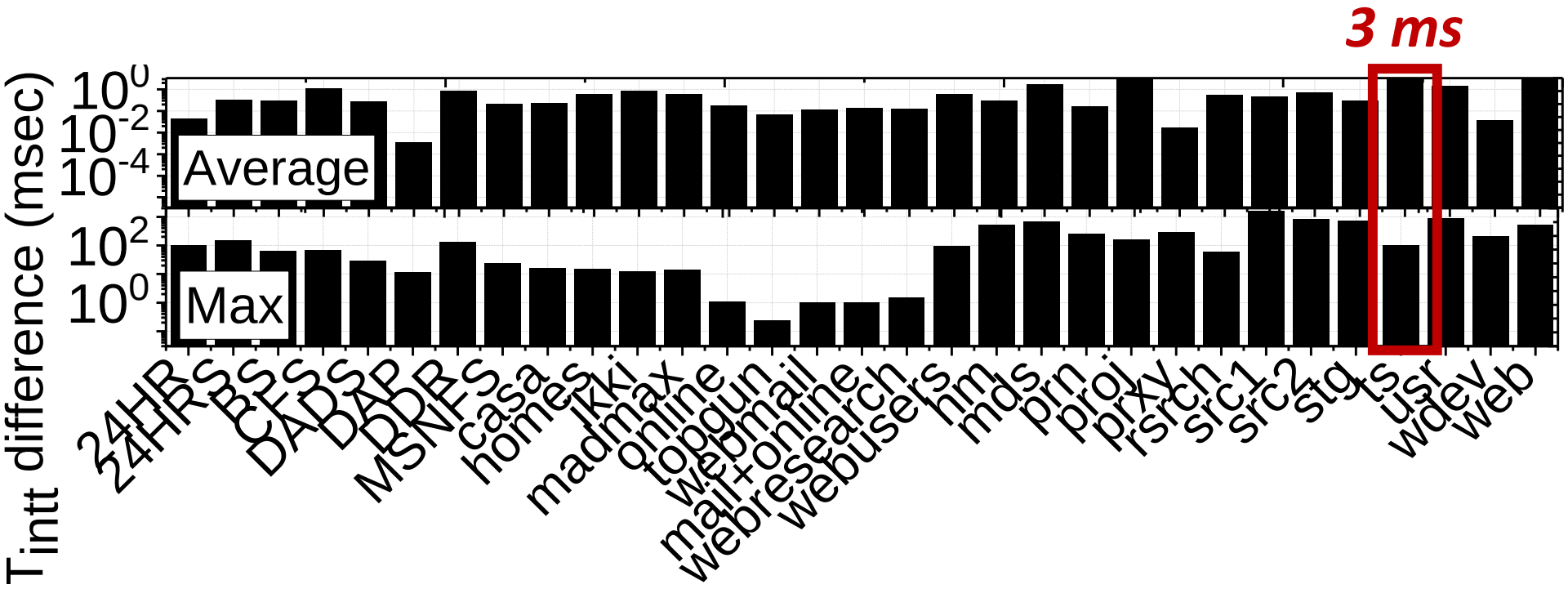
- **Evaluation node**
  - All-flash array (by grouping four NVM Express SSDs)
- **Target block traces**
  - 577 of large-scale block traces (around 2007~2009)
- **Reconstruction techniques**
  - ***Acceleration***: Reconstruction by shortening T_{intt}
  - ***Revision***: Replaying block trace on all-flash array
  - ***Fixed-th***: An advanced revision method by inferring T_{idle} with a fixed threshold
  - ***Dynamic***: Reconstructions using our inference model, but with no post-processing.
  - ***TraceTracker***: Hardware/software co-evalution

Evaluation #1: Different I/O timing



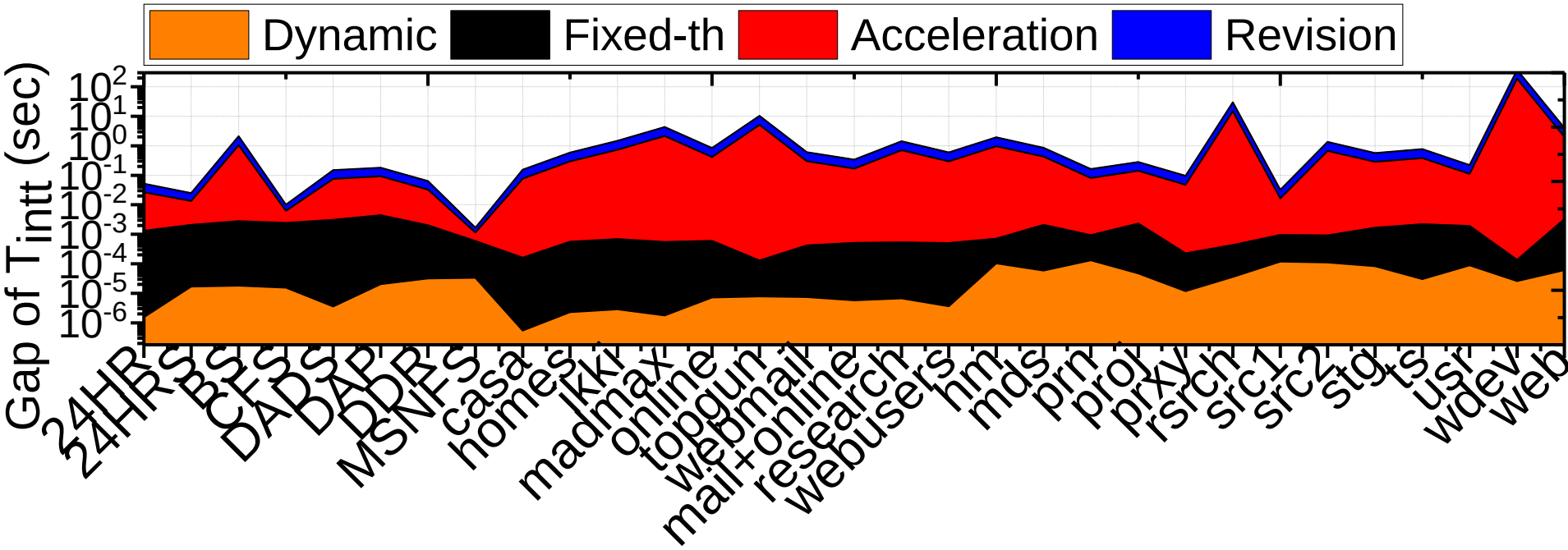
- New traces have, on average, 0.677 ms **shorter** inter-arrival time than old block traces
- *Ex) 'ts'* has 3 ms shorter T_{intt}, on average.

Evaluation #1: Different I/O timing



- New traces have, on average, 0.677 ms **shorter** inter-arrival time than old block traces
- *Ex) 'ts'* has 3 ms shorter T_{intt}, on average.

Evaluation #2: Inaccuracy of reconstruction

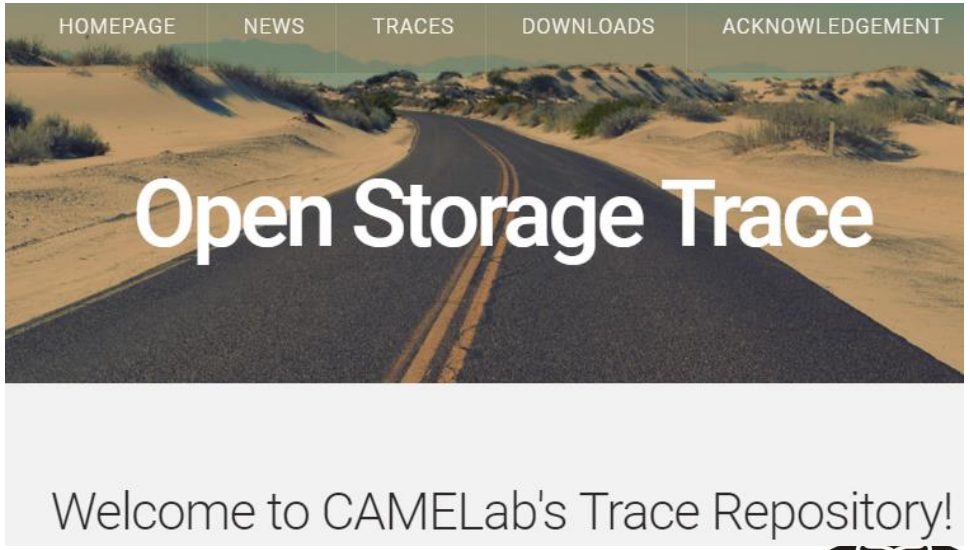


- ***Acceleration*** and ***Revision***: 7.08, 7.15 sec due to no information for T_{idle}
- ***Fixed-th*** : 1.3 ms (inaccurate T_{idle})
- ***Dynamic*** : 0.035 ms (inaccurate IO-mode)

Conclusion

- ***TraceTracker*** is trace reconstruction method which remasters the storage latency while maintaining runtime contexts of target traces.
- **We're preparing brand new traces for open-license!**

Trace download is available at *trace.camelab.org*



KANDEMIR : All-Flash Array based HPC Testbed

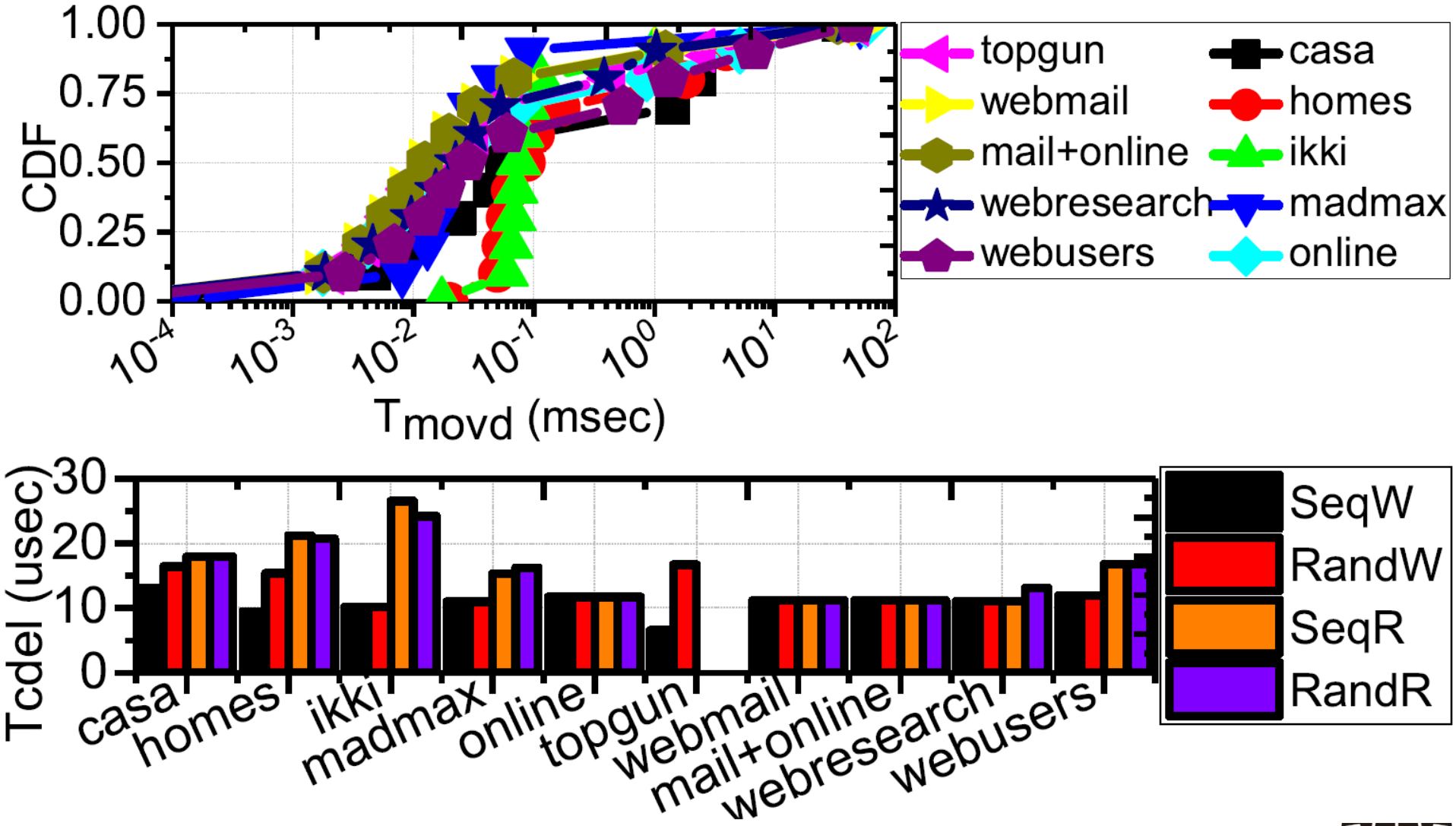
TraceTracker: Hardware/Software Co-Evaluation for Large-Scale I/O Workload Reconstruction

Miryeong Kwon, Jie Zhang, Gyuyoung Park, Wonil Choi, David Donofrio, John Shalf, Mahmut Kandemir, and Myoungsoo Jung

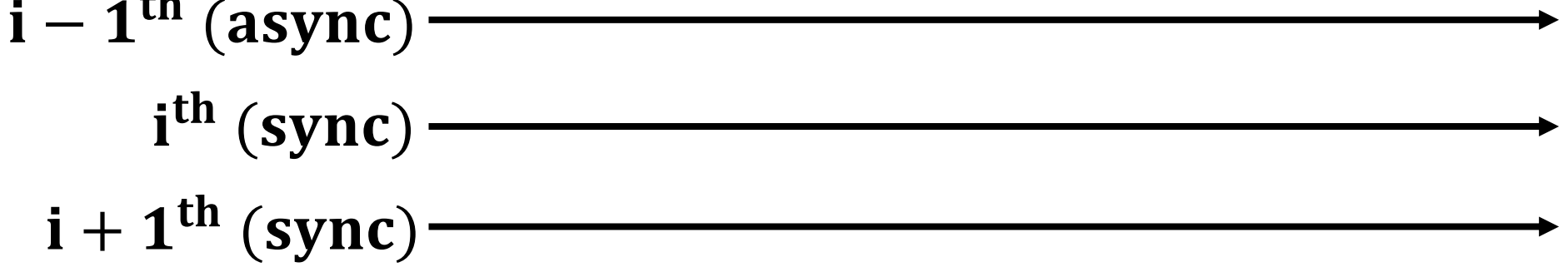
YONSEI UNIVERSITY

CAMELab.org
Computer Architecture and Memory Systems Laboratory

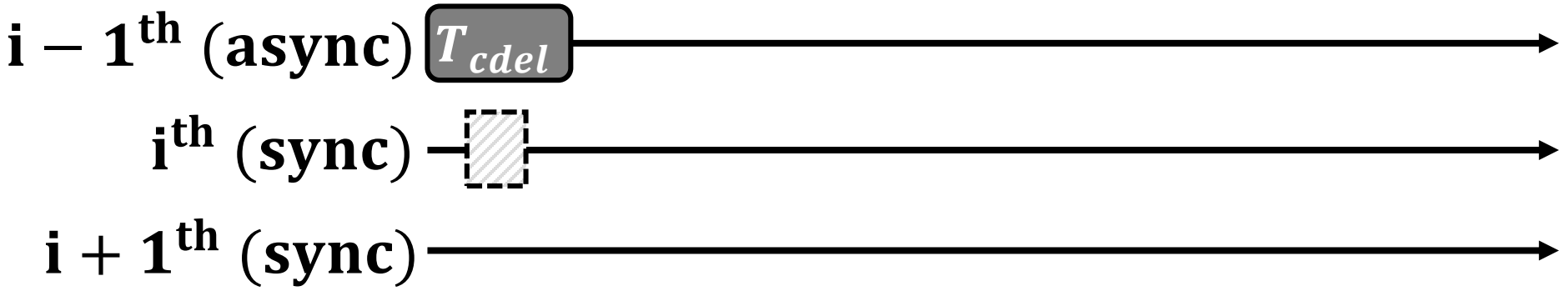
CDF distribution of T_{movd}, T_{cdel}



Block Request Timing



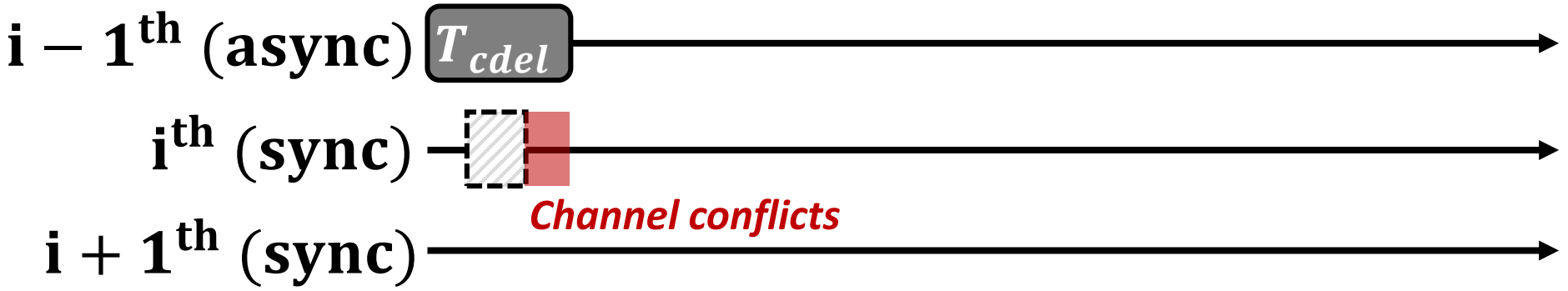
Block Request Timing



: Computation cycles by user/kernel

T_{cdel} : Delay due to storage interface (channel) data movement

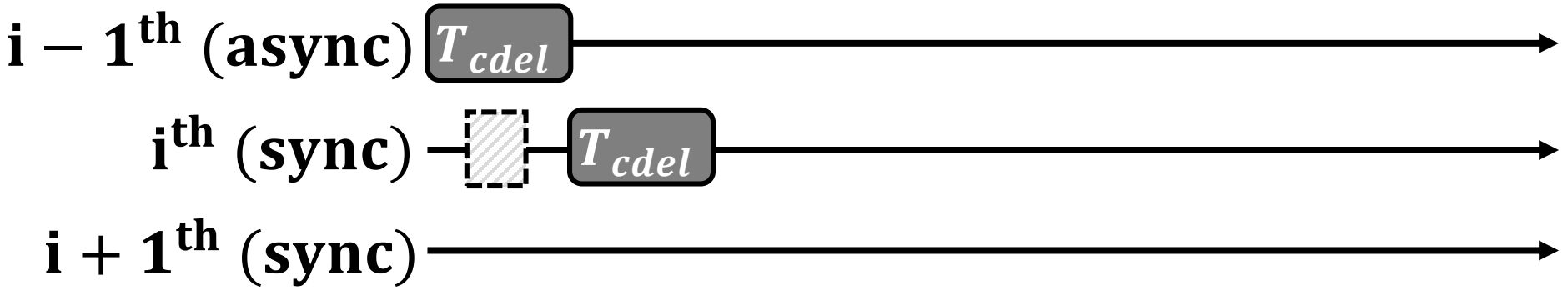
Block Request Timing



: Computation cycles by user/kernel

T_{cdel} : Delay due to storage interface (channel) data movement

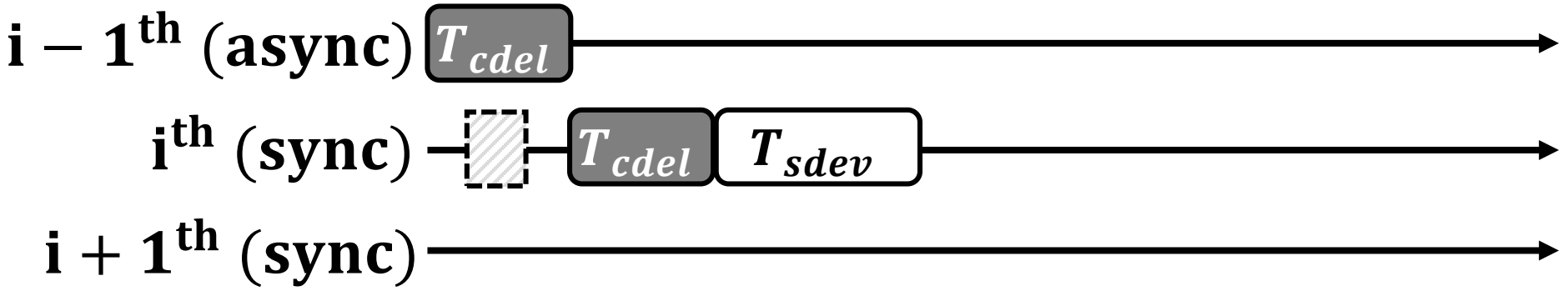
Block Request Timing



: Computation cycles by user/kernel

T_{cdel} : Delay due to storage interface (channel) data movement

Block Request Timing

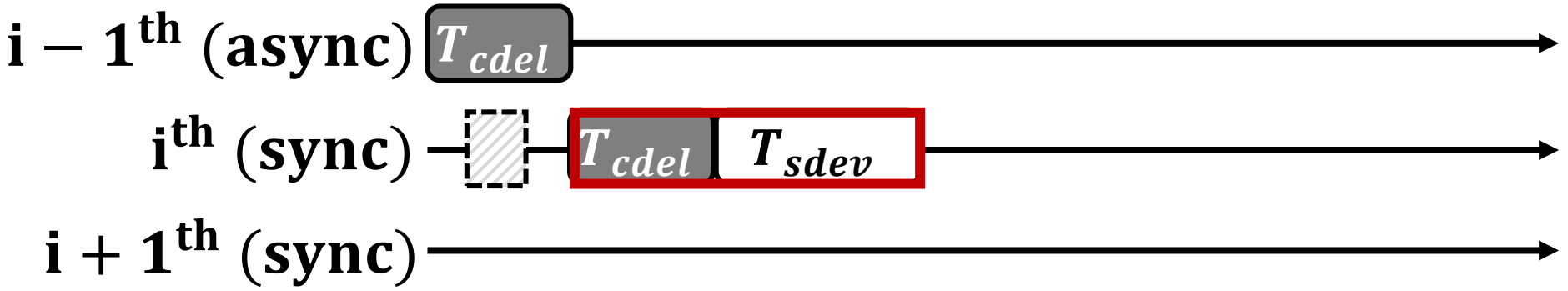


: Computation cycles by user/kernel

T_{cdel} : Delay due to storage interface (channel) data movement

T_{sdev} : Actual device time taken by the storage to service the IO

Block Request Timing



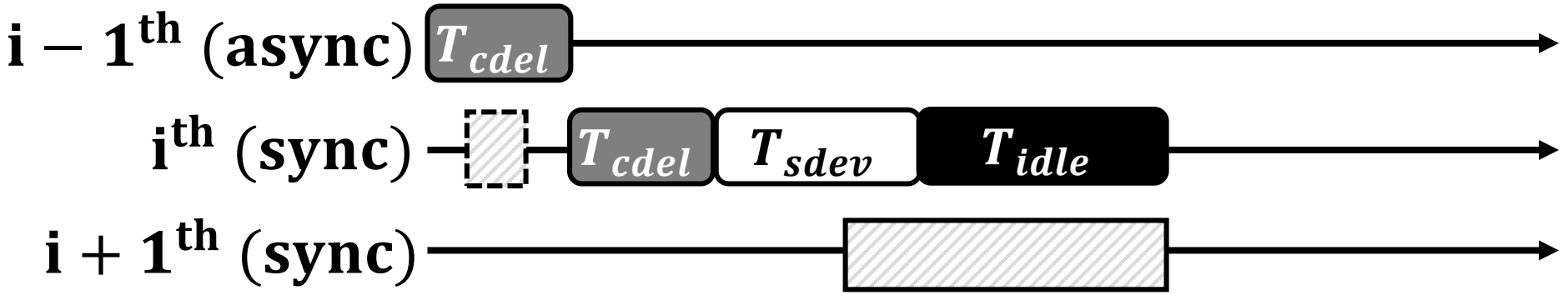
: Computation cycles by user/kernel

T_{cdel} : Delay due to storage interface (channel) data movement

T_{sdev} : Actual device time taken by the storage to service the IO

I/O subsystem latency (T_{slat})

Block Request Timing



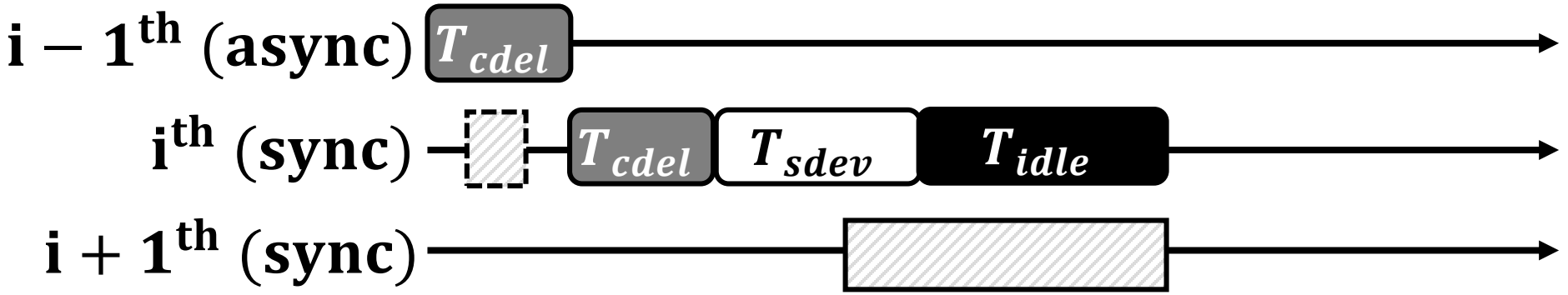
: Computation cycles by user/kernel

T_{cdel} : Delay due to storage interface (channel) data movement

T_{sdev} : Actual device time taken by the storage to service the IO

T_{idle} : user/application does nothing

Block Request Timing



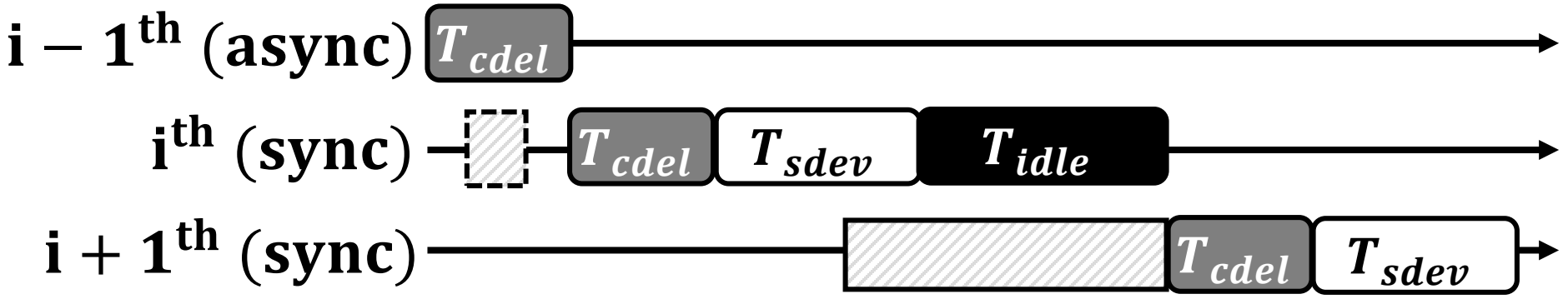
: Computation cycles by user/kernel

T_{cdel} : Delay due to storage interface (channel) data movement

T_{sdev} : Actual device time taken by the storage to service the IO

T_{idle} : user/application does nothing

Block Request Timing



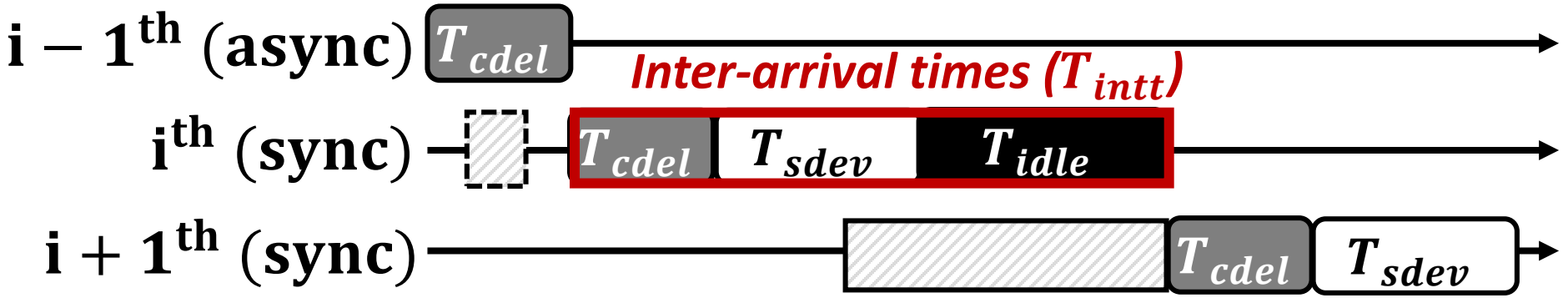
: Computation cycles by user/kernel

T_{cdel} : Delay due to storage interface (channel) data movement

T_{sdev} : Actual device time taken by the storage to service the IO

T_{idle} : user/application does nothing

Block Request Timing



: Computation cycles by user/kernel

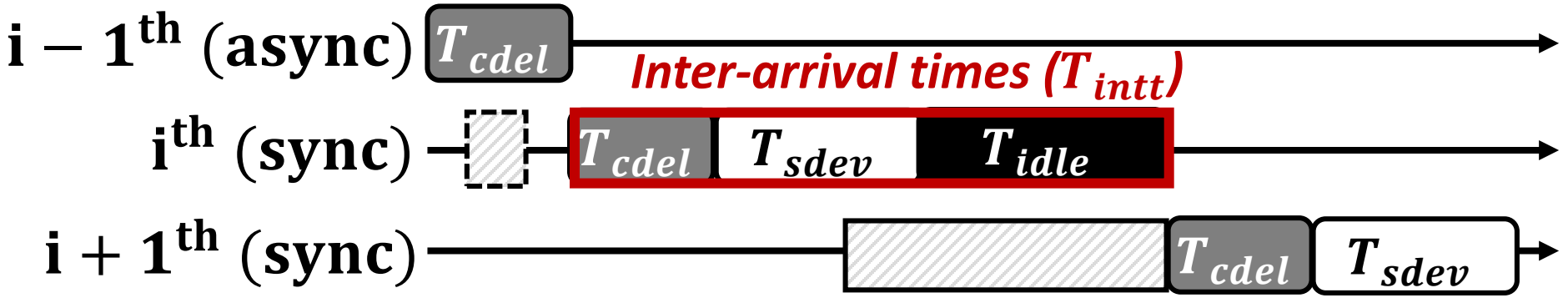
T_{cdel} : Delay due to storage interface (channel) data movement

T_{sdev} : Actual device time taken by the storage to service the IO

T_{idle} : user/application does nothing

Inter-arrival times can be decomposed into T_{cdel}, T_{sdev}, and T_{idle}

Block Request Timing



: Computation cycles by user/kernel

T_{cdel} : Delay due to storage interface (channel) data movement

T_{sdev} : Actual device time taken by the storage to service the IO

T_{idle} : user/application does nothing

Why inter-arrival times decomposition is important to infer runtime context?

Overview of TraceTracker

Overview of TraceTracker

Pre-software evaluation

Overview of TraceTracker

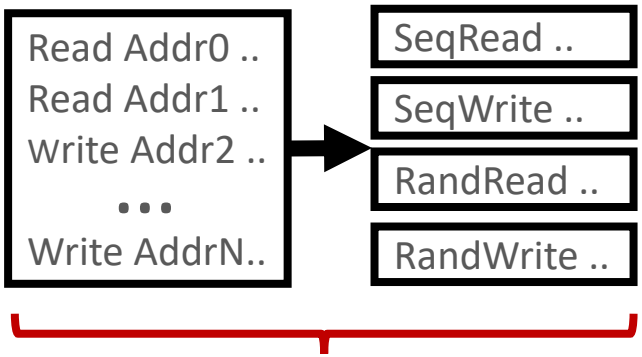
Pre-software evaluation  ***Hardware evaluation***

Overview of TraceTracker

Overview of TraceTracker

Pre-software evaluation → ***Hardware evaluation*** → ***Post-software evaluation***

Old traces

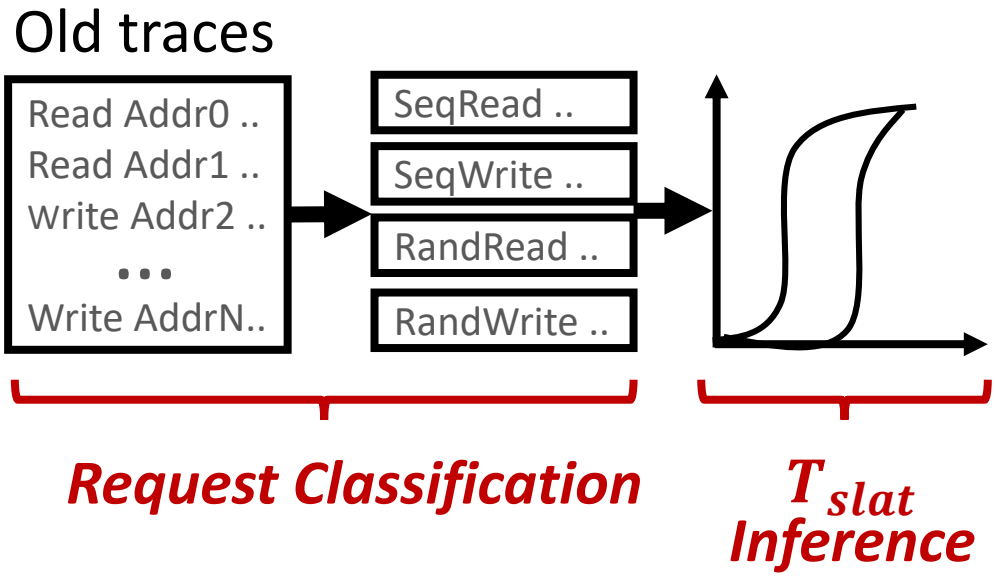


Request Classification

Each T_{cdel} and T_{sdev} have similar values for same operation type and request size

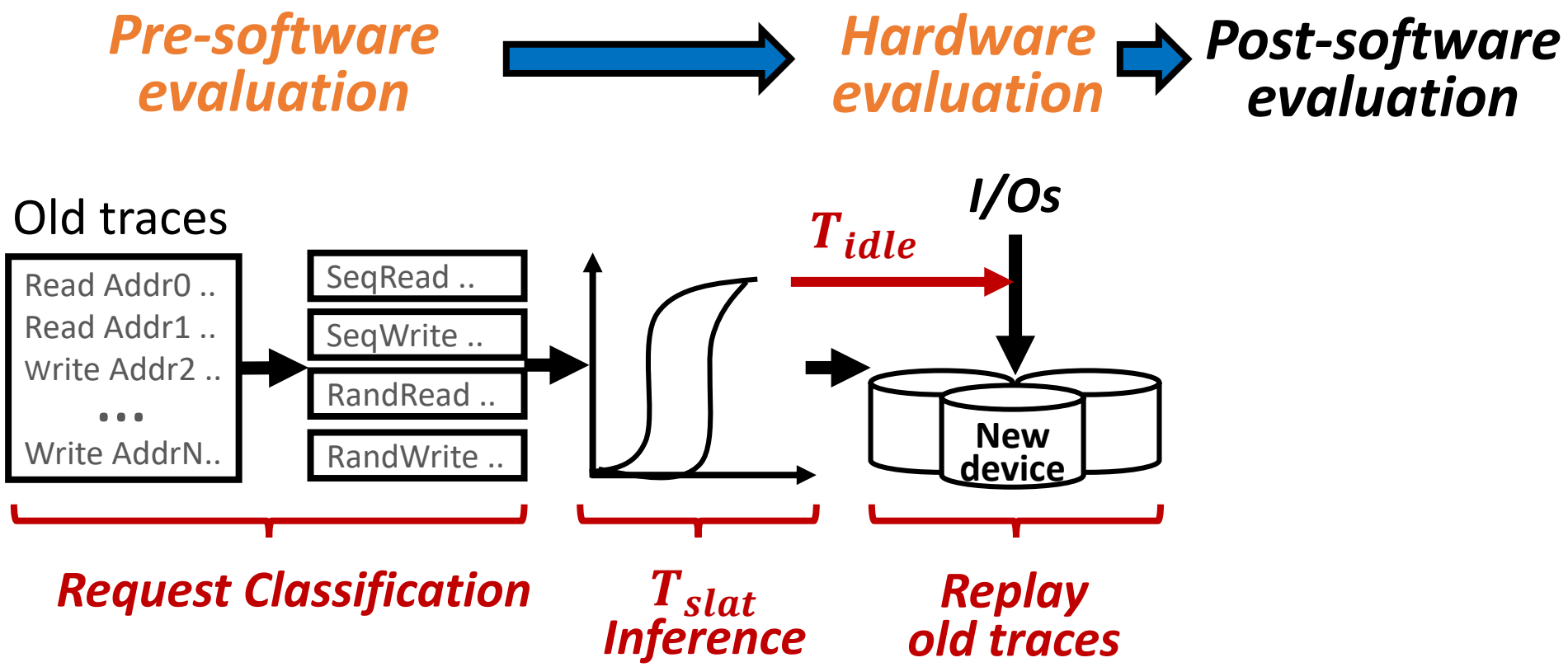
Overview of TraceTracker

Pre-software evaluation → ***Hardware evaluation*** → ***Post-software evaluation***



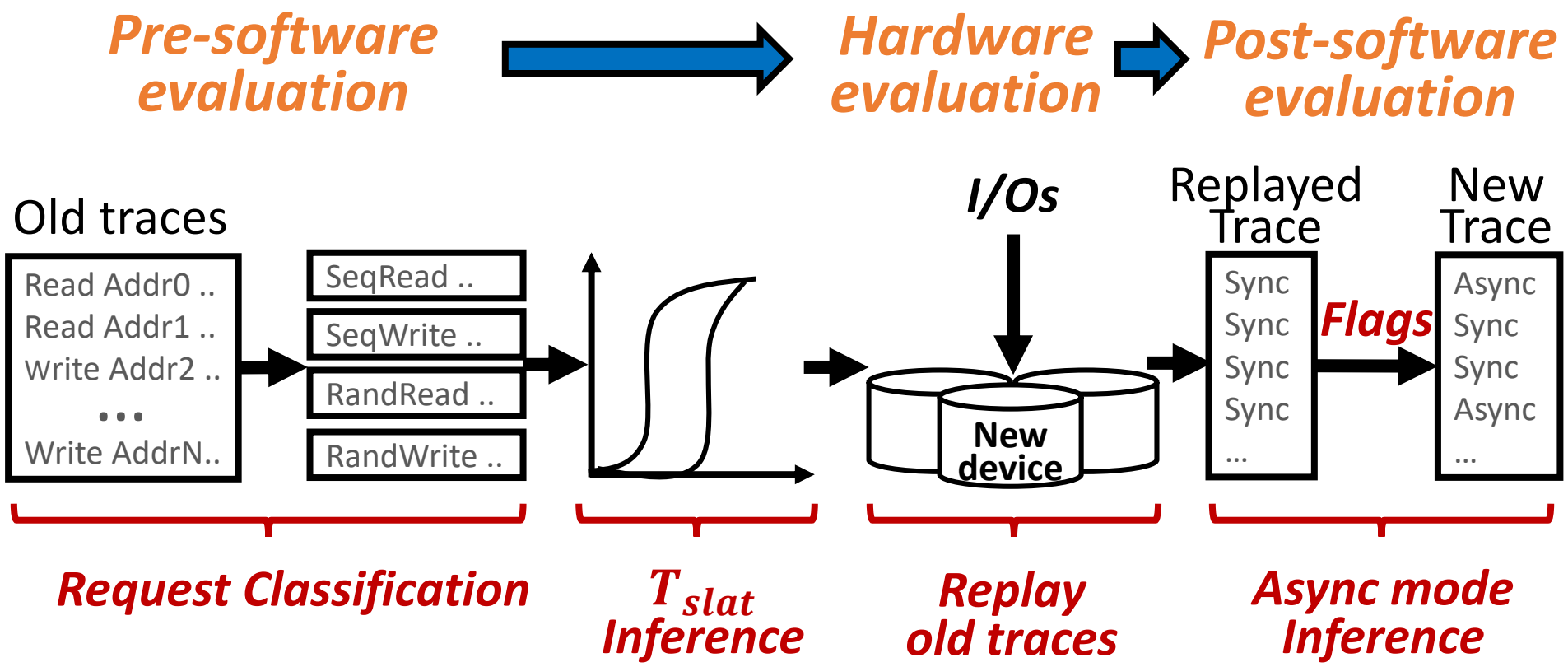
Infer the T_{slat} (: T_{cdel} and T_{sdev}) and calculate T_{idle}(=T_{intt}-T_{slat})

Overview of TraceTracker



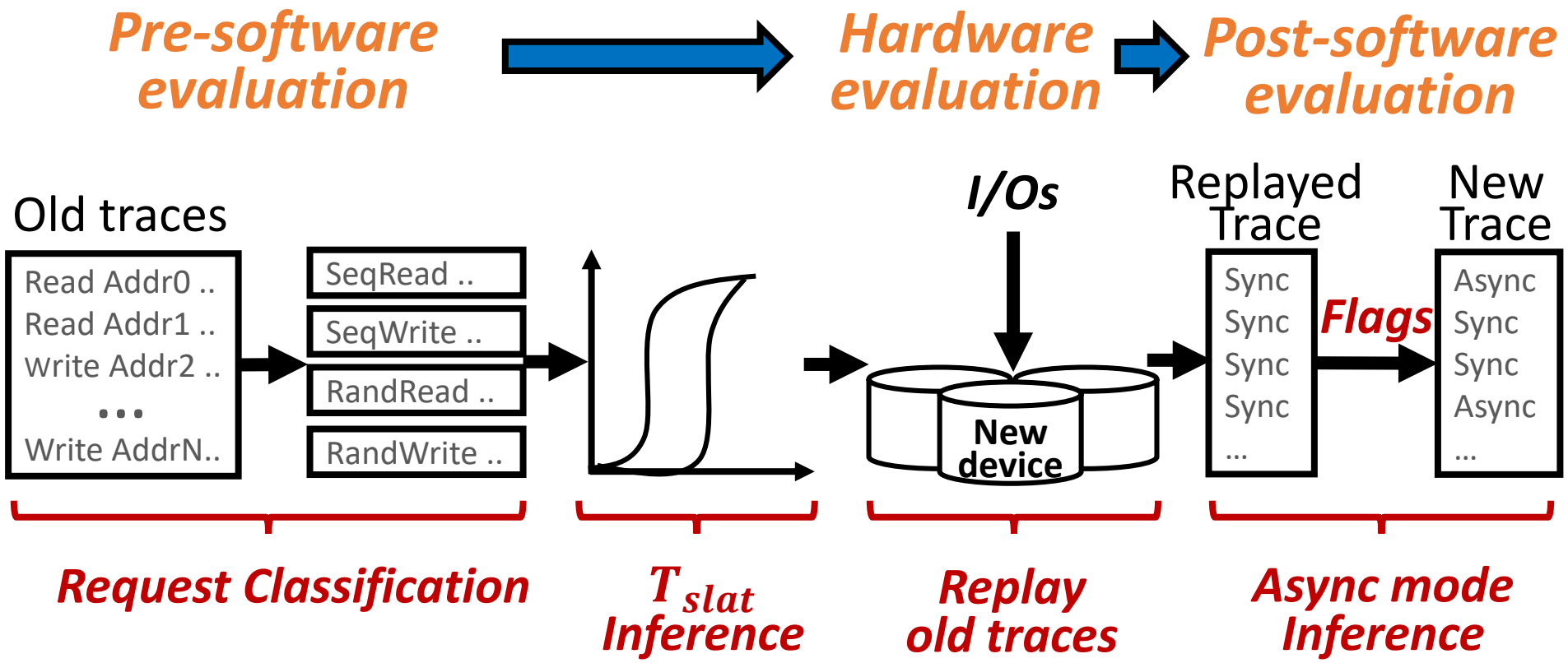
Replay old traces' I/Os with inferred T_{idle} while remastering T_{slat} from new system

Overview of TraceTracker



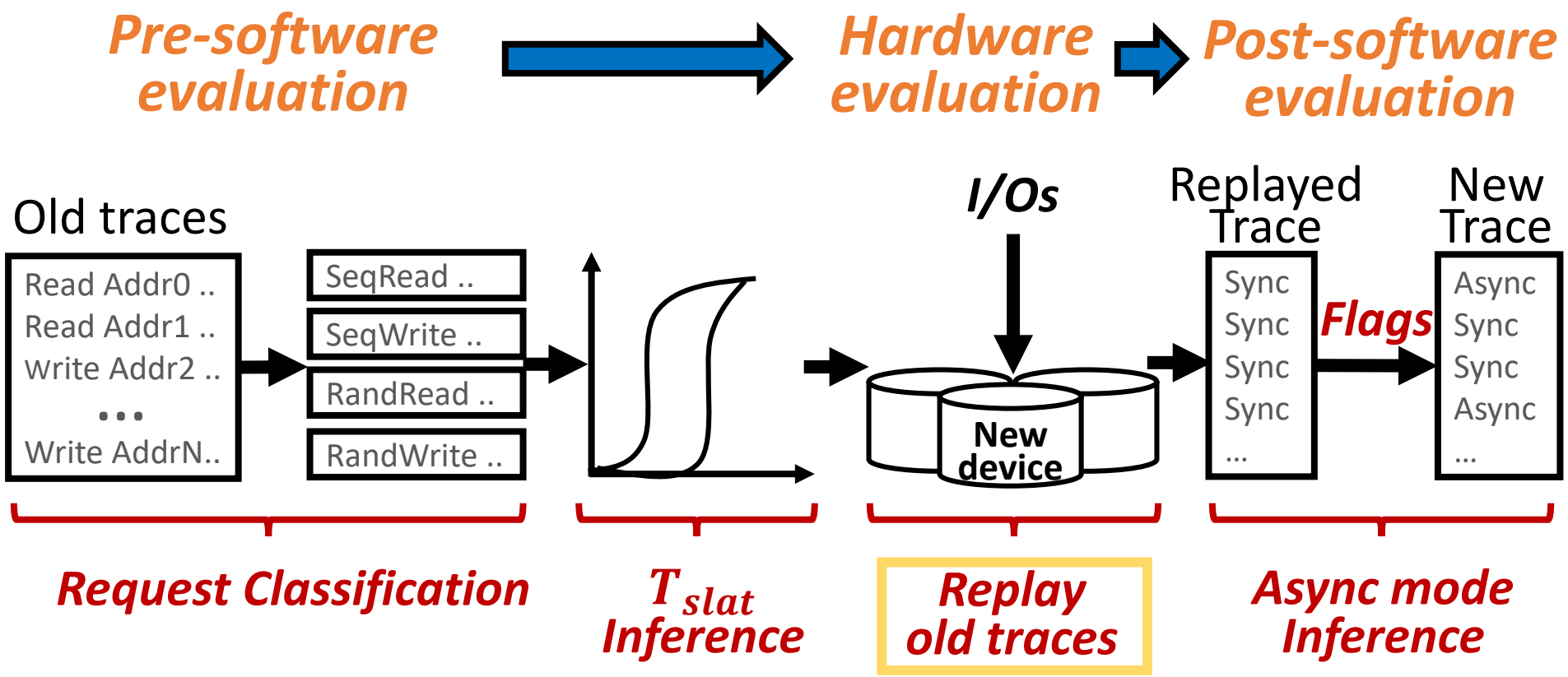
Infer I/O mode (sync or async) from old traces and revise replayed trace

Overview of TraceTracker



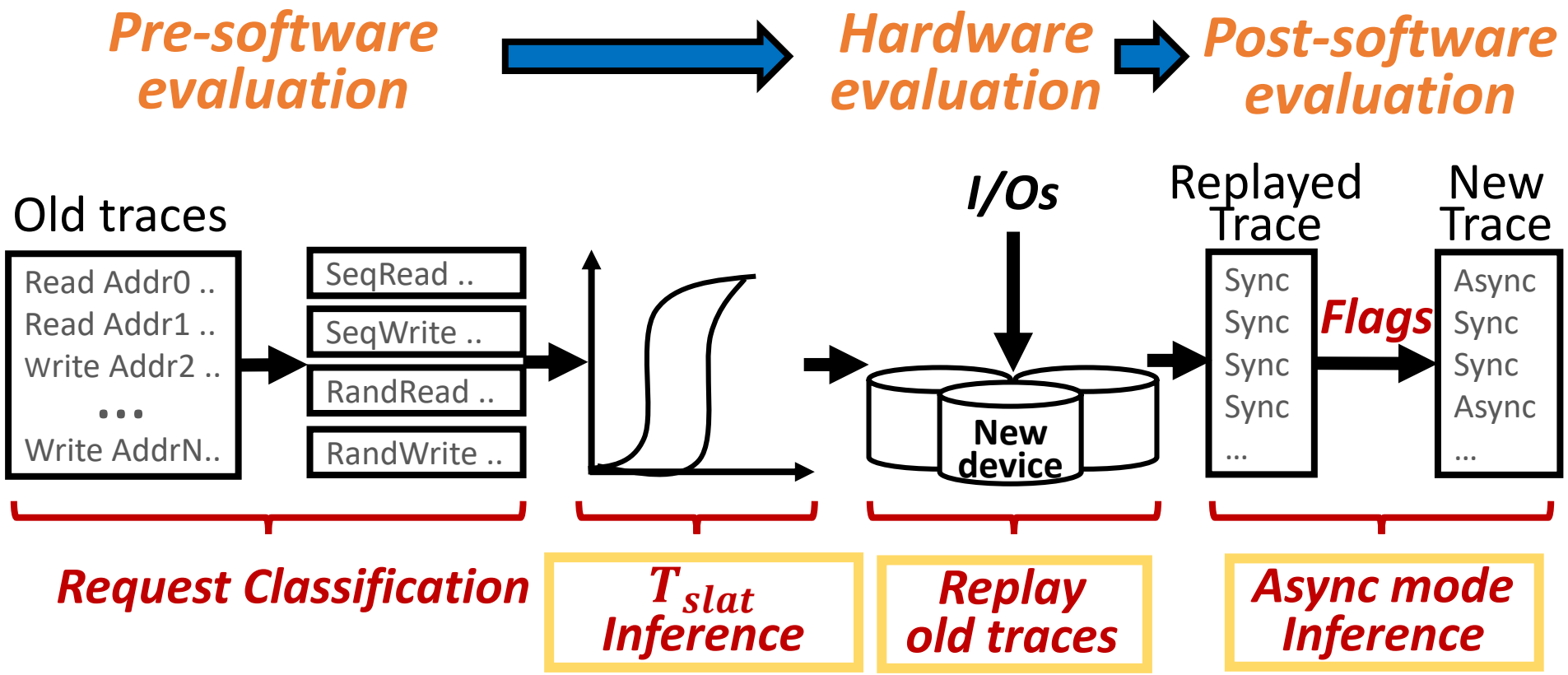
1) Accurate device latency
2) Traces include runtime contexts

Overview of TraceTracker



1) Accurate device latency
2) Traces include runtime contexts

Overview of TraceTracker



1) Accurate device latency
2) Traces include runtime contexts

Request Classification

Key idea:

1) Each request group has similar T_{slat}
2) T_{sdev} can be expressed as linear model

Request Classification

Key idea:

1) Each request group has similar T_{slat}

2) T_{sdev} can be expressed as linear model

$$T_{slat} = T_{cdel} + T_{sdev}$$

Sequential/Random Read

$$T_{slat} = T_{cdel\_read} + Coeff_{read} \times REQSIZE + T_{movd}$$

Sequential/Random Write

$$T_{slat} = T_{cdel\_write} + Coeff_{write} \times REQSIZE + T_{movd}$$

Request Classification

Key idea:

1) Each request group has similar T_{slat}

2) T_{sdev} can be expressed as linear model

$$T_{slat} = T_{cdel} + T_{sdev}$$

Sequential/Random Read

$$T_{slat} = T_{cdel\_read} + Coeff_{read} \times REQSIZE + T_{movd}$$

Sequential/Random Write

$$T_{slat} = T_{cdel\_write} + Coeff_{write} \times REQSIZE + T_{movd}$$

Request Classification

Sequential/Random Read

$$T_{slat} = T_{cdel\_read} + Coeff_{read} \times REQSIZE + T_{movd}$$

Request Classification

Sequential/Random Read

$$T_{slat} = T_{cdel\_read} + Coeff_{read} \times REQSIZE + T_{movd}$$

$$T_{slat}1 = T_{cdel\_read} + Coeff_{read} \times REQSIZE1 + T_{movd}$$

$$T_{slat}2 = T_{cdel\_read} + Coeff_{read} \times REQSIZE2 + T_{movd}$$

Request Classification

Sequential/Random Read

$$T_{slat} = T_{cdel\_read} + Coeff_{read} \times REQSIZE + T_{movd}$$

$$T_{slat}1 = T_{cdel\_read} + Coeff_{read} \times REQSIZE1 + T_{movd}$$

$$T_{slat}2 = T_{cdel\_read} + Coeff_{read} \times REQSIZE2 + T_{movd}$$

Request Classification

Sequential/Random Read

$$T_{slat} = T_{cdel\_read} + Coeff_{read} \times REQSIZE + T_{movd}$$

$$T_{slat}1 = T_{cdel\_read} + Coeff_{read} \times REQSIZE1 + T_{movd}$$
$$T_{slat}2 = T_{cdel\_read} + Coeff_{read} \times REQSIZE2 + T_{movd}$$

$$T_{slat}1 - T_{slat}2 = Coeff_{read} \times Diff(SIZE1 - SIZE2)$$

Request Classification

Sequential/Random Read

$$T_{slat} = T_{cdel\_read} + Coeff_{read} \times REQSIZE + T_{movd}$$

$$T_{slat}1 = T_{cdel\_read} + Coeff_{read} \times REQSIZE1 + T_{movd}$$
$$T_{slat}2 = T_{cdel\_read} + Coeff_{read} \times REQSIZE2 + T_{movd}$$

$$T_{slat}1 - T_{slat}2 = Coeff_{read} \times Diff(SIZE1 - SIZE2)$$

Inferred from max slope of CDF — *Known info.*

Request Classification

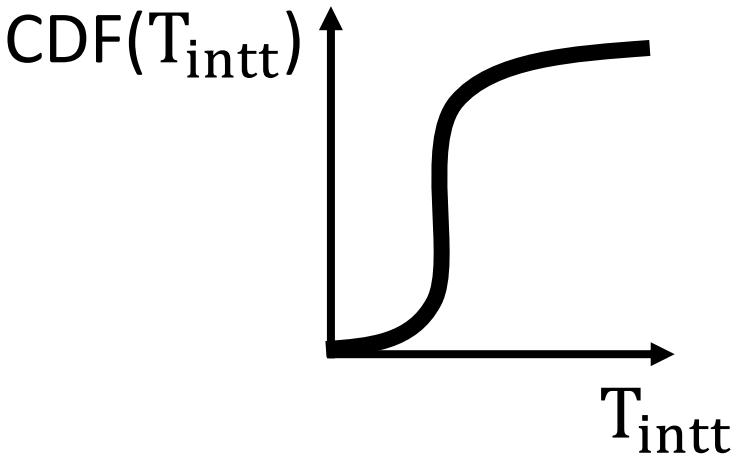
Sequential/Random Read

$$T_{slat} = T_{cdel\_read} + Coeff_{read} \times REQSIZE + T_{movd}$$

$$T_{slat}1 = T_{cdel\_read} + Coeff_{read} \times REQSIZE1 + T_{movd}$$
$$T_{slat}2 = T_{cdel\_read} + Coeff_{read} \times REQSIZE2 + T_{movd}$$

$$T_{slat}1 - T_{slat}2 = Coeff_{read} \times Diff(SIZE1 - SIZE2)$$

Inferred from max slope of CDF *Known info.*



Request Classification

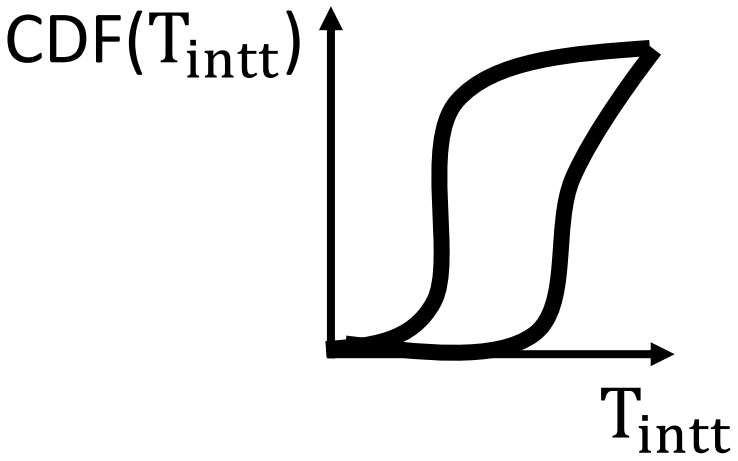
Sequential/Random Read

$$T_{slat} = T_{cdel\_read} + Coeff_{read} \times REQSIZE + T_{movd}$$

$$T_{slat}1 = T_{cdel\_read} + Coeff_{read} \times REQSIZE1 + T_{movd}$$
$$T_{slat}2 = T_{cdel\_read} + Coeff_{read} \times REQSIZE2 + T_{movd}$$

$$T_{slat}1 - T_{slat}2 = Coeff_{read} \times Diff(SIZE1 - SIZE2)$$

Inferred from max slope of CDF *Known info.*



Request Classification

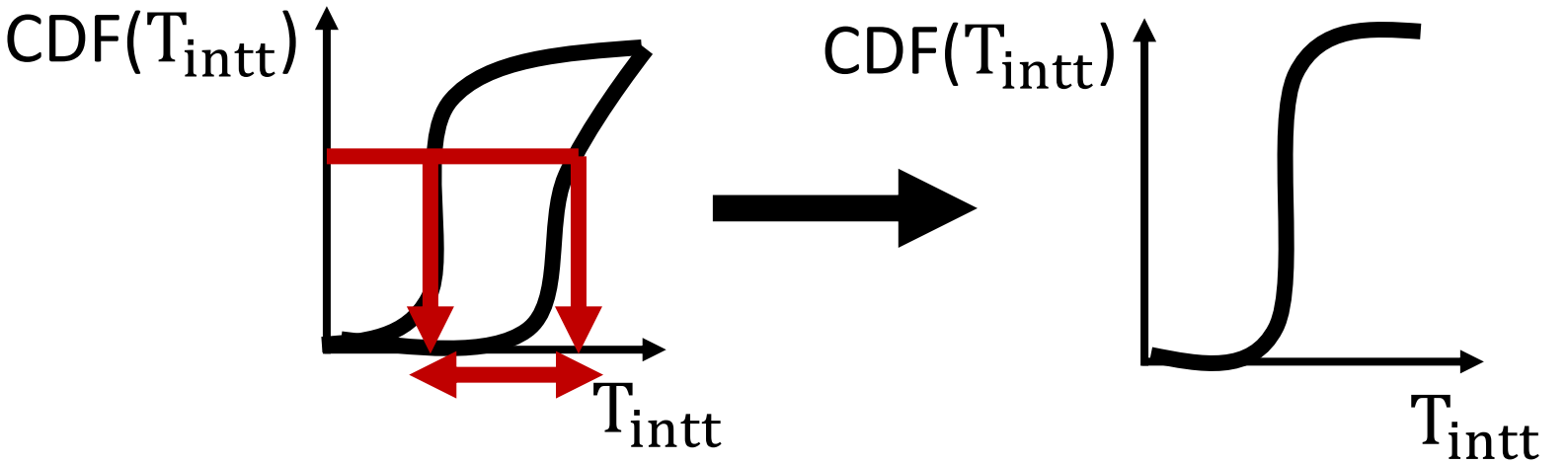
Sequential/Random Read

$$T_{slat} = T_{cdel\_read} + Coeff_{read} \times REQSIZE + T_{movd}$$

$$T_{slat}1 = T_{cdel\_read} + Coeff_{read} \times REQSIZE1 + T_{movd}$$
$$T_{slat}2 = T_{cdel\_read} + Coeff_{read} \times REQSIZE2 + T_{movd}$$

$$T_{slat}1 - T_{slat}2 = Coeff_{read} \times Diff(SIZE1 - SIZE2)$$

Inferred from max slope of CDF *Known info.*



Request Classification

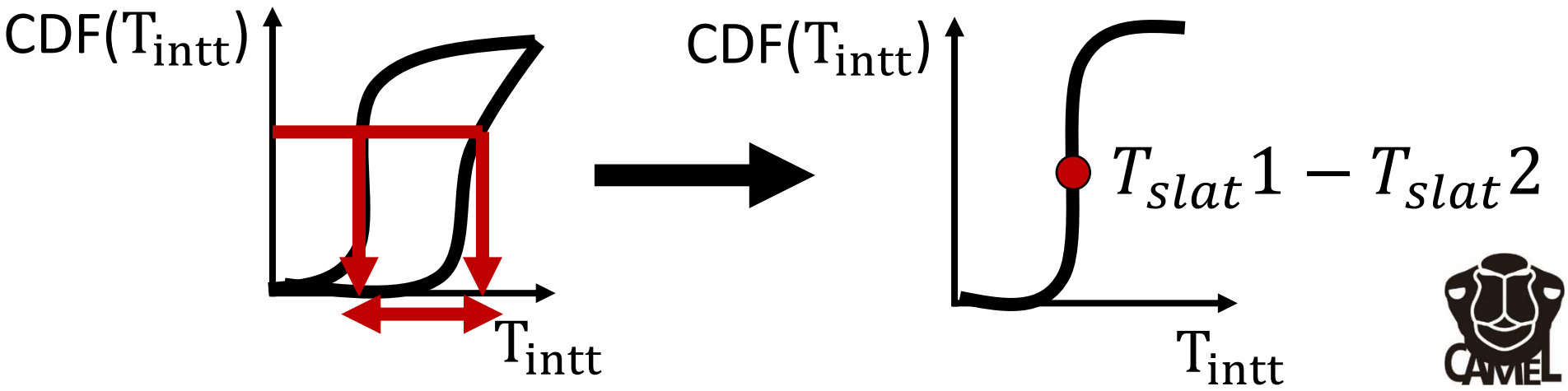
Sequential/Random Read

$$T_{slat} = T_{cdel\_read} + Coeff_{read} \times REQSIZE + T_{movd}$$

$$T_{slat}1 = T_{cdel\_read} + Coeff_{read} \times REQSIZE1 + T_{movd}$$
$$T_{slat}2 = T_{cdel\_read} + Coeff_{read} \times REQSIZE2 + T_{movd}$$

$$T_{slat}1 - T_{slat}2 = Coeff_{read} \times Diff(SIZE1 - SIZE2)$$

Inferred from max slope of CDF *Known info.*



Request Classification

Sequential/Random Read

$$T_{slat} = T_{cdel\_read} + Coeff_{read} \times REQSIZE + T_{movd}$$

$$T_{slat}1 = T_{cdel\_read} + Coeff_{read} \times REQSIZE1 + T_{movd}$$
$$T_{slat}2 = T_{cdel\_read} + Coeff_{read} \times REQSIZE2 + T_{movd}$$

$$T_{slat}1 - T_{slat}2 = Coeff_{read} \times Diff(SIZE1 - SIZE2)$$

Inferred from max slope of CDF *Known info.*

Then, which two CDF will be best?

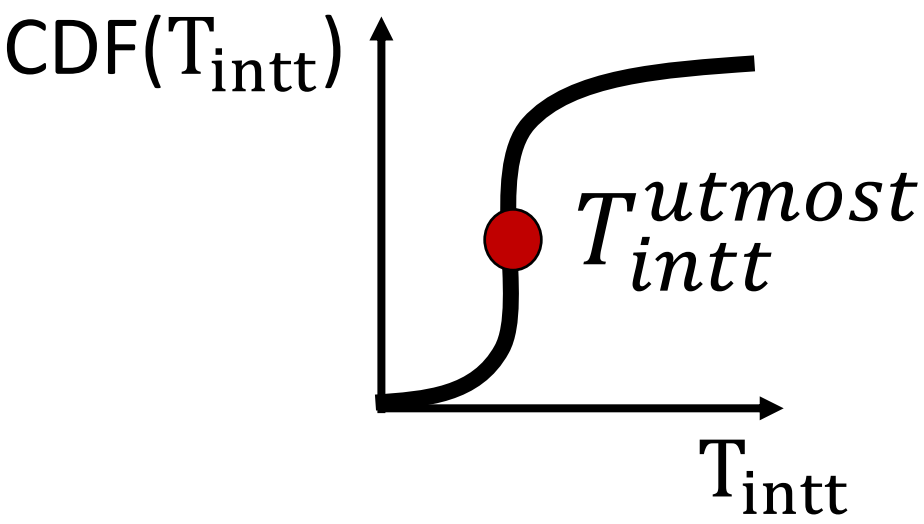
Inference Model

Key idea: Select two **steepest** CDF

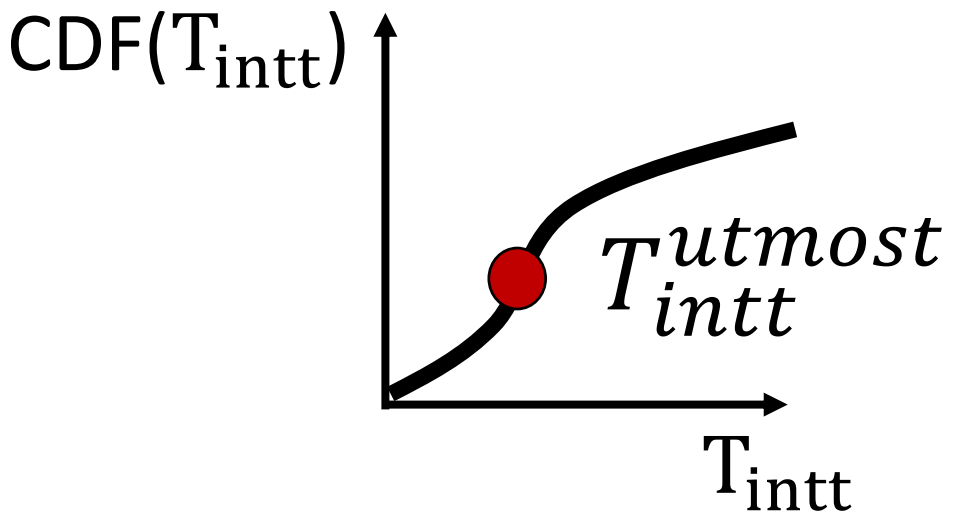
Inference Model

Key idea: Select two **steepest** CDF

∵ Steep CDF is more accurate to infer T_{slat}



REQ_SIZE 1

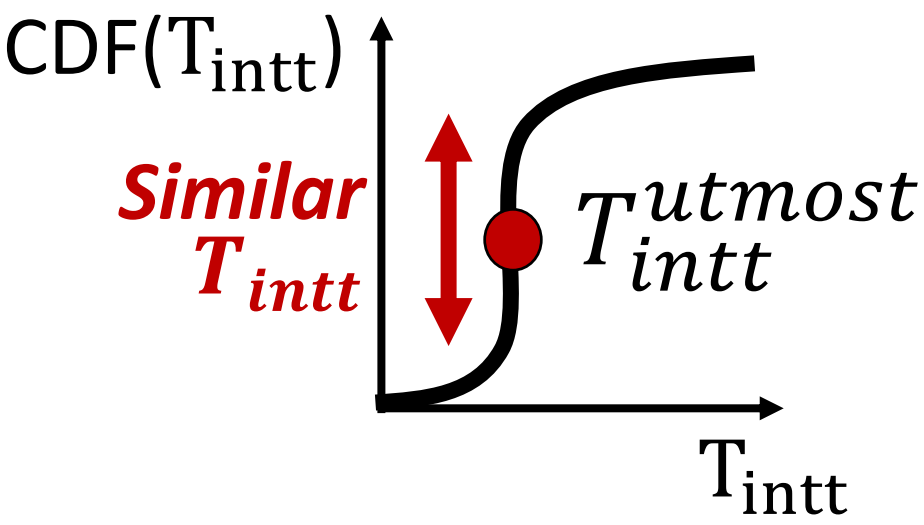


REQ_SIZE 2

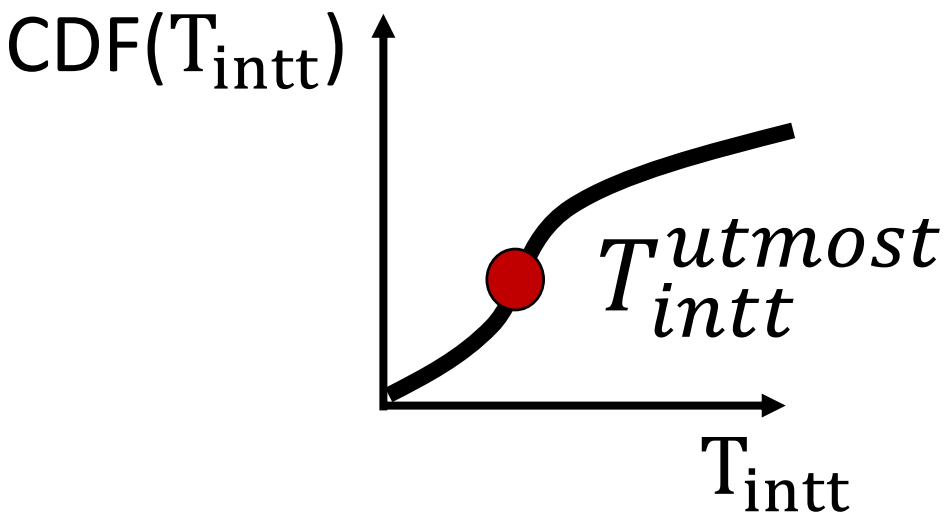
Inference Model

Key idea: Select two **steepest** CDF

∵ Steep CDF is more accurate to infer T_{slat}



REQ_SIZE 1

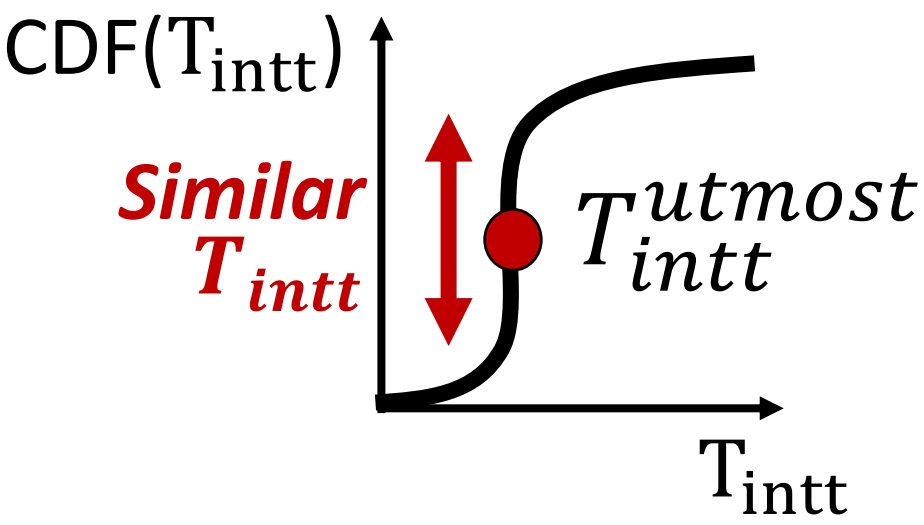


REQ_SIZE 2

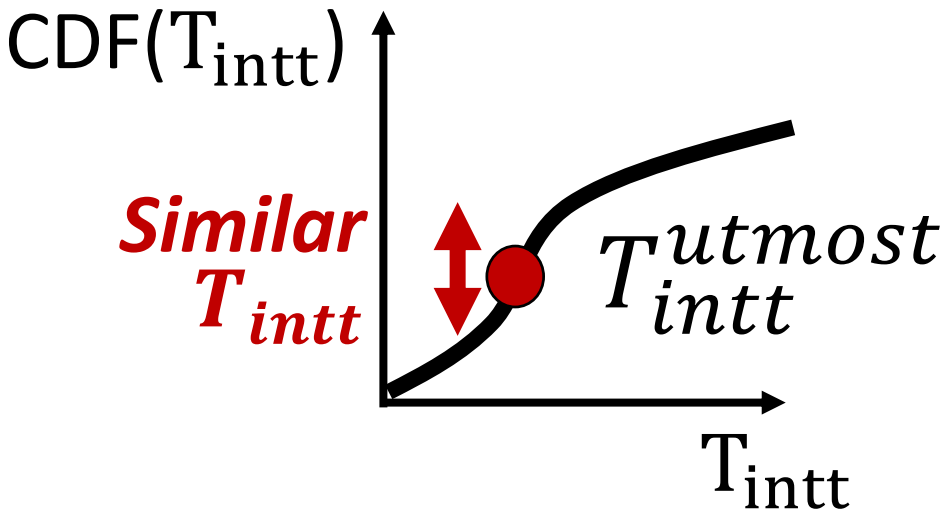
Inference Model

Key idea: Select two **steepest** CDF

∵ Steep CDF is more accurate to infer T_{slat}



REQ_SIZE 1



REQ_SIZE 2

Inference Automation

Key idea: Mathematical analysis;

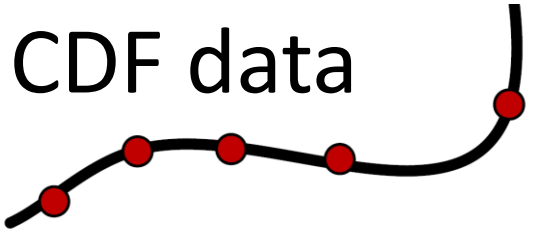
Maximum slope of CDF => Maximum of CDF’

Inference Automation

Key idea: Mathematical analysis;
Maximum slope of CDF => Maximum of CDF'

Challenge: Non-derivative discrete CDF data

Solution: Use pchip-interpolation



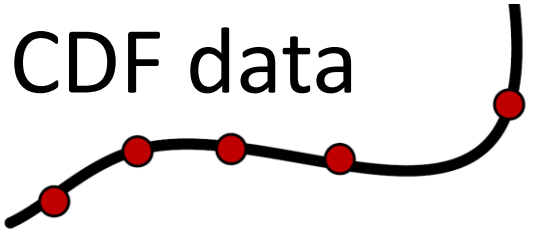
Pchip interpolation

Inference Automation

Key idea: Mathematical analysis;
Maximum slope of CDF => Maximum of CDF'

Challenge: Non-derivative discrete CDF data

Solution: Use pchip-interpolation



Pchip interpolation

Challenge: High cost of interpolation

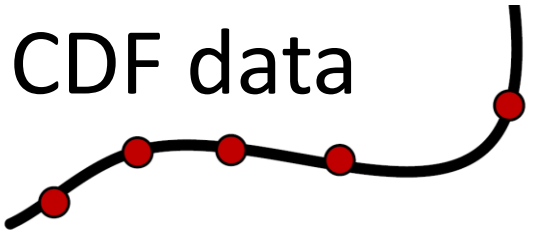
Solution: Use both PDF and CDF method

Inference Automation

Key idea: Mathematical analysis;
Maximum slope of CDF => Maximum of CDF'

Challenge: Non-derivative discrete CDF data

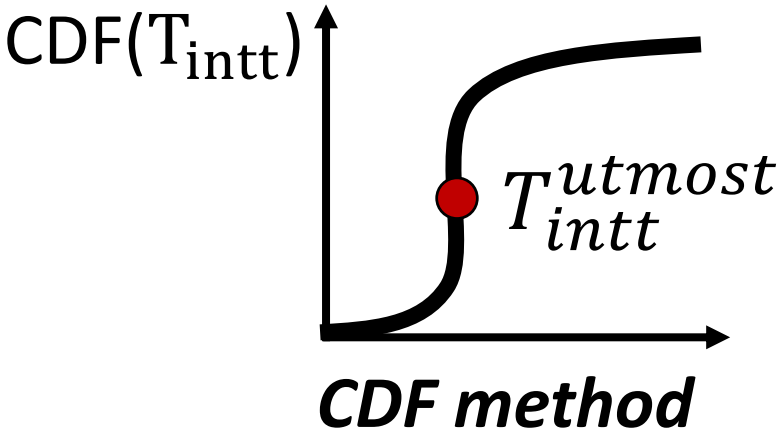
Solution: Use pchip-interpolation



Pchip interpolation

Challenge: High cost of interpolation

Solution: Use both PDF and CDF method

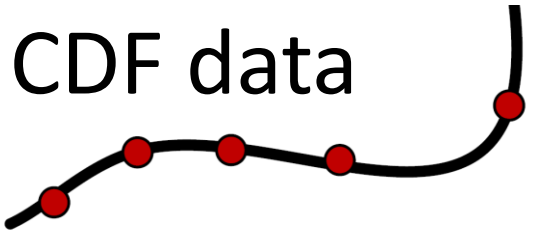


Inference Automation

Key idea: Mathematical analysis;
Maximum slope of CDF => Maximum of CDF'

Challenge: Non-derivative discrete CDF data

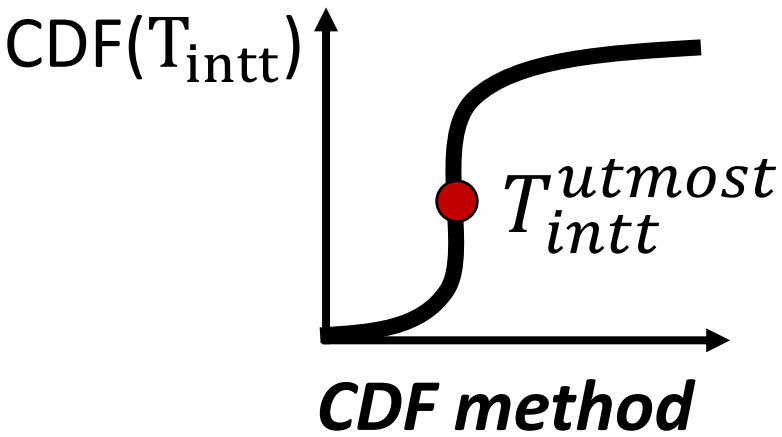
Solution: Use pchip-interpolation

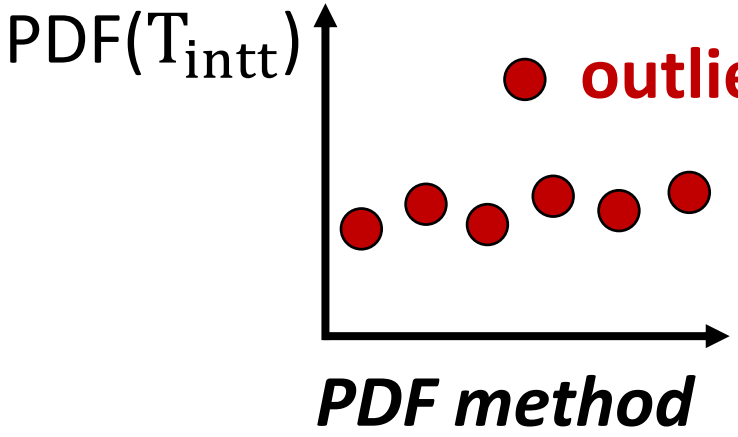


Pchip interpolation

Challenge: High cost of interpolation

Solution: Use both PDF and CDF method



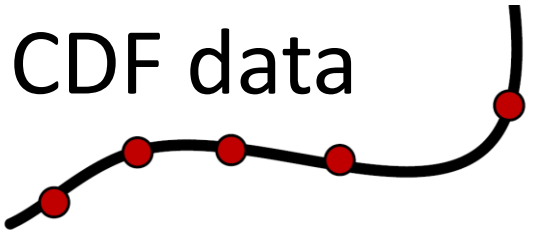


Inference Automation

Key idea: Mathematical analysis;
Maximum slope of CDF => Maximum of CDF'

Challenge: Non-derivative discrete CDF data

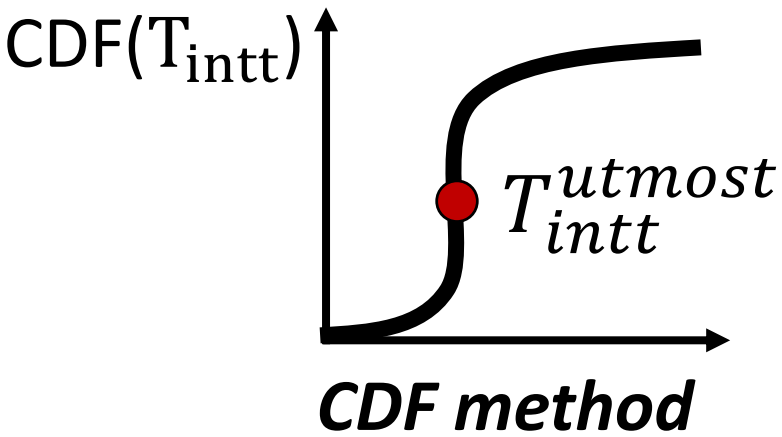
Solution: Use pchip-interpolation

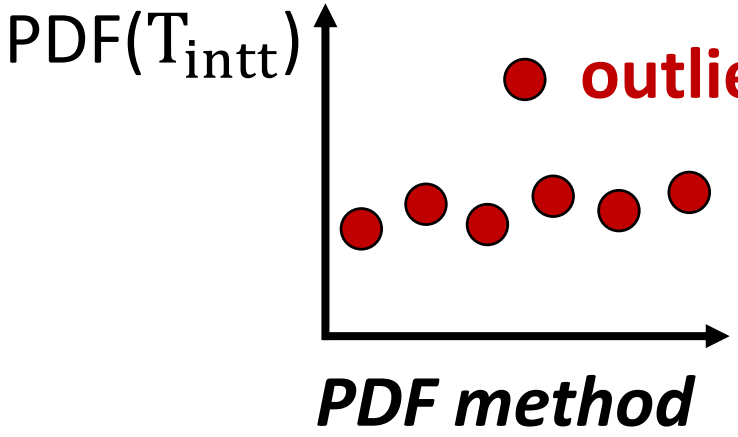


Pchip interpolation

Challenge: High cost of interpolation

Solution: Use both PDF and CDF method



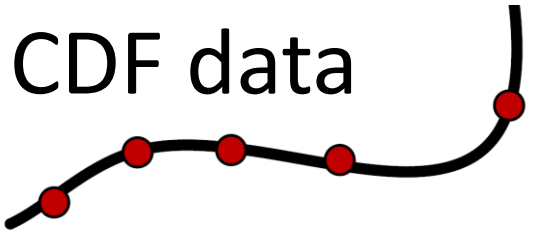


Inference Automation

Key idea: Mathematical analysis;
Maximum slope of CDF => Maximum of CDF'

Challenge: Non-derivative discrete CDF data

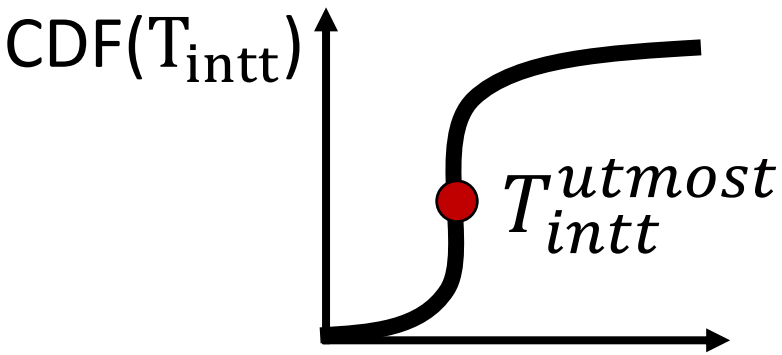
Solution: Use pchip-interpolation



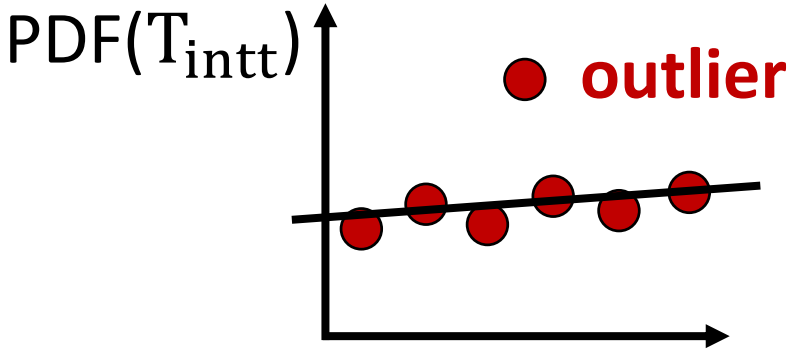
Pchip interpolation

Challenge: High cost of interpolation

Solution: Use both PDF and CDF method



CDF method



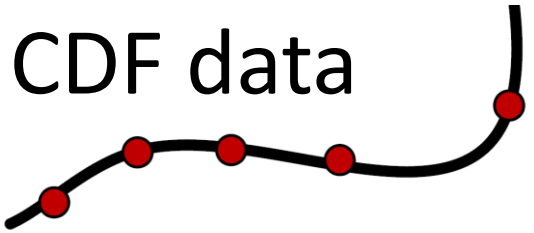
PDF method

Inference Automation

Key idea: Mathematical analysis;
Maximum slope of CDF => Maximum of CDF'

Challenge: Non-derivative discrete CDF data

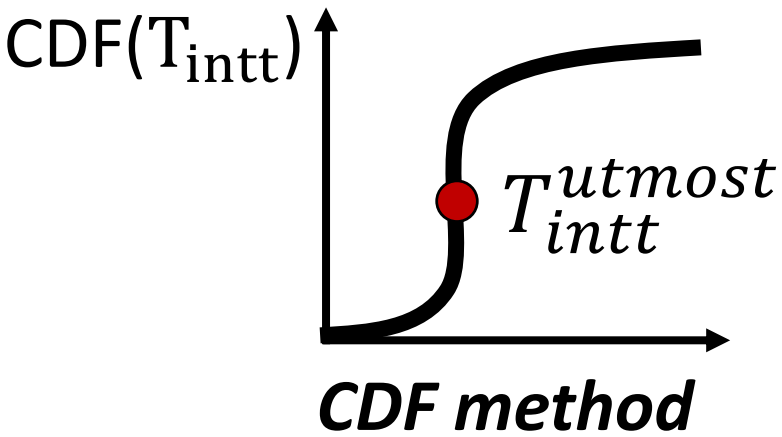
Solution: Use pchip-interpolation



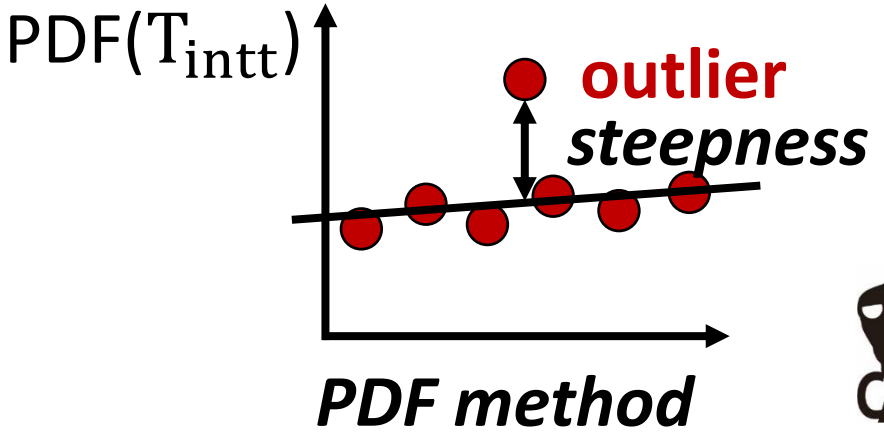
Pchip interpolation

Challenge: High cost of interpolation

Solution: Use both PDF and CDF method



CDF method



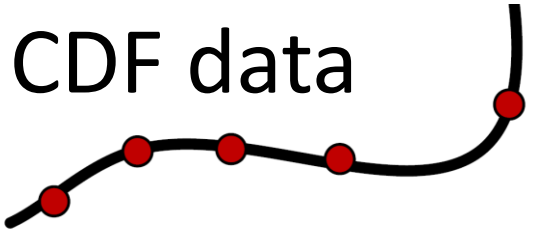
PDF method

Inference Automation

Key idea: Mathematical analysis;
Maximum slope of CDF => Maximum of CDF'

Challenge: Non-derivative discrete CDF data

Solution: Use pchip-interpolation



Pchip interpolation

Challenge: High cost of interpolation

Solution:

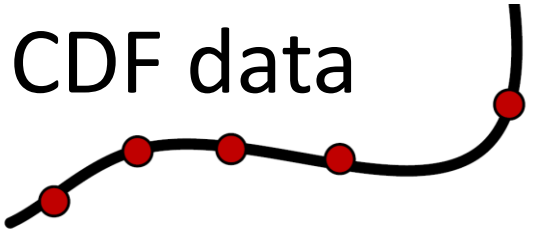
	PDF method	CDF method
Accuracy	<	
Efficiency	>	

Inference Automation

Key idea: Mathematical analysis;
Maximum slope of CDF => Maximum of CDF'

Challenge: Non-derivative discrete CDF data

Solution: Use pchip-interpolation



Pchip interpolation

Challenge: High cost of interpolation

Solution:

	PDF method	CDF method
Accuracy	<	
Efficiency	>	

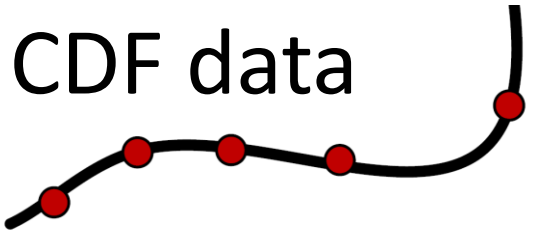
Steepness analysis

Inference Automation

Key idea: Mathematical analysis;
Maximum slope of CDF => Maximum of CDF'

Challenge: Non-derivative discrete CDF data

Solution: Use pchip-interpolation



Pchip interpolation

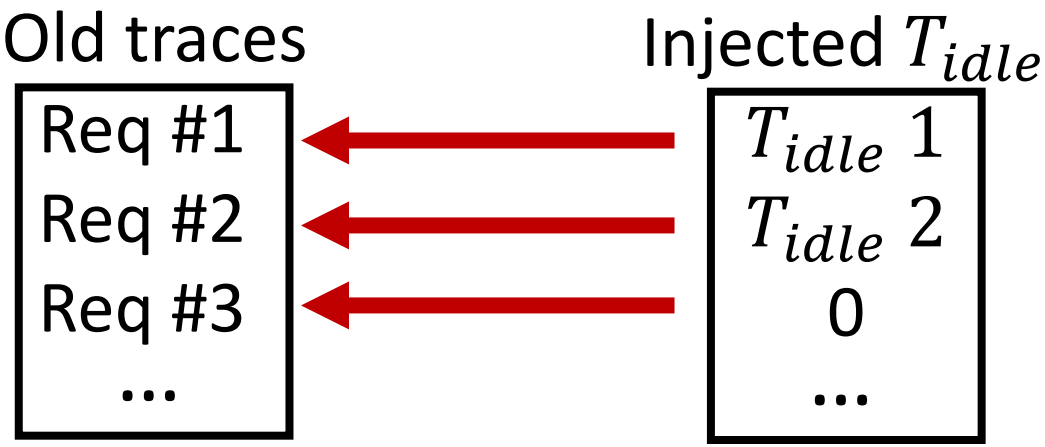
Challenge: High cost of interpolation

Solution:

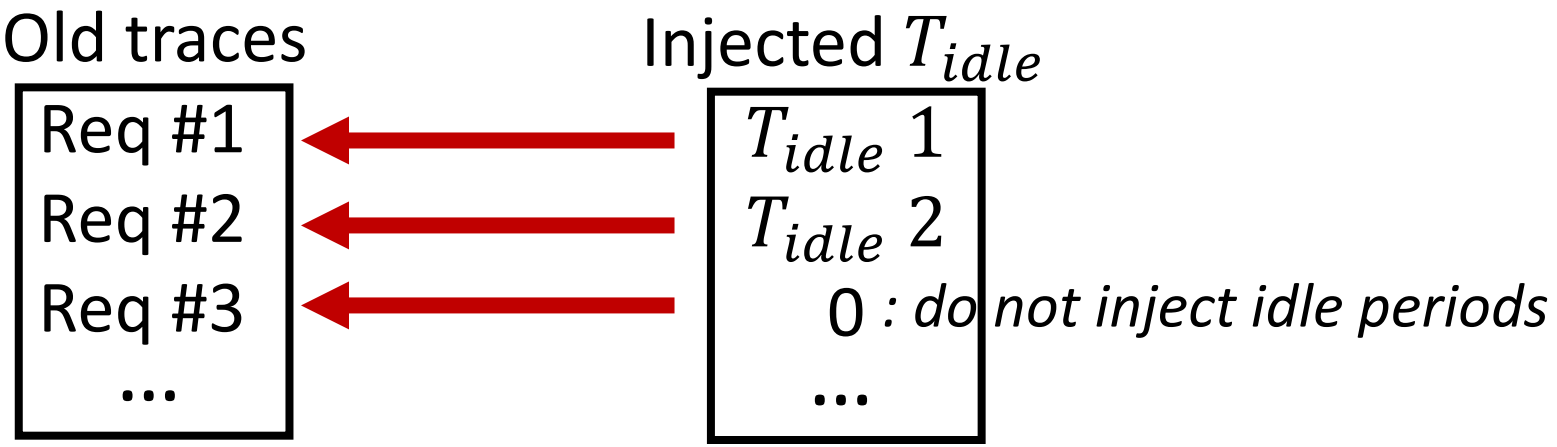
	PDF method	CDF method
Accuracy	<	
Efficiency	>	

Find max slope of CDF(diff)

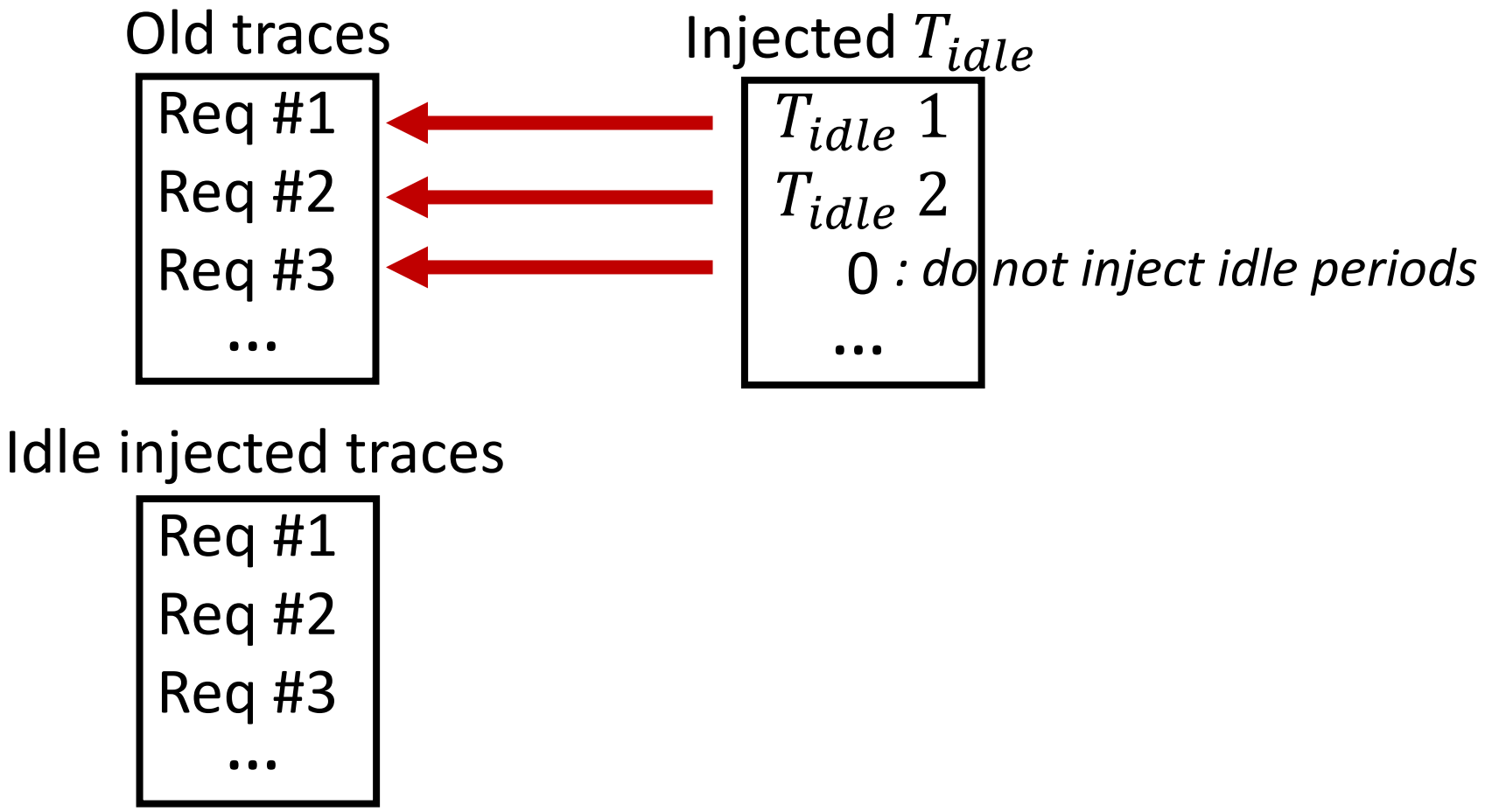
Verification



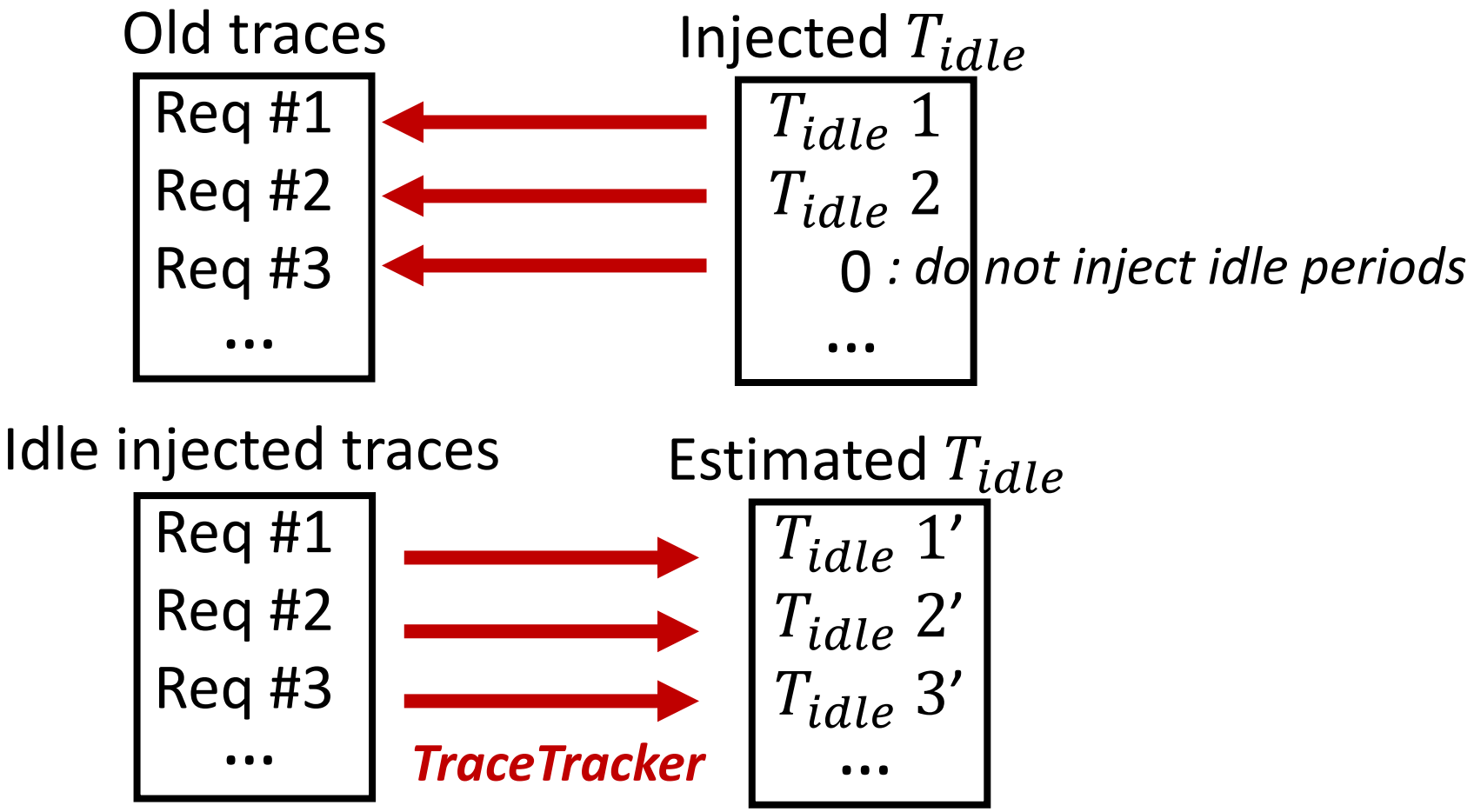
Verification



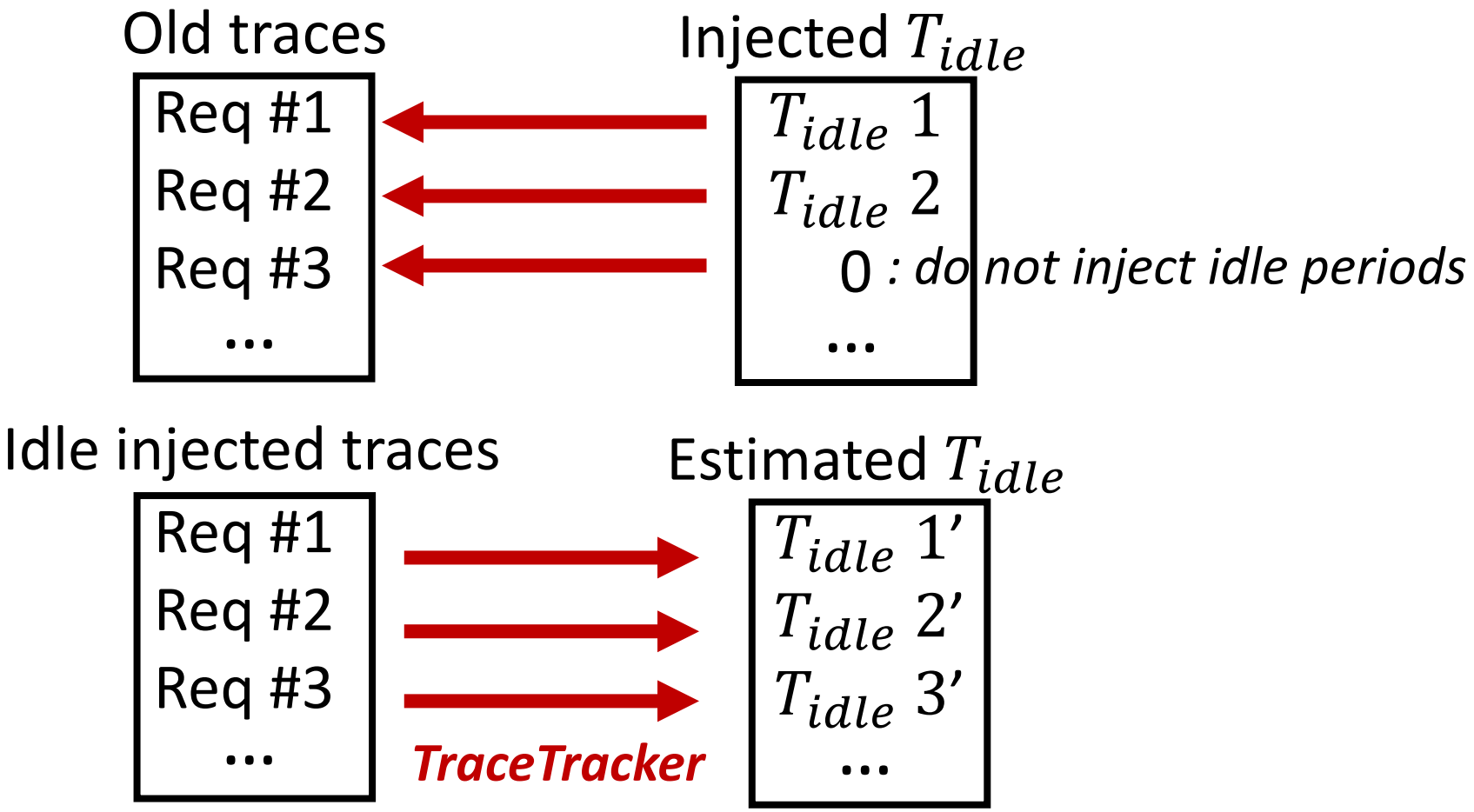
Verification



Verification



Verification



1) Number of T_{idle} occurrence: 99%

2) Total periods of T_{idle}: 96%