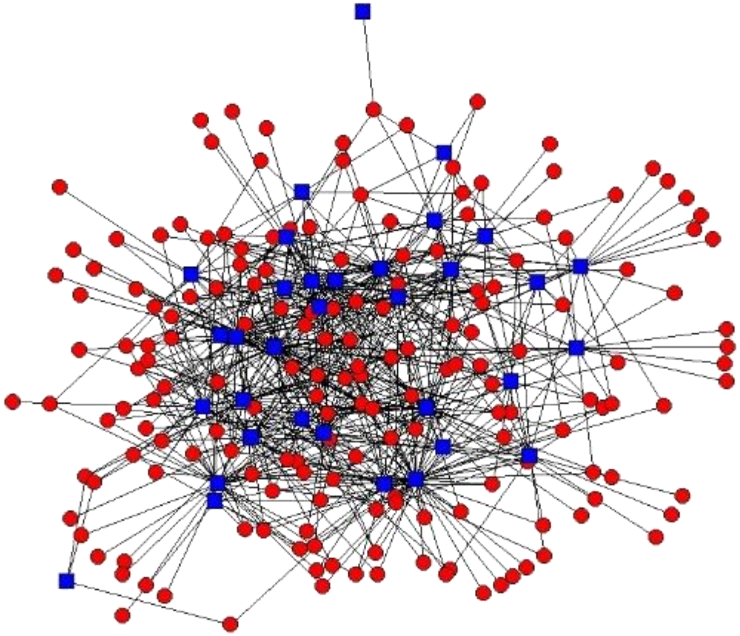
Accelerating Irregular Computations with Hardware Transactional Memory and Active Messages

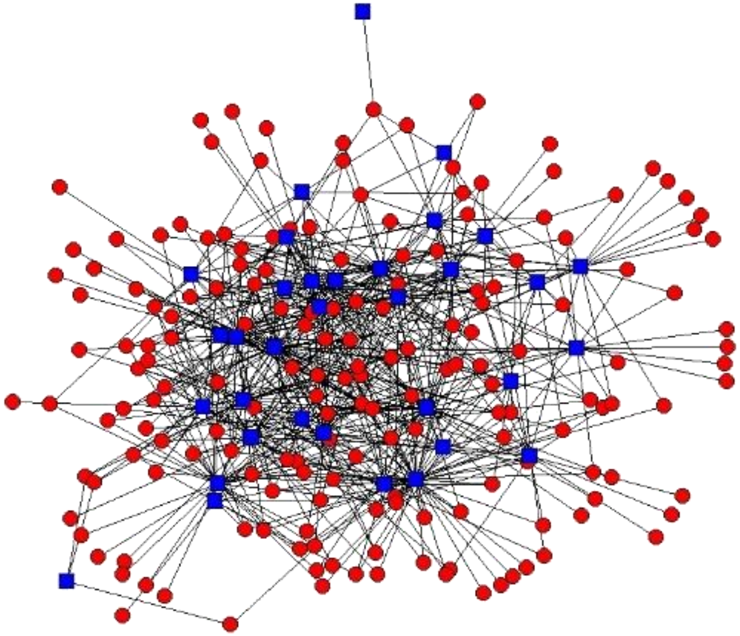
MACIEJ BESTA, TORSTEN HOEFLER

LARGE-SCALE IRREGULAR GRAPH PROCESSING



LARGE-SCALE IRREGULAR GRAPH PROCESSING

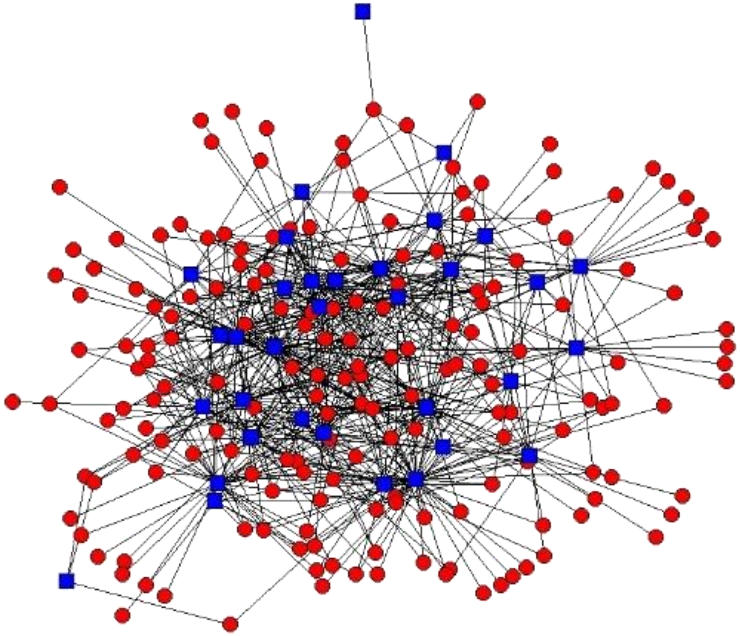
- Becoming more important [1]



[1] A. Lumsdaine et al. Challenges in Parallel Graph Processing. Parallel Processing Let. 2007.

Large-Scale Irregular Graph Processing

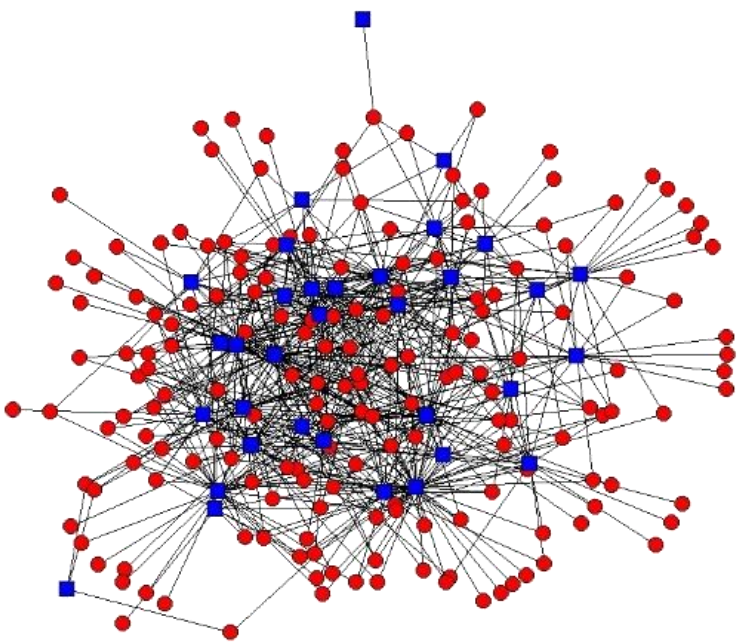
- Becoming more important [1]
  - Machine learning

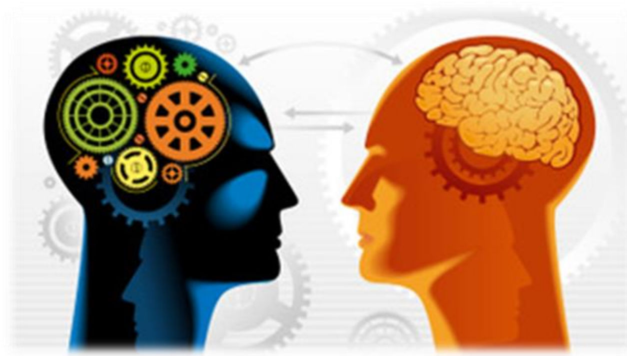


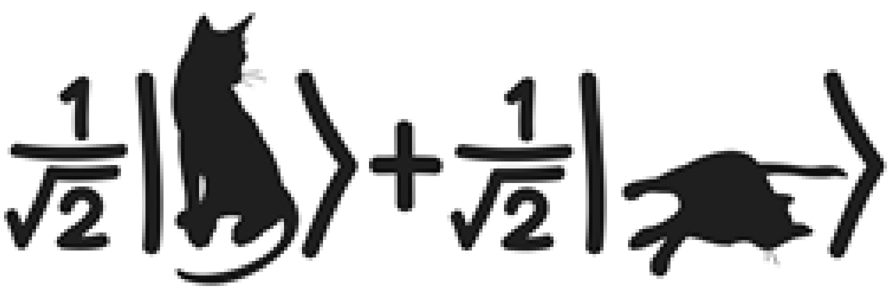
[1] A. Lumsdaine et al. Challenges in Parallel Graph Processing. Parallel Processing Let. 2007.

LARGE-SCALE IRREGULAR GRAPH PROCESSING

- Becoming more important [1]
  - Machine learning
  - Computational science



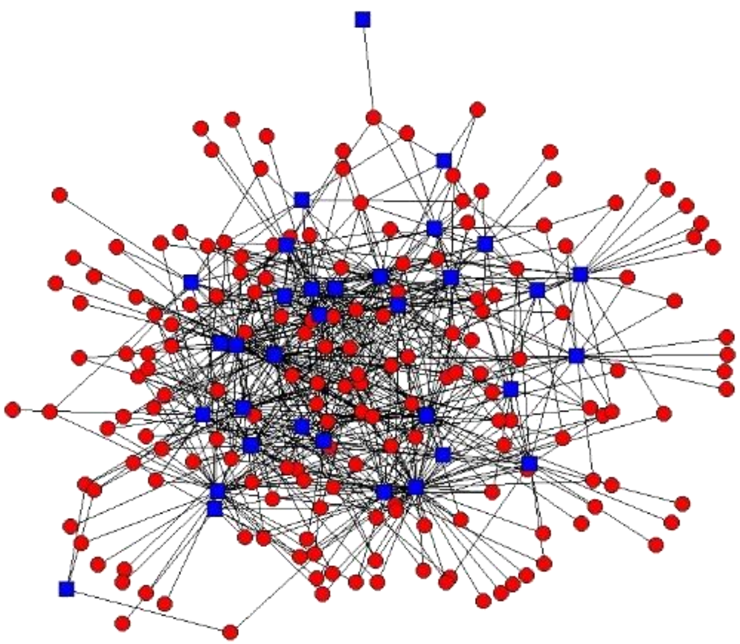


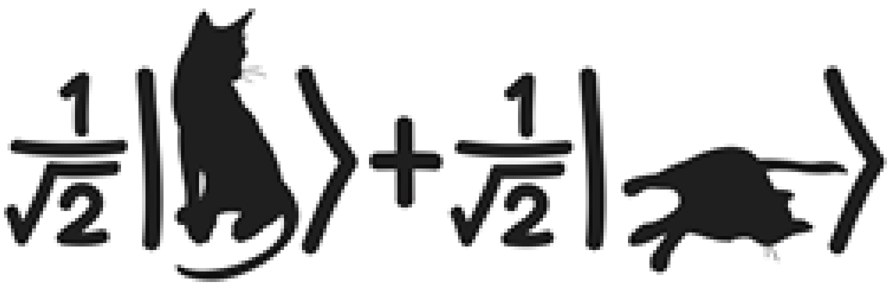


[1] A. Lumsdaine et al. Challenges in Parallel Graph Processing. Parallel Processing Let. 2007.

LARGE-SCALE IRREGULAR GRAPH PROCESSING

- Becoming more important [1]
  - Machine learning
  - Computational science
  - Social network analysis



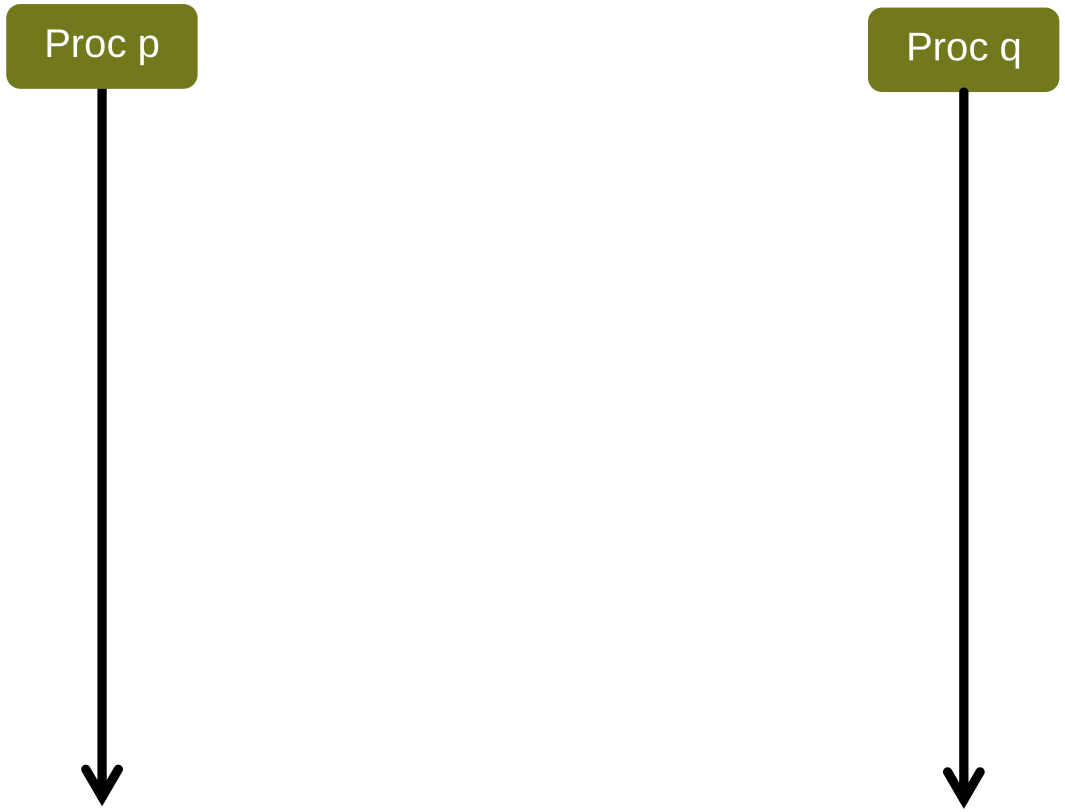


[1] A. Lumsdaine et al. Challenges in Parallel Graph Processing. Parallel Processing Let. 2007.

Synchronization Mechanisms

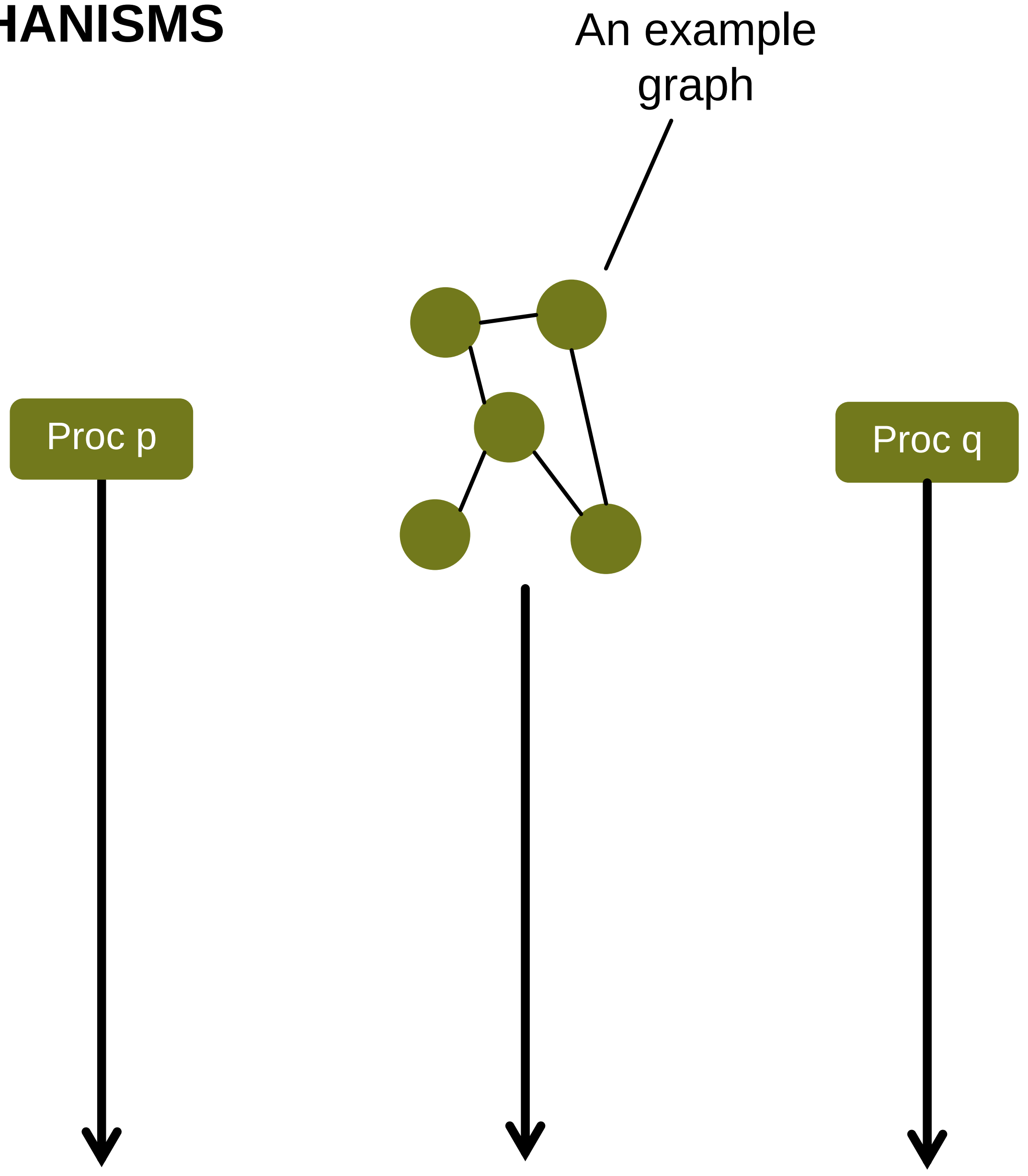
Coarse locks

Synchronization Mechanisms
Coarse locks



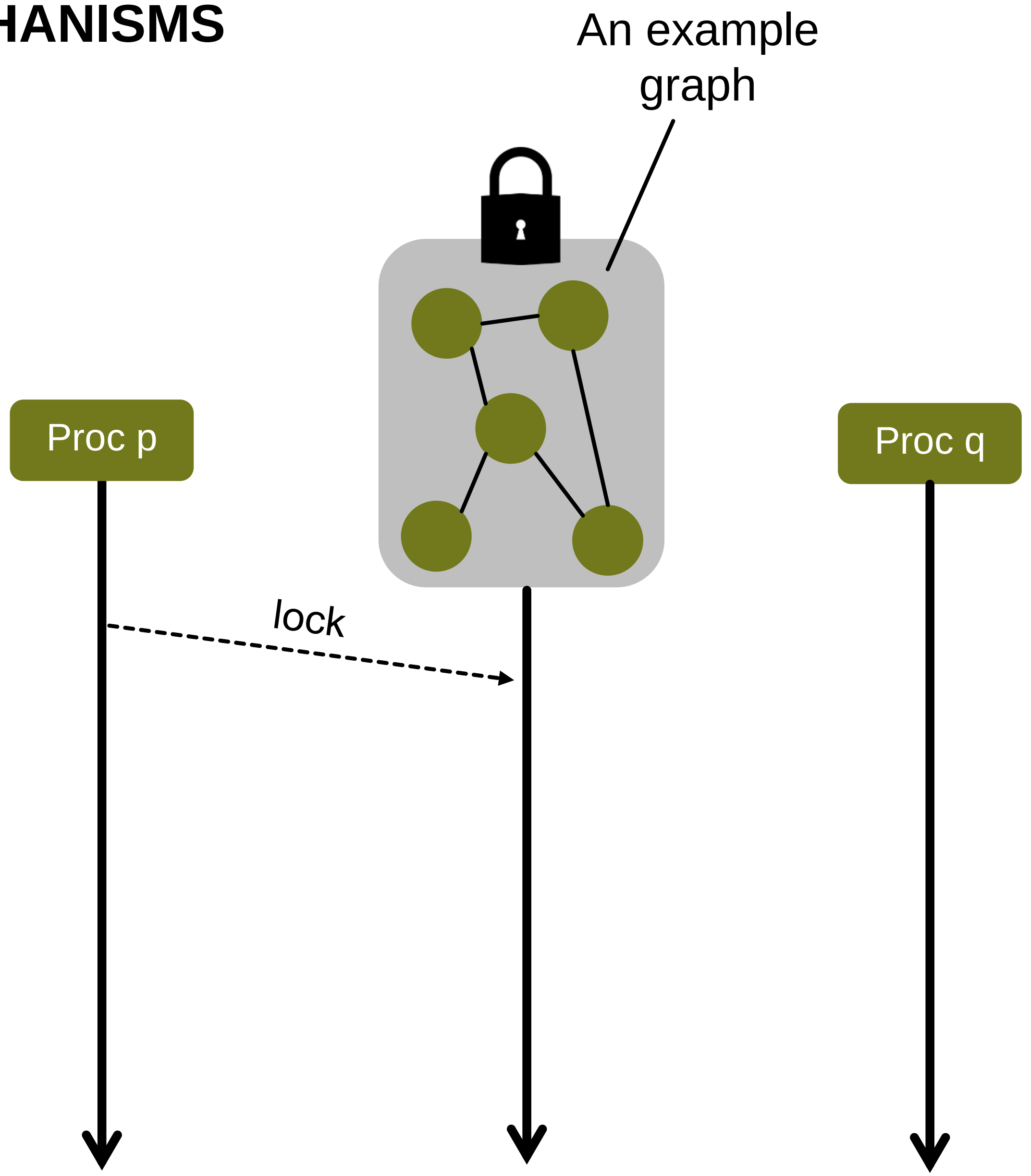
Synchronization Mechanisms

Coarse locks

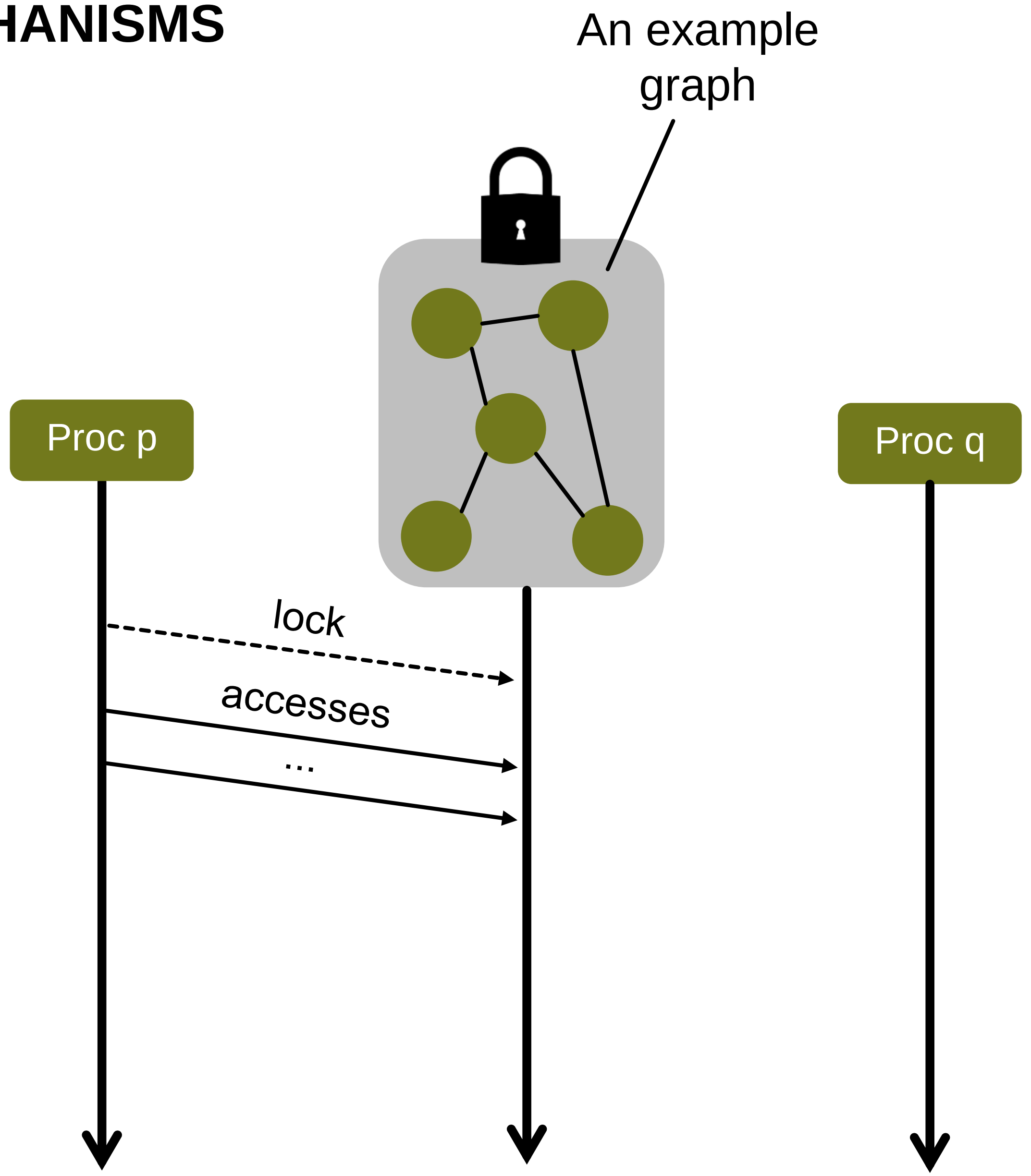


Synchronization Mechanisms

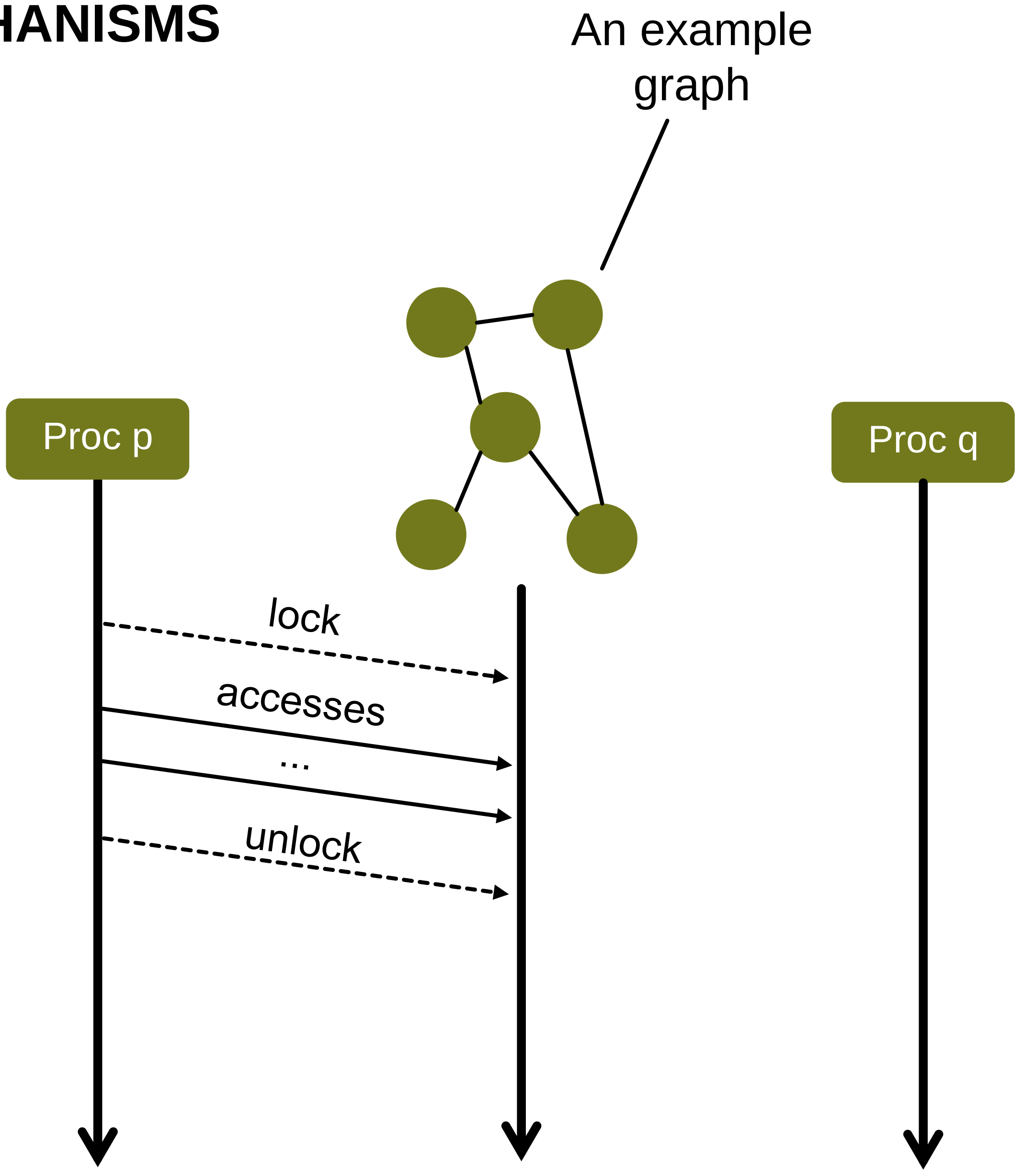
Coarse locks



Synchronization Mechanisms
Coarse locks

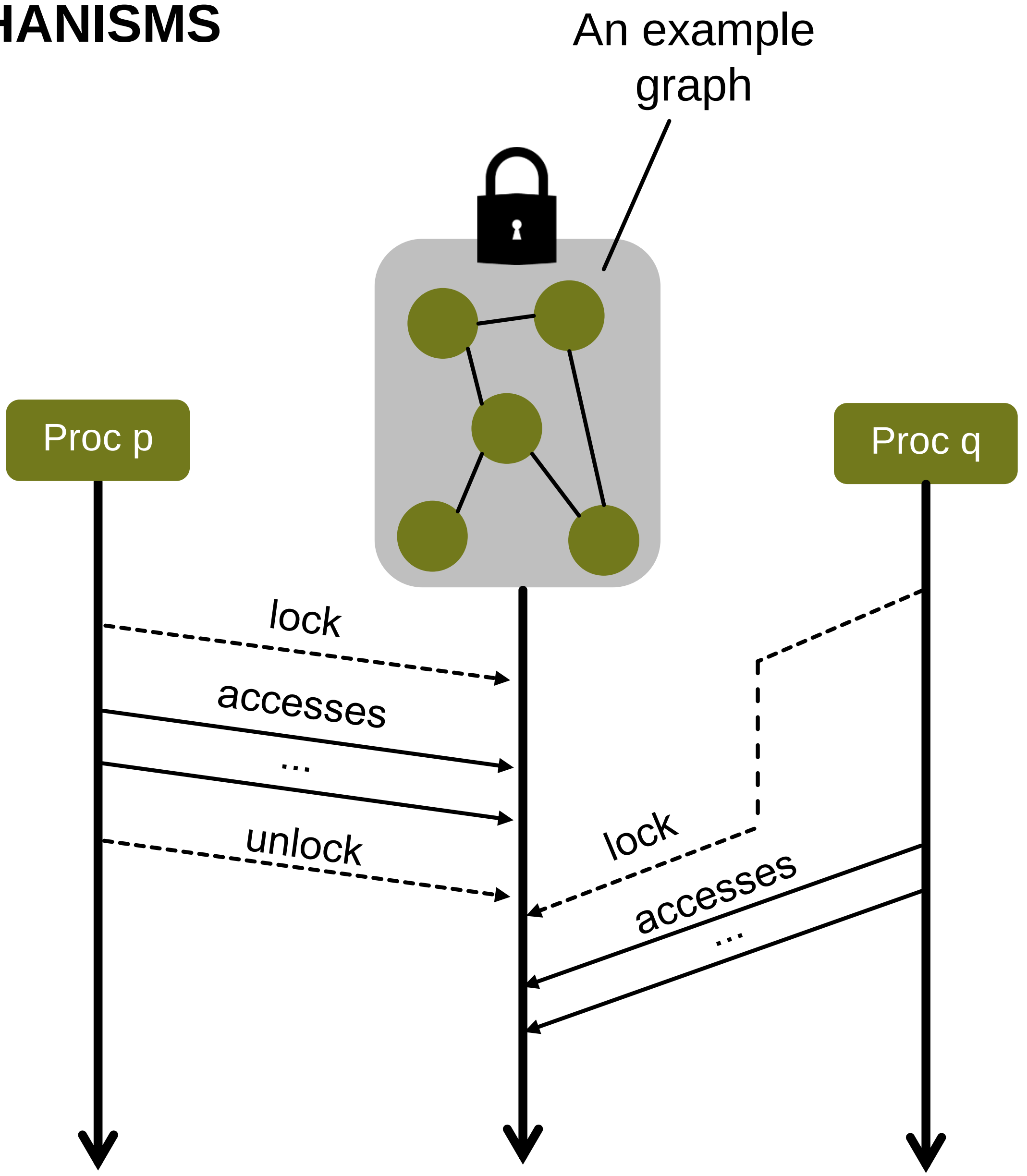


Synchronization Mechanisms
Coarse locks



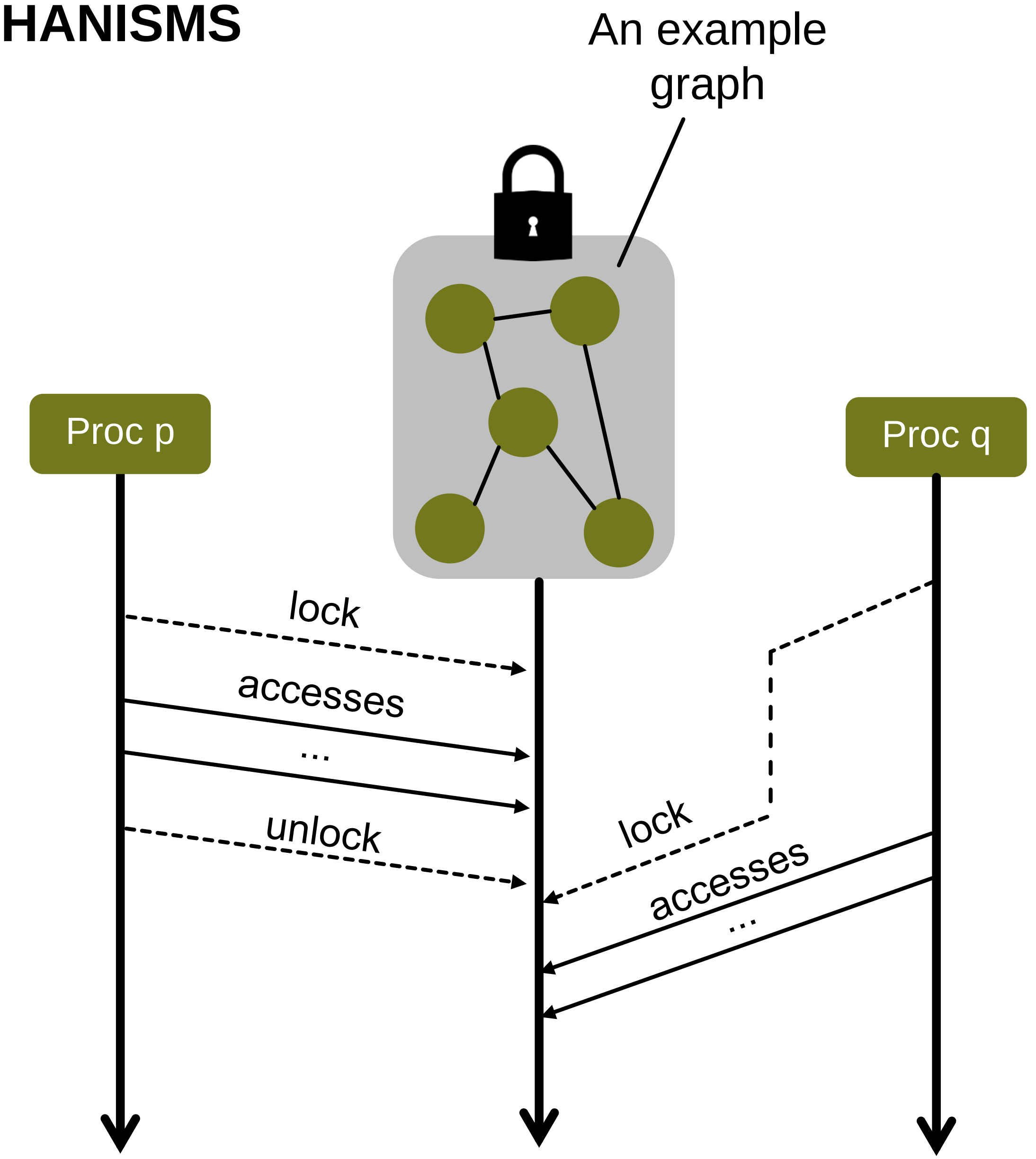
Synchronization Mechanisms

Coarse locks



Synchronization Mechanisms
Coarse locks

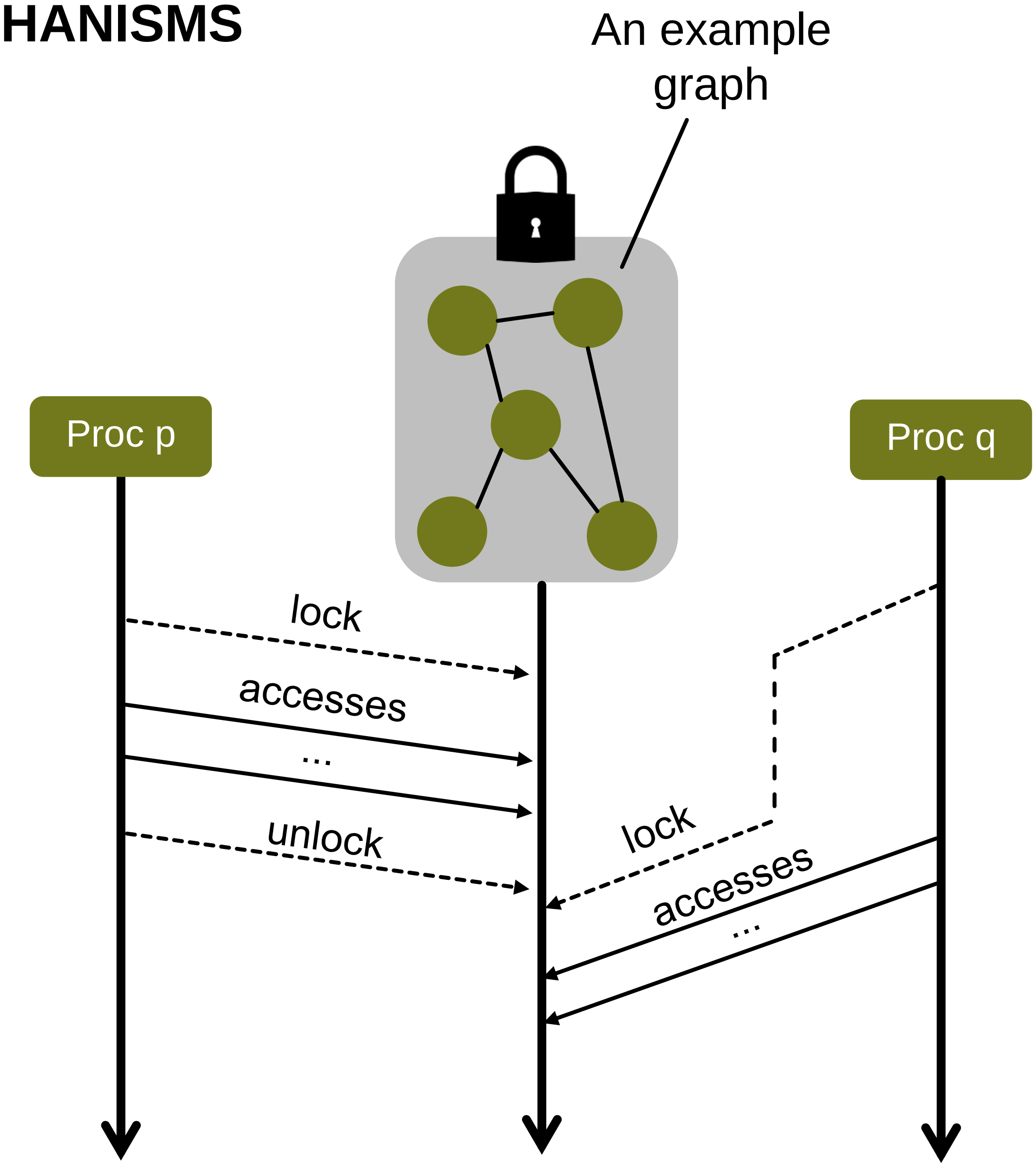
Simple protocols



Synchronization Mechanisms
Coarse locks

Simple protocols

Serialization

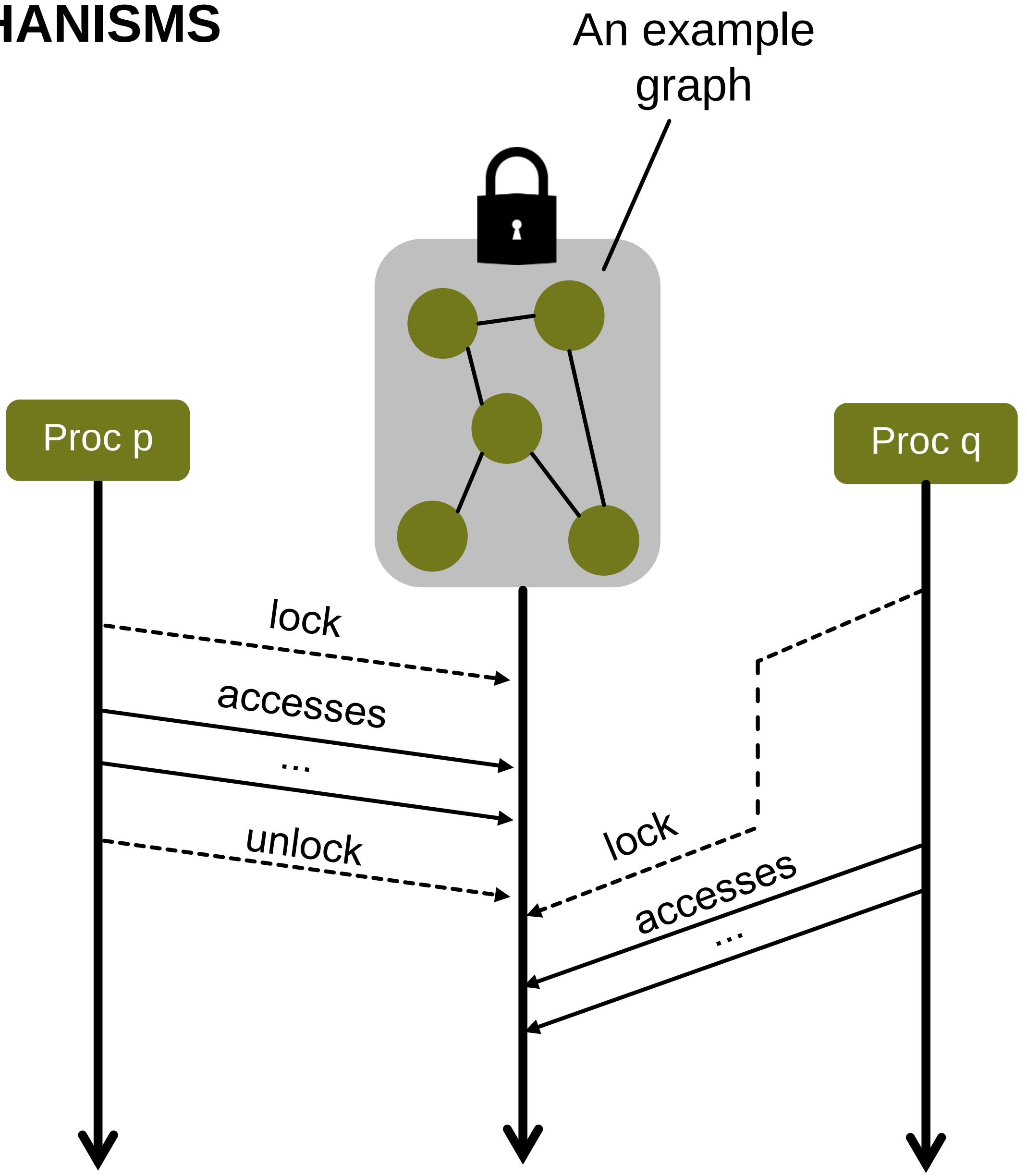


Synchronization Mechanisms
Coarse locks

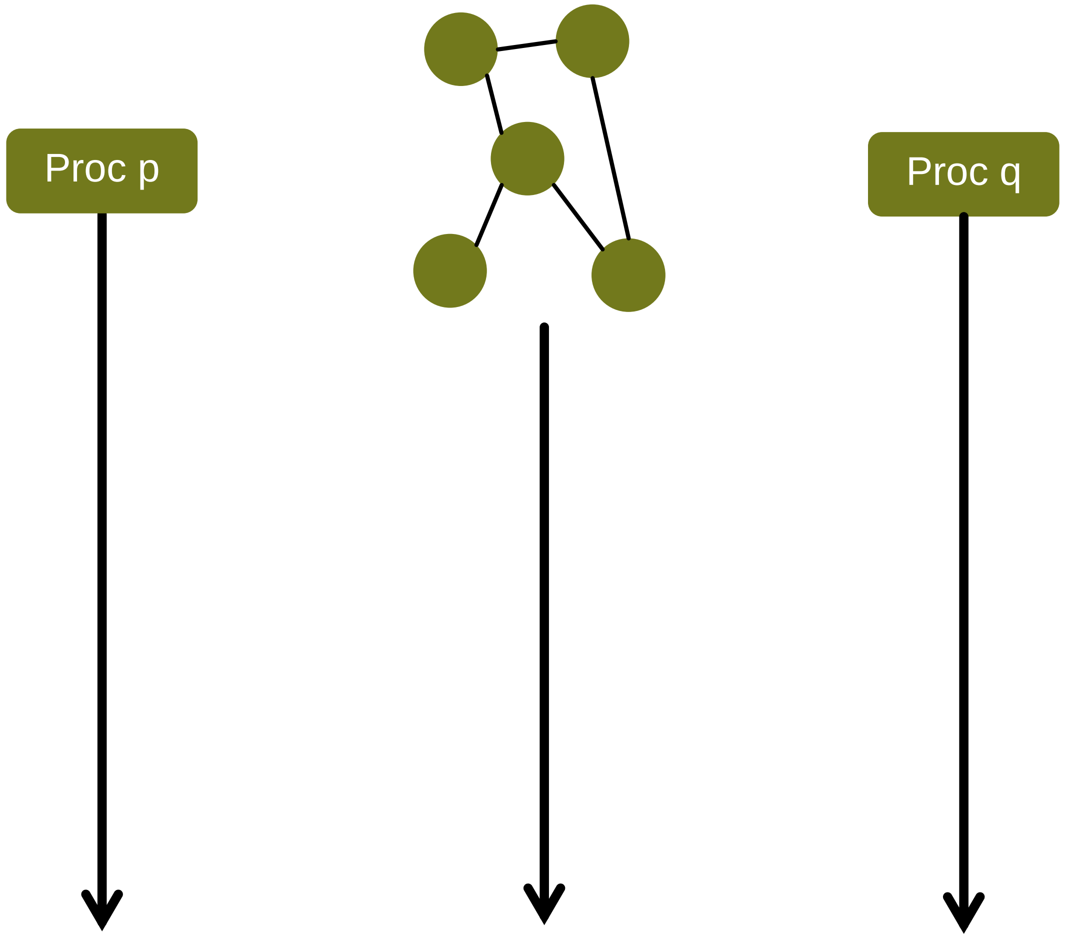
Simple protocols

Serialization

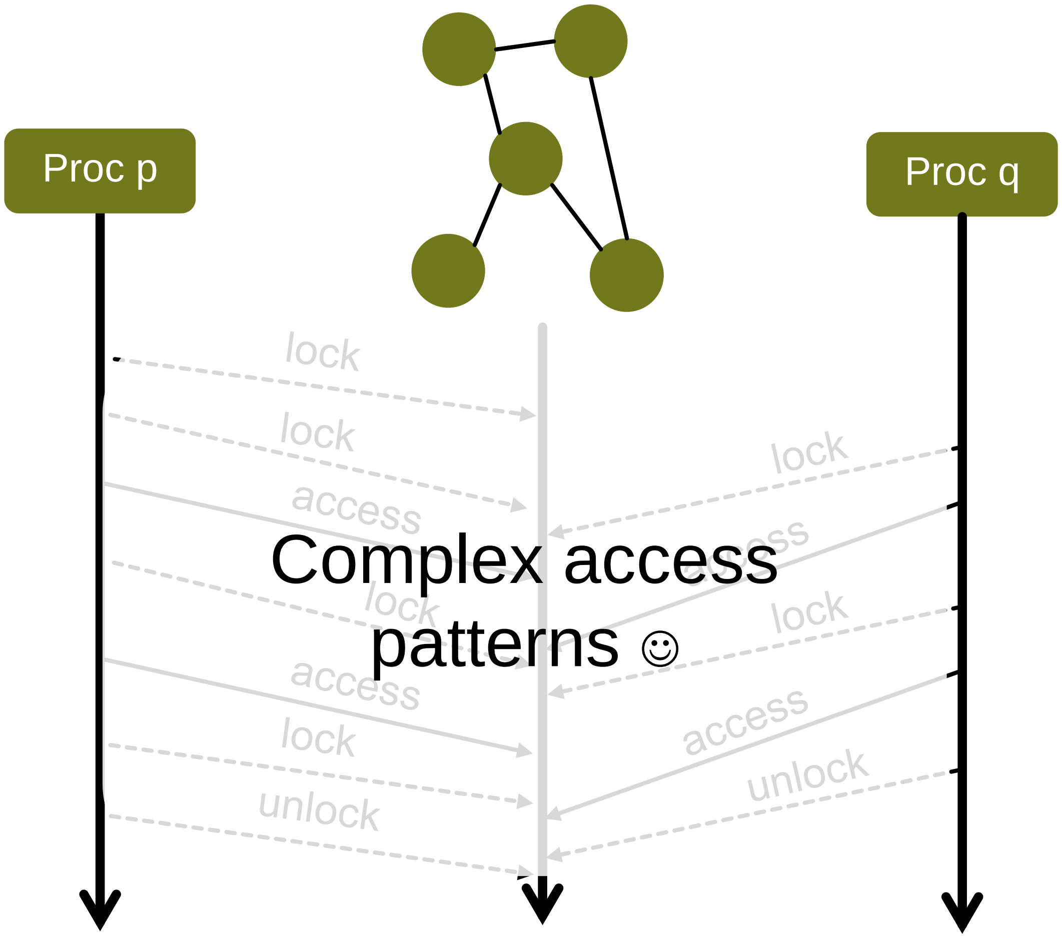
Detrimental performance



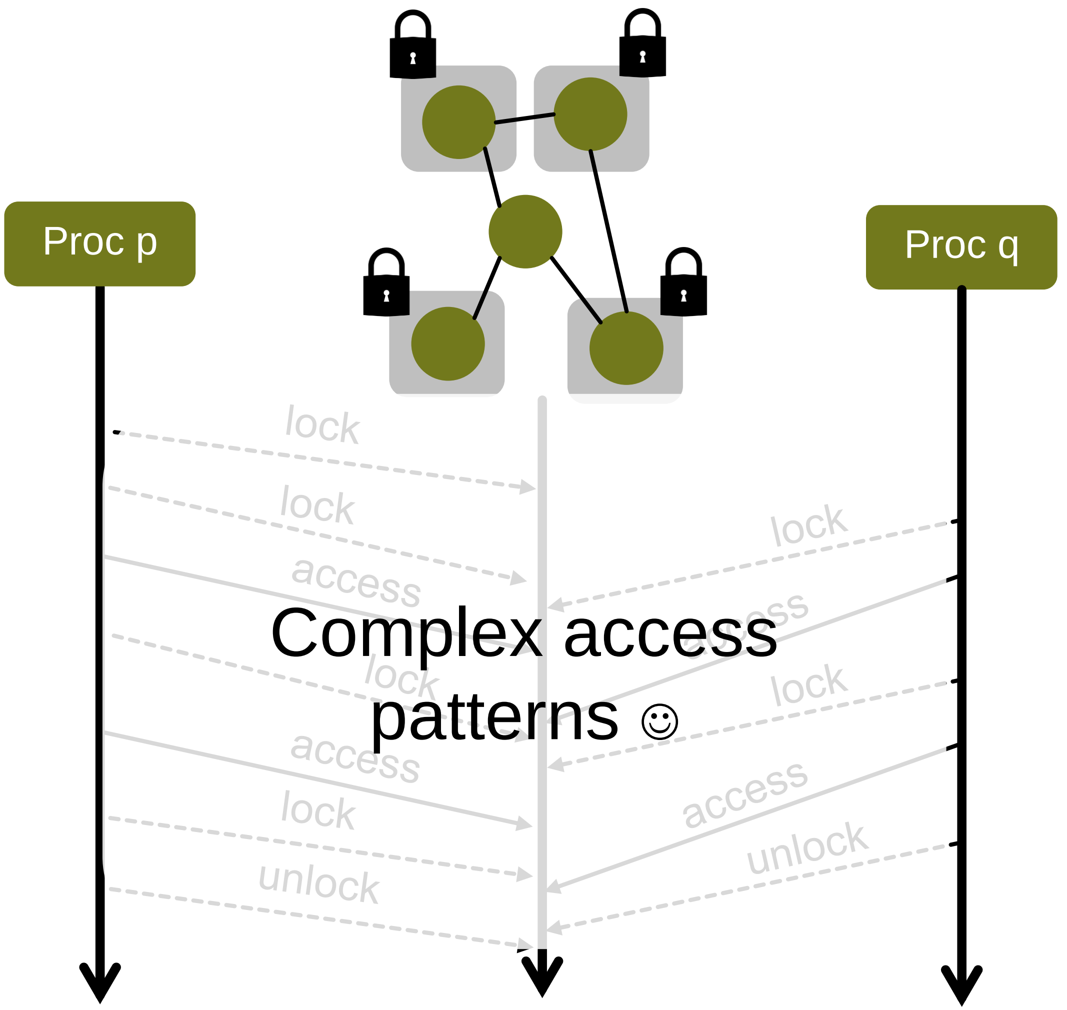
Synchronization Mechanisms
Fine locks



Synchronization Mechanisms
Fine locks

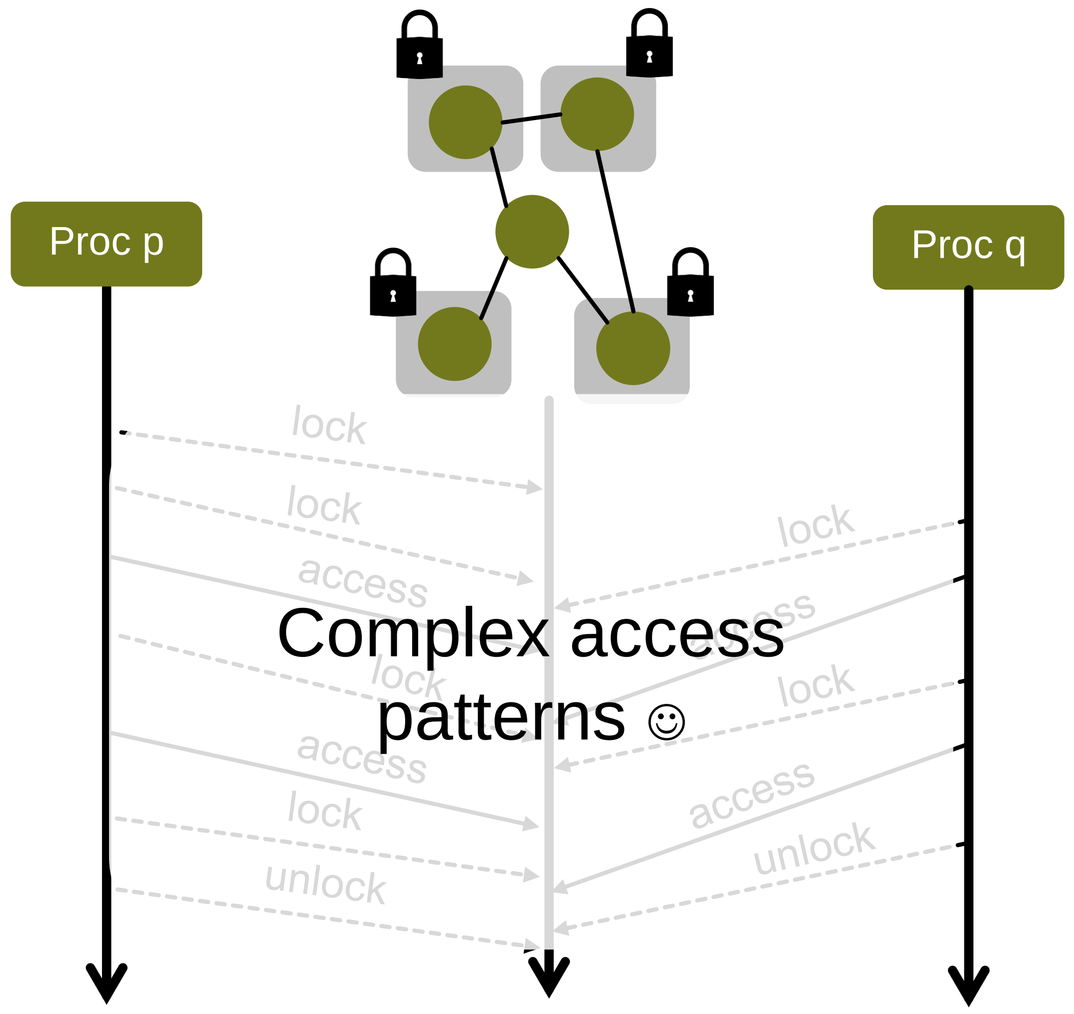


Synchronization Mechanisms
Fine locks



Synchronization Mechanisms
Fine locks

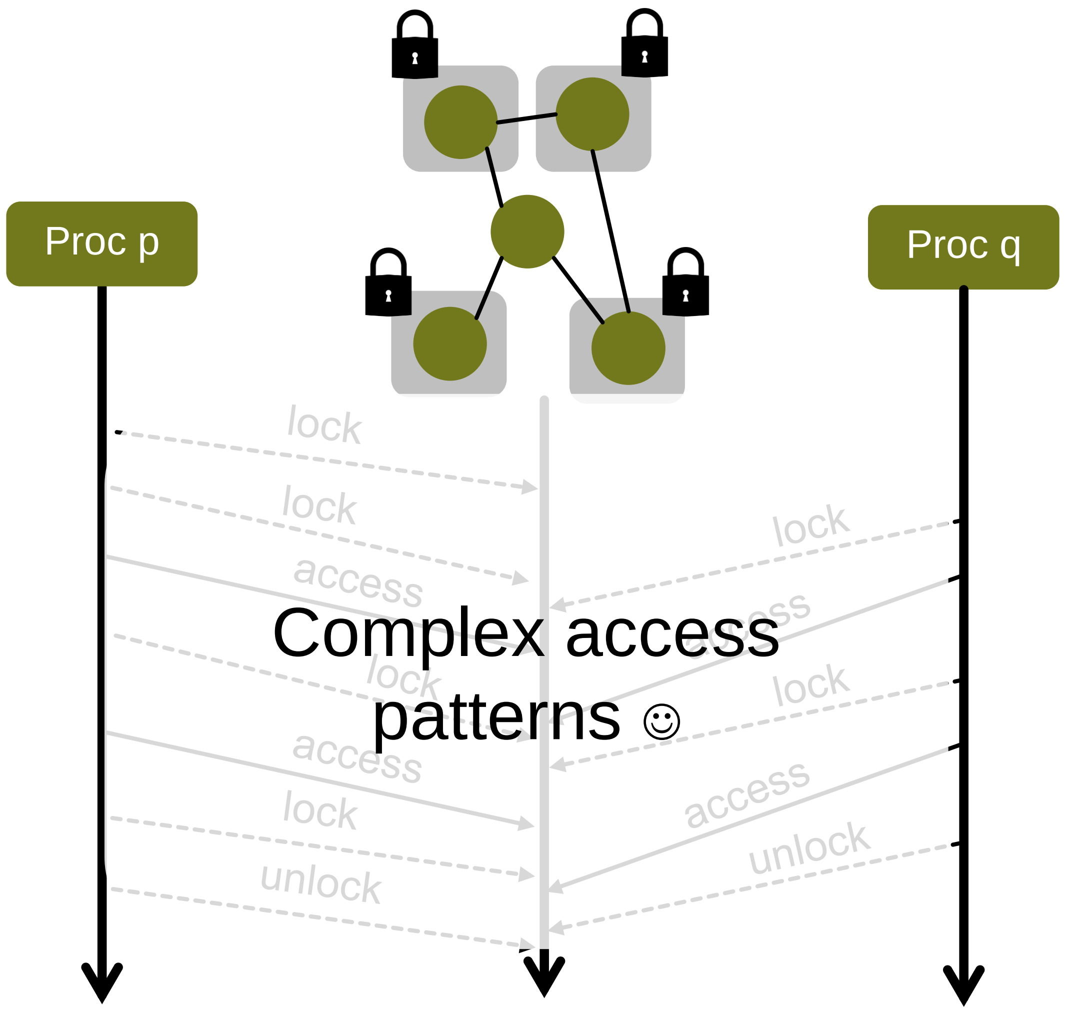
Higher performance possible



Synchronization Mechanisms
Fine locks

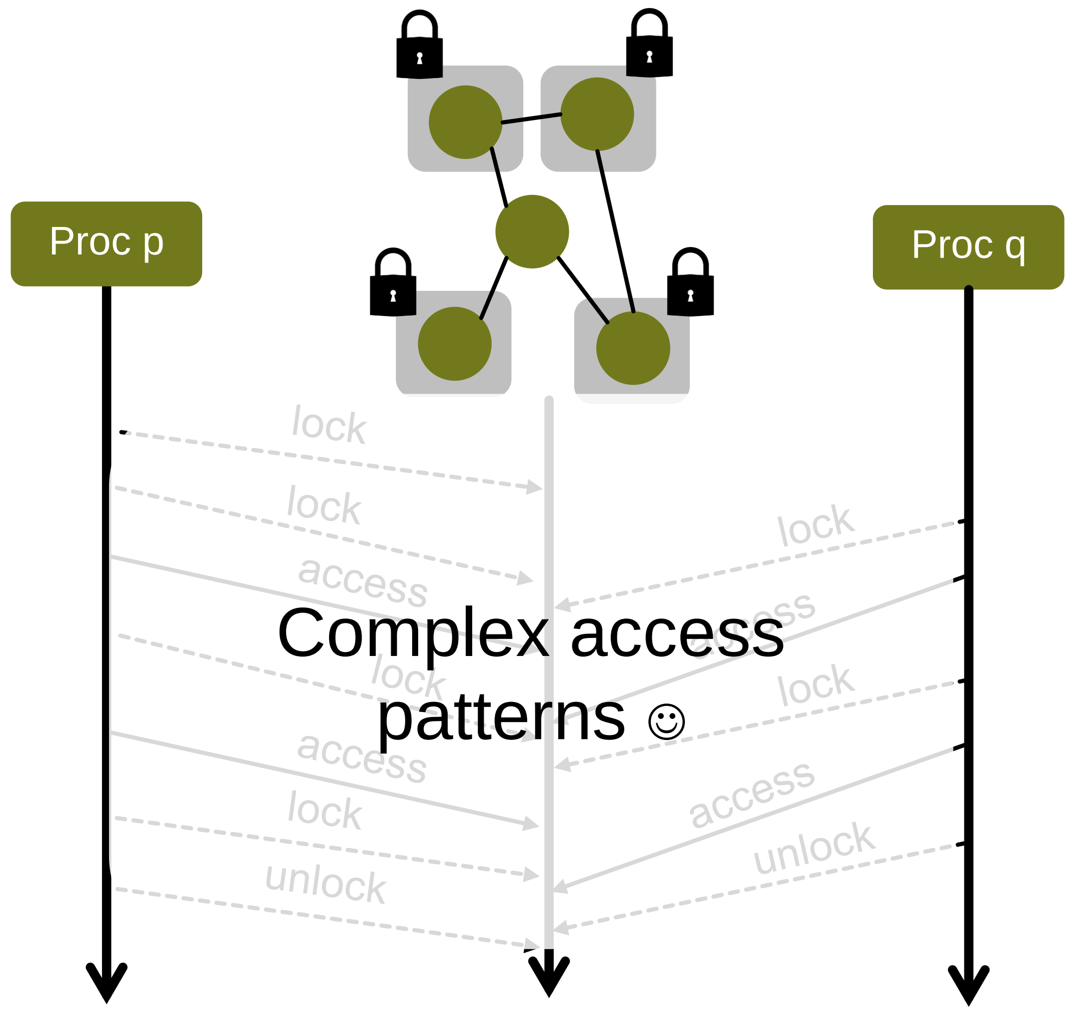
Higher performance possible

Complex protocols

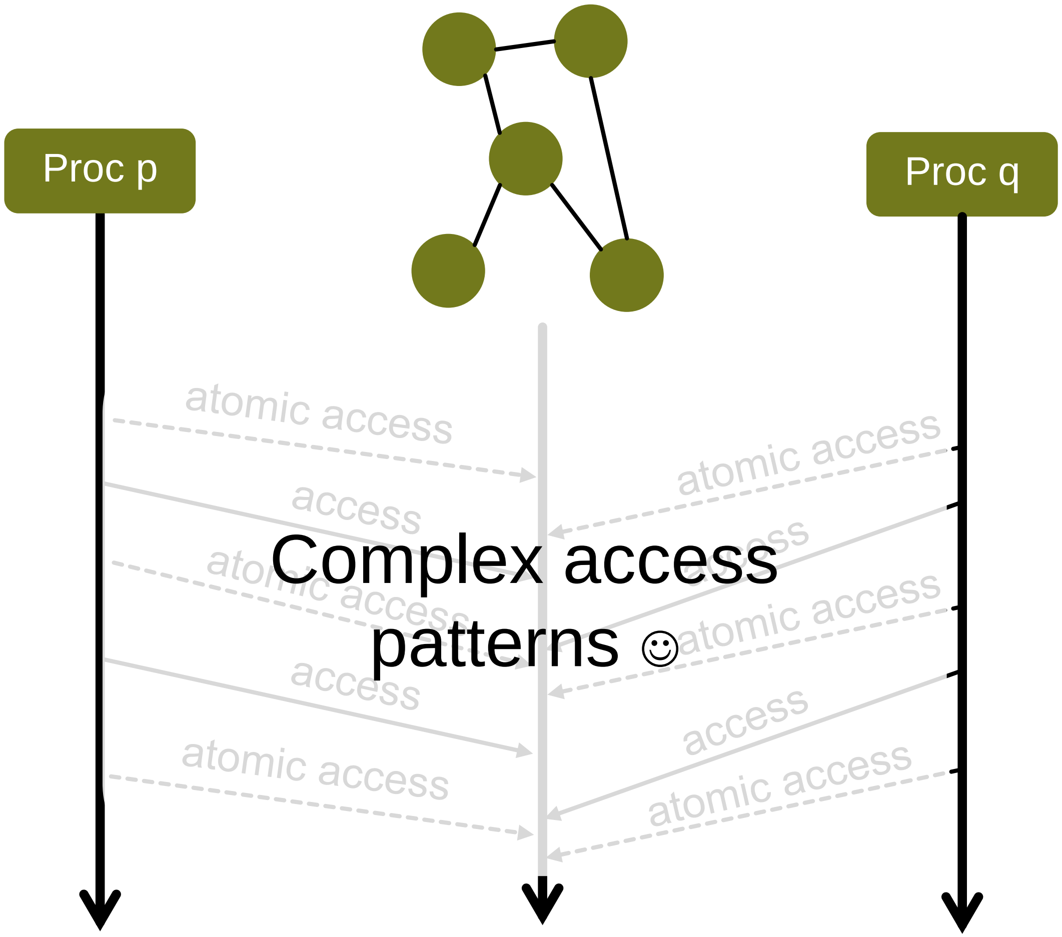


Synchronization Mechanisms
Fine Locks

- ✔ Higher performance possible
- ✘ Complex protocols
- ✘ Risk of deadlocks

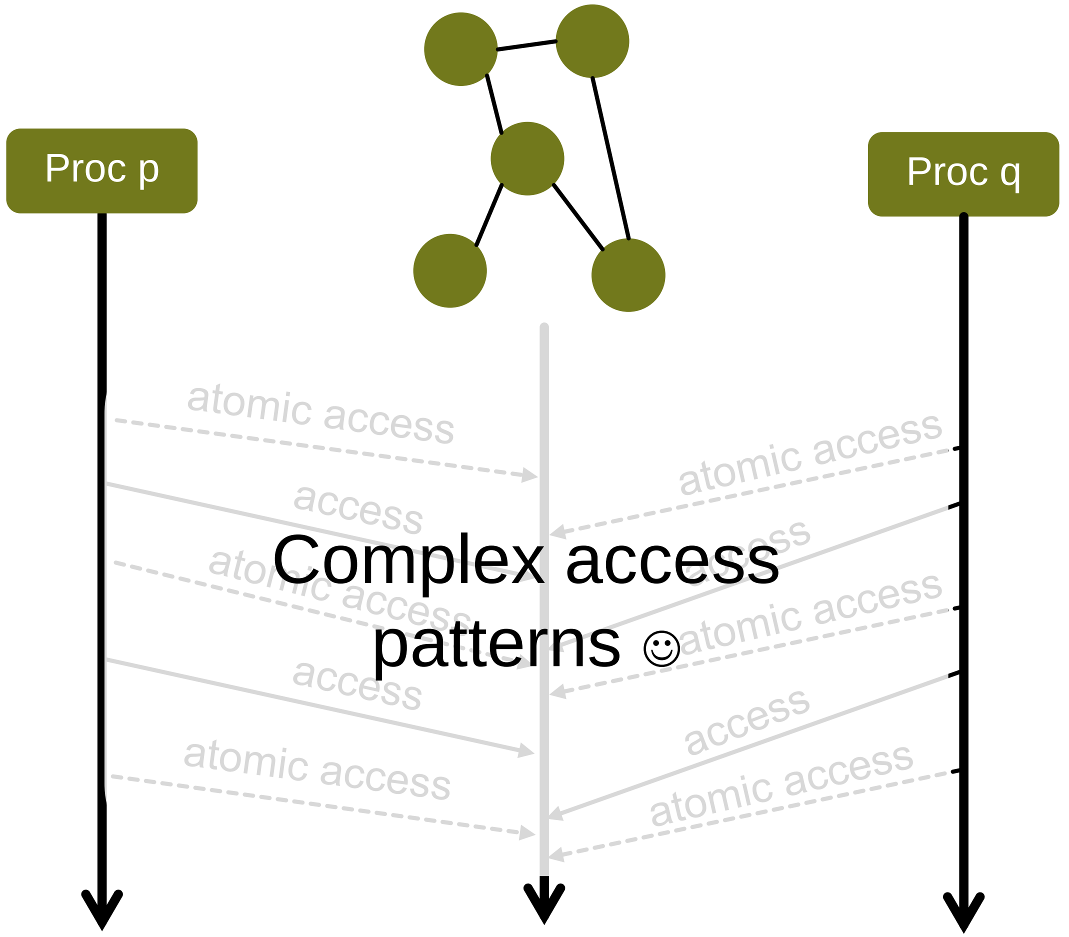


Synchronization Mechanisms
Atomic operations



Synchronization Mechanisms
Atomic operations

High performance (may be challenging to get)

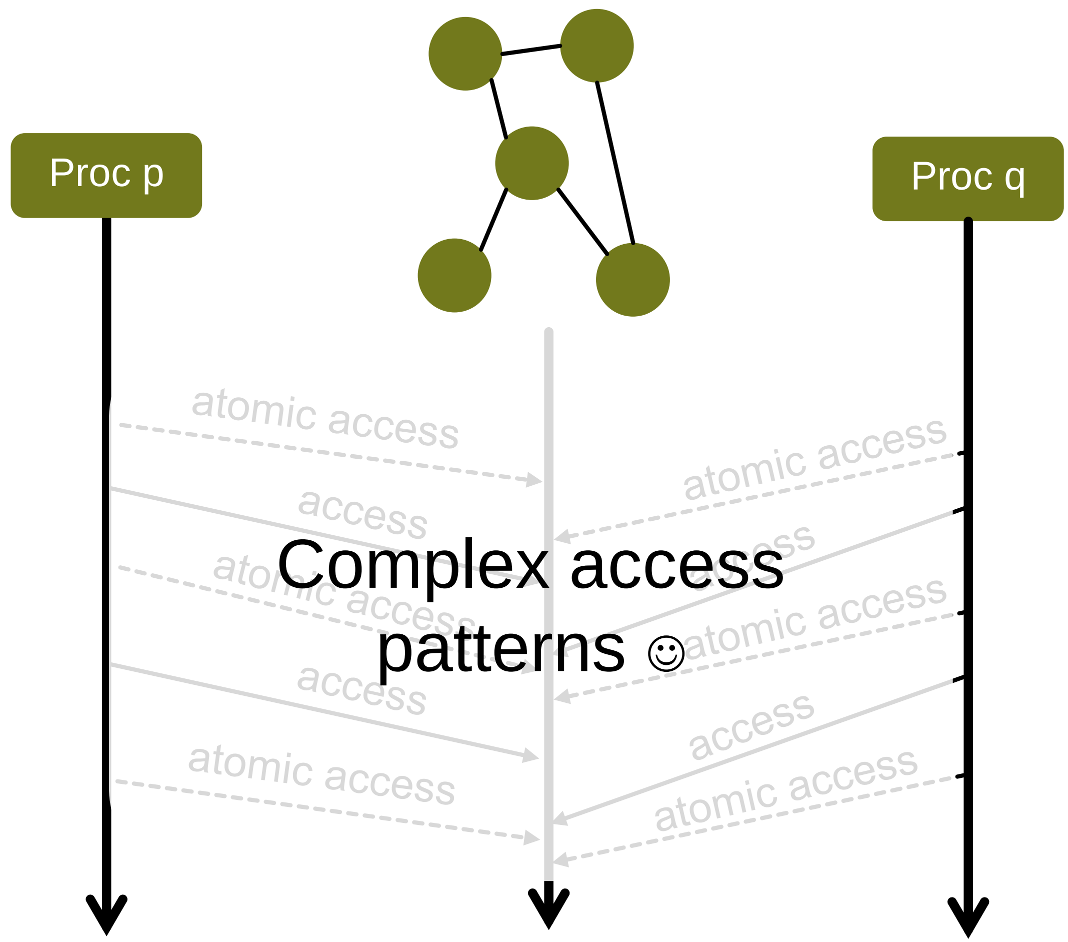


Synchronization Mechanisms

Atomic operations

High performance (may be challenging to get)

Complex protocols



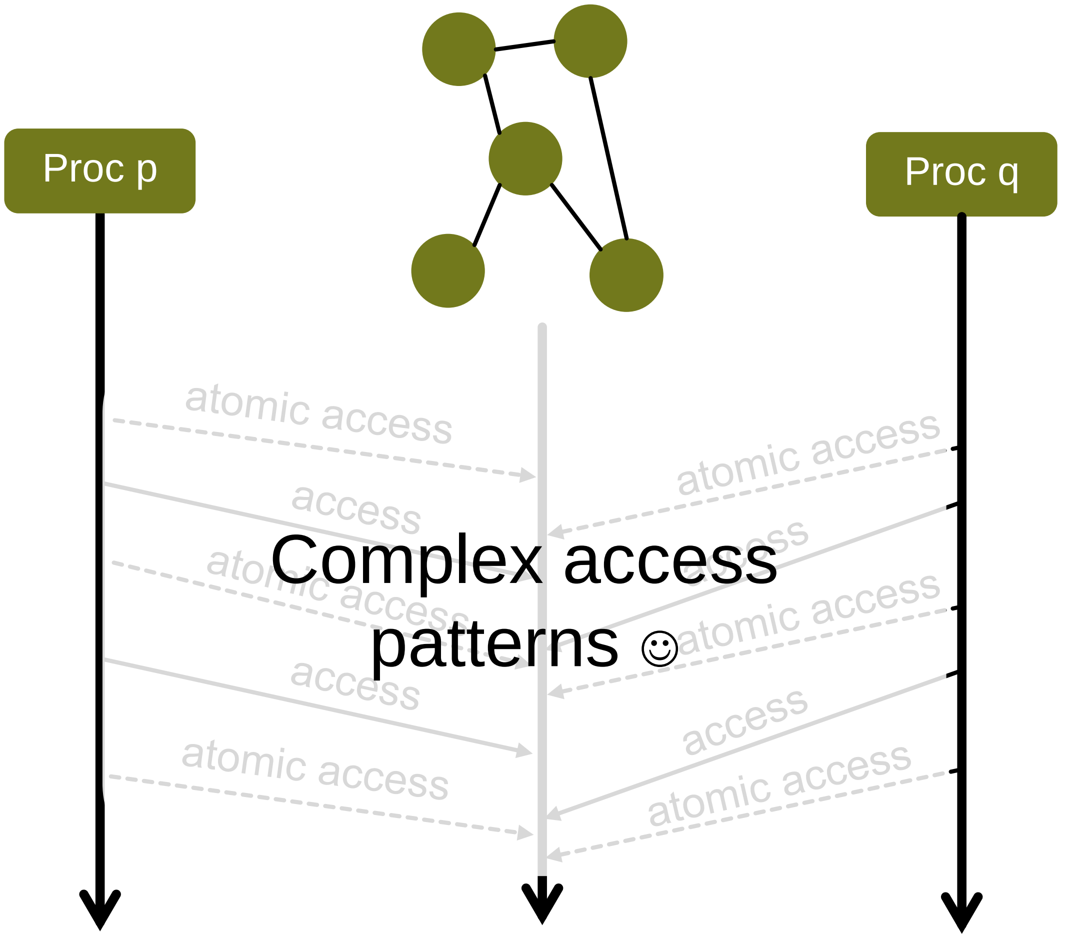
Complex access patterns ☺

Synchronization Mechanisms
Atomic operations

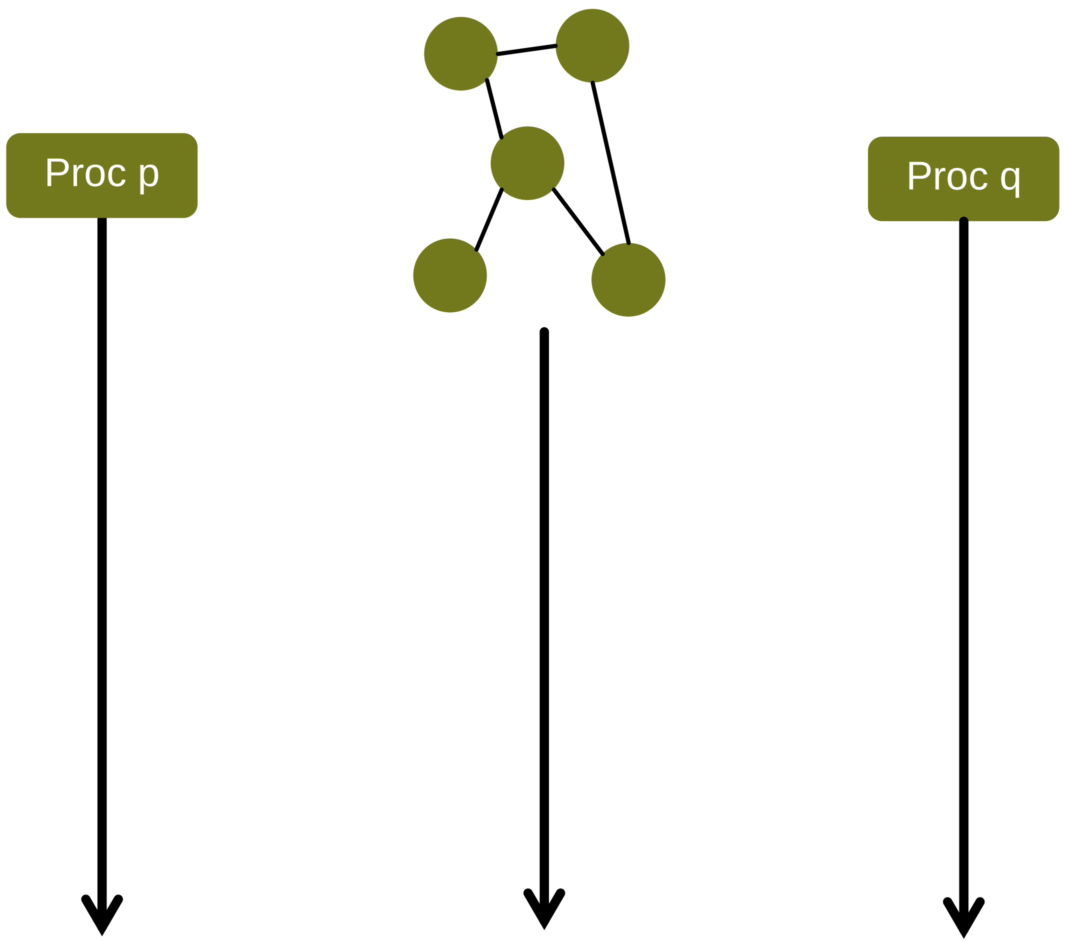
High performance (may be challenging to get)

Complex protocols

Subtle issues (ABA, …)

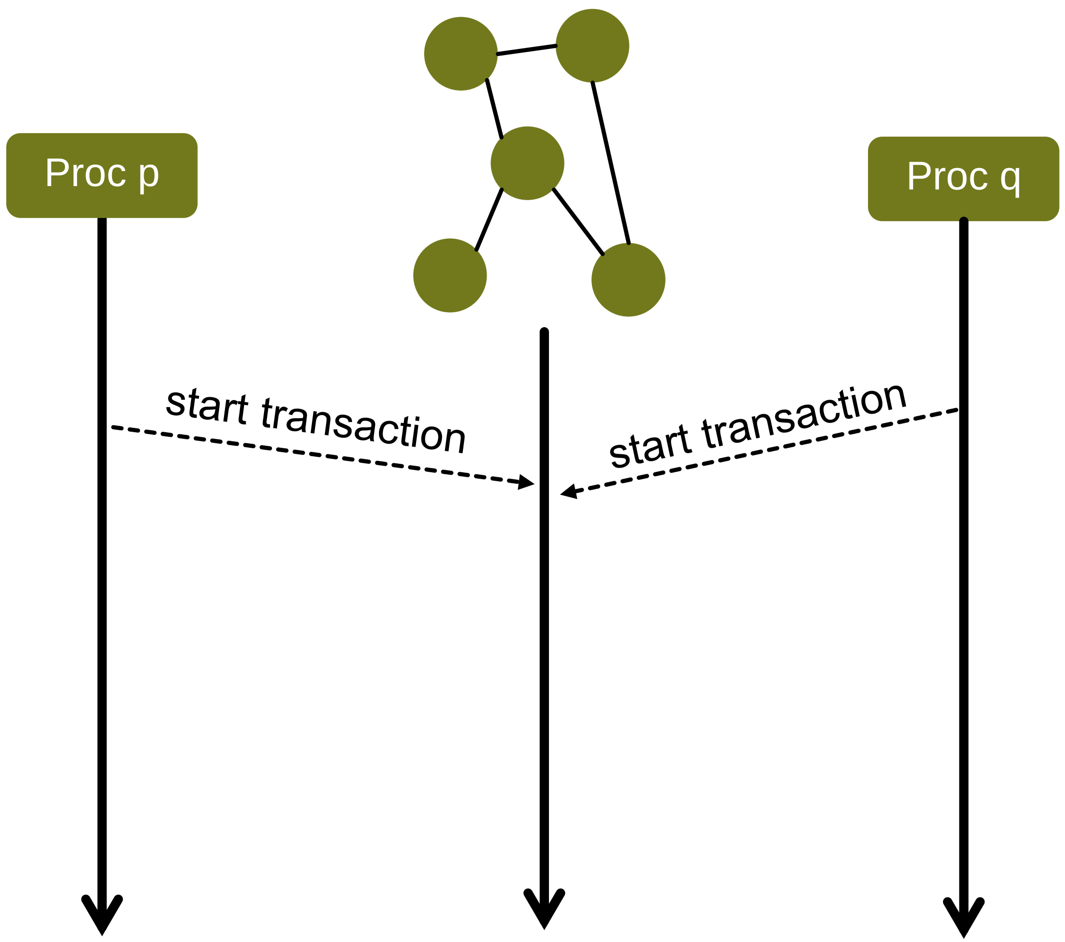


Synchronization Mechanisms
Software transactional memory (STM) [1]



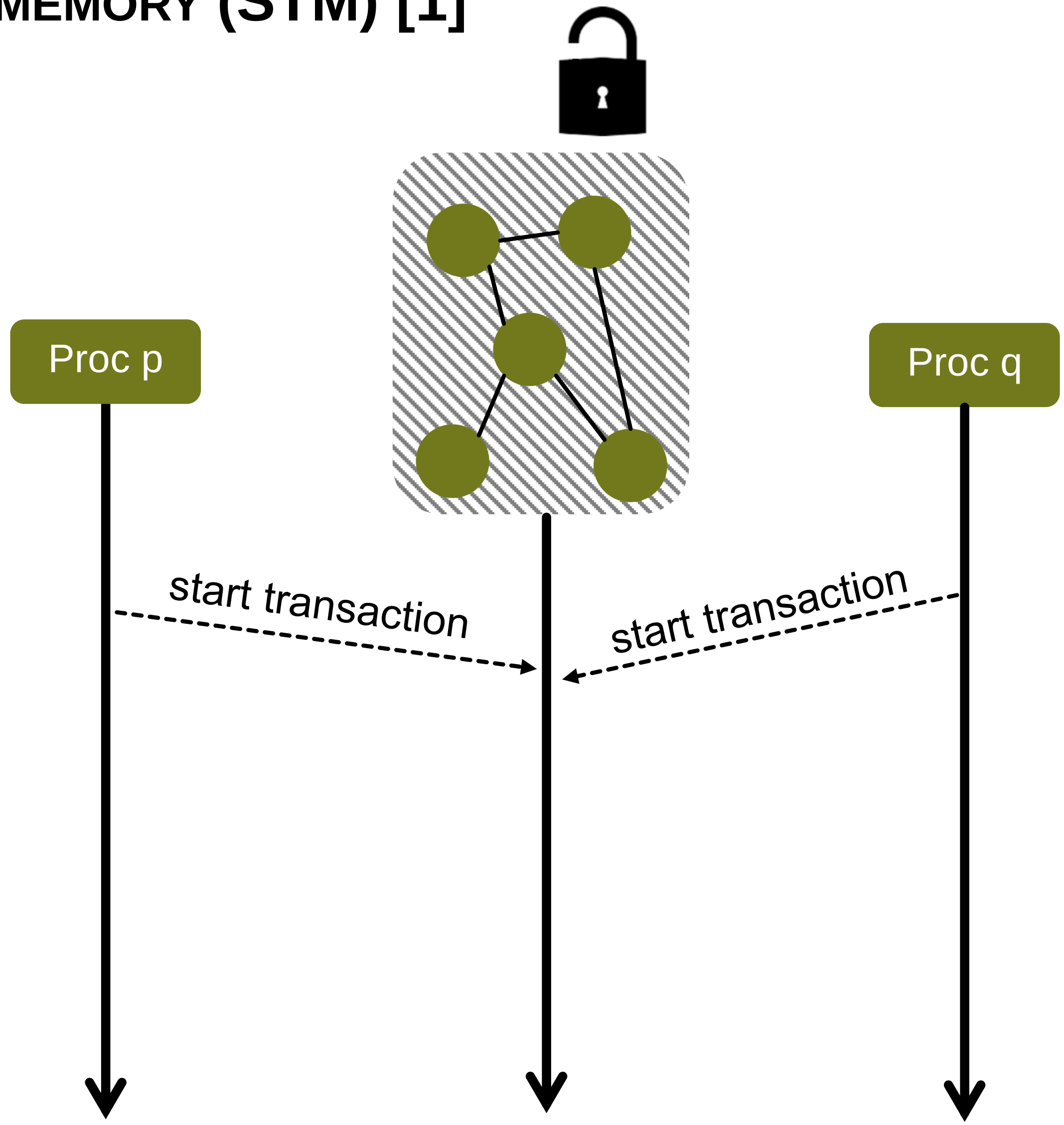
[1] N. Shavit and D. Touitou. Software transactional memory. PODC’95.

Synchronization Mechanisms
Software transactional memory (STM) [1]



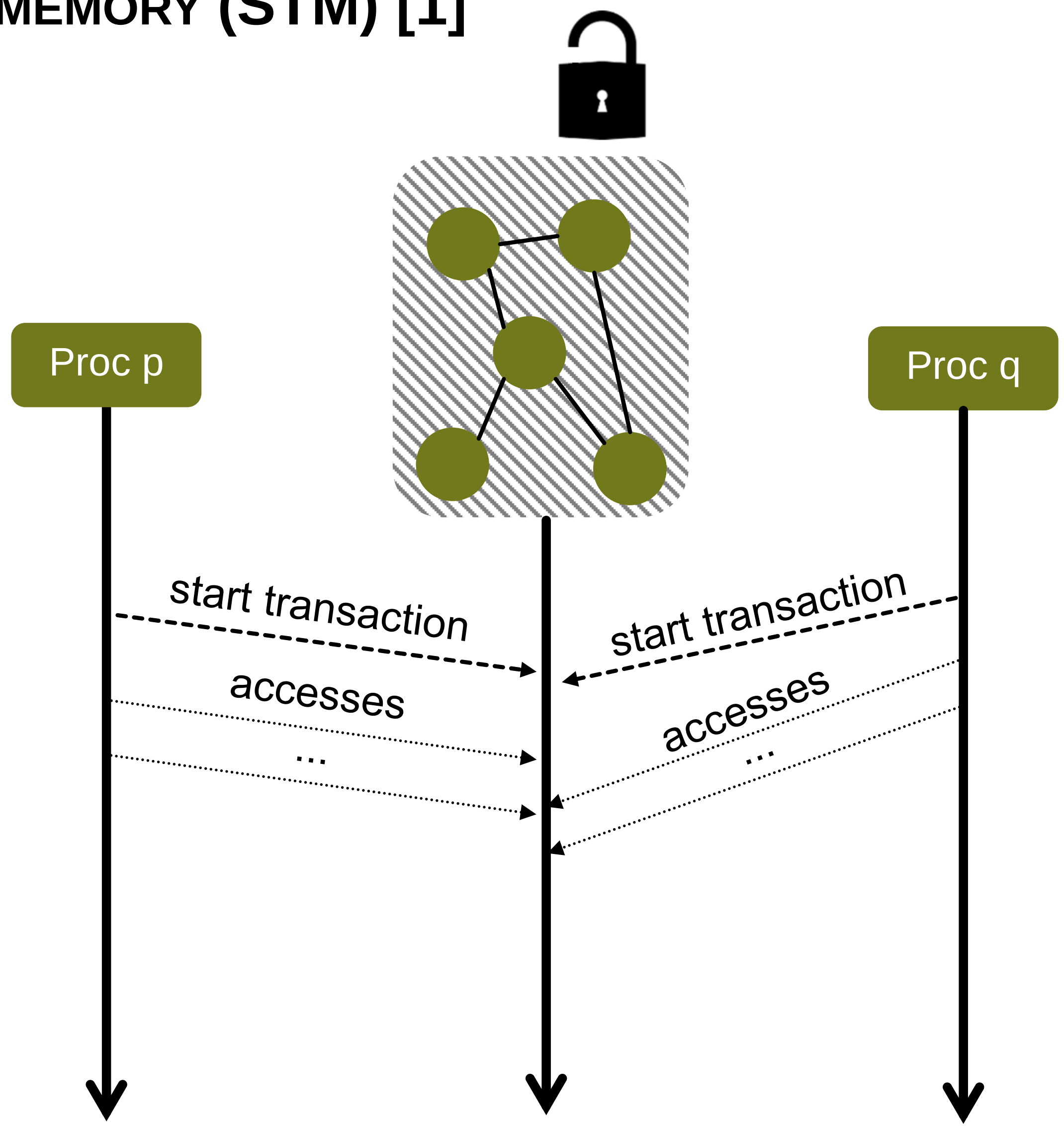
[1] N. Shavit and D. Touitou. Software transactional memory. PODC'95.

Synchronization Mechanisms
Software transactional memory (STM) [1]



[1] N. Shavit and D. Touitou. Software transactional memory. PODC'95.

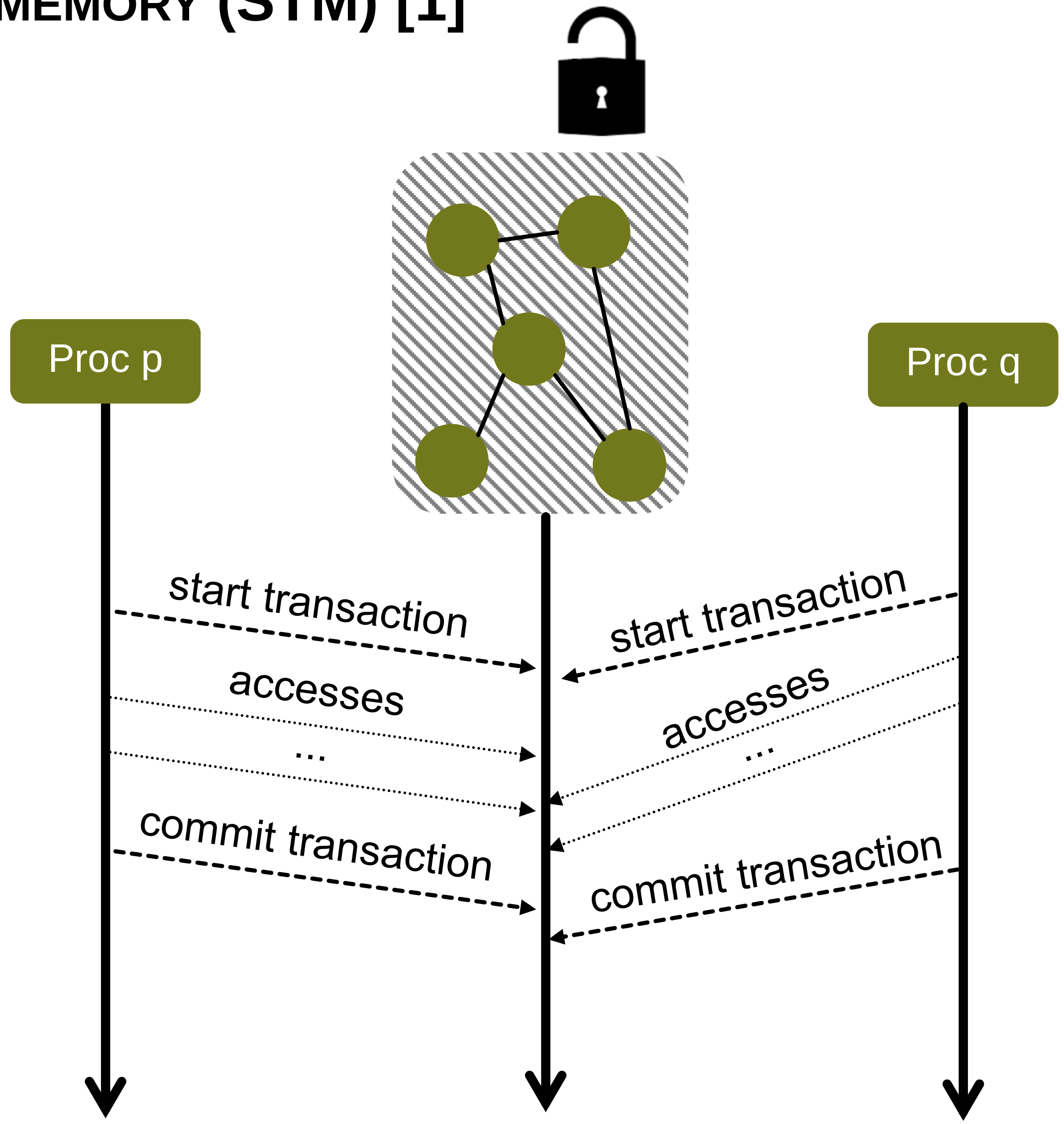
Synchronization Mechanisms
Software transactional memory (STM) [1]



[1] N. Shavit and D. Touitou. Software transactional memory. PODC'95.

Synchronization Mechanisms

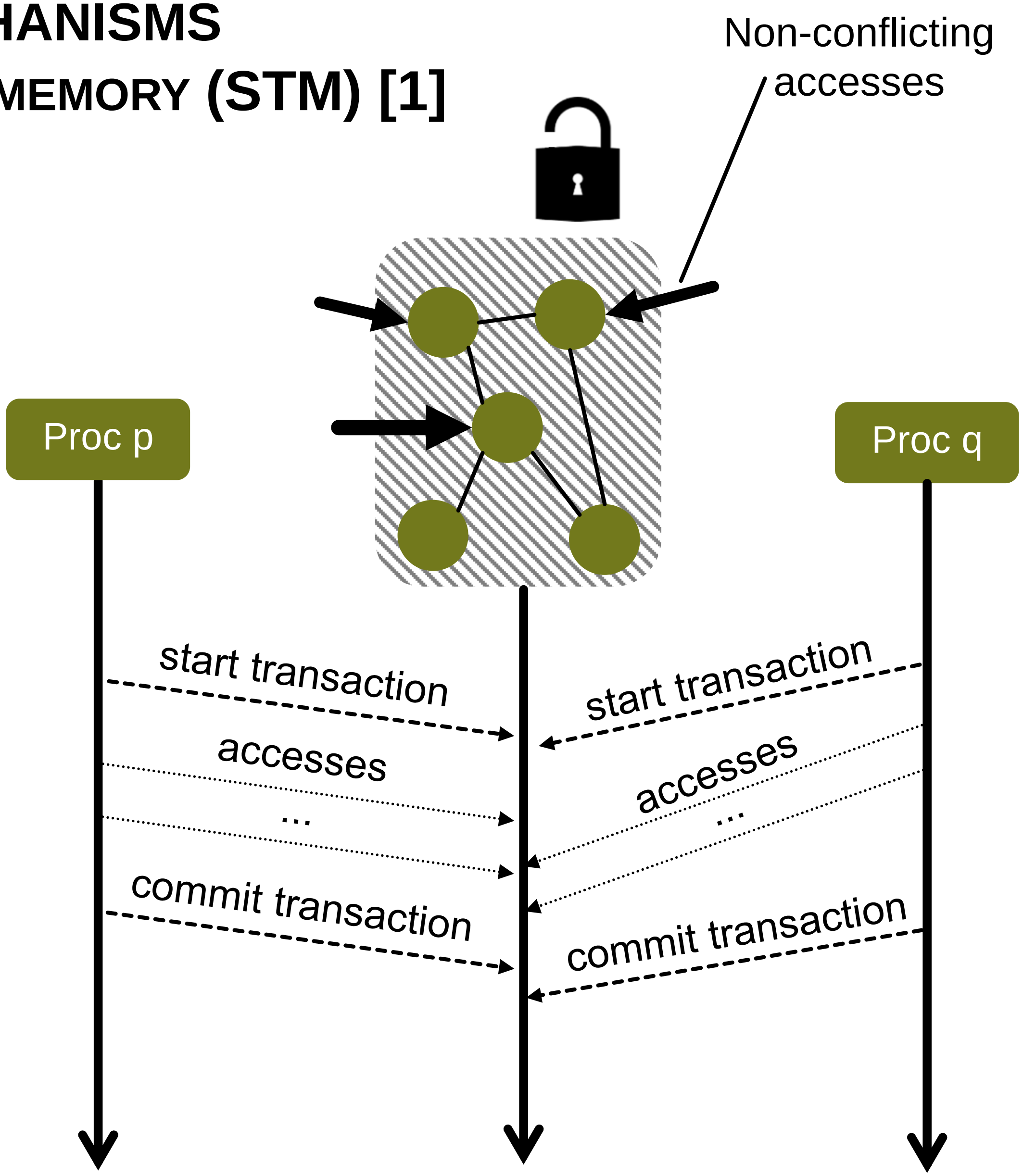
Software transactional memory (STM) [1]



[1] N. Shavit and D. Touitou. Software transactional memory. PODC'95.

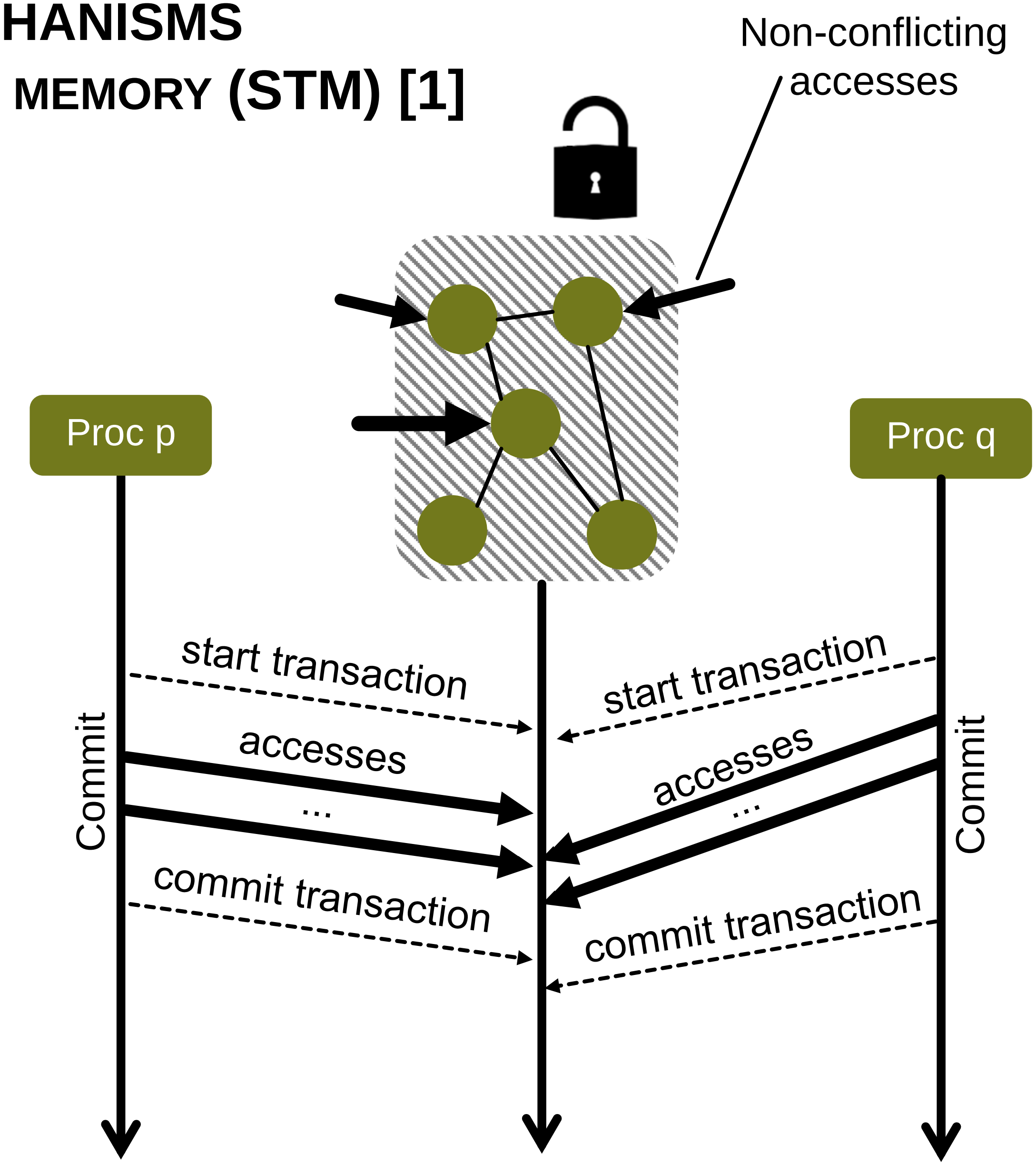
Synchronization Mechanisms

Software transactional memory (STM) [1]



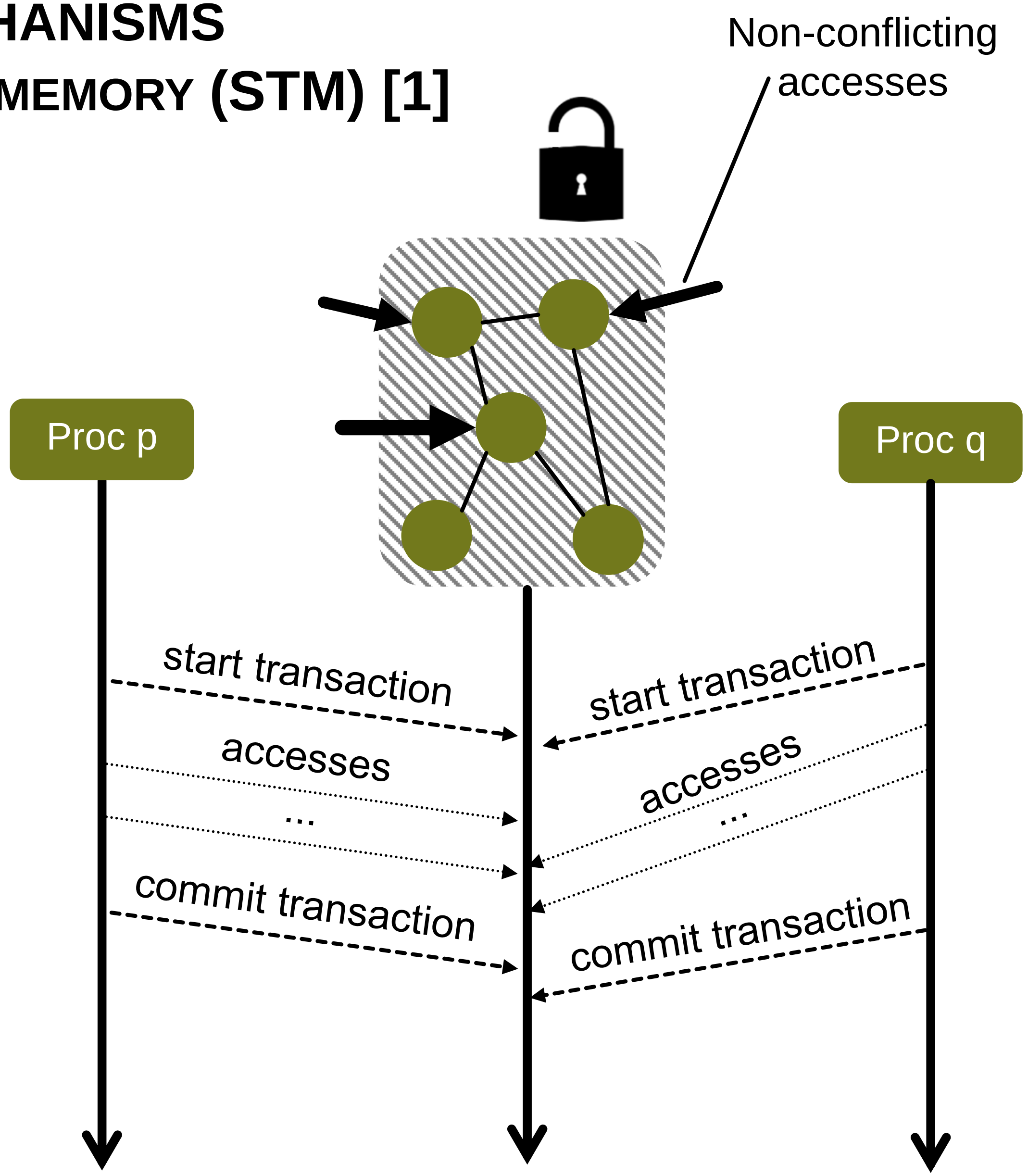
[1] N. Shavit and D. Touitou. Software transactional memory. PODC'95.

SYNCHRONIZATION MECHANISMS
SOFTWARE TRANSACTIONAL MEMORY (STM) [1]



[1] N. Shavit and D. Touitou. Software transactional memory. PODC'95.

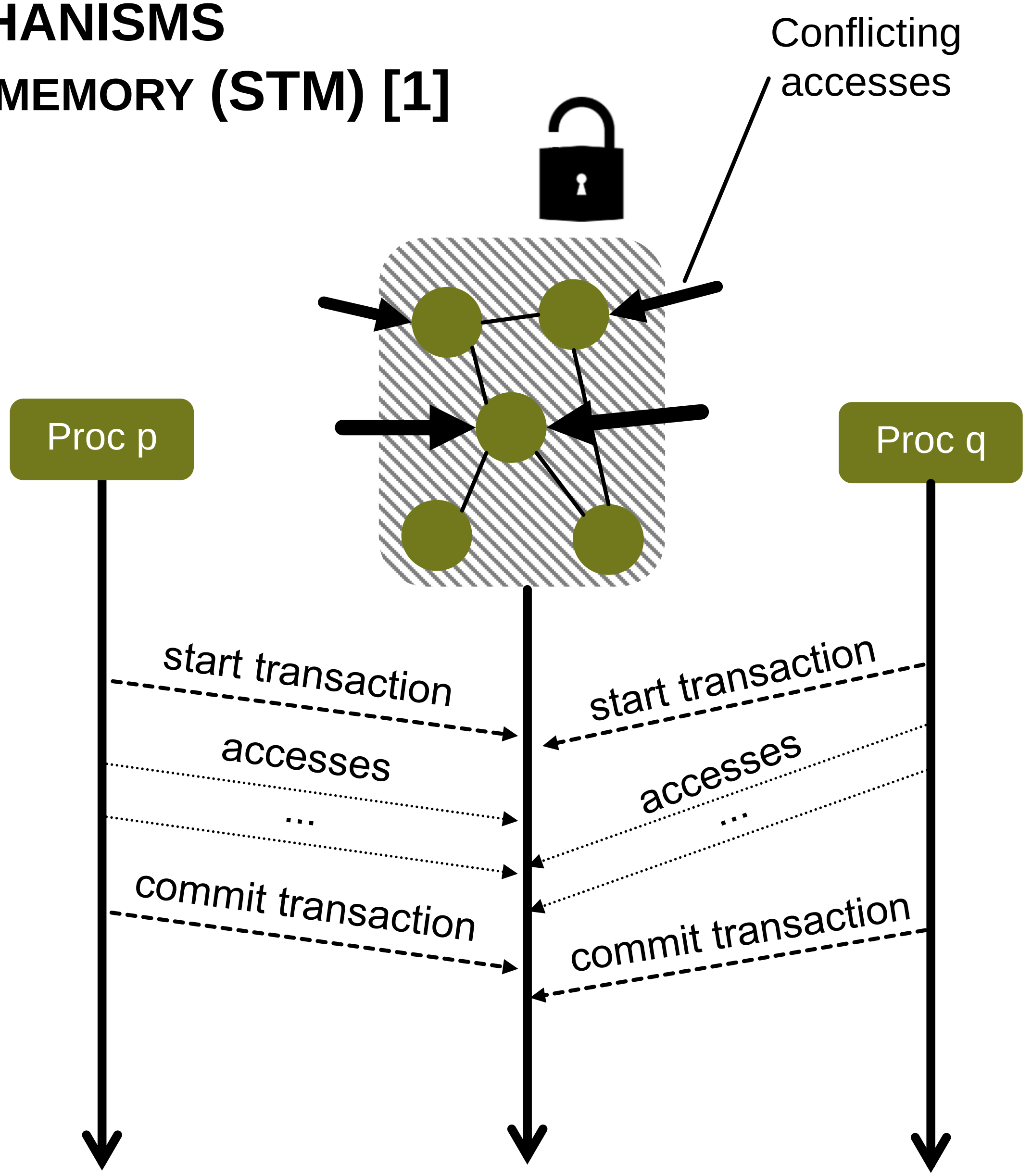
SYNCHRONIZATION MECHANISMS
SOFTWARE TRANSACTIONAL MEMORY (STM) [1]



[1] N. Shavit and D. Touitou. Software transactional memory. PODC'95.

Synchronization Mechanisms

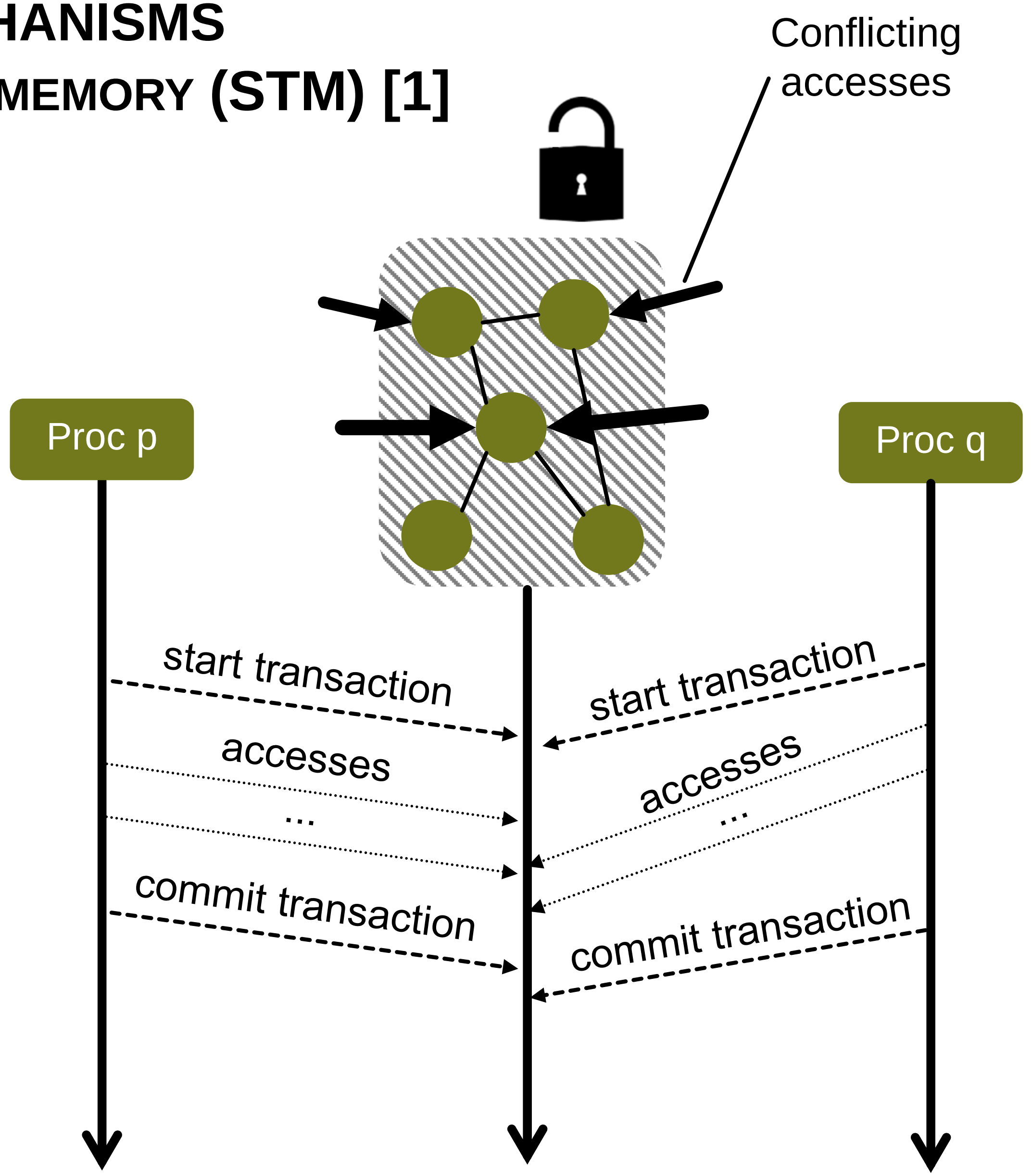
Software transactional memory (STM) [1]



[1] N. Shavit and D. Touitou. Software transactional memory. PODC'95.

Synchronization Mechanisms
Software transactional memory (STM) [1]

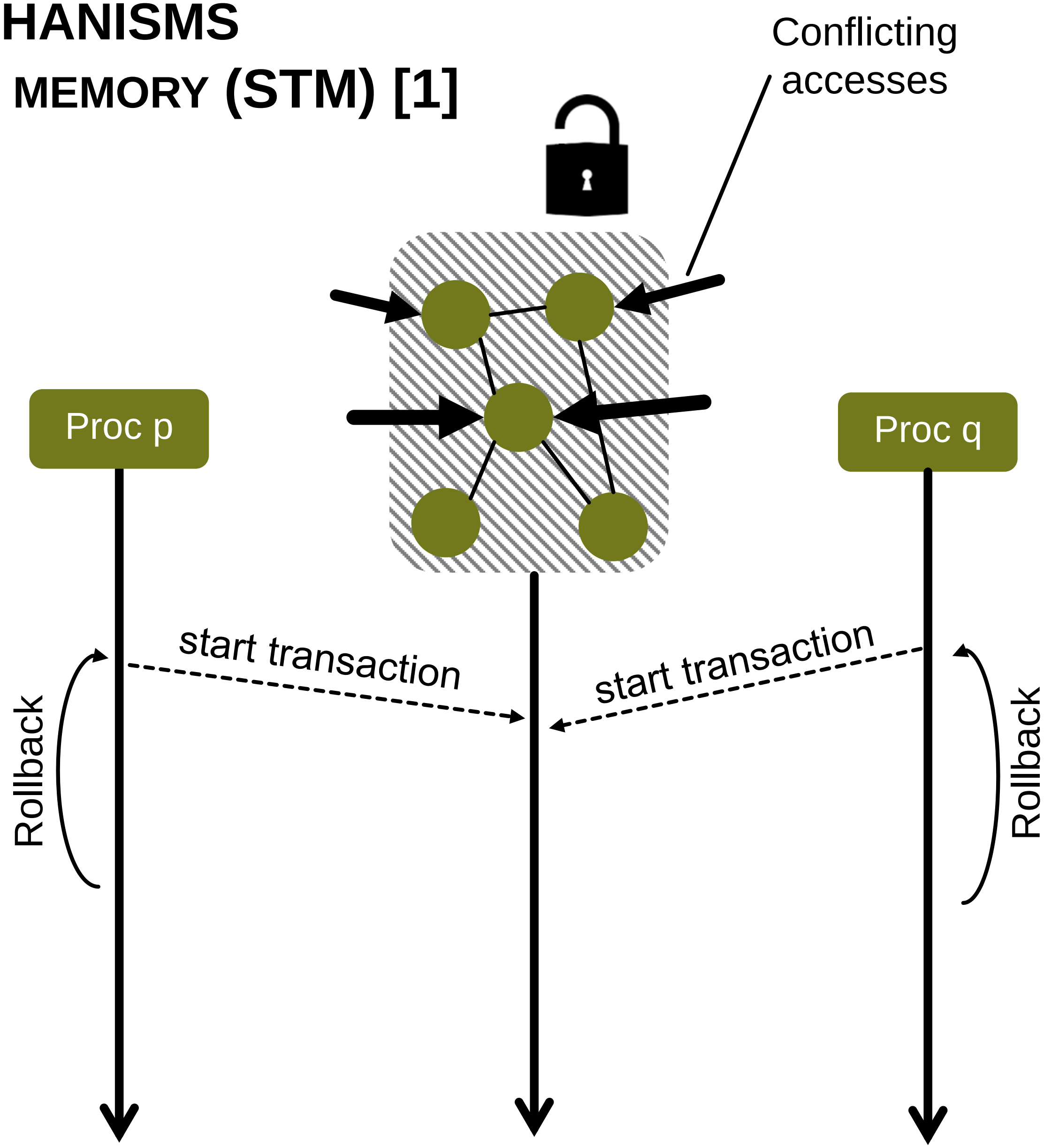
Conflicts solved with rollbacks and/or serialization.



[1] N. Shavit and D. Touitou. Software transactional memory. PODC'95.

Synchronization Mechanisms
Software transactional memory (STM) [1]

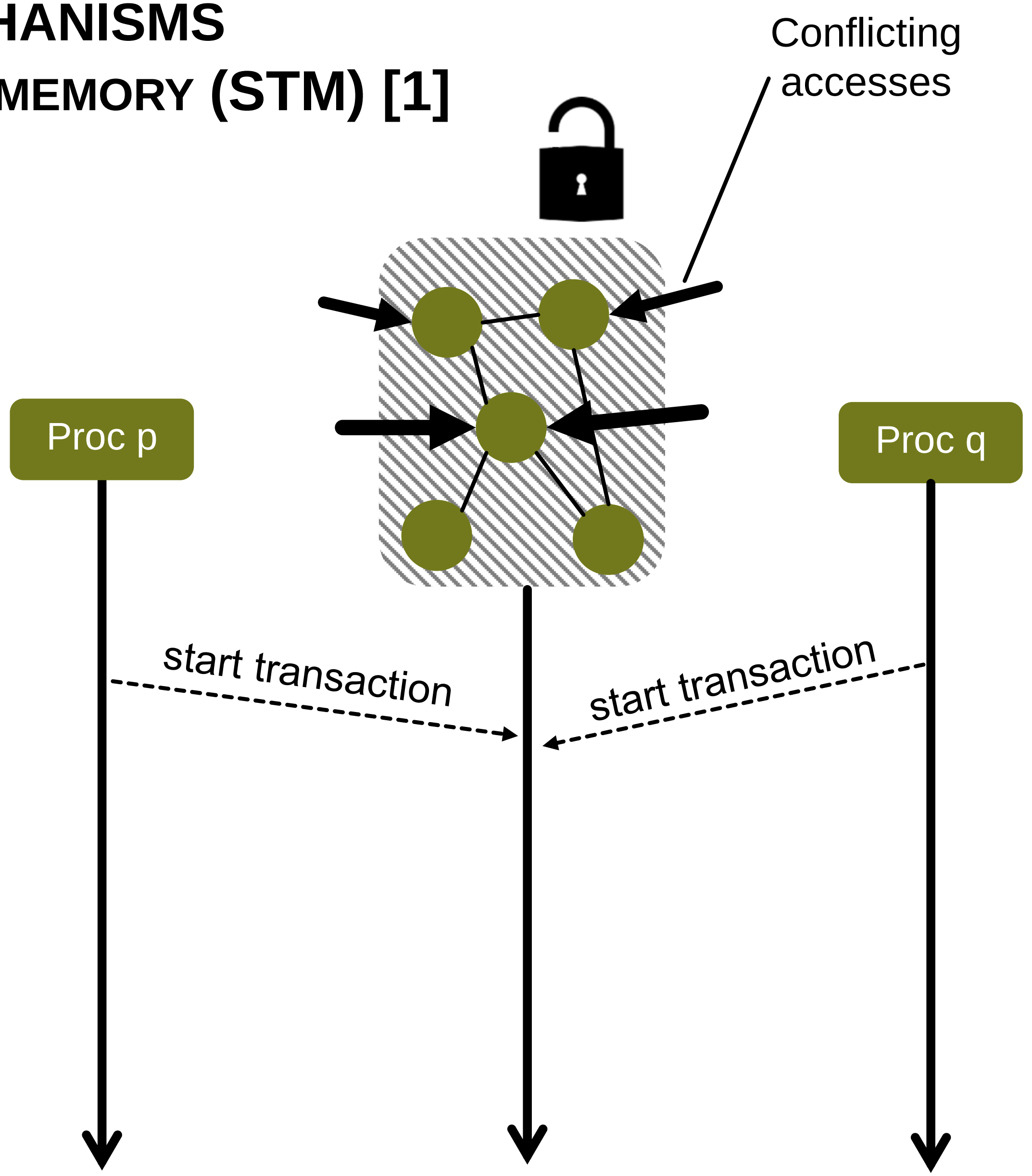
Conflicts solved with rollbacks and/or serialization.



[1] N. Shavit and D. Touitou. Software transactional memory. PODC’95.

SYNCHRONIZATION MECHANISMS
SOFTWARE TRANSACTIONAL MEMORY (STM) [1]

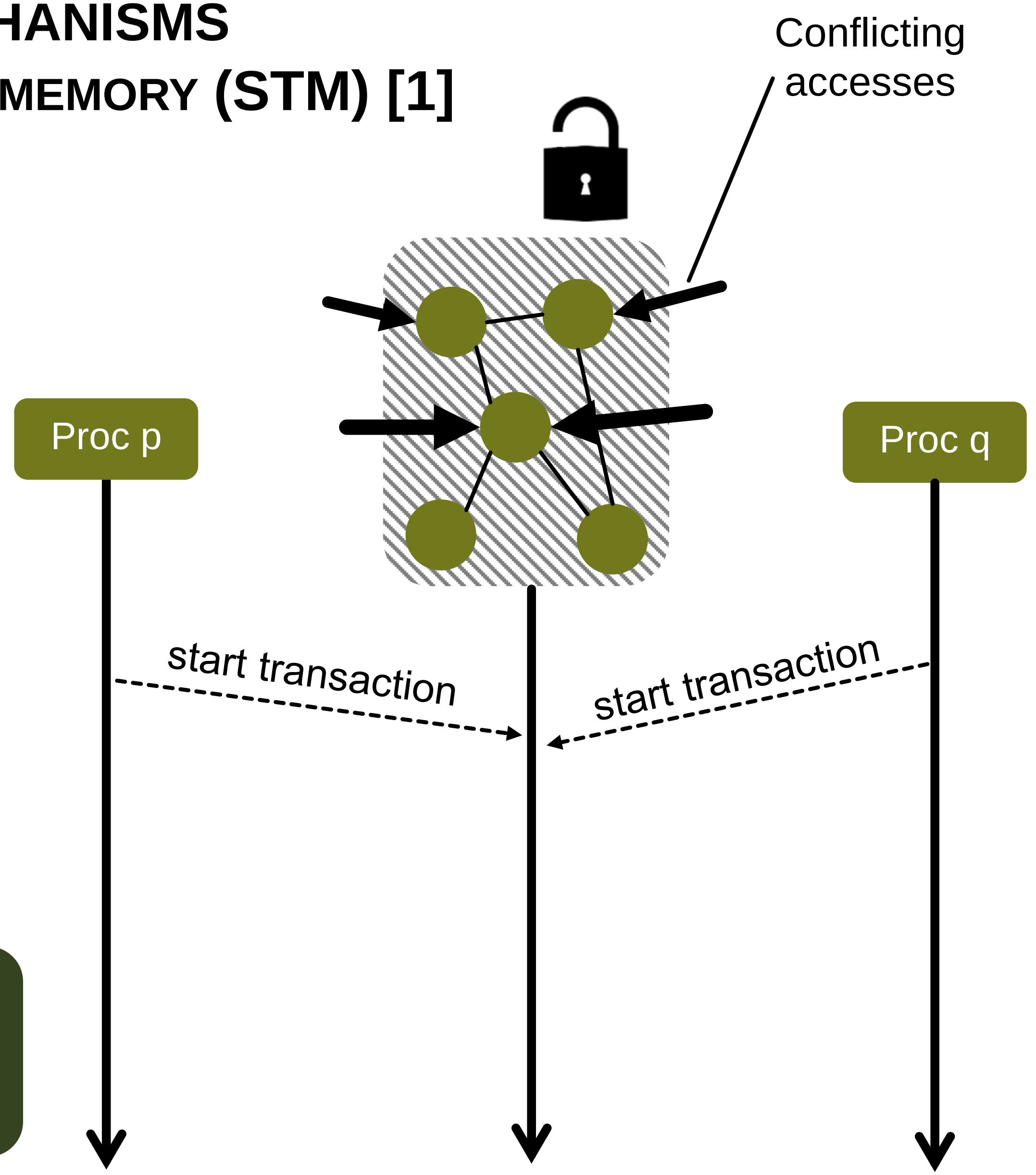
Conflicts solved with rollbacks and/or serialization.



[1] N. Shavit and D. Touitou. Software transactional memory. PODC'95.

SYNCHRONIZATION MECHANISMS
SOFTWARE TRANSACTIONAL MEMORY (STM) [1]

Conflicts solved with rollbacks and/or serialization.

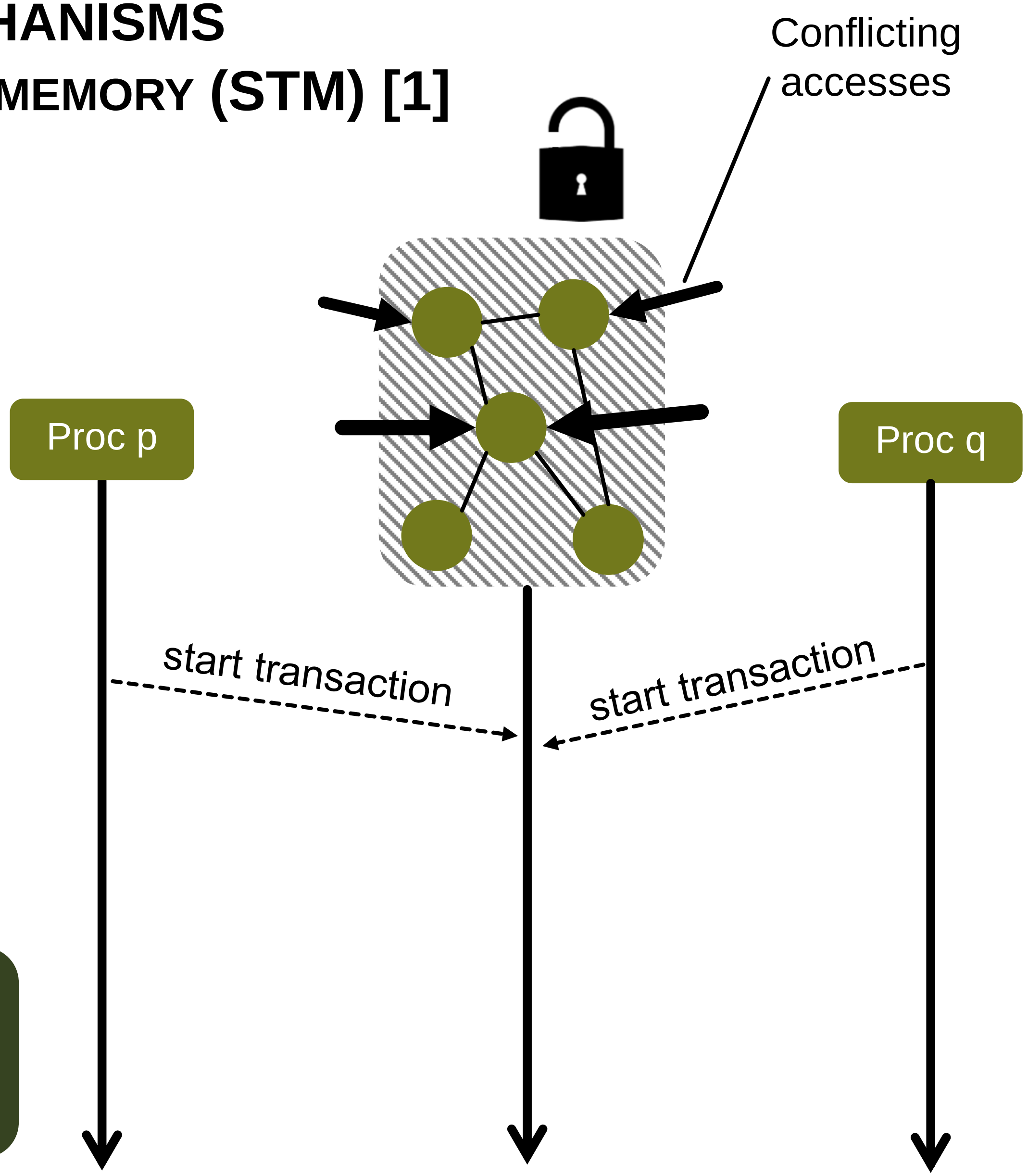


Simple protocols

[1] N. Shavit and D. Touitou. Software transactional memory. PODC'95.

SYNCHRONIZATION MECHANISMS
SOFTWARE TRANSACTIONAL MEMORY (STM) [1]

Conflicts solved with rollbacks and/or serialization.



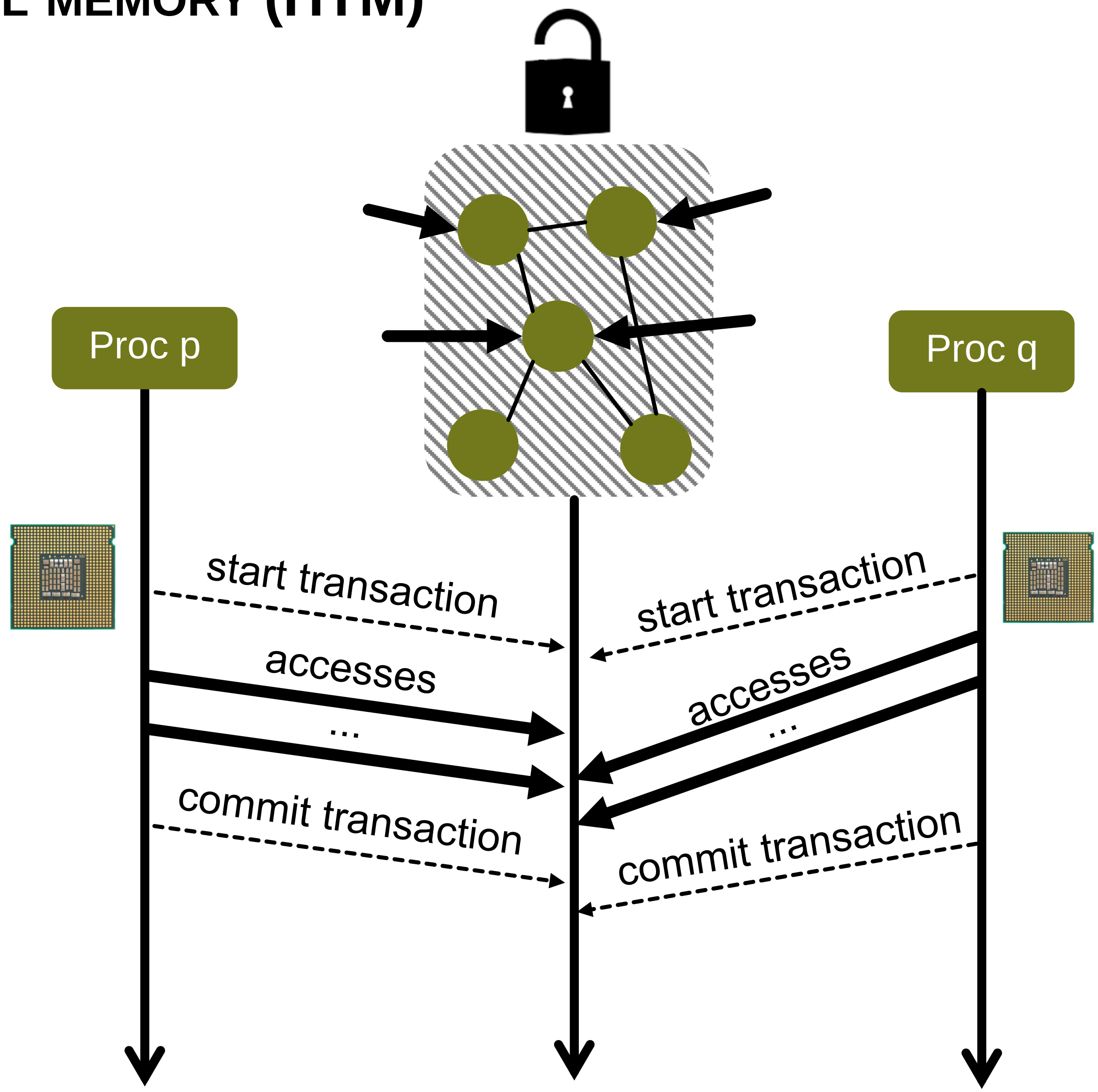
Software overheads

Simple protocols

[1] N. Shavit and D. Touitou. Software transactional memory. PODC'95.

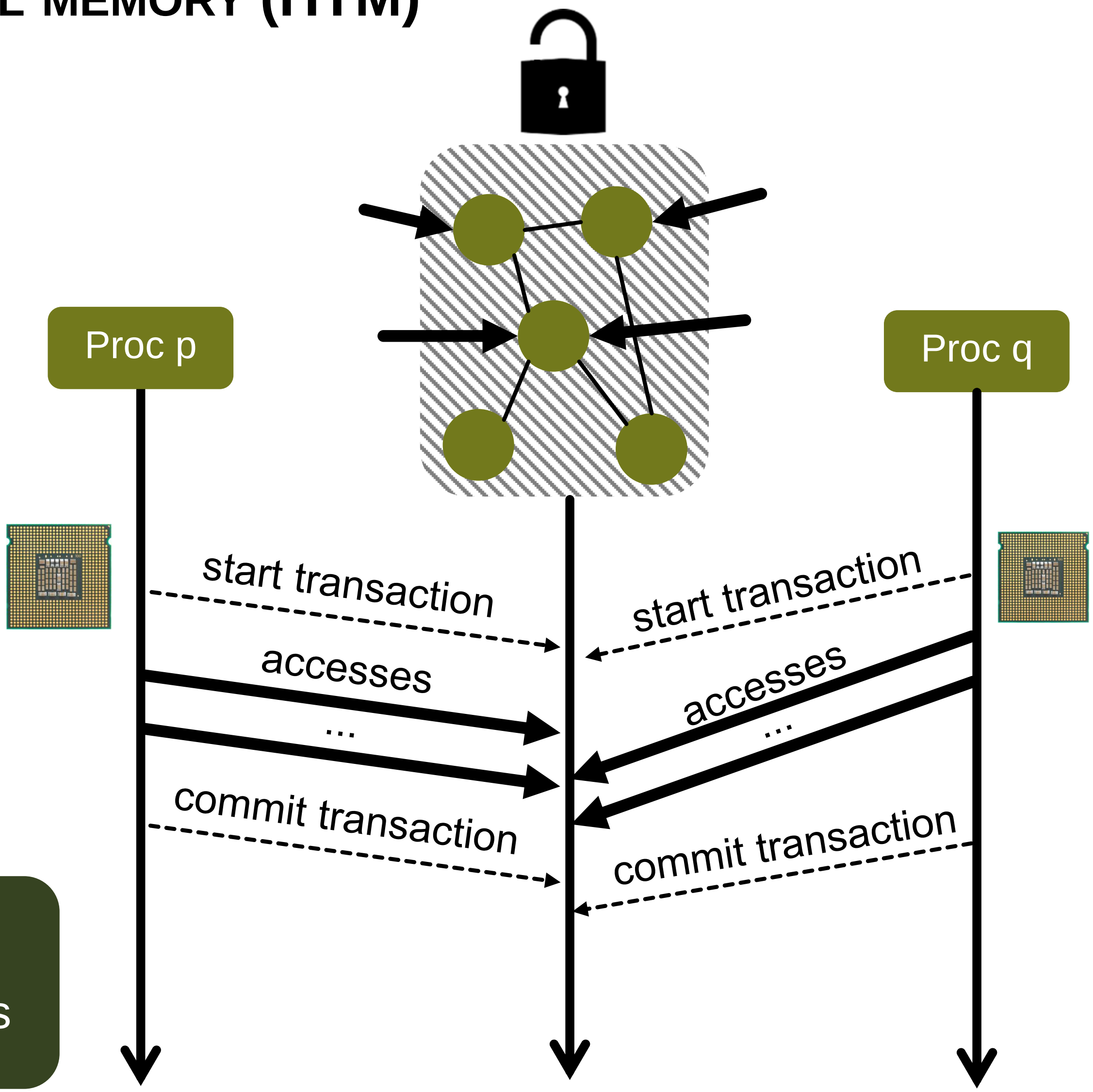
Synchronization Mechanisms
Hardware transactional memory (HTM)

Conflicts solved with rollbacks and/or HW serialization.



Synchronization Mechanisms
Hardware transactional memory (HTM)

Conflicts solved with rollbacks and/or HW serialization.



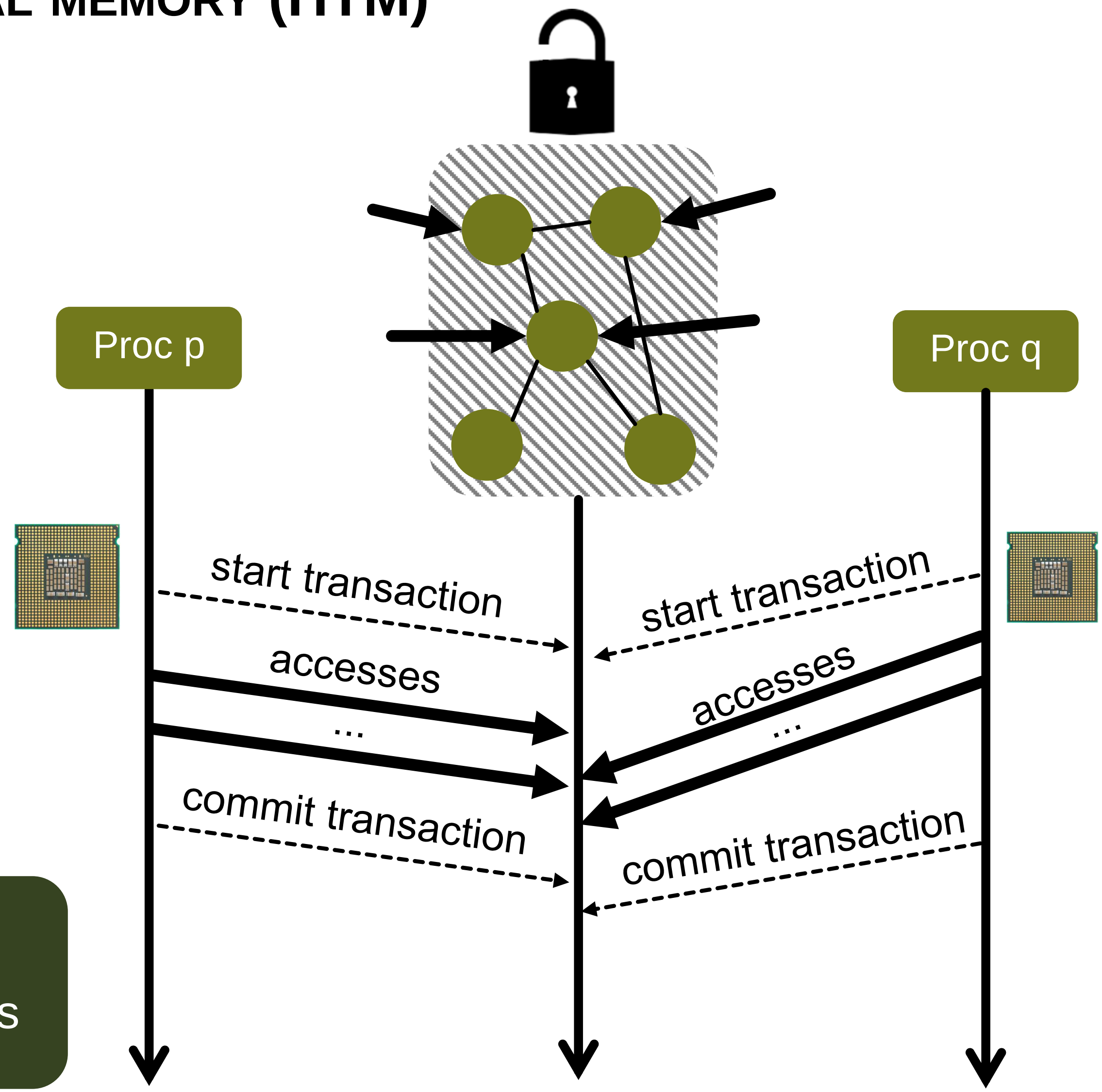
Simple protocols

Synchronization Mechanisms
Hardware transactional memory (HTM)

Conflicts solved with rollbacks and/or HW serialization.

High performance? For graphs?

Simple protocols



Hardware Transactional Memory

HARDWARE TRANSACTIONAL MEMORY

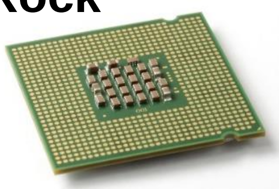
AZUL SYSTEMS®

Vega

HARDWARE TRANSACTIONAL MEMORY

Sun microsystems

Rock



AZUL SYSTEMS

Vega

Hardware Transactional Memory

AZUL SYSTEMS

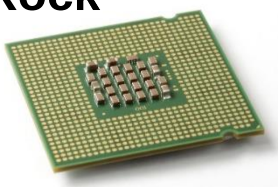
Vega

IBM

BlueGene/Q

Sun microsystems

Rock

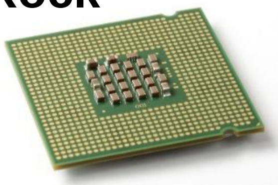


POWER8

Hardware Transactional Memory

Sun microsystems

Rock



AZUL SYSTEMS

Vega

IBM

BlueGene/Q

POWER8

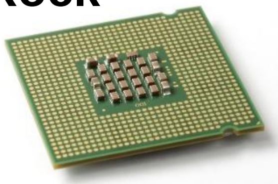
Haswell

intel

Hardware Transactional Memory

Sun microsystems

Rock



AZUL SYSTEMS

Vega

IBM

BlueGene/Q

POWER8

Haswell

intel

HARDWARE TRANSACTIONAL MEMORY

Rock

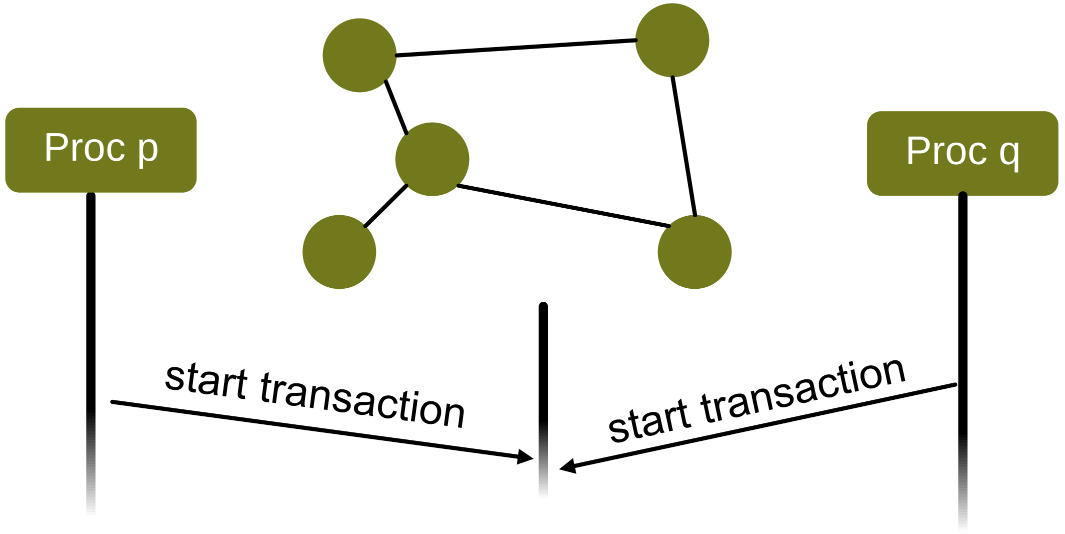
BlueGene/Q

POWER8

They offer programmability, how about performance?

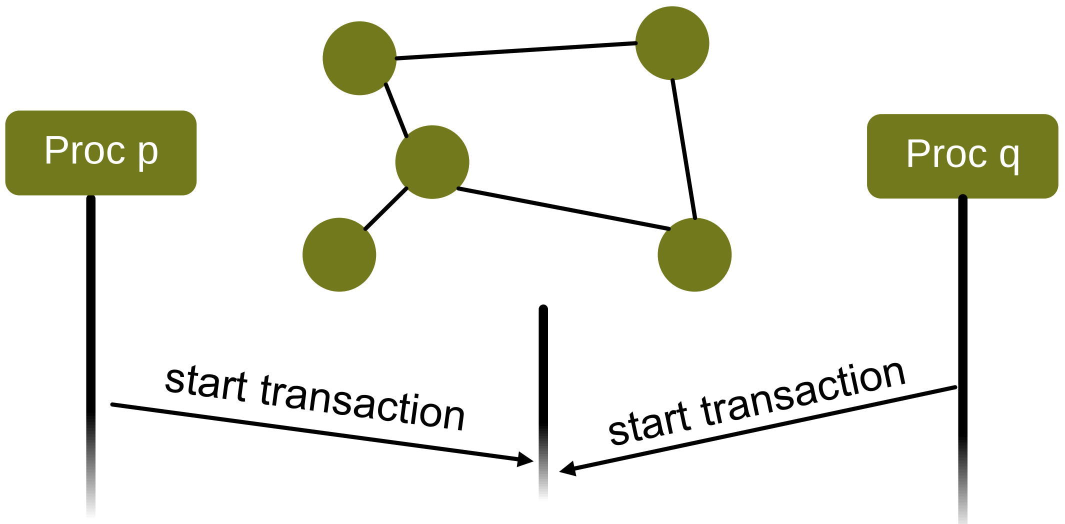
Haswell

Shared- & Distributed-memory Machines



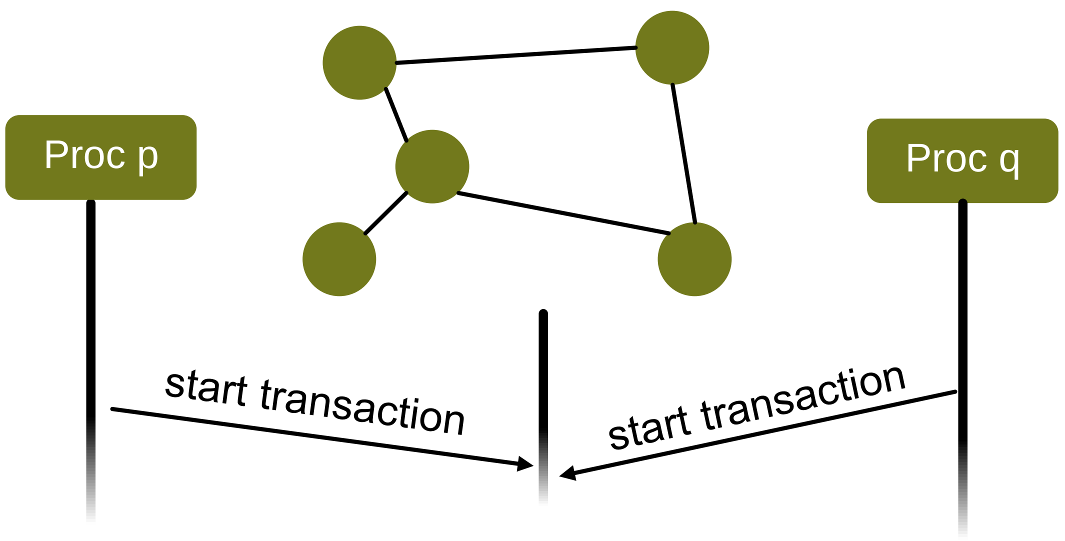
SHARED- & DISTRIBUTED-MEMORY MACHINES

- HTM works fine for single shared-memory domains



Shared- & Distributed-memory Machines

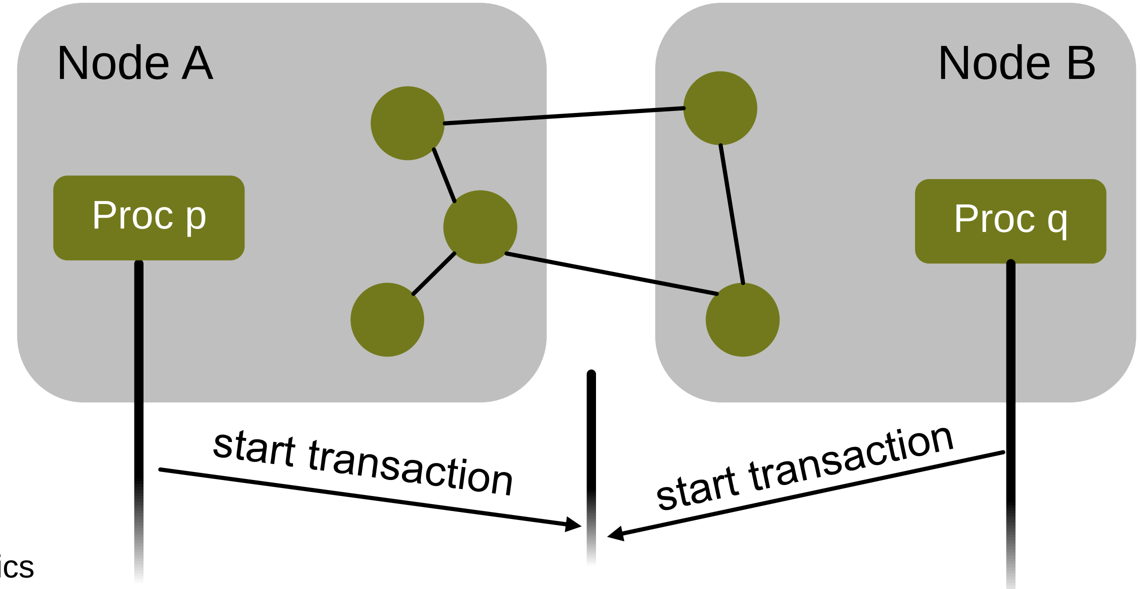
- HTM works fine for single shared-memory domains
  - Most graphs fit in such machines [1]



[1] Y. Perez et al. Ringo: Interactive Graph Analytics on Big-Memory Machines. SIGCOMM'14.

Shared- & Distributed-memory Machines

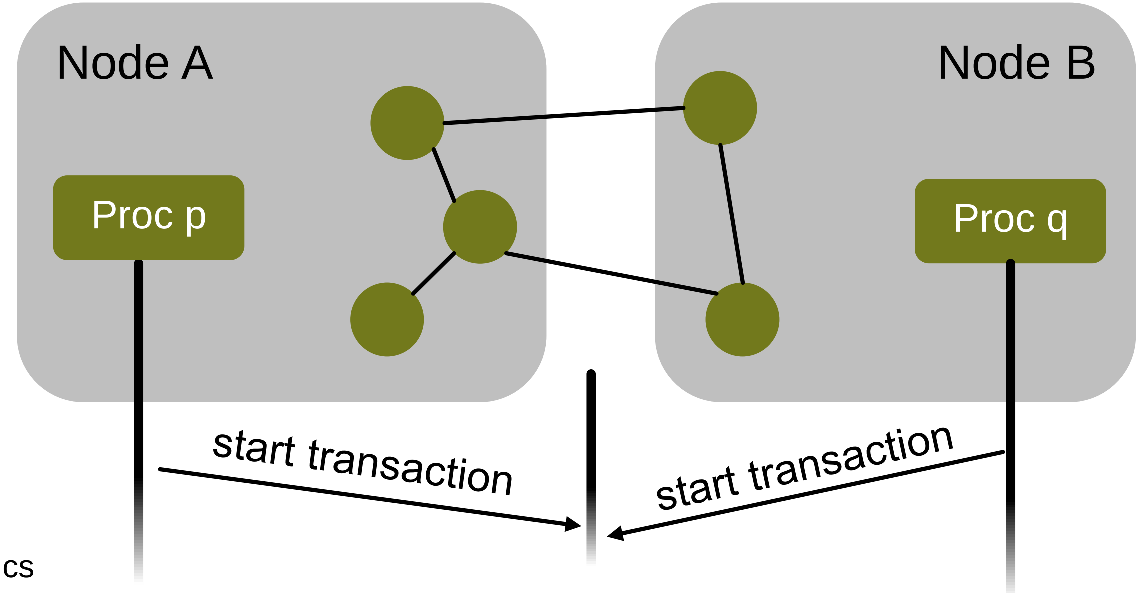
- HTM works fine for single shared-memory domains
  - Most graphs fit in such machines [1]



[1] Y. Perez et al. Ringo: Interactive Graph Analytics on Big-Memory Machines. SIGCOMM'14.

SHARED- & DISTRIBUTED-MEMORY MACHINES

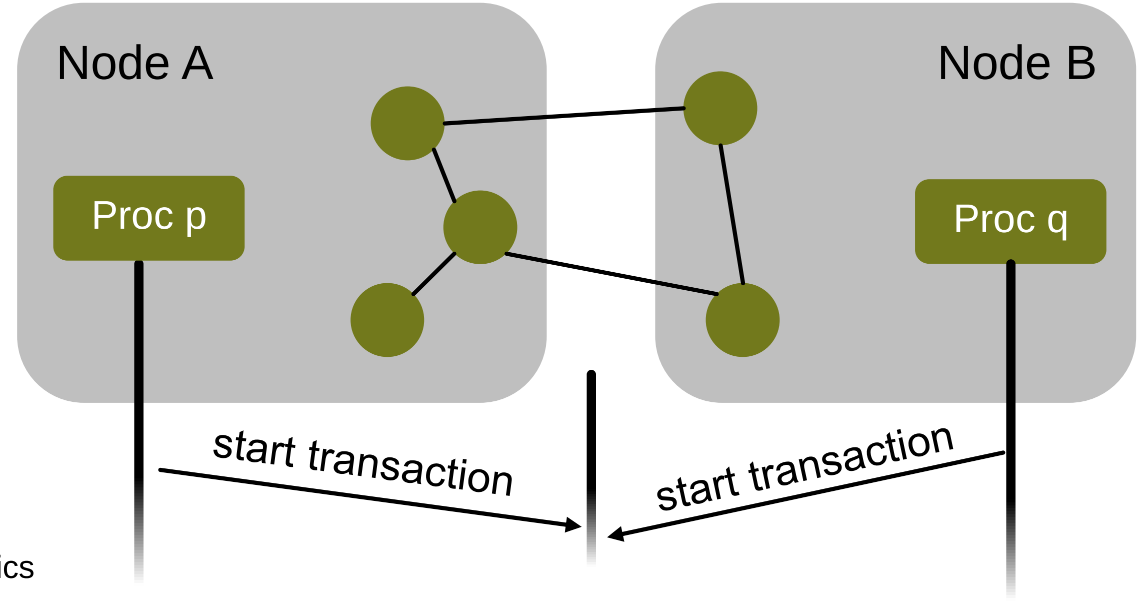
- HTM works fine for single shared-memory domains
  - Most graphs fit in such machines [1]
- However, some do not:



[1] Y. Perez et al. Ringo: Interactive Graph Analytics on Big-Memory Machines. SIGCOMM'14.

Shared- & Distributed-memory Machines

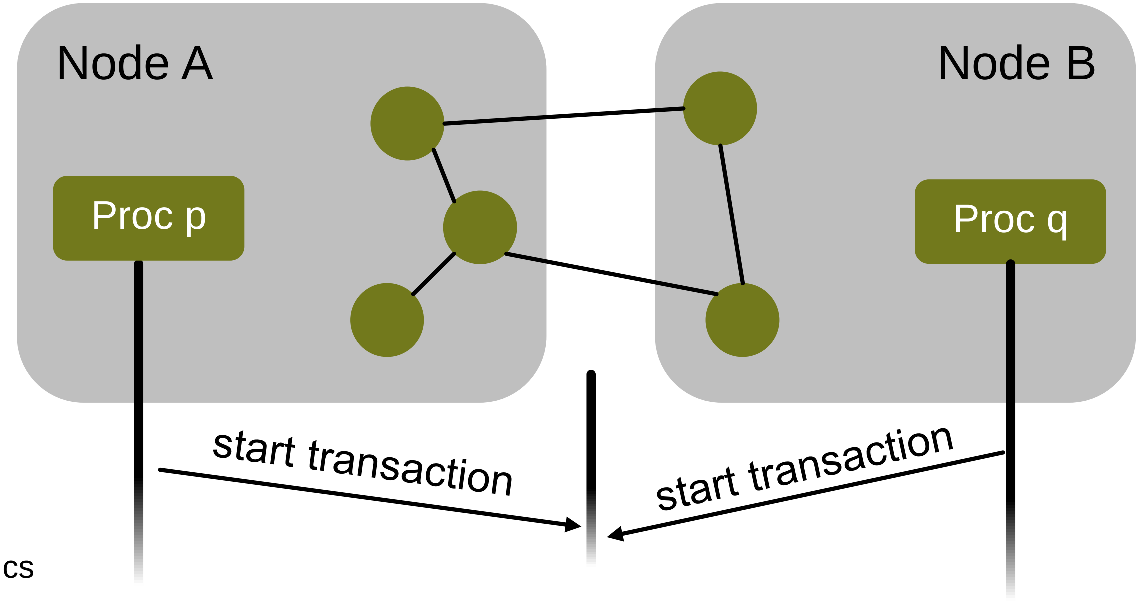
- HTM works fine for single shared-memory domains
  - Most graphs fit in such machines [1]
- However, some do not:
  - Very large instances



[1] Y. Perez et al. Ringo: Interactive Graph Analytics on Big-Memory Machines. SIGCOMM'14.

SHARED- & DISTRIBUTED-MEMORY MACHINES

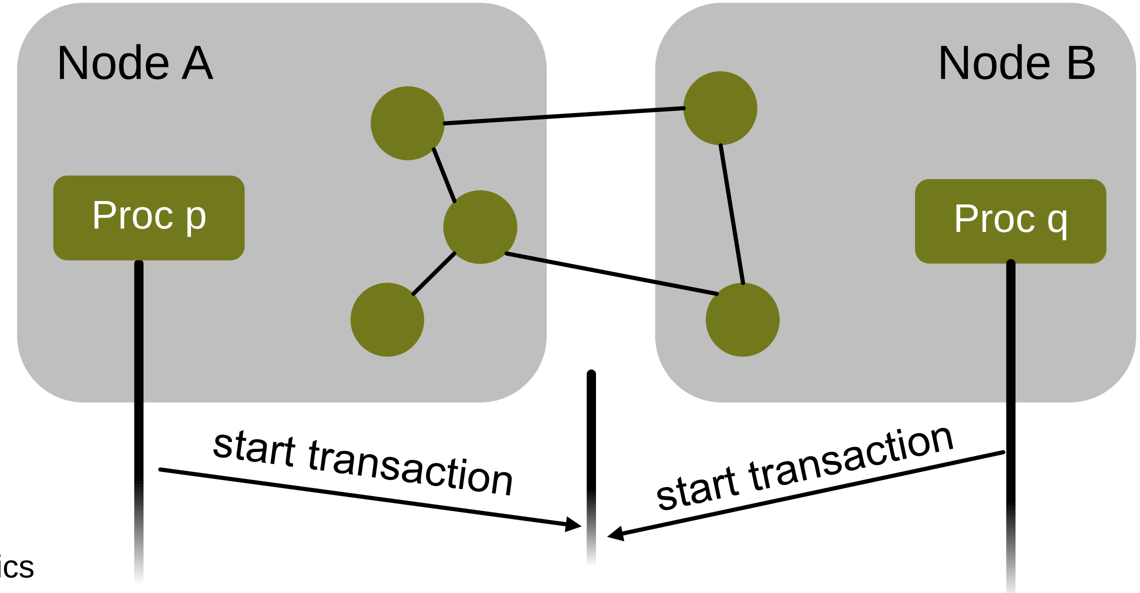
- HTM works fine for single shared-memory domains
  - Most graphs fit in such machines [1]
- However, some do not:
  - Very large instances
  - Rich vertex/edge data



[1] Y. Perez et al. Ringo: Interactive Graph Analytics on Big-Memory Machines. SIGCOMM'14.

SHARED- & DISTRIBUTED-MEMORY MACHINES

- HTM works fine for single shared-memory domains
  - Most graphs fit in such machines [1]
- However, some do not:
  - Very large instances
  - Rich vertex/edge data
- Fat nodes with lots of RAM still expensive ($35K for a machine with 1TB of RAM [1])

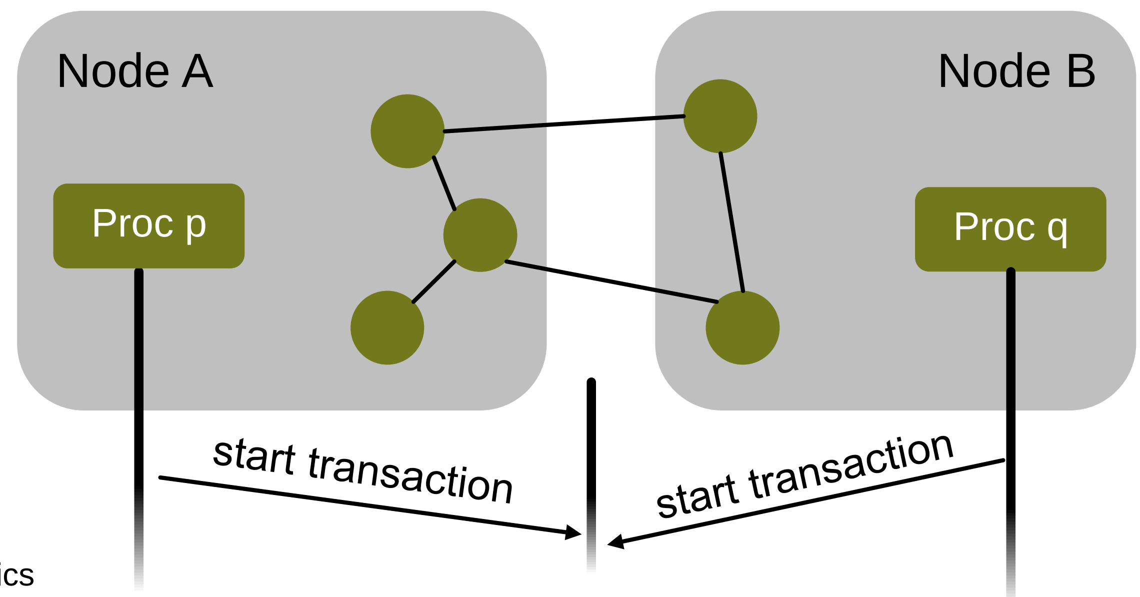


[1] Y. Perez et al. Ringo: Interactive Graph Analytics on Big-Memory Machines. SIGCOMM'14.

Shared- & Distributed-memory Machines

- HTM works fine for single shared-memory domains
  - Most graphs fit in such machines [1]
- However, some do not:
  - Very large instances
  - Rich vertex/edge data
- Fat nodes with lots of RAM still expensive ($35K for a machine with 1TB of RAM [1])

How to apply HTM in such a setting?



[1] Y. Perez et al. Ringo: Interactive Graph Analytics on Big-Memory Machines. SIGCOMM'14.

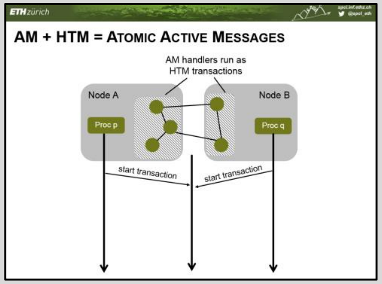
Overview of our Research

OVERVIEW OF OUR RESEARCH

HTM for graphs in SM & DM environments

Overview of our Research

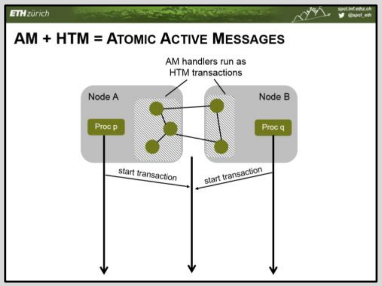
HTM for graphs in SM & DM environments



HTM + Active Messages
= Atomic Active Messages

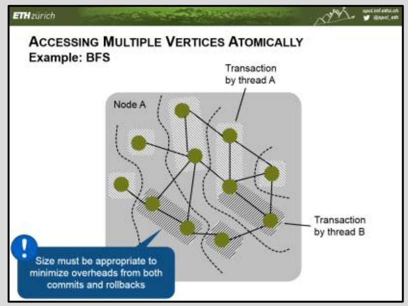
Overview of our Research

HTM for graphs in SM & DM environments



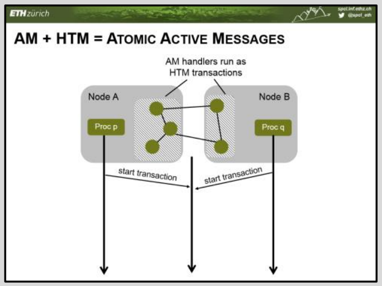
HTM + Active Messages
= Atomic Active Messages

Coarsening & coalescing



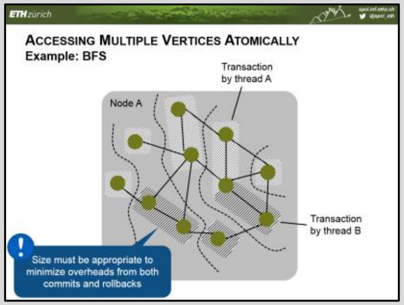
Overview of our Research

HTM for graphs in SM & DM environments



HTM + Active Messages
= Atomic Active Messages

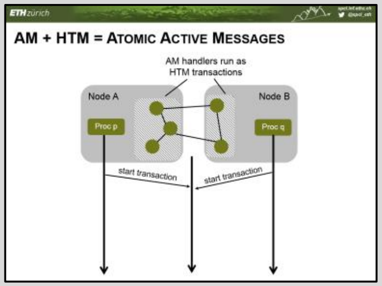
Coarsening & coalescing



Performance Modeling & Analysis

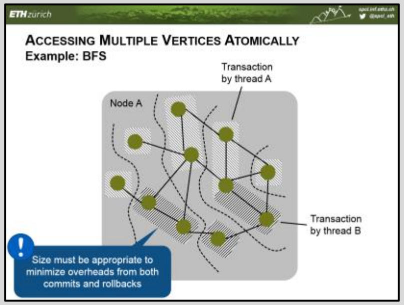
Overview of our Research

HTM for graphs in SM & DM environments



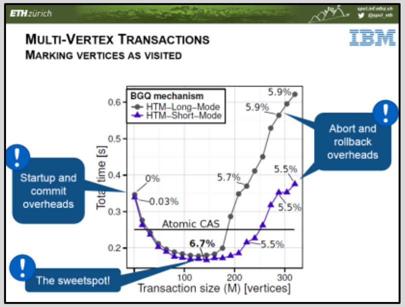
HTM + Active Messages
= Atomic Active Messages

Coarsening & coalescing



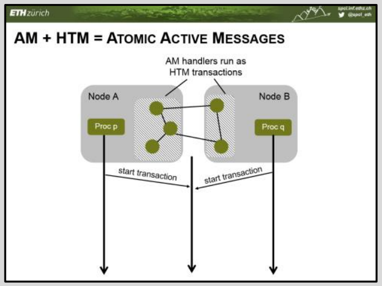
Performance Modeling & Analysis

Haswell & BG/Q Analysis



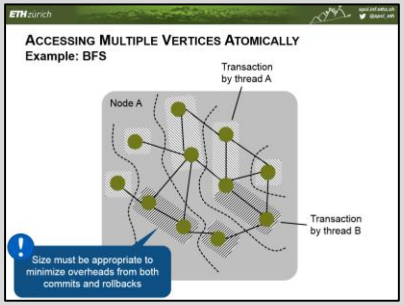
Overview of our Research

HTM for graphs in SM & DM environments



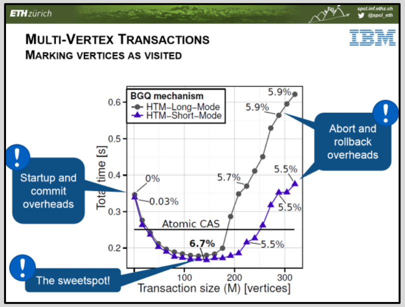
HTM + Active Messages
= Atomic Active Messages

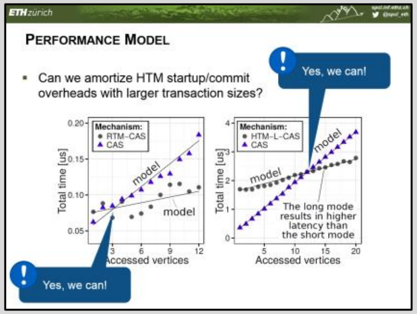
Coarsening & coalescing



Performance Modeling & Analysis

Haswell & BG/Q Analysis

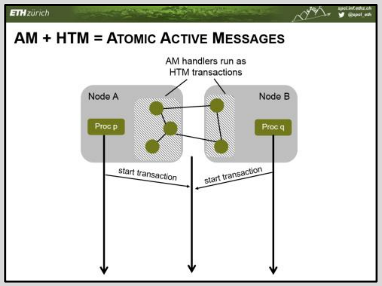




Performance model

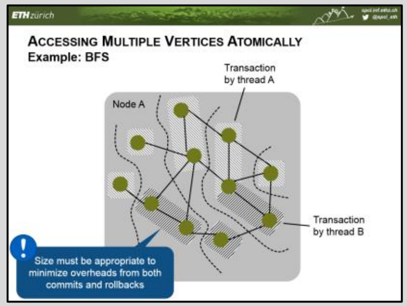
Overview of our Research

HTM for graphs in SM & DM environments



HTM + Active Messages
= Atomic Active Messages

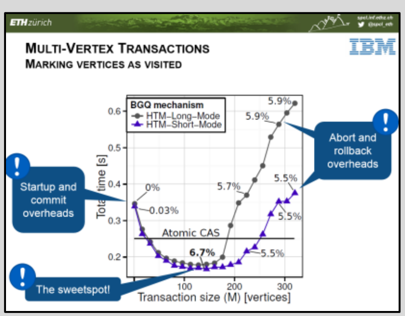
Coarsening & coalescing

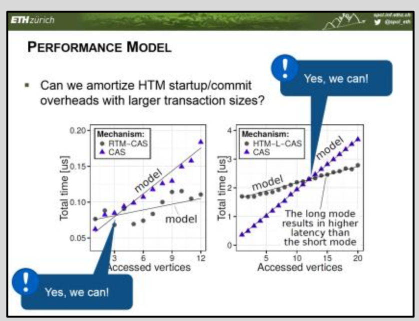


Evaluation

Performance Modeling & Analysis

Haswell & BG/Q Analysis

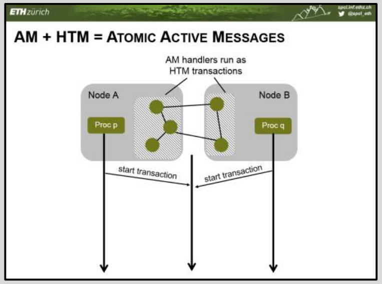




Performance model

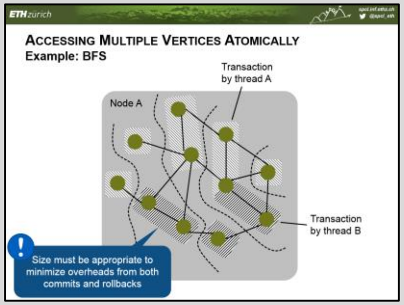
Overview of our Research

HTM for graphs in SM & DM environments

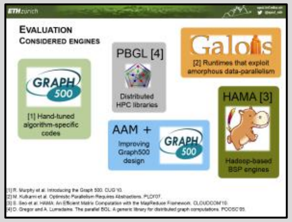


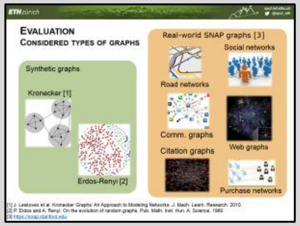
HTM + Active Messages = Atomic Active Messages

Coarsening & coalescing



Evaluation

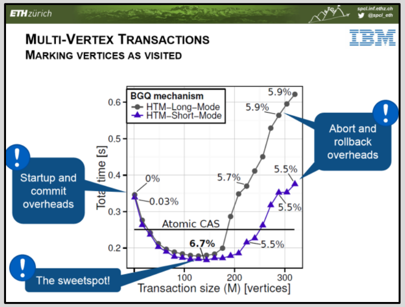


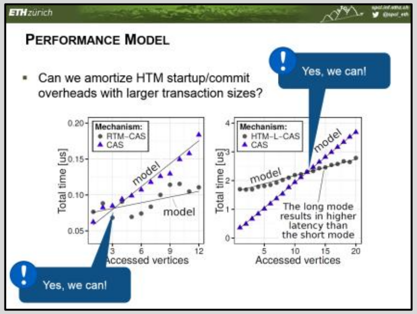


Considered engines and graphs

Performance Modeling & Analysis

Haswell & BG/Q Analysis

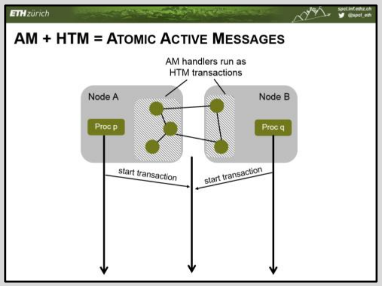




Performance model

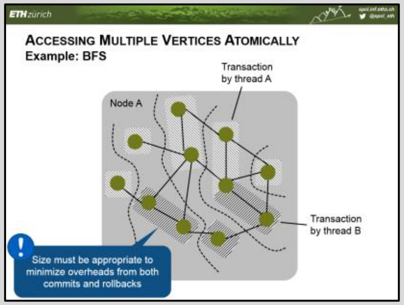
Overview of our Research

HTM for graphs in SM & DM environments

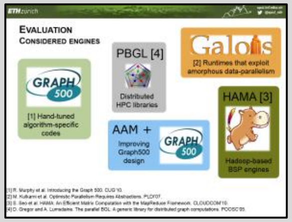


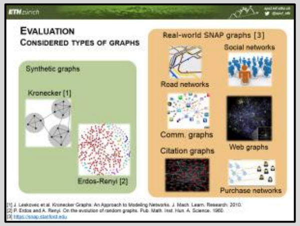
HTM + Active Messages
= Atomic Active Messages

Coarsening & coalescing

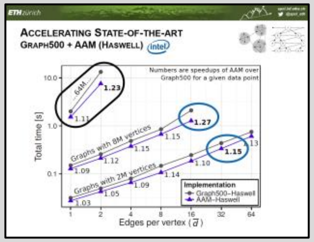


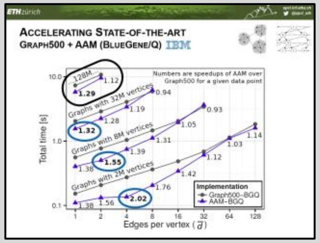
Evaluation





Considered engines and graphs

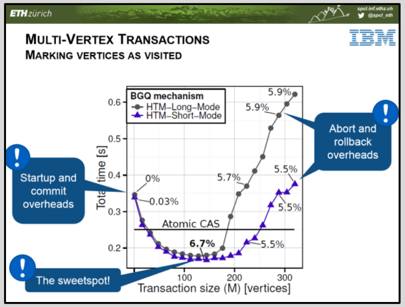


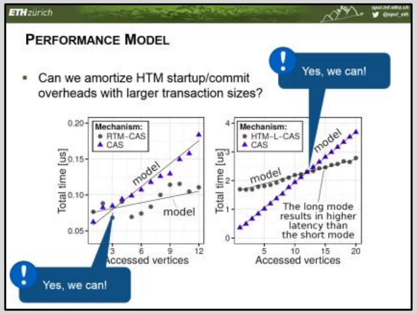


Accelerating state-of-the-art

Performance Modeling & Analysis

Haswell & BG/Q Analysis

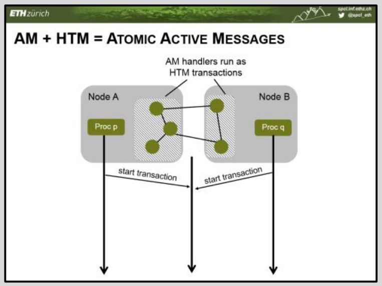




Performance model

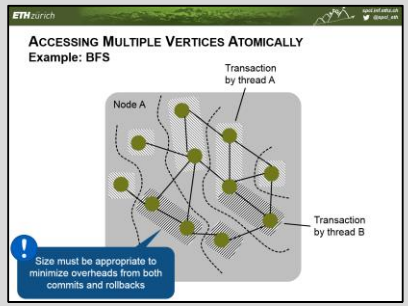
Overview of our Research

HTM for graphs in SM & DM environments



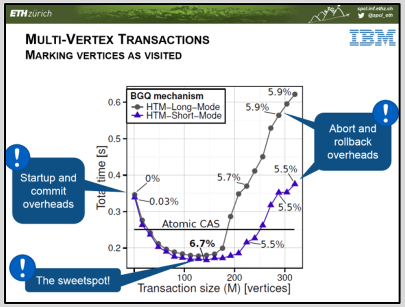
HTM + Active Messages
= Atomic Active Messages

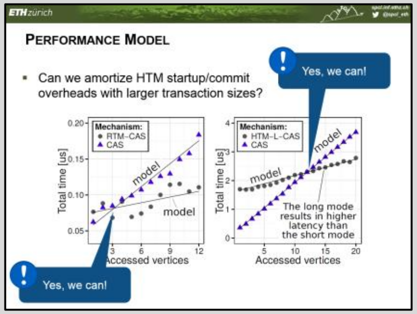
Coarsening & coalescing



Performance Modeling & Analysis

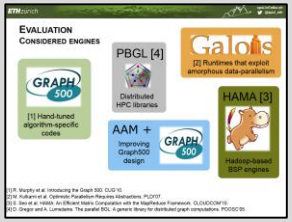
Haswell & BG/Q Analysis

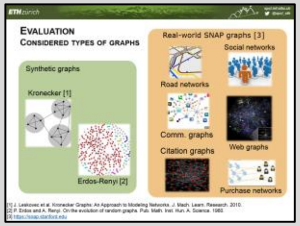




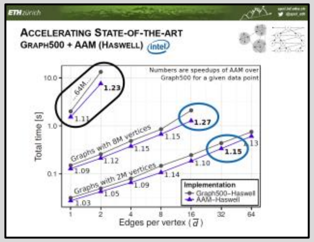
Performance model

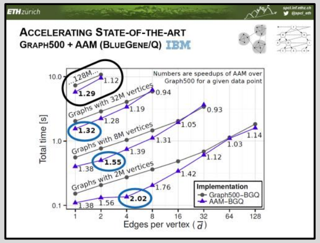
Evaluation



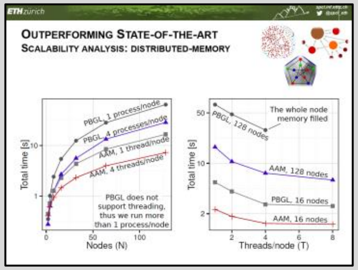


Considered engines and graphs





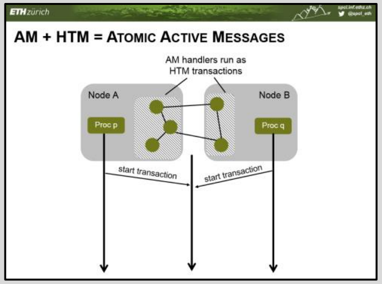
Accelerating state-of-the-art



Scalability

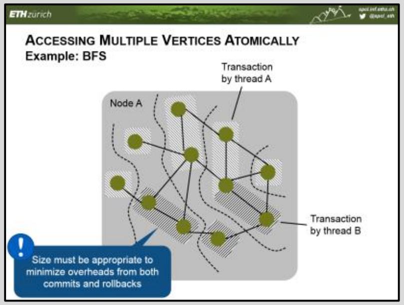
Overview of our Research

HTM for graphs in SM & DM environments



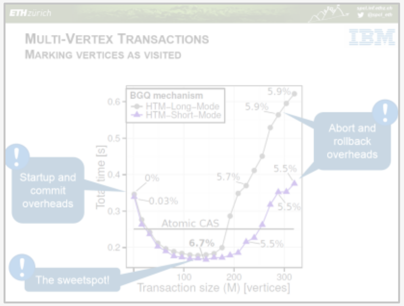
HTM + Active Messages
= Atomic Active Messages

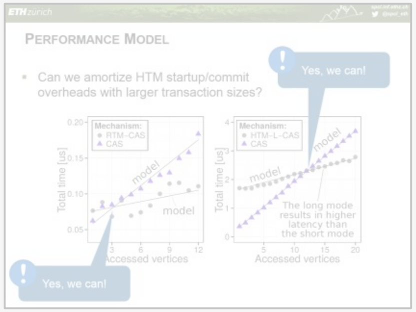
Coarsening & coalescing



Performance Modeling & Analysis

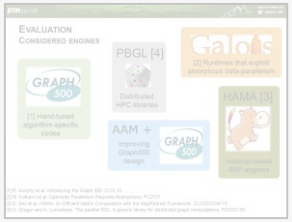
Haswell & BG/Q Analysis

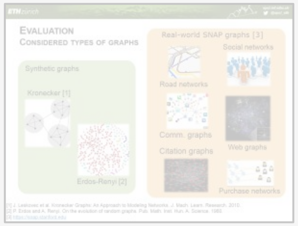




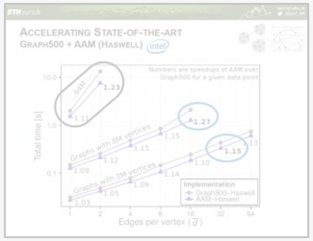
Performance model

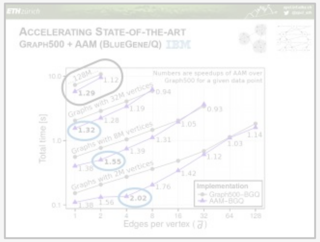
Evaluation



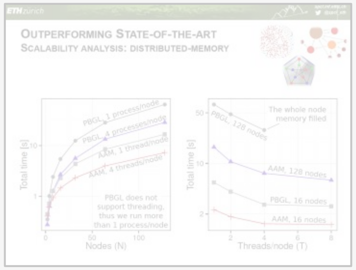


Considered engines and graphs





Accelerating state-of-the-art



Scalability

Active Messages (AM)

ACTIVE MESSAGES (AM)

Process p

Process q

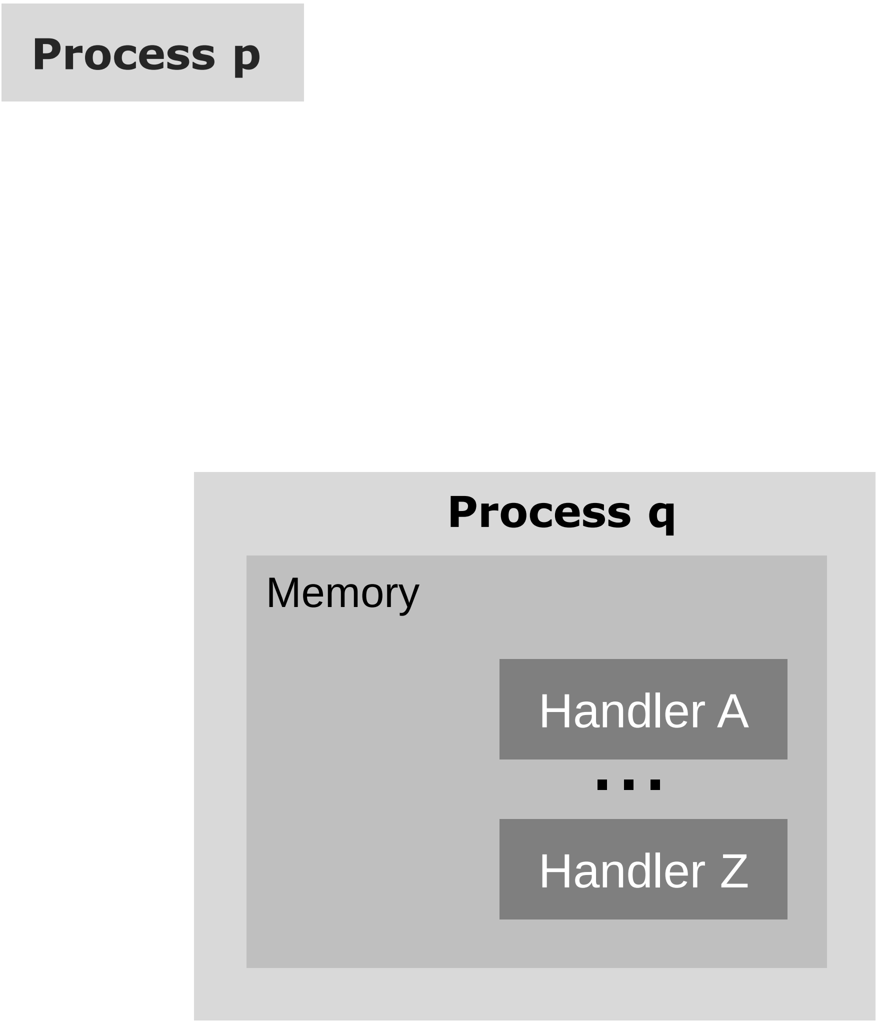
ACTIVE MESSAGES (AM)

Process p

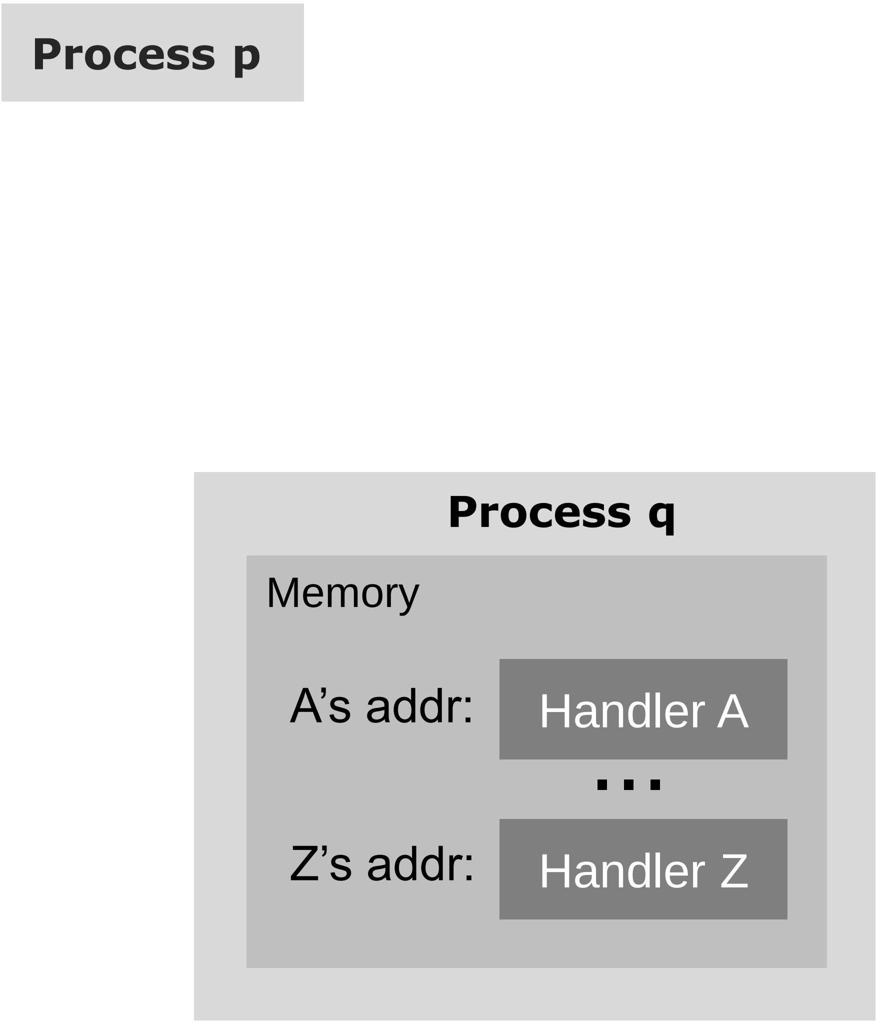
Process q

Memory

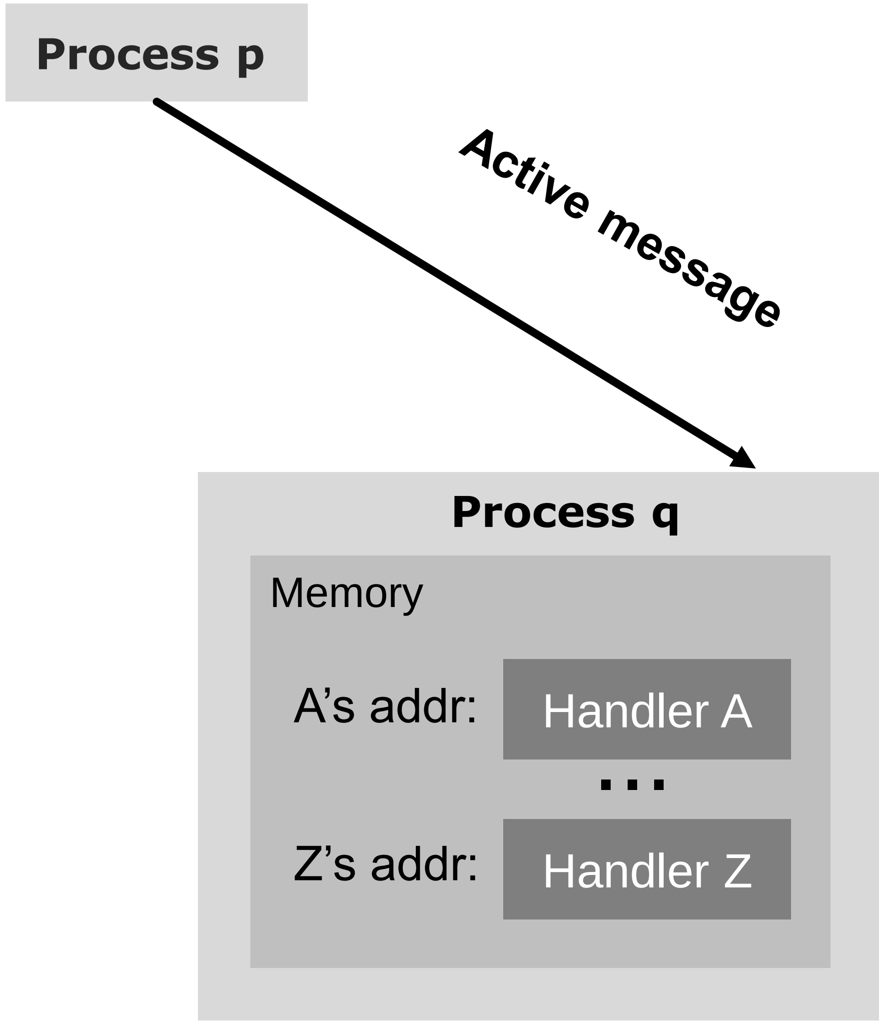
ACTIVE MESSAGES (AM)



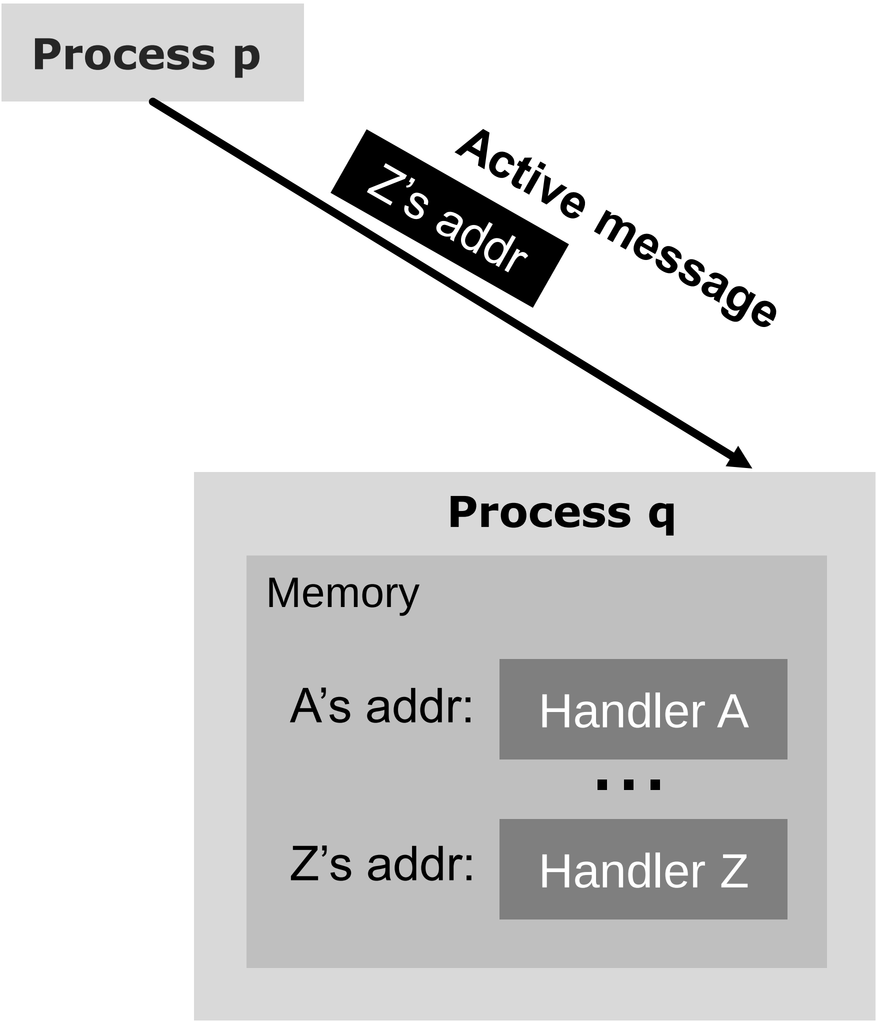
ACTIVE MESSAGES (AM)



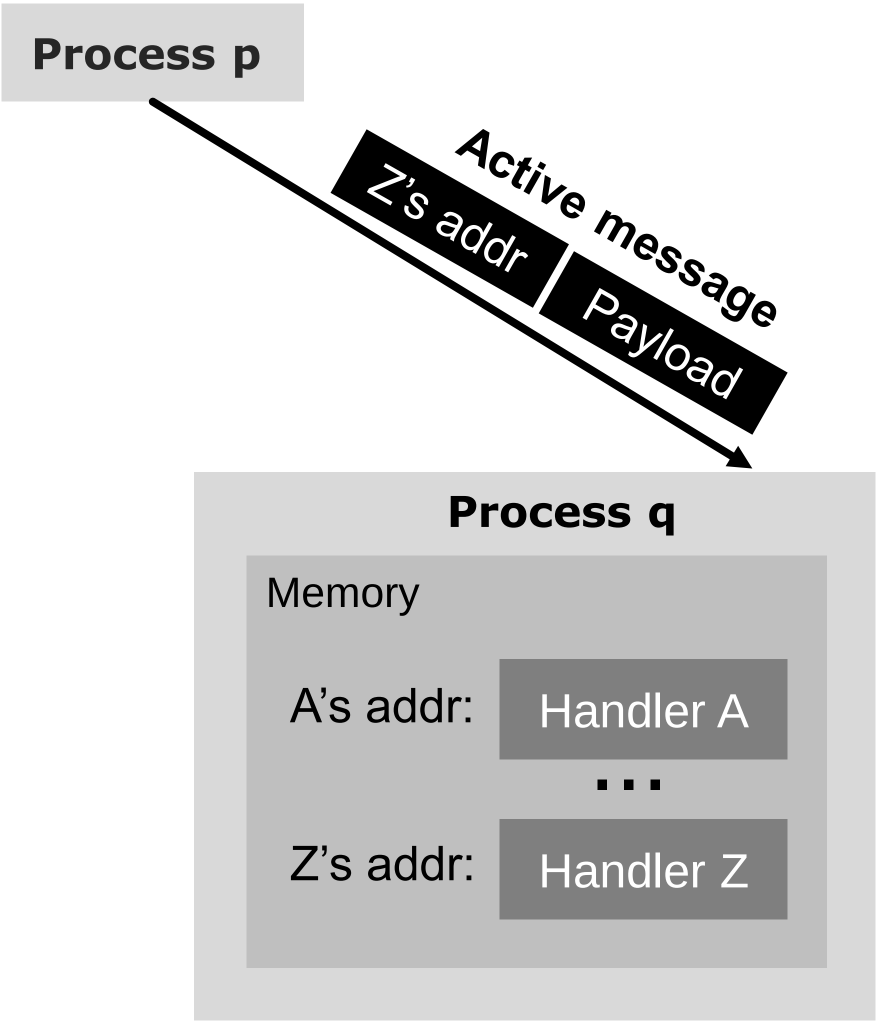
ACTIVE MESSAGES (AM)



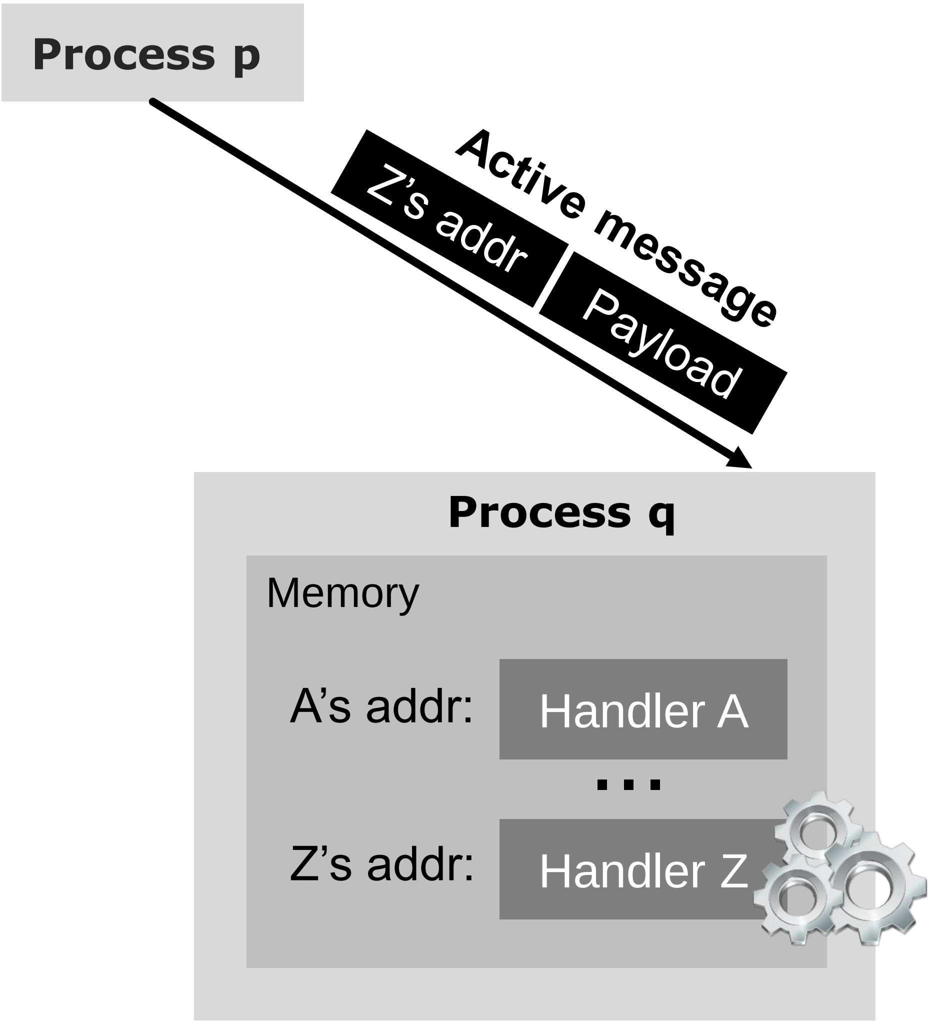
ACTIVE MESSAGES (AM)



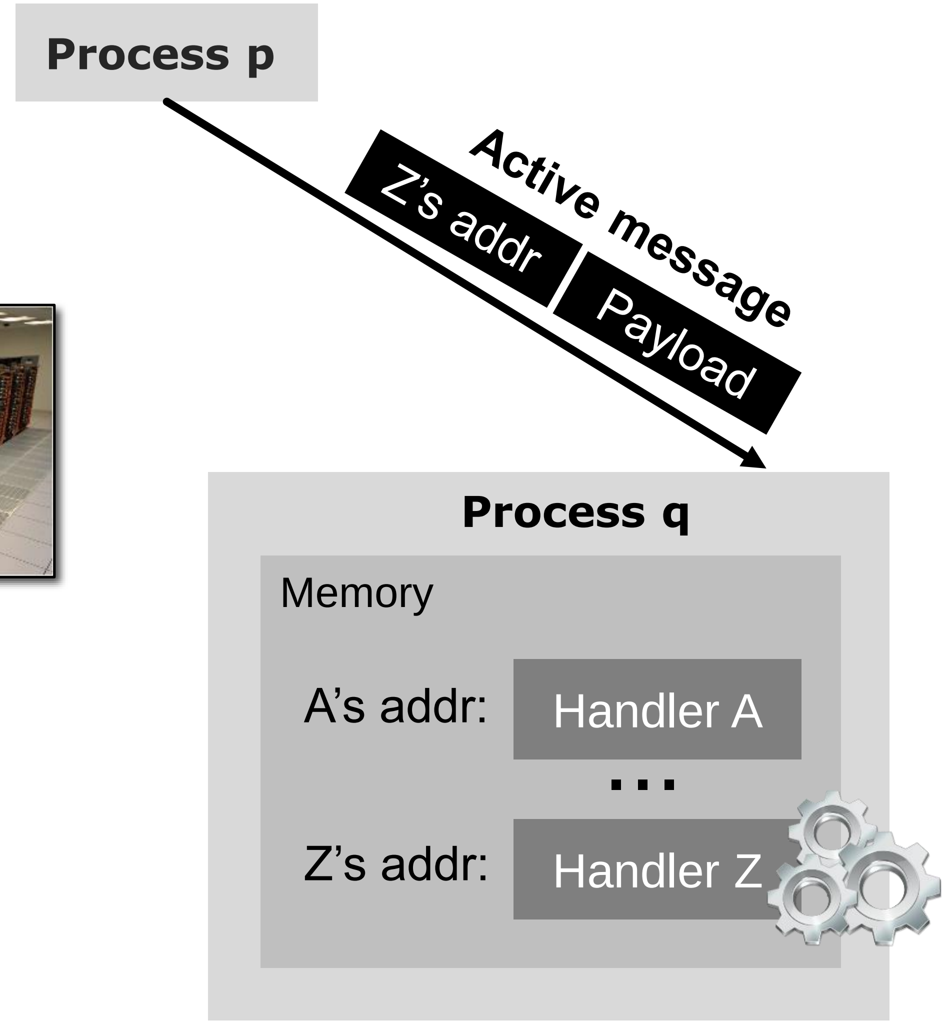
ACTIVE MESSAGES (AM)



ACTIVE MESSAGES (AM)



ACTIVE MESSAGES (AM)

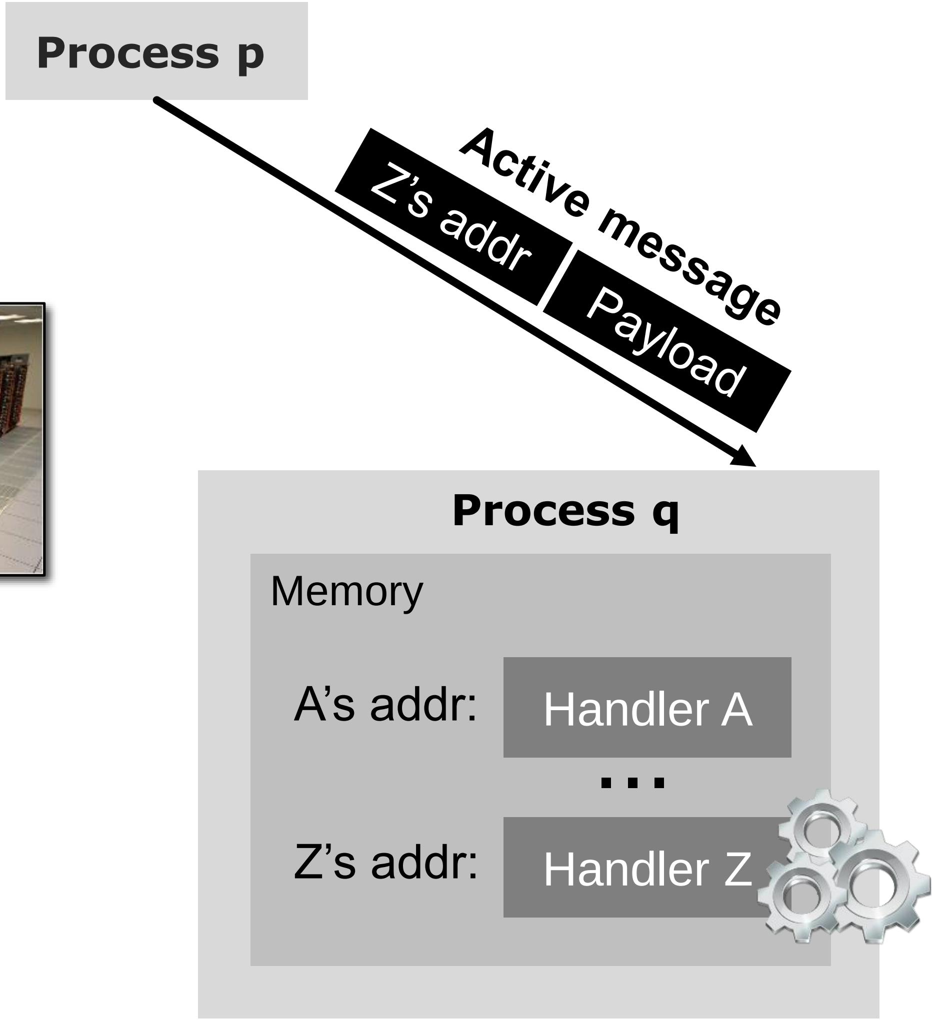


ACTIVE MESSAGES (AM)

IBM

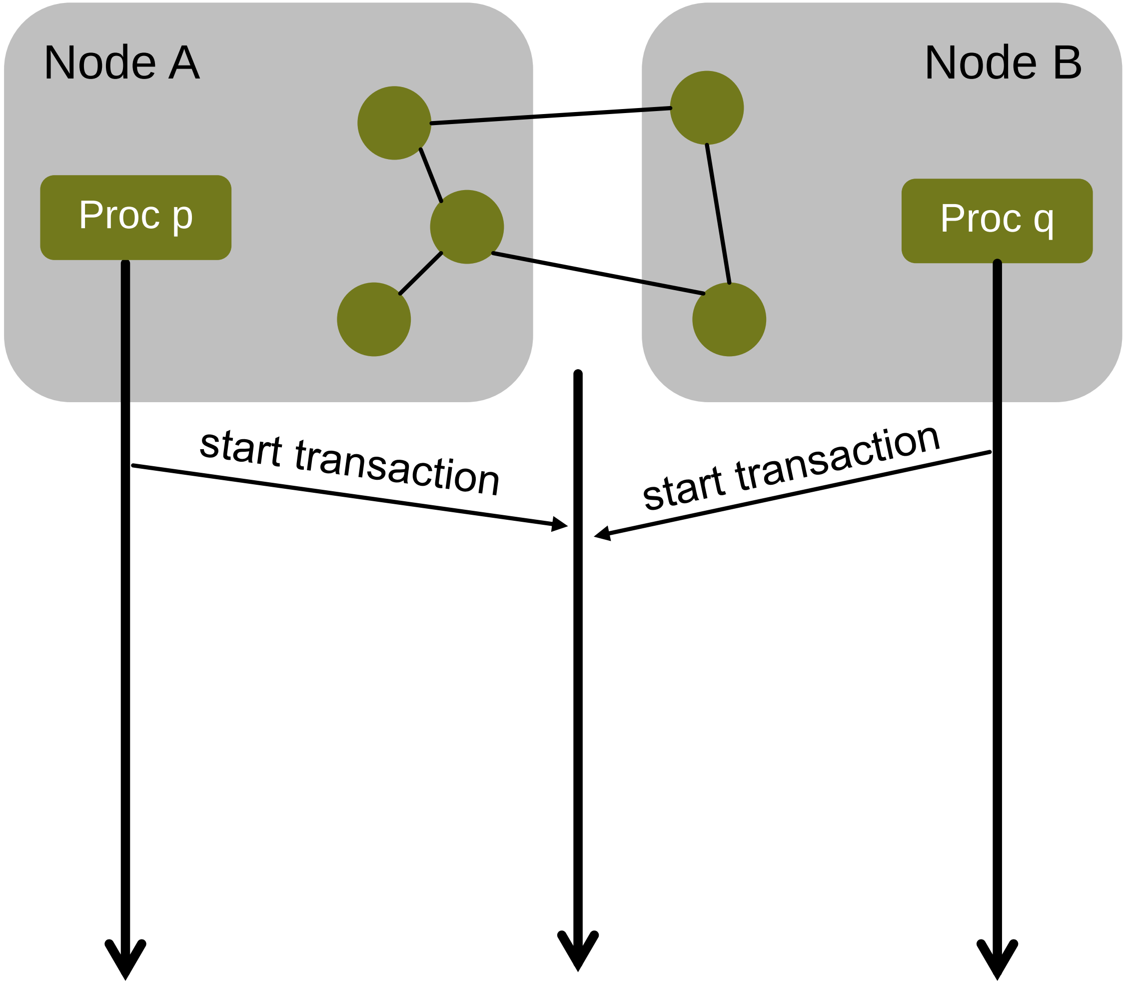
AM++[1]

GASNet [2]

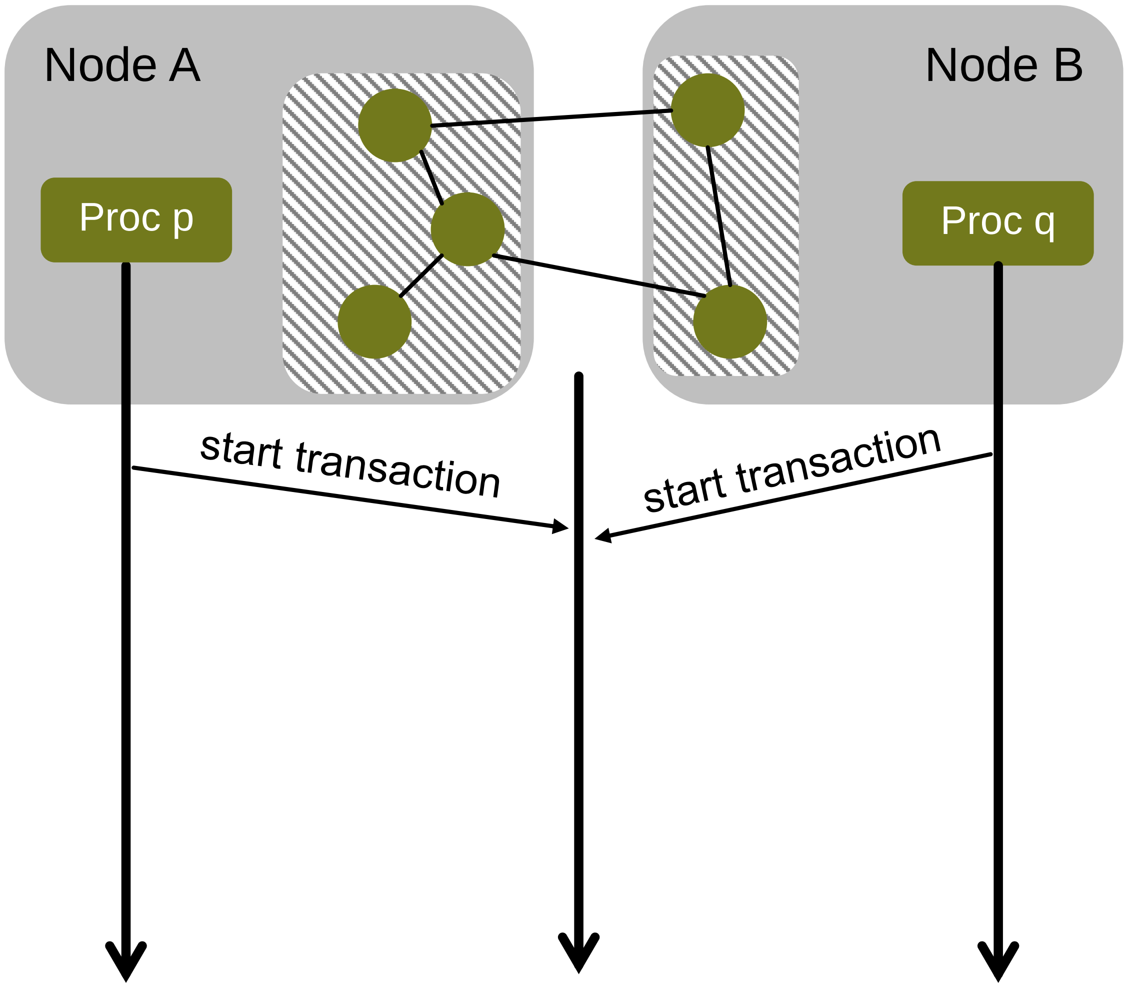


[1] J. J. Willcock et al. AM++: A generalized active message framework. PACT'10.
[2] D. Bonachea, GASNet Specification, v1.1. Berkeley Technical Report. 2002.

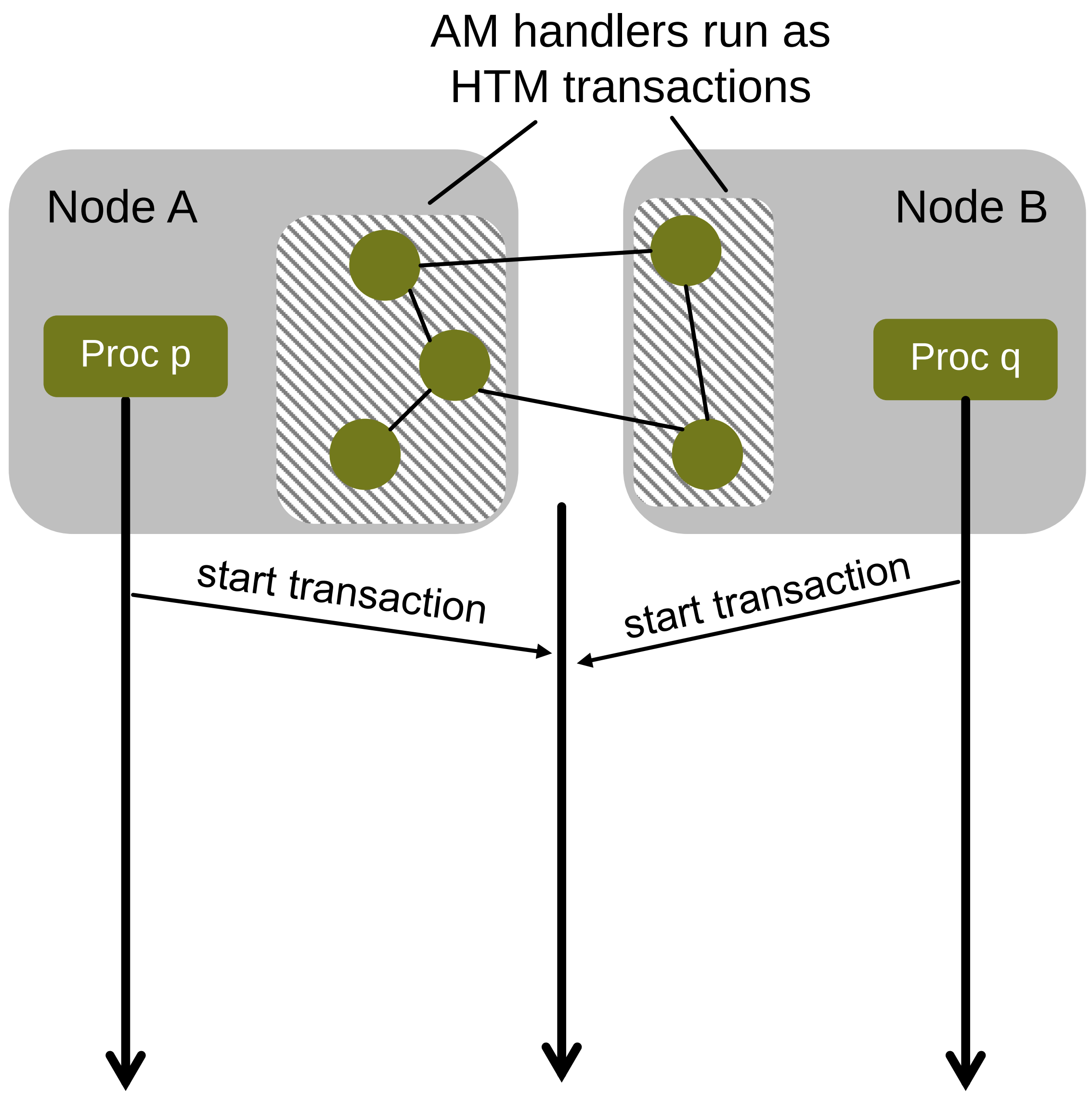
AM + HTM = ...


Node A
Node B
Proc p
Proc q
start transaction
start transaction

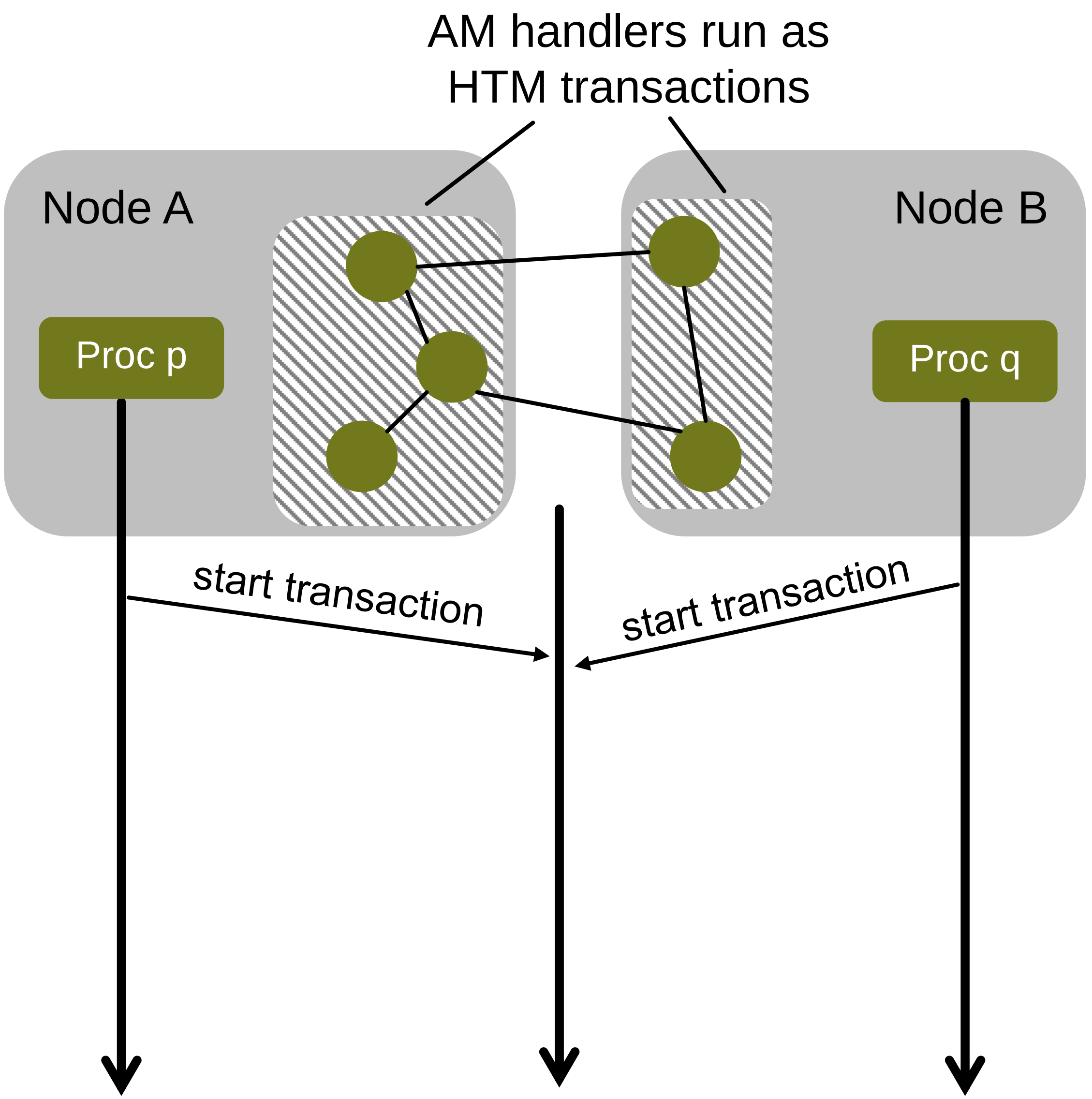
AM + HTM = ...


Node A
Node B
Proc p
Proc q
start transaction
start transaction

AM + HTM = ...

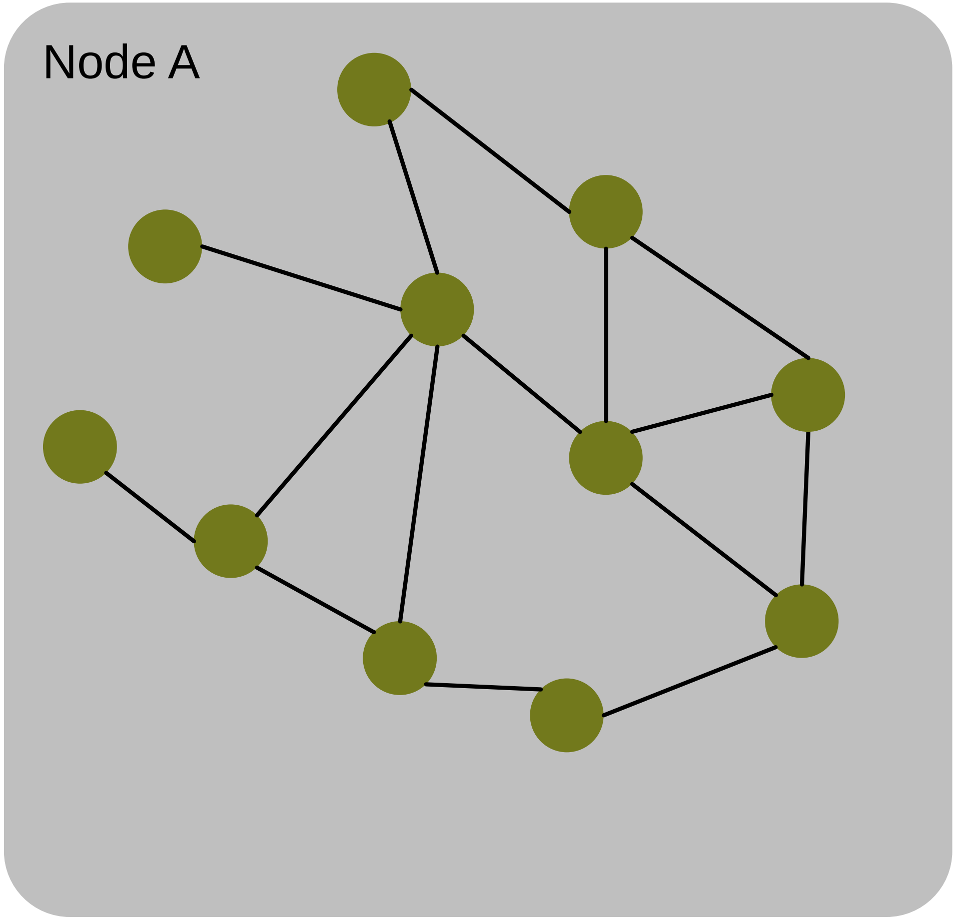

AM handlers run as
HTM transactions
Node A
Node B
Proc p
Proc q
start transaction
start transaction

AM + HTM = Atomic Active Messages

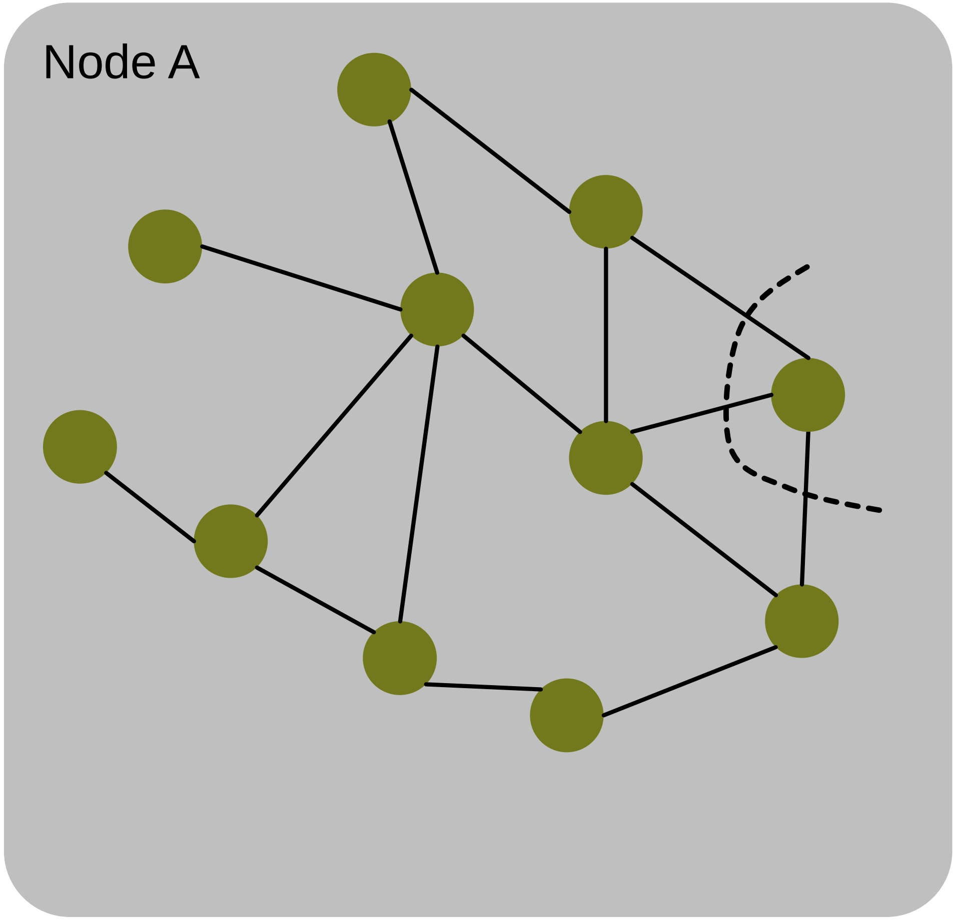


ACCESSING MULTIPLE VERTICES ATOMICALLY

Example: BFS

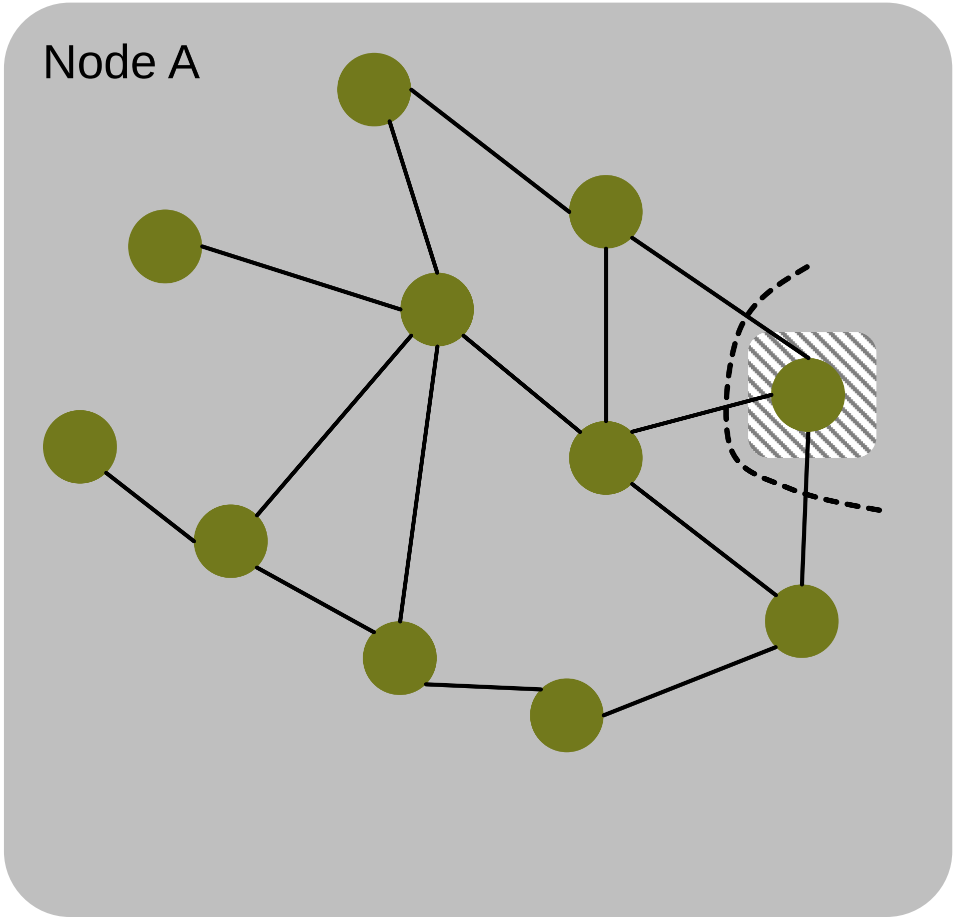


Accessing Multiple Vertices Atomically
Example: BFS



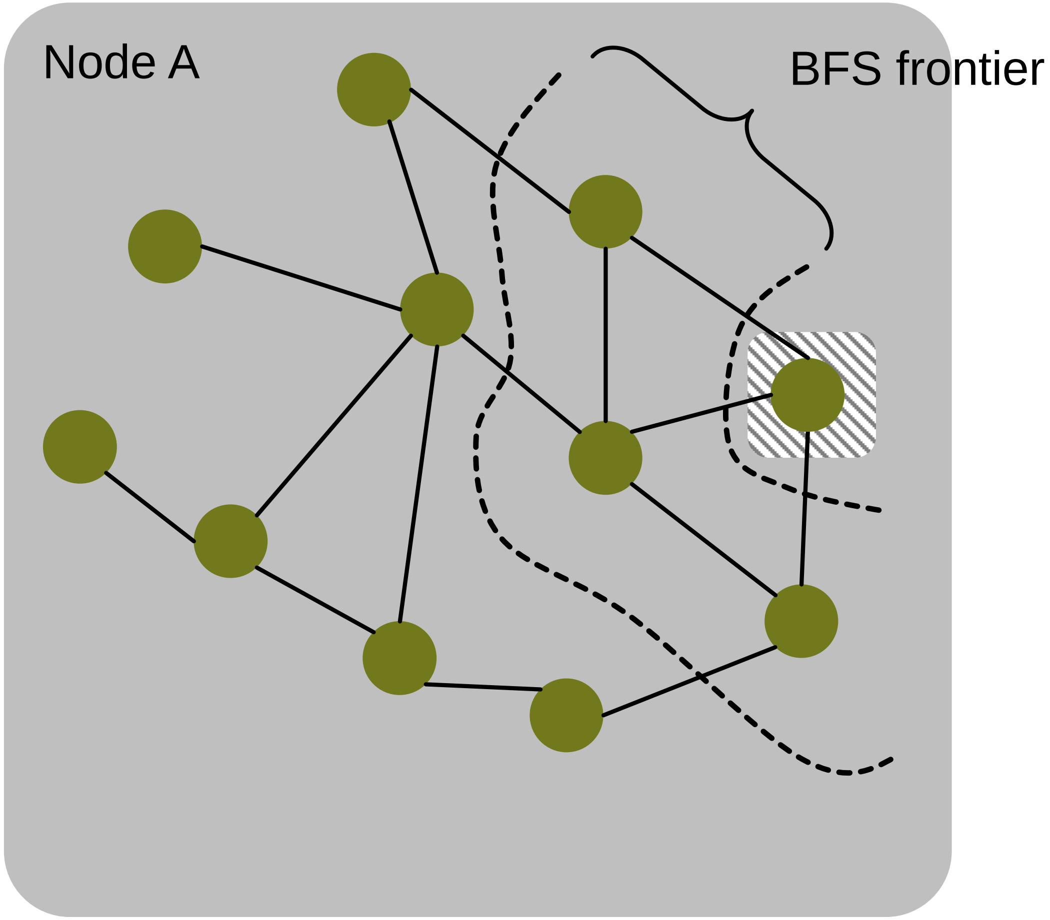
Accessing Multiple Vertices Atomically

Example: BFS



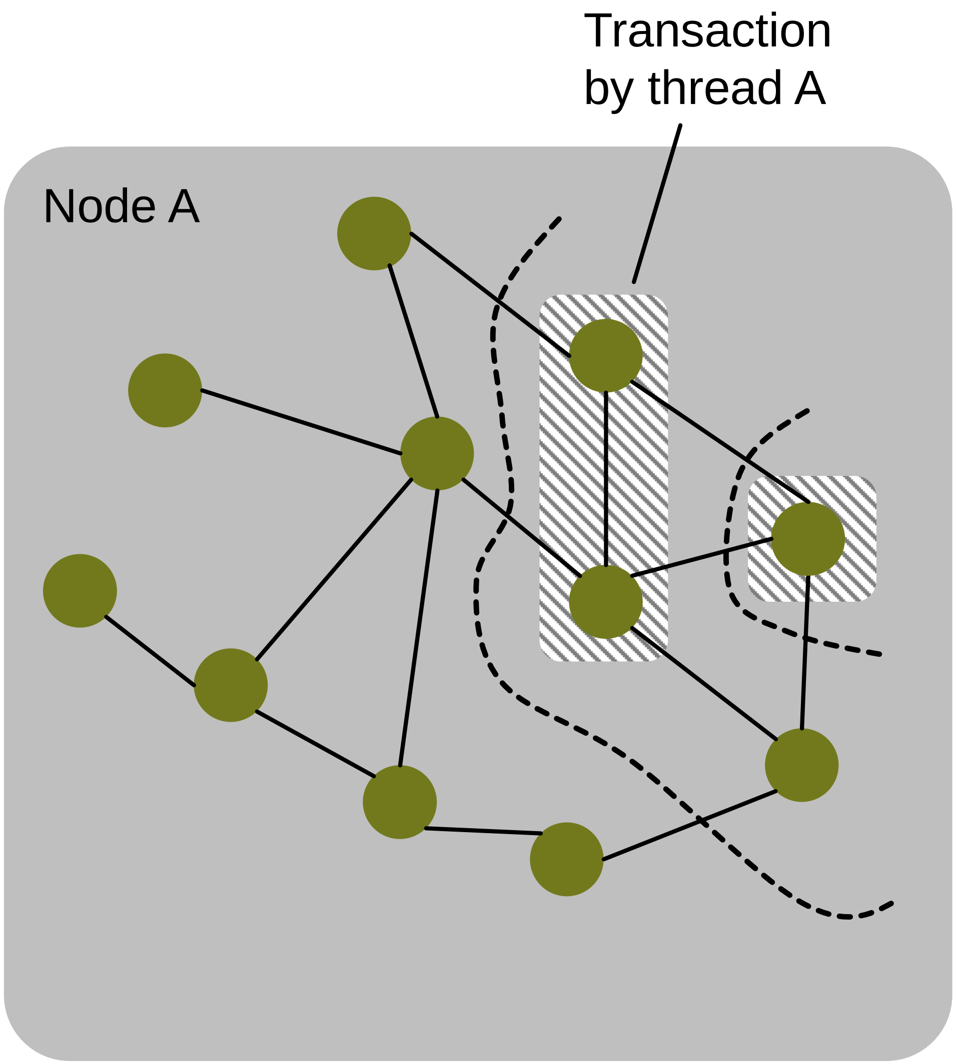
Accessing Multiple Vertices Atomically

Example: BFS



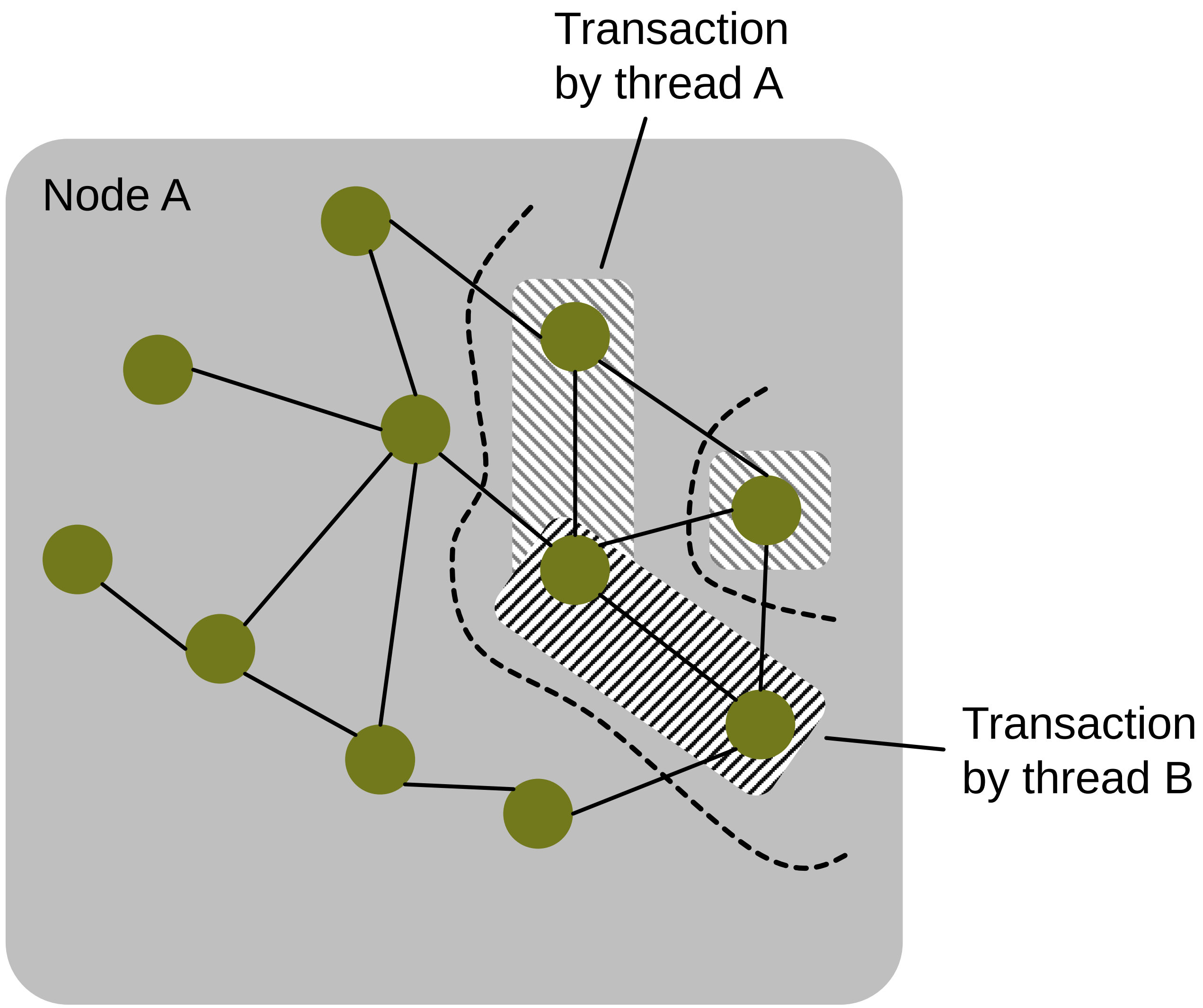
ACCESSING MULTIPLE VERTICES ATOMICALLY

Example: BFS



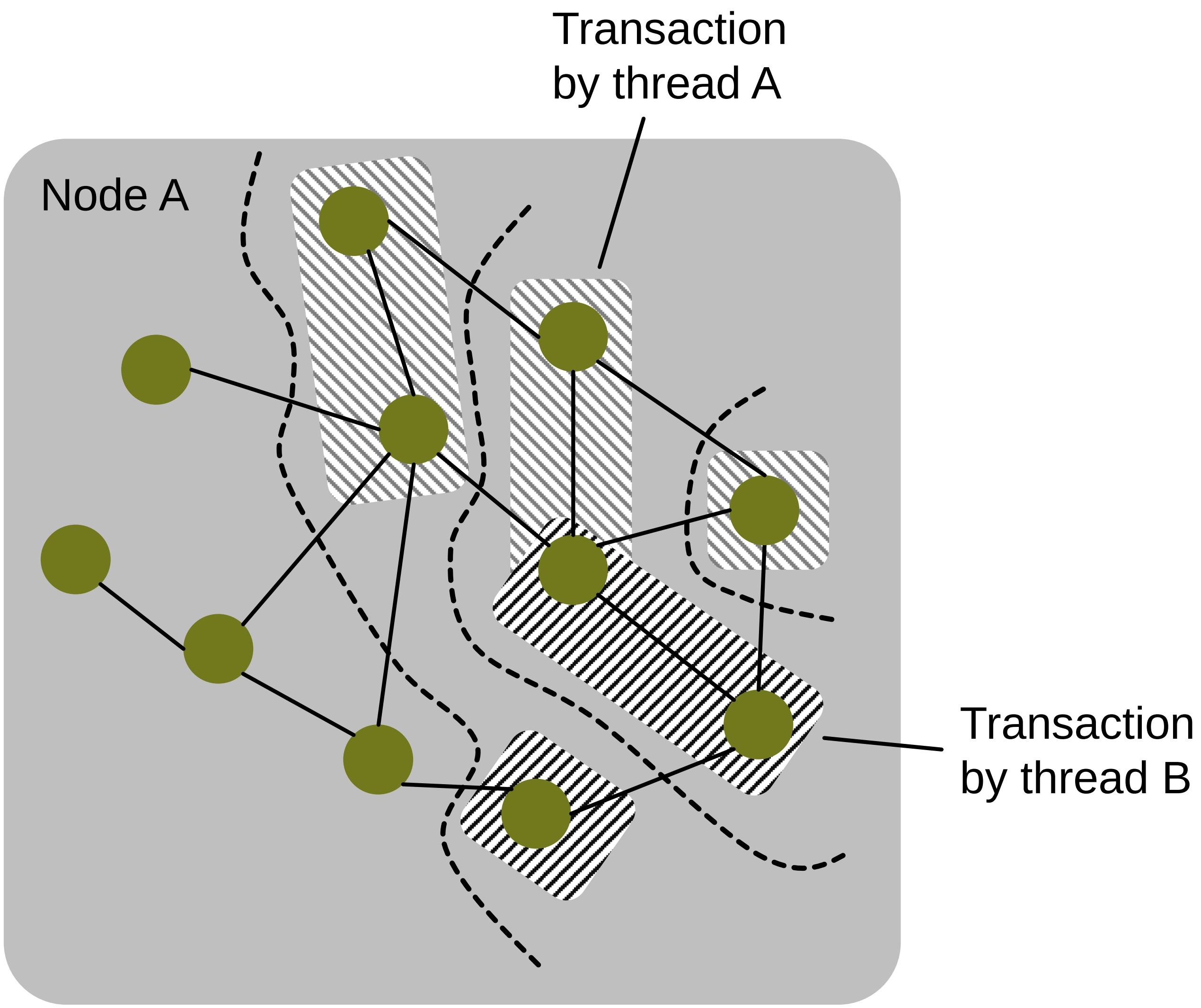
Accessing Multiple Vertices Atomically

Example: BFS

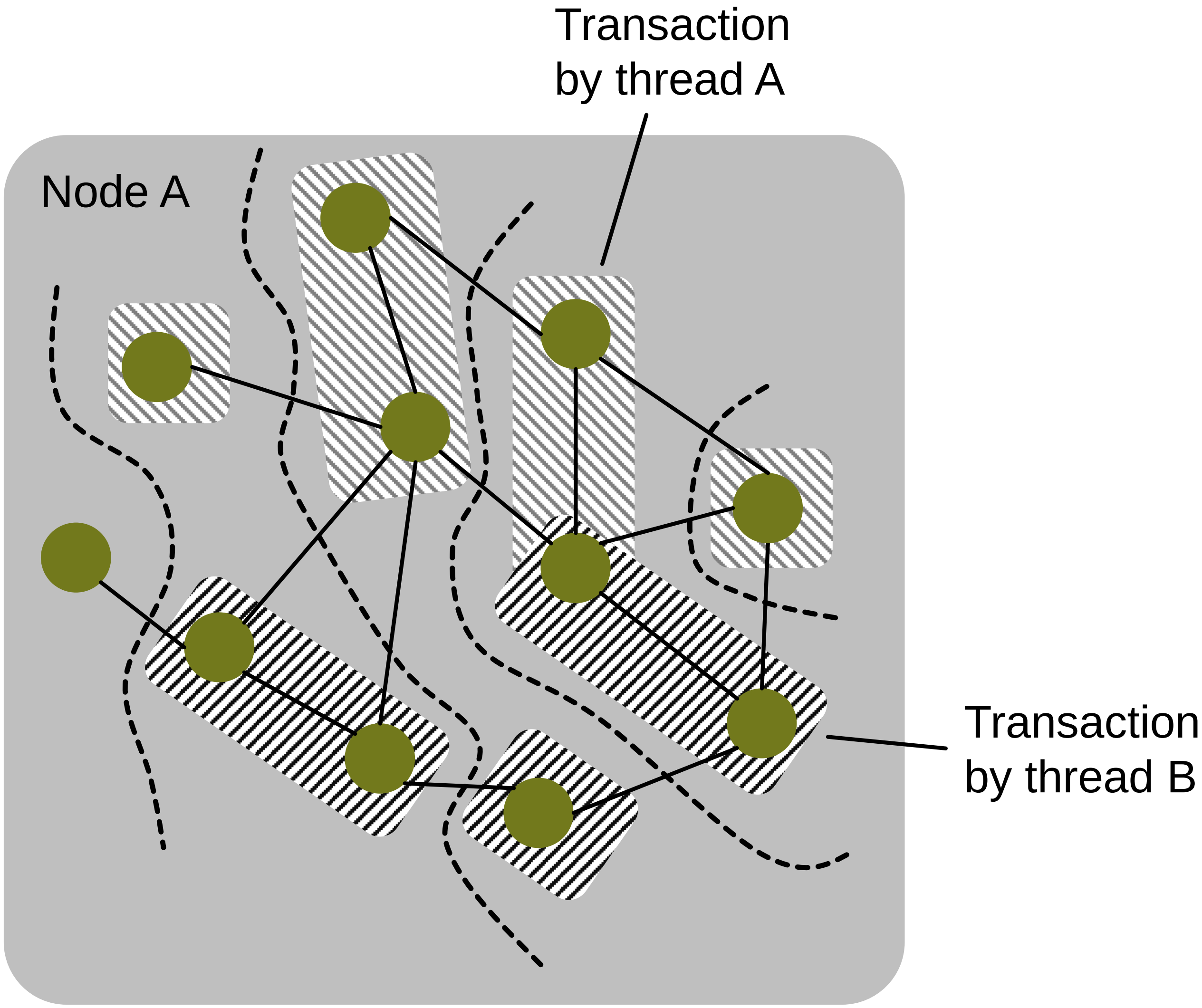


Accessing Multiple Vertices Atomically

Example: BFS

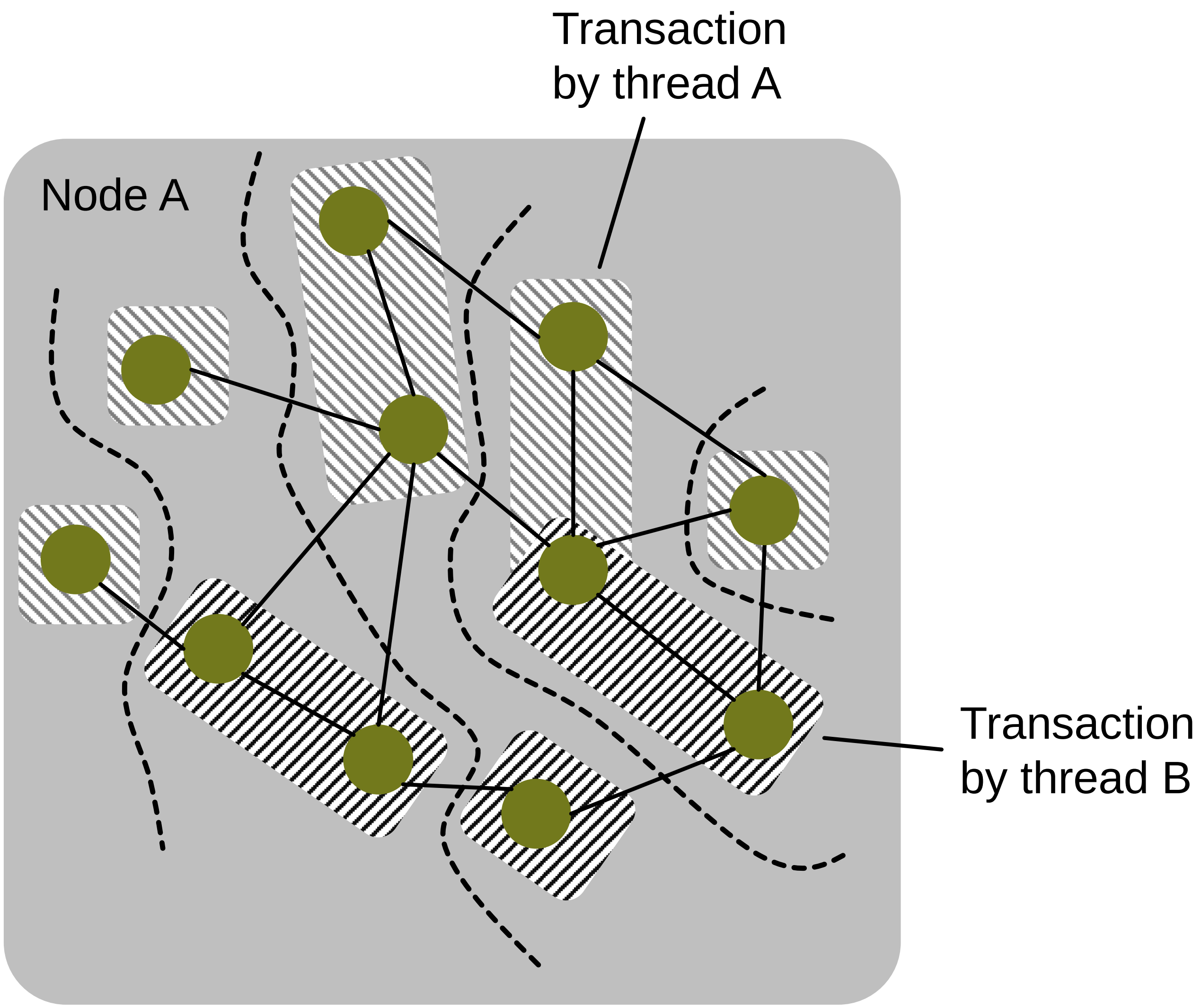


Accessing Multiple Vertices Atomically
Example: BFS



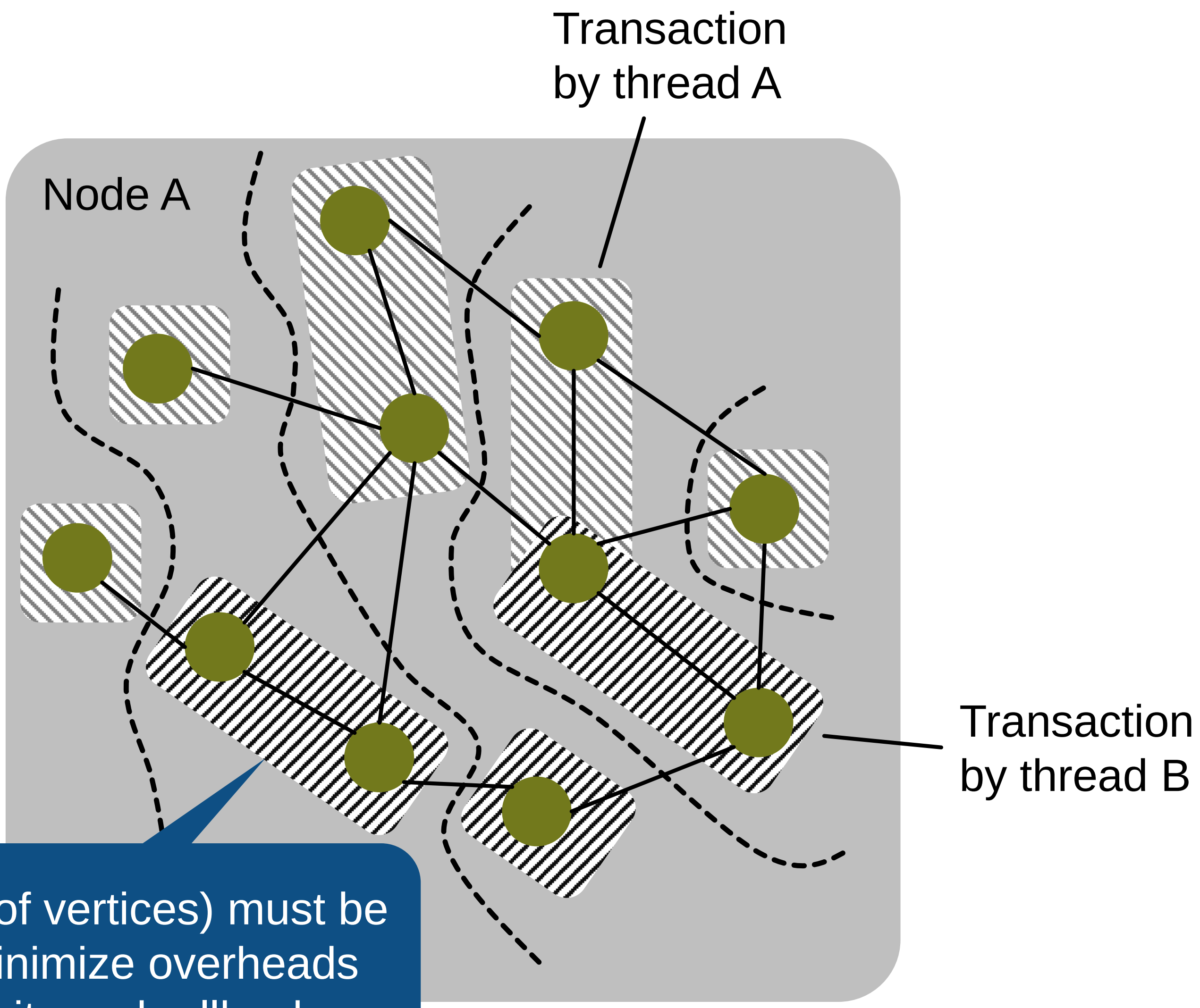
Accessing Multiple Vertices Atomically

Example: BFS



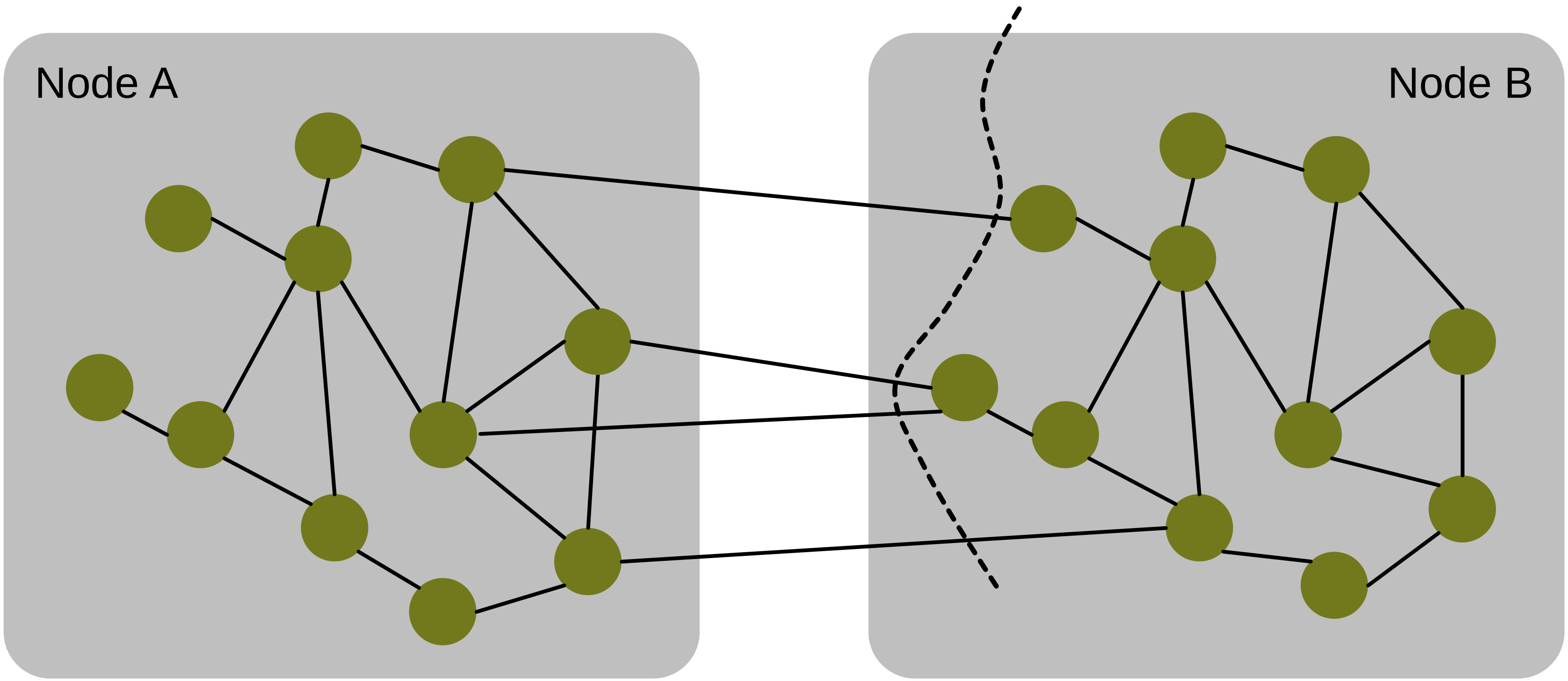
Accessing Multiple Vertices Atomically

Example: BFS

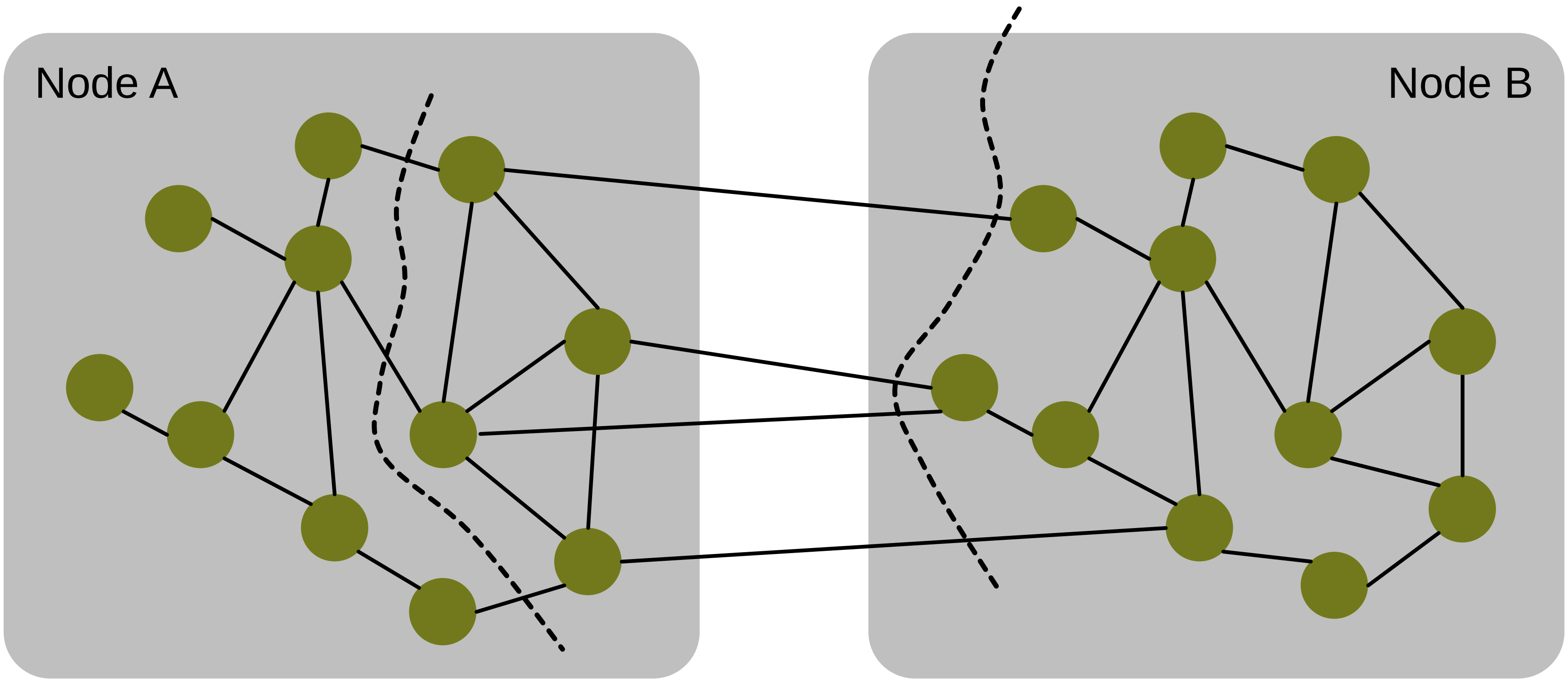


Size (the number of vertices) must be appropriate to minimize overheads from both commits and rollbacks

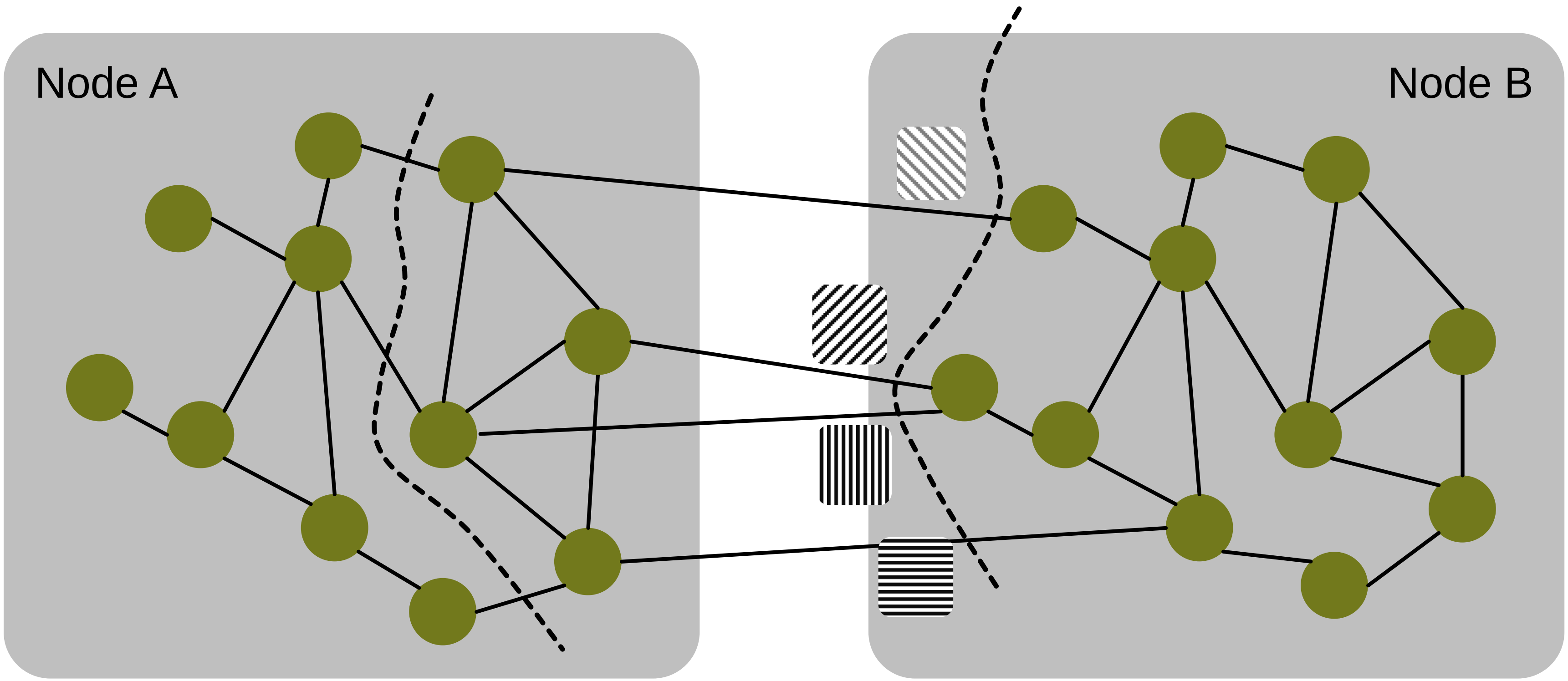
Transferring Transactions Across Nodes



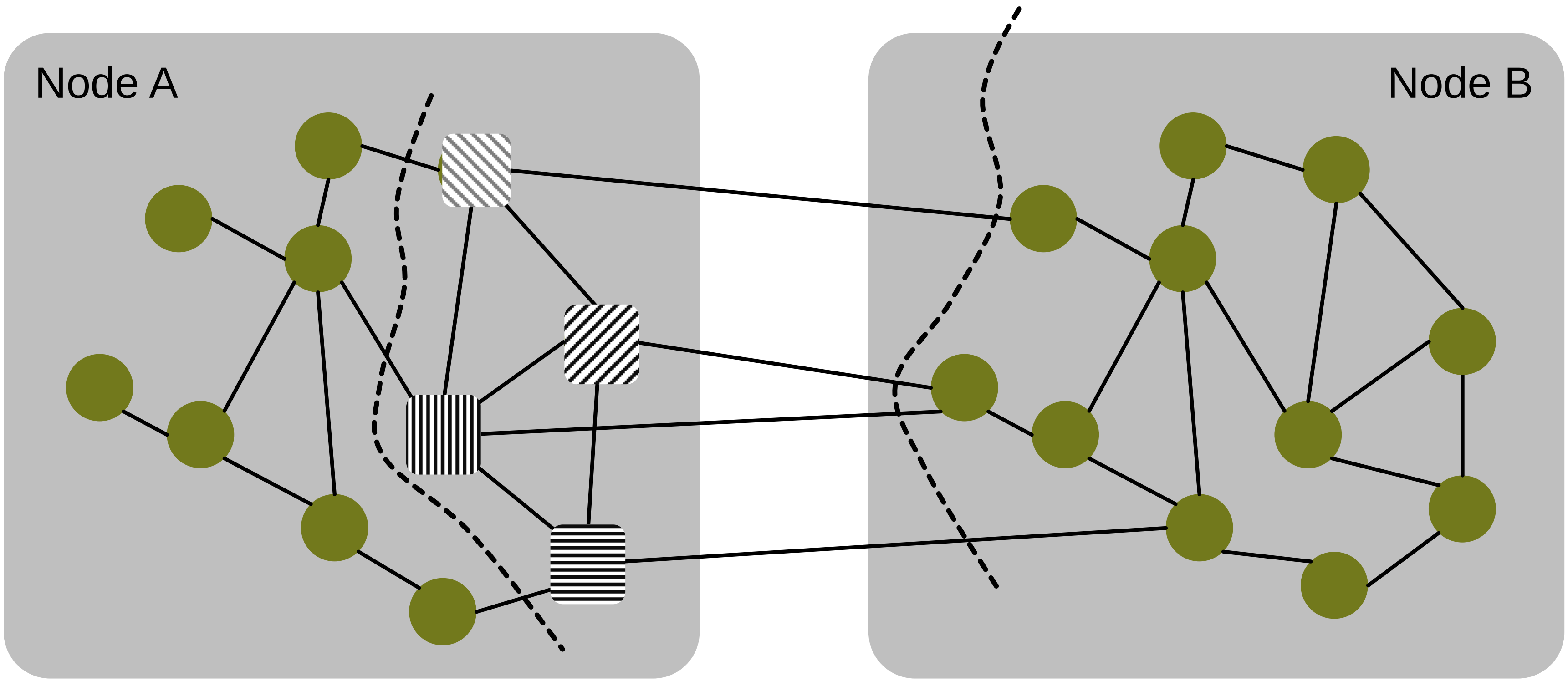
Transferring Transactions Across Nodes



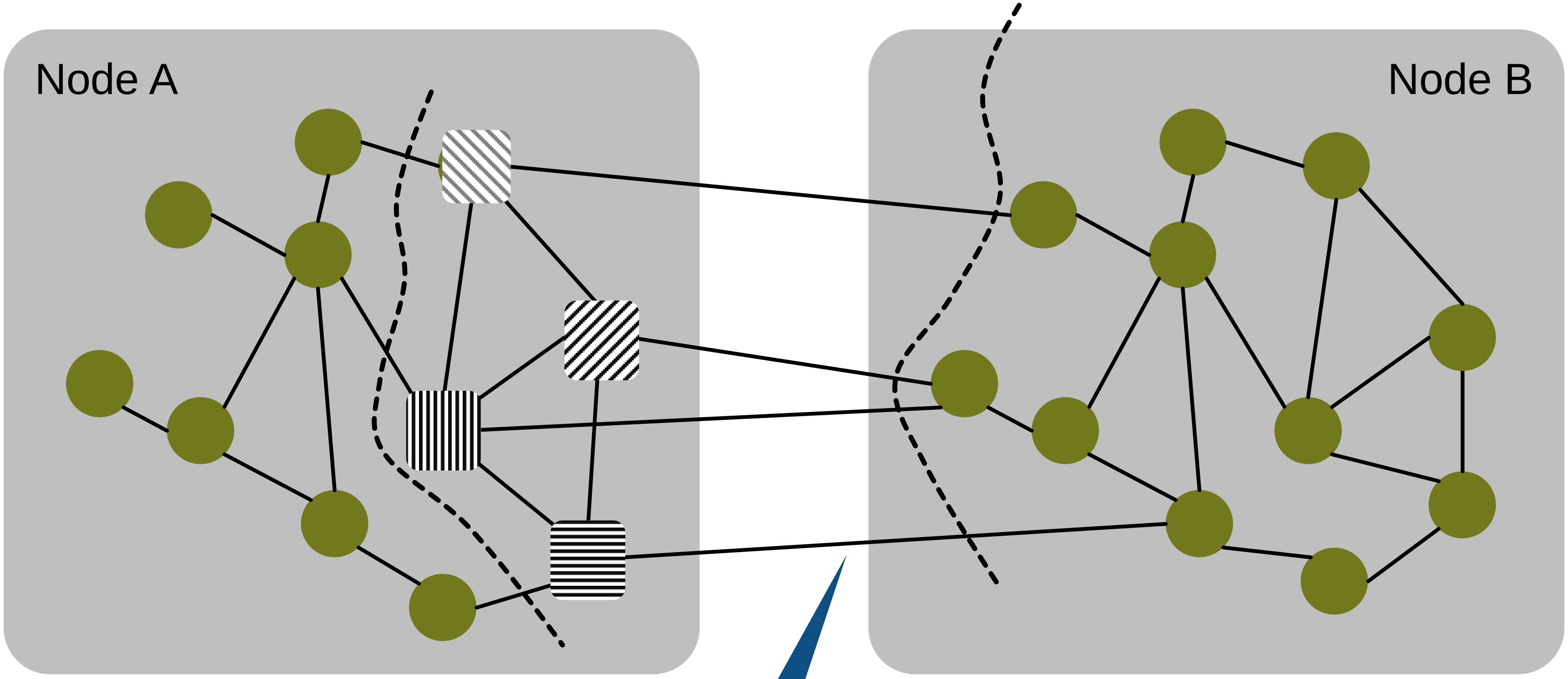
TRANSFERRING TRANSACTIONS ACROSS NODES



TRANSFERRING TRANSACTIONS ACROSS NODES

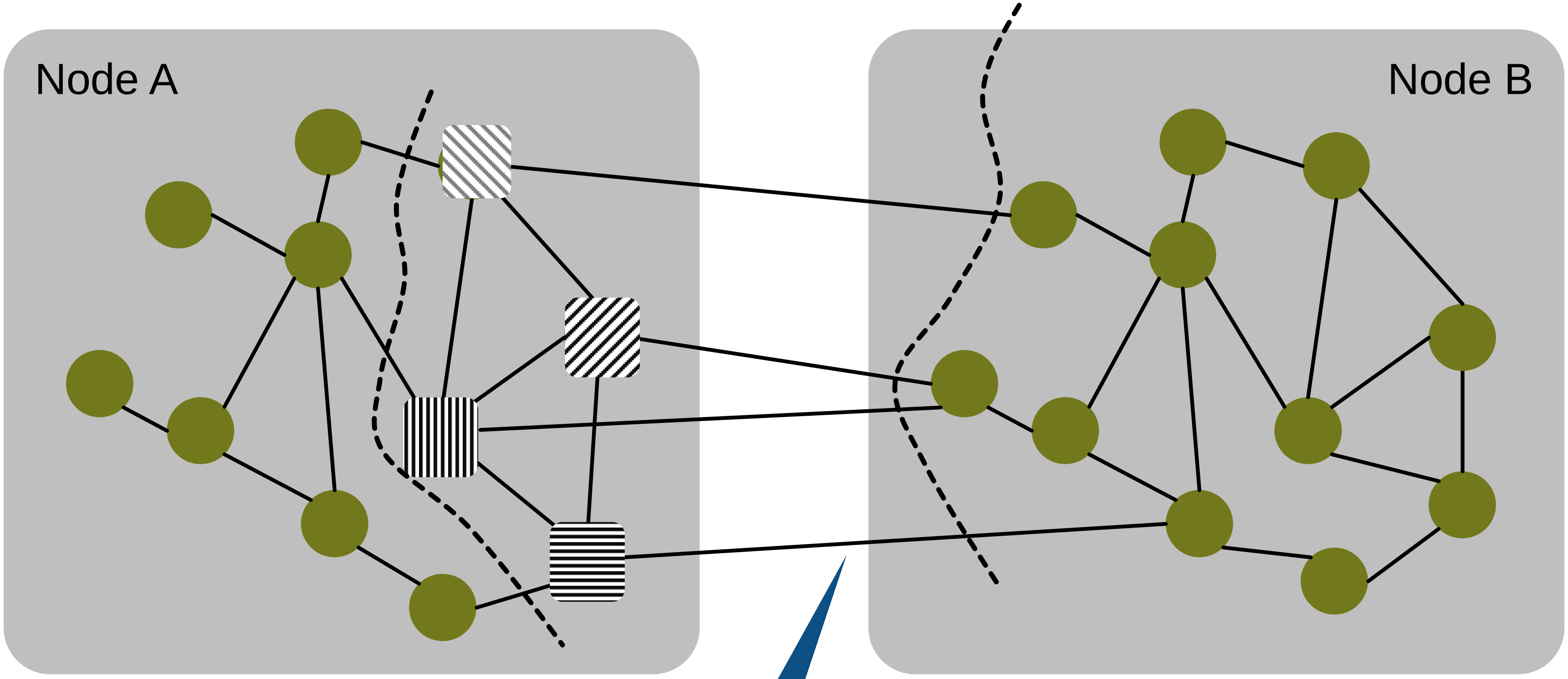


Transferring Transactions Across Nodes



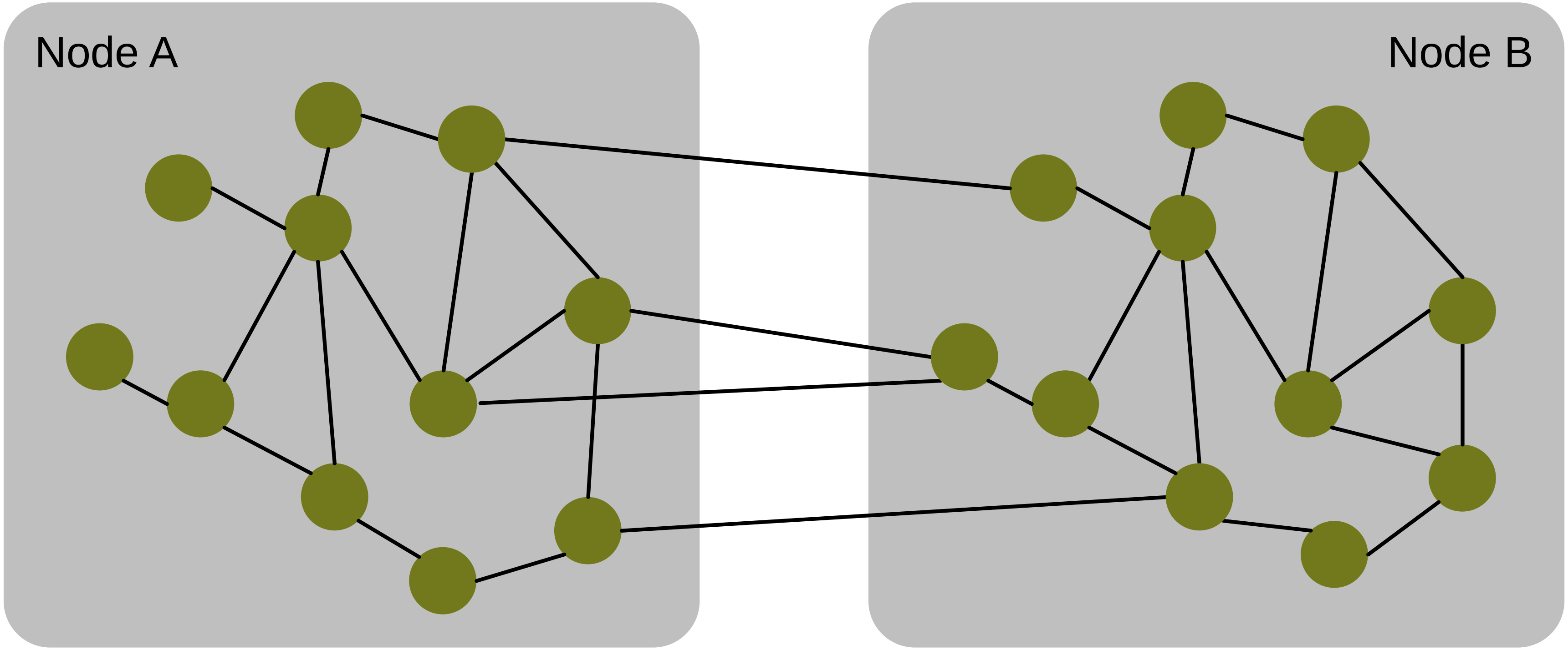
Transactions must be appropriately coalesced to minimize communication overheads

Transferring Transactions Across Nodes

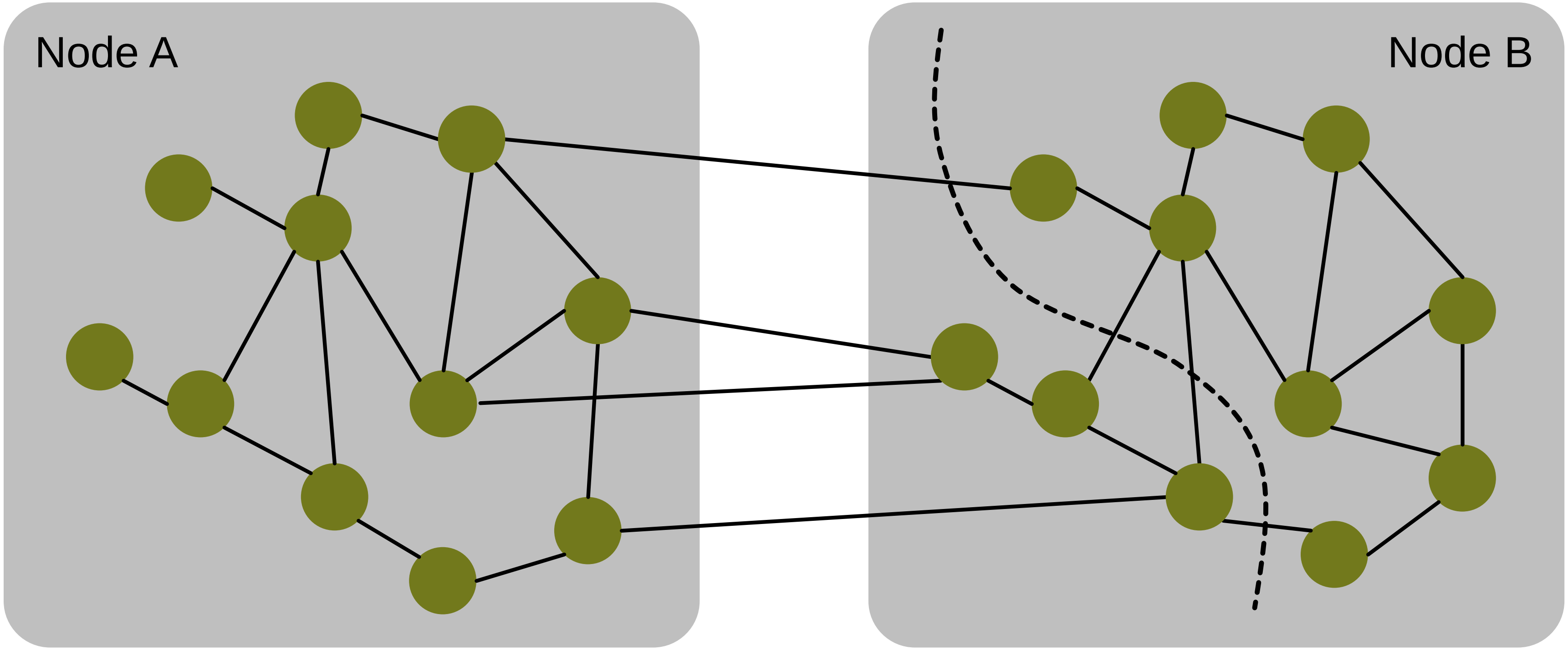


Transactions must be appropriately coalesced to minimize communication overheads

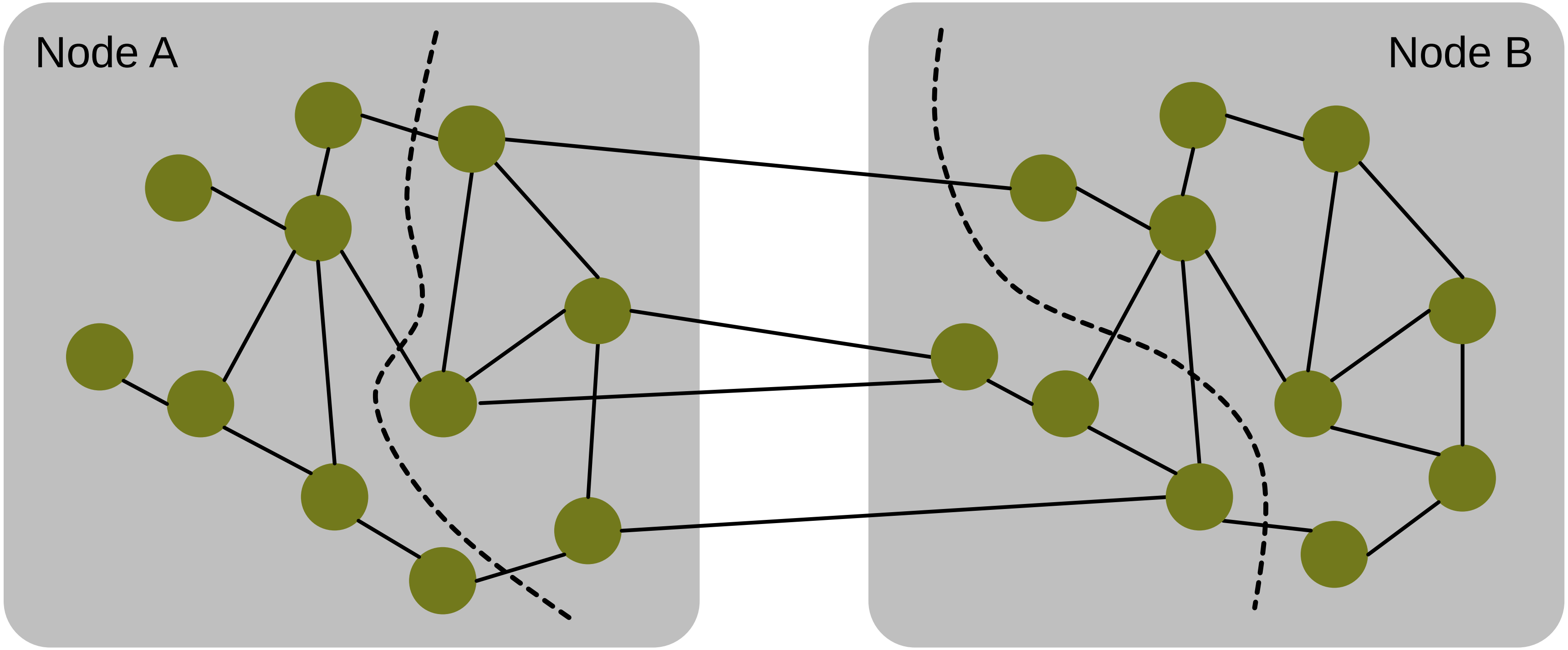
Executing Transactions on Multiple Nodes



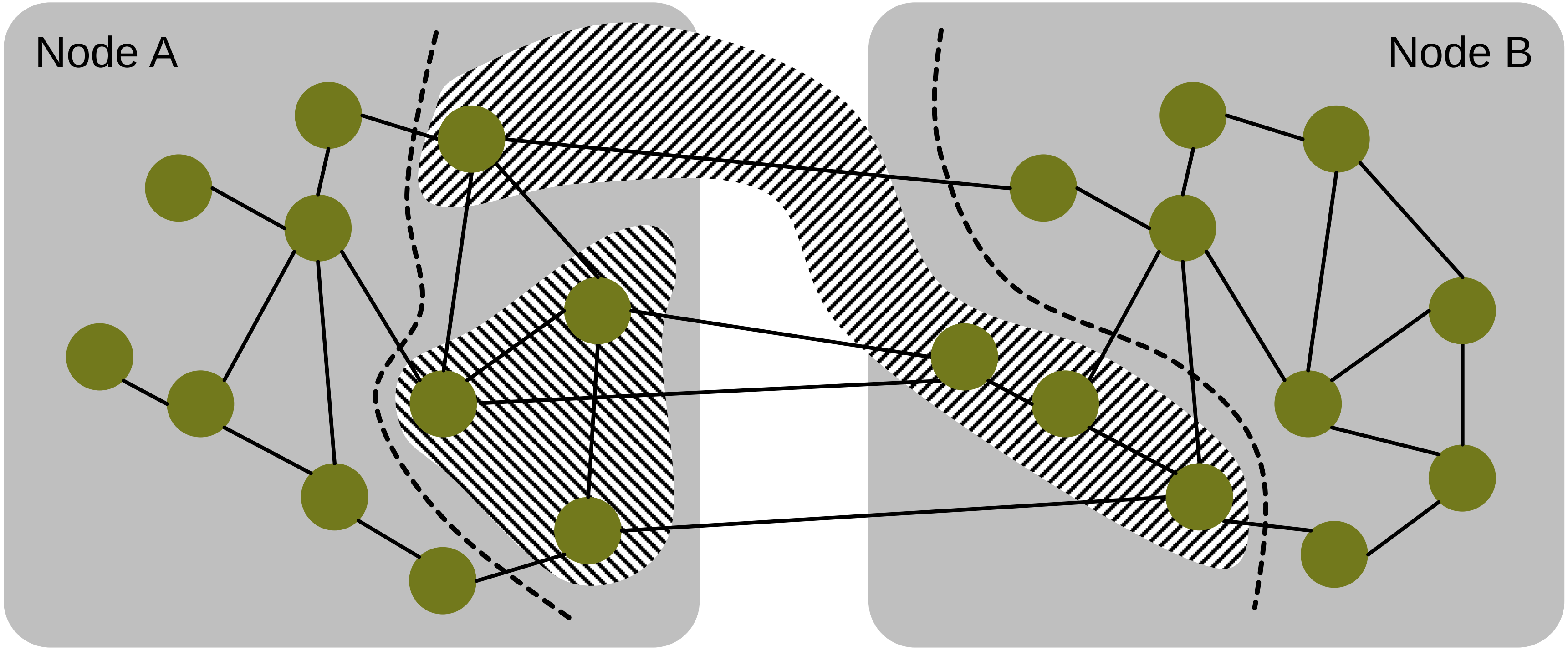
Executing Transactions on Multiple Nodes



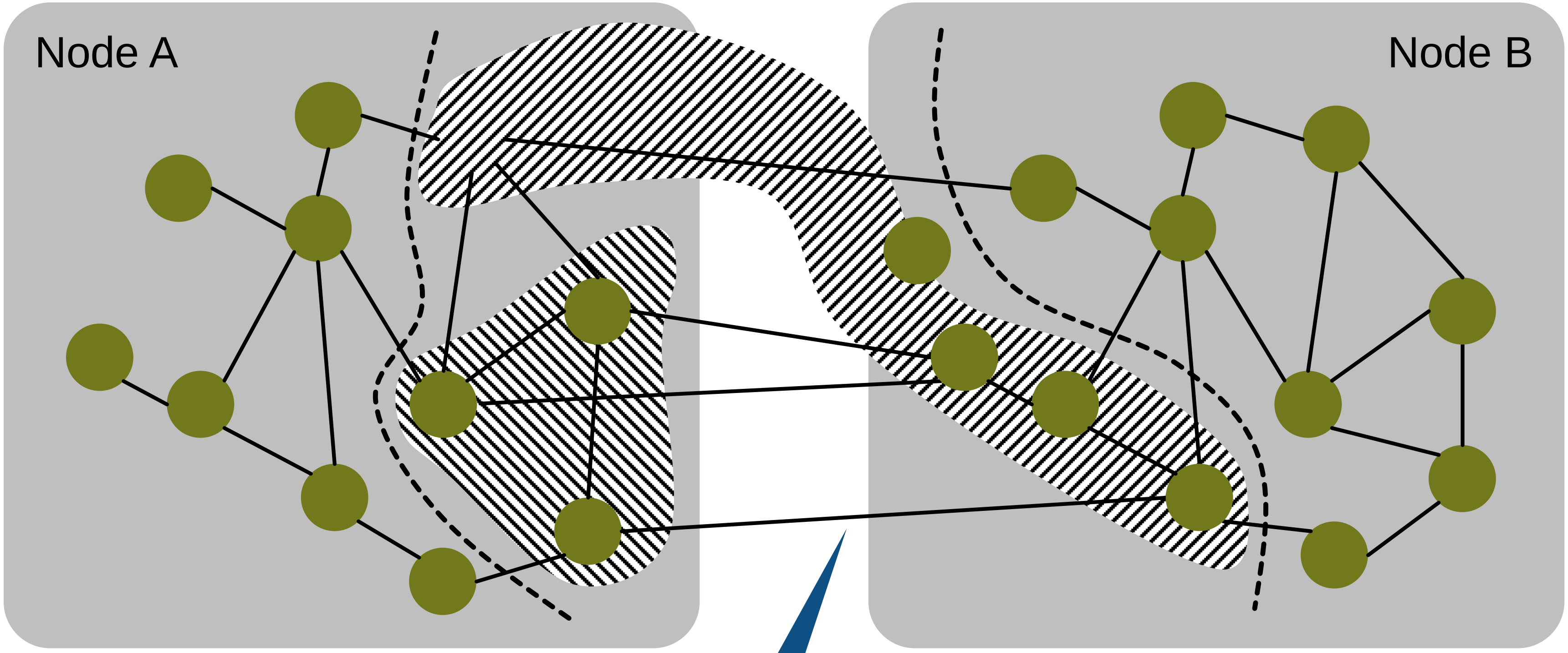
Executing Transactions on Multiple Nodes



Executing Transactions on Multiple Nodes

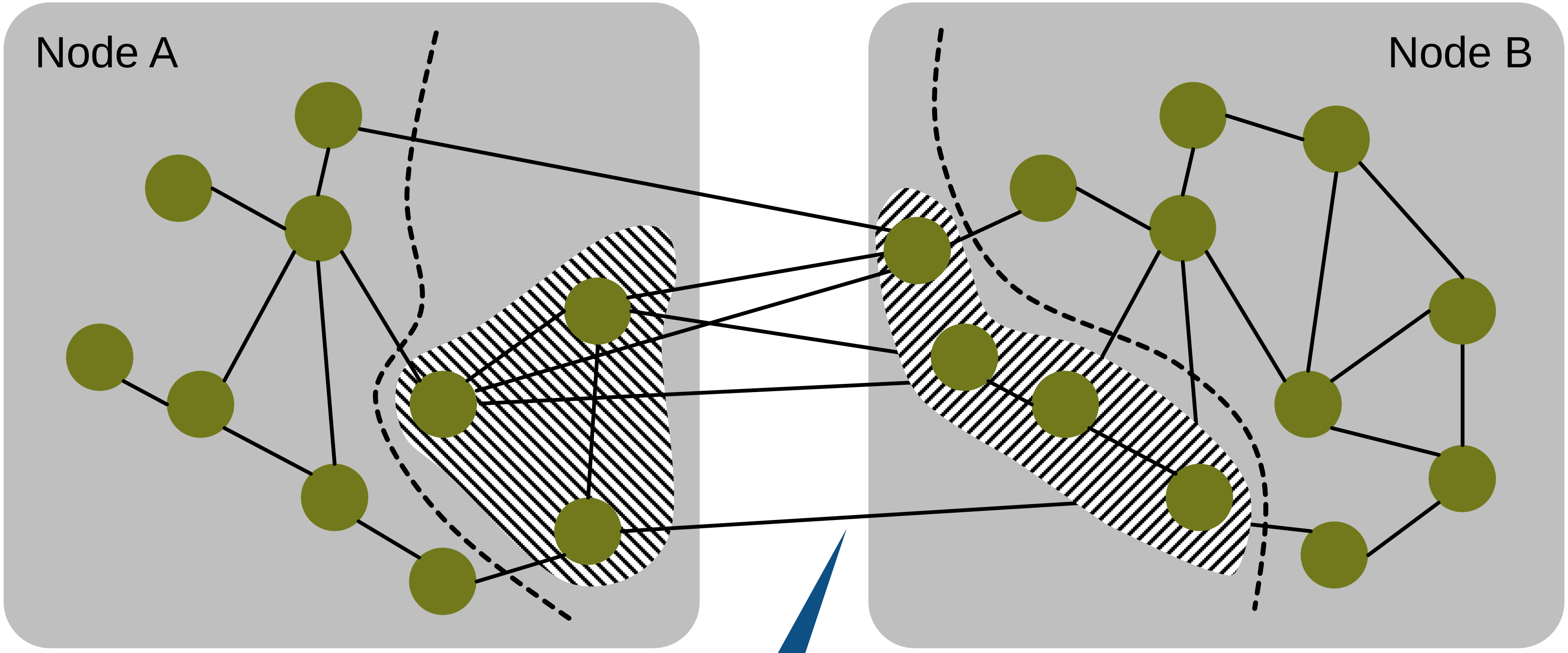


Executing Transactions on Multiple Nodes



Vertices must be appropriately relocated to enable execution of a hardware transaction

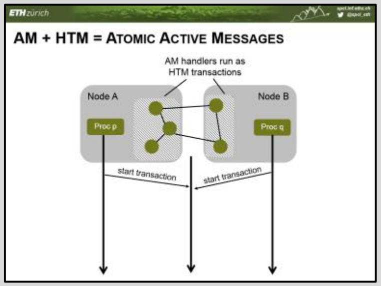
Executing Transactions on Multiple Nodes



Vertices must be appropriately relocated to enable execution of a hardware transaction

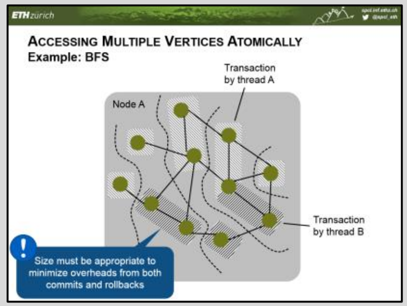
Overview of our Research

HTM for graphs in SM & DM environments

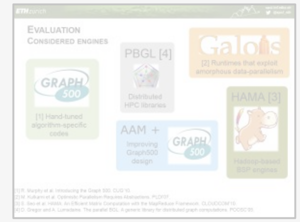


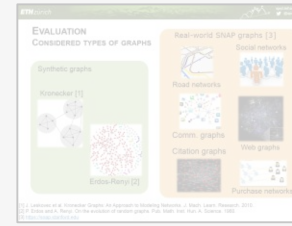
HTM + Active Messages
= Atomic Active Messages

Coarsening & coalescing

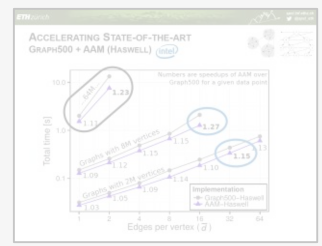


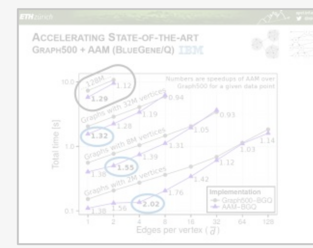
Evaluation



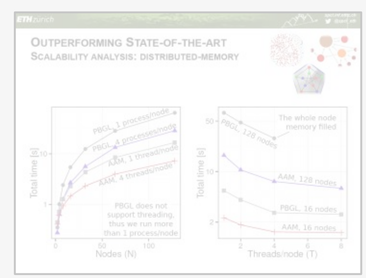


Considered engines and graphs





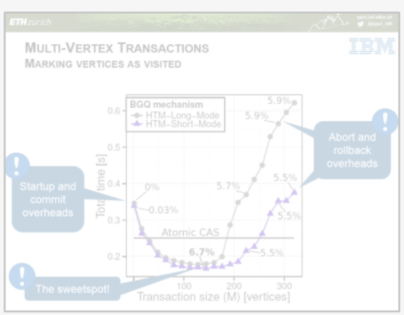
Accelerating state-of-the-art

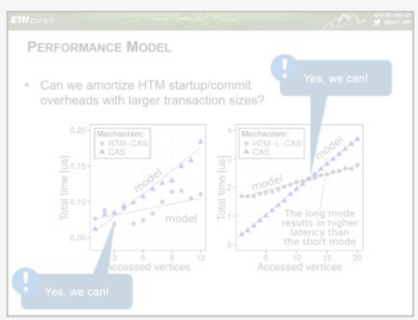


Scalability

Performance Modeling & Analysis

Haswell & BG/Q Analysis

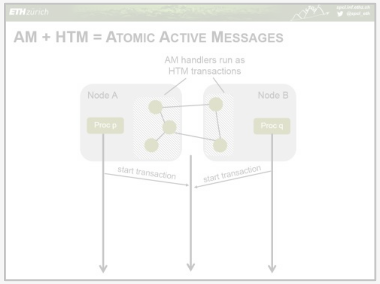




Performance model

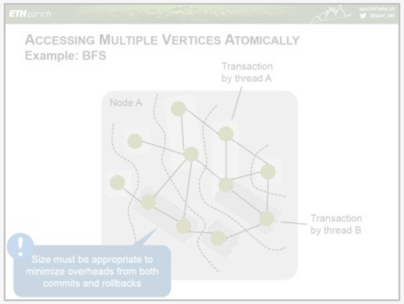
Overview of our Research

HTM for graphs in SM & DM environments

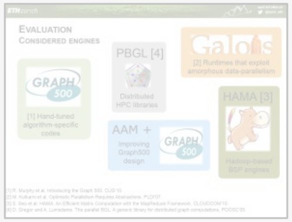


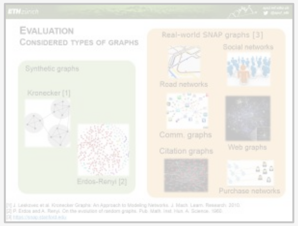
HTM + Active Messages = Atomic Active Messages

Coarsening & coalescing

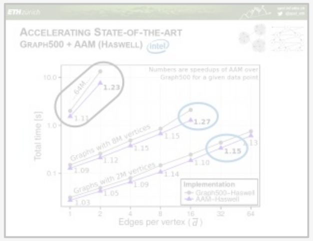


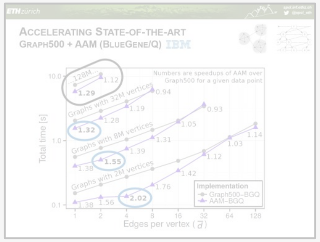
Evaluation



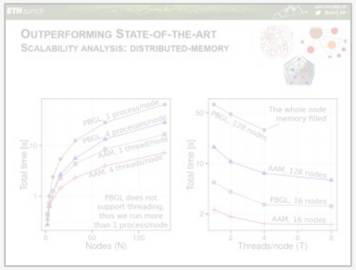


Considered engines and graphs





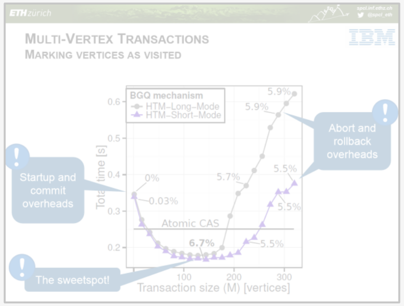
Accelerating state-of-the-art

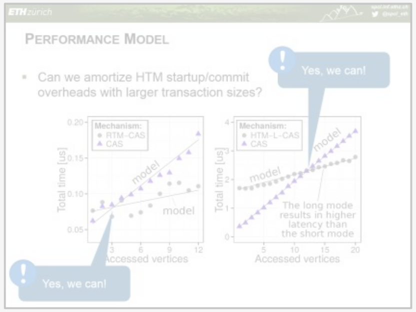


Scalability

Performance Modeling & Analysis

Haswell & BG/Q Analysis

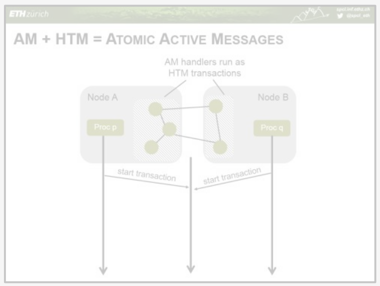




Performance model

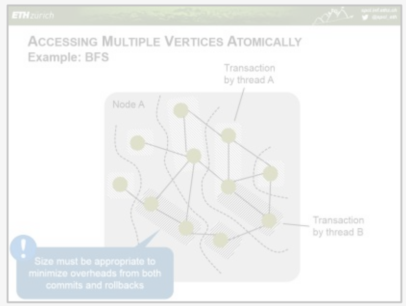
Overview of our Research

HTM for graphs in SM & DM environments

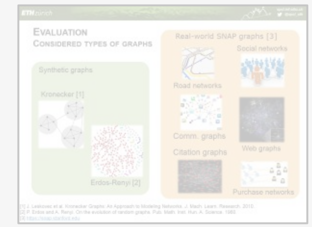


HTM + Active Messages
= Atomic Active Messages

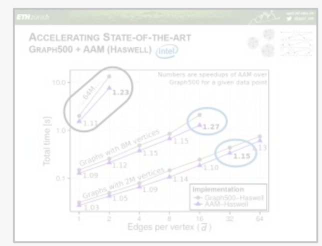
Coarsening & coalescing

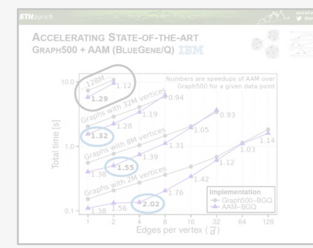


Evaluation

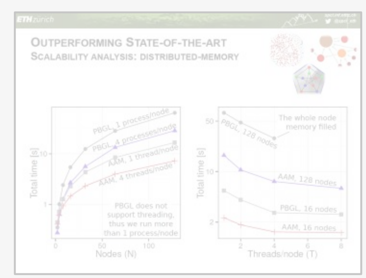


Considered engines and graphs





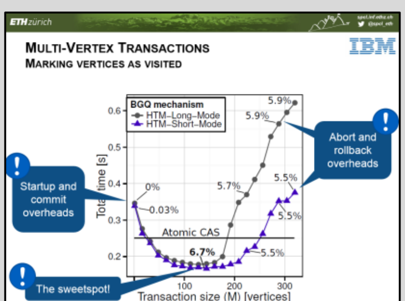
Accelerating state-of-the-art

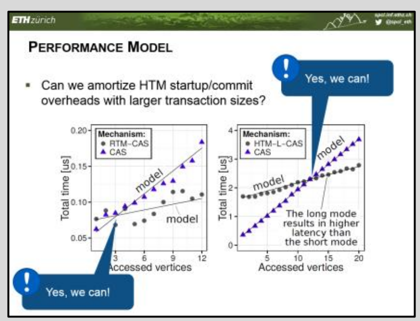


Scalability

Performance Modeling & Analysis

Haswell & BG/Q Analysis





Performance model

PERFORMANCE ANALYSIS

RESEARCH QUESTIONS

Performance Analysis
Research questions

How can we implement AAM handlers to run most efficiently?

Performance Analysis

Research questions

How can we implement AAM handlers to run most efficiently?

What are performance tradeoffs related to HTM?

Performance Analysis
Research questions

How can we implement AAM handlers to run most efficiently?

What are performance tradeoffs related to HTM?

What are advantages of HTM over atomics for AAM?

Performance Analysis
Research questions

How can we implement AAM handlers to run most efficiently?

What are performance tradeoffs related to HTM?

What are advantages of HTM over atomics for AAM?

What are the optimal transaction sizes? Can we amortize transaction overheads?

Performance Analysis

Types of machines

PERFORMANCE ANALYSIS
TYPES OF MACHINES

- Evaluation on 3 machines

PERFORMANCE ANALYSIS
TYPES OF MACHINES

- Evaluation on 3 machines
  - Intel Haswell server

Performance Analysis
Types of machines

- Evaluation on 3 machines
  - Intel Haswell server
  - InfiniBand cluster

HPC clusters

Commodity machines

Performance Analysis
Types of machines

- Evaluation on 3 machines
  - Intel Haswell server
  - InfiniBand cluster
  - IBM BlueGene/Q


HPC clusters

Commodity machines


Supercomputing machines

Performance Analysis

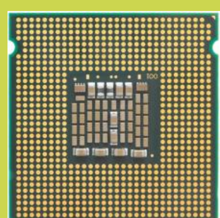
Considered mechanisms

Performance Analysis
Considered mechanisms

Haswell HTM

Performance Analysis
Considered mechanisms

Haswell HTM

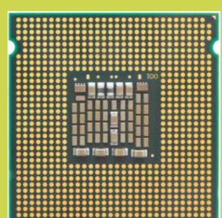


L1 L1

Deployed in L1

Performance Analysis
Considered mechanisms

Haswell HTM



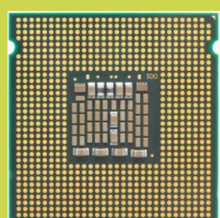
L1 L1

32KB per core

Deployed in L1

Performance Analysis
Considered mechanisms

Haswell HTM



L1 L1

32KB per core

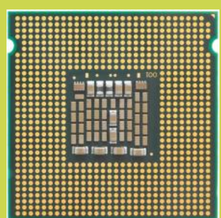
Deployed in L1

8-way associative

Performance Analysis

Considered mechanisms

Haswell HTM



L1 L1

32KB per core

Deployed in L1

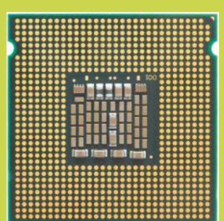
8-way associative

RTM (Restricted Transactional Memory)

HLE (Hardware Lock Elision)

Performance Analysis
Considered mechanisms

Haswell HTM



L1 L1

32KB per core

Deployed in L1

8-way associative

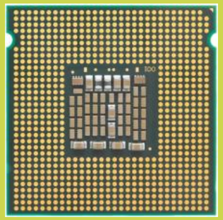
RTM (Restricted Transactional Memory)

HLE (Hardware Lock Elision)

BlueGene/Q HTM

Performance Analysis
Considered mechanisms

Haswell HTM



L1 L1

32KB per core

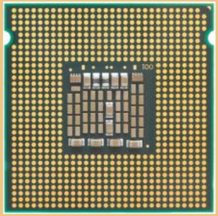
Deployed in L1

8-way associative

RTM (Restricted Transactional Memory)

HLE (Hardware Lock Elision)

BlueGene/Q HTM



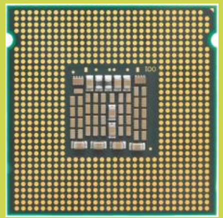
L1 L1

L2

Deployed in L2

Performance Analysis
Considered mechanisms

Haswell HTM



L1 L1

32KB per core

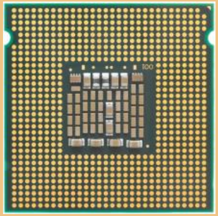
Deployed in L1

8-way associative

RTM (Restricted Transactional Memory)

HLE (Hardware Lock Elision)

BlueGene/Q HTM



L1 L1

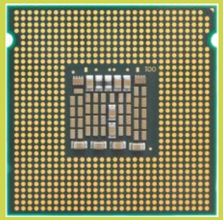
L2

2MB per core

Deployed in L2

Performance Analysis
Considered mechanisms

Haswell HTM



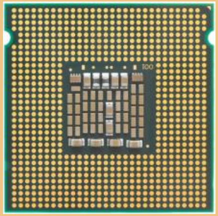
L1 | L1

- 32KB per core
- Deployed in L1
- 8-way associative

RTM (Restricted Transactional Memory)

HLE (Hardware Lock Elision)

BlueGene/Q HTM



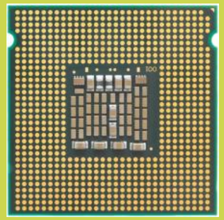
L1 | L1
L2

- 2MB per core
- Deployed in L2
- 16-way associative

Performance Analysis

Considered mechanisms

Haswell HTM



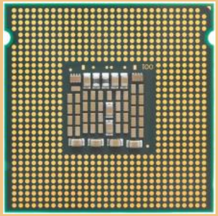
L1 L1

- 32KB per core
- Deployed in L1
- 8-way associative

RTM (Restricted Transactional Memory)

HLE (Hardware Lock Elision)

BlueGene/Q HTM



L1 L1

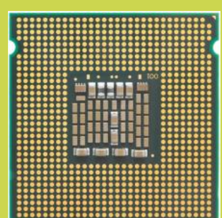
L2

- 2MB per core
- Deployed in L2
- 16-way associative

The Long Running Mode

Performance Analysis
Considered mechanisms

Haswell HTM



L1 L1

32KB per core

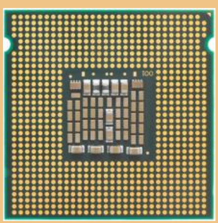
Deployed in L1

8-way associative

RTM (Restricted Transactional Memory)

HLE (Hardware Lock Elision)

BlueGene/Q HTM



L1 L1

L2

2MB per core

Deployed in L2

16-way associative

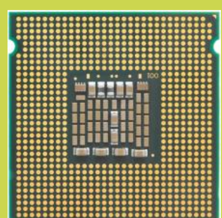
The Long Running Mode

The Short Running Mode

Performance Analysis
Considered mechanisms

Atomics

Haswell HTM



L1 L1

32KB per core

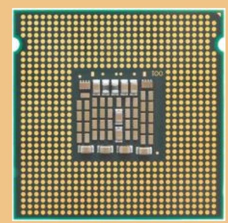
Deployed in L1

8-way associative

RTM (Restricted Transactional Memory)

HLE (Hardware Lock Elision)

BlueGene/Q HTM



L1 L1

L2

2MB per core

Deployed in L2

16-way associative

The Long Running Mode

The Short Running Mode

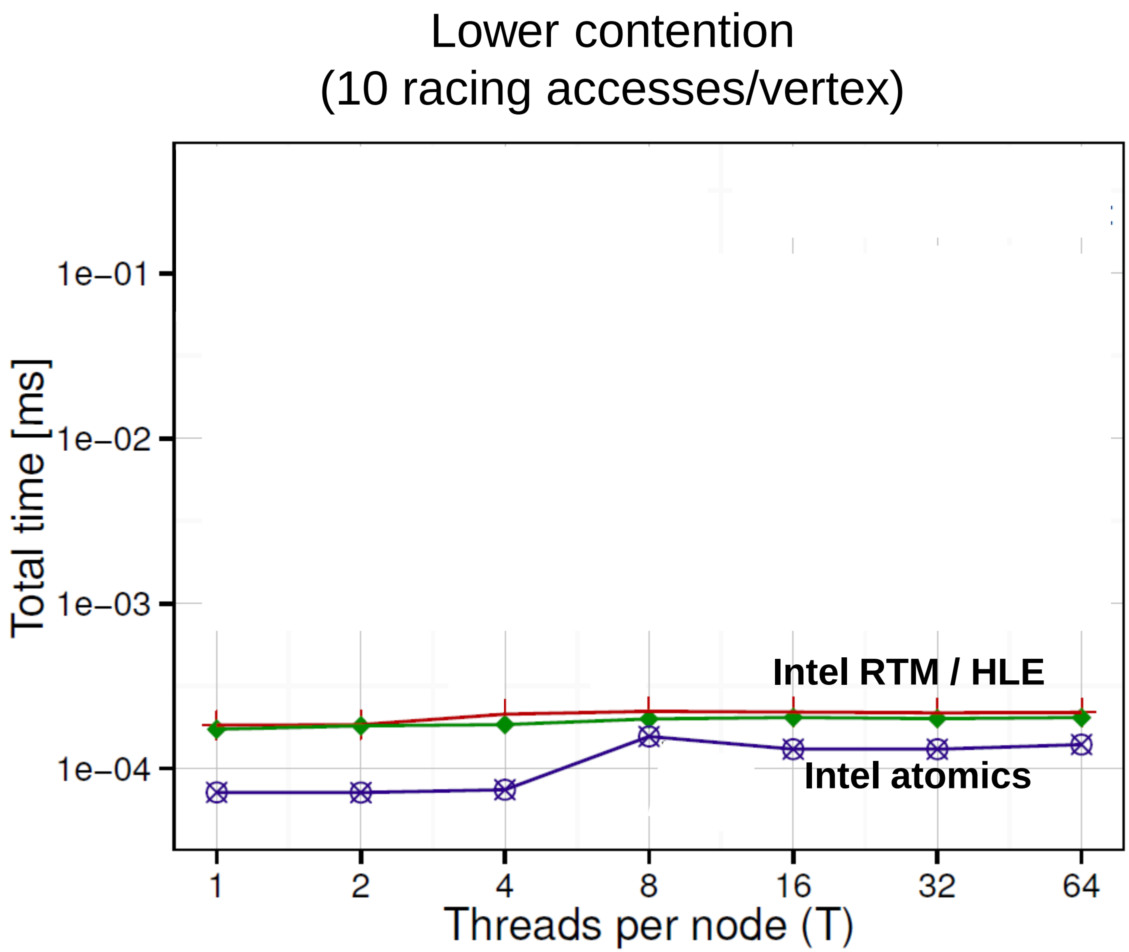
SINGLE-VERTEX TRANSACTIONS
MARKING A VERTEX AS VISITED

Used in BFS, SSSP, ...

```
// start handler
if(!v.visited) {
 v.visited = 1;
}
// finish handler
```

SINGLE-VERTEX TRANSACTIONS
MARKING A VERTEX AS VISITED

Used in BFS, SSSP, ...



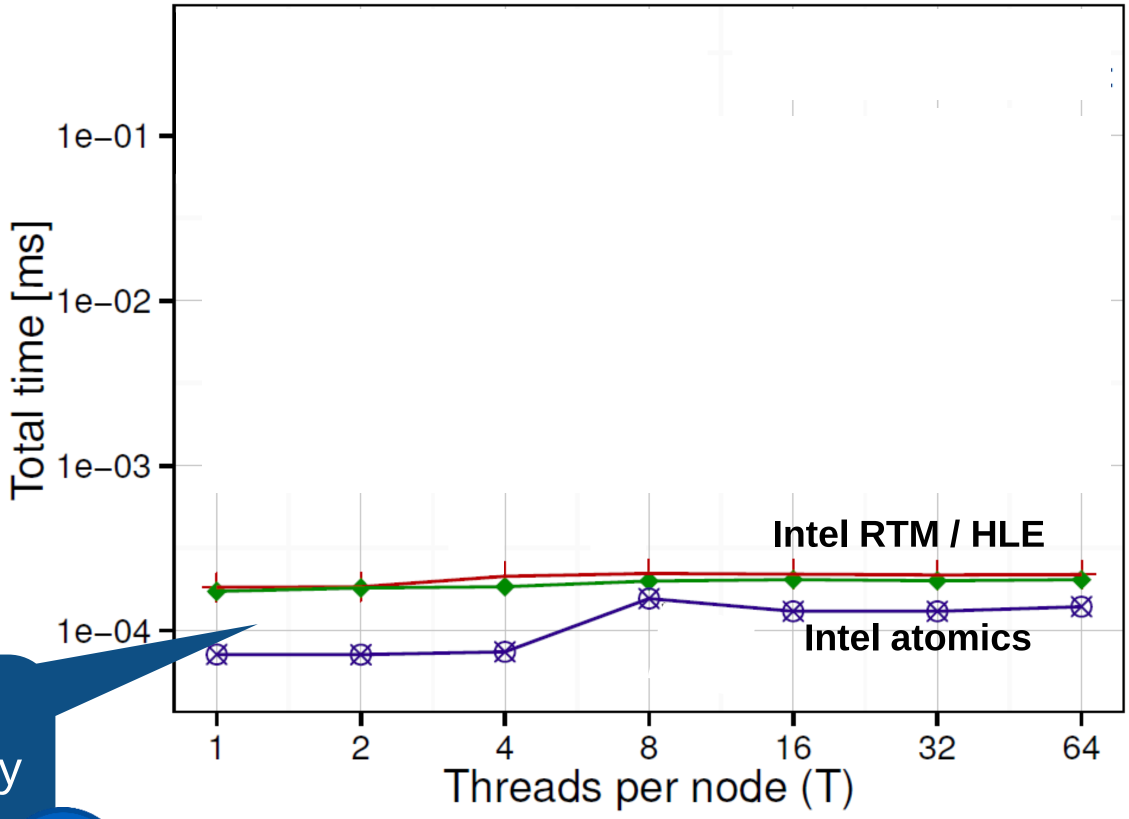

```
// start handler
if(!v.visited) {
 v.visited = 1;
}
// finish handler
```

Single-Vertex Transactions
Marking a vertex as visited

Used in BFS, SSSP, ...

```
// start handler
if(!v.visited) {
 v.visited = 1;
}
// finish handler
```

Lower contention
(10 racing accesses/vertex)

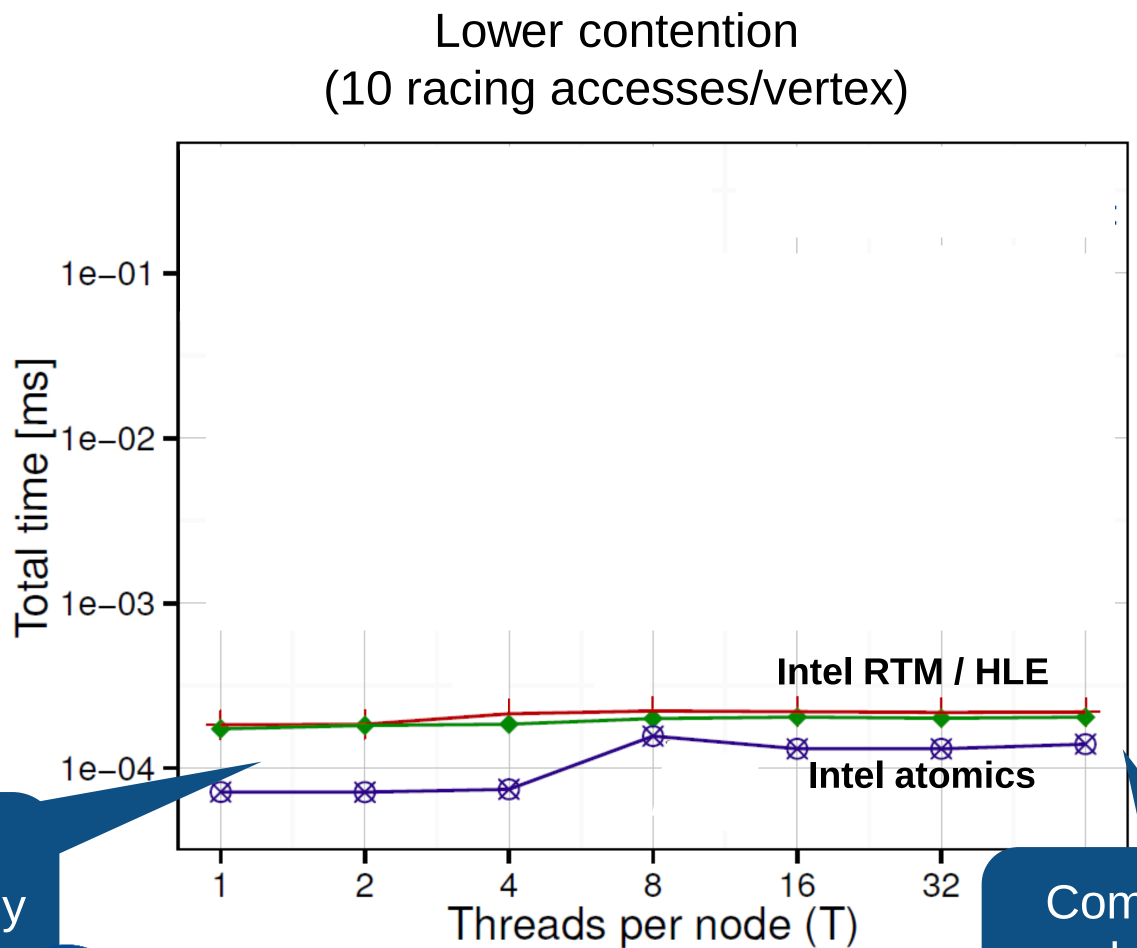


Atomics (CAS) slightly faster than HTM

Single-Vertex Transactions
Marking a vertex as visited

Used in BFS, SSSP, ...

```
// start handler
if(!v.visited) {
 v.visited = 1;
}
// finish handler
```



Atomics (CAS) slightly faster than HTM

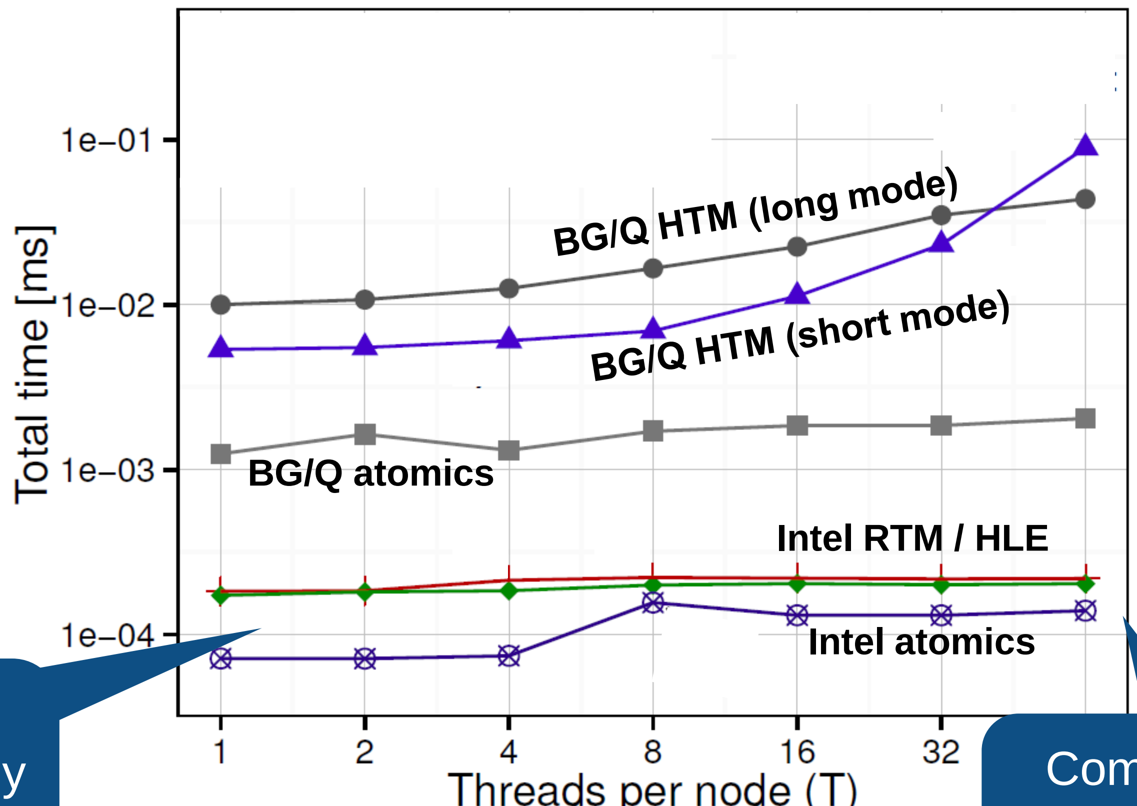
Commit overheads dominate

SINGLE-VERTEX TRANSACTIONS
MARKING A VERTEX AS VISITED

Used in BFS, SSSP, ...

```
// start handler
if(!v.visited) {
 v.visited = 1;
}
// finish handler
```

Lower contention
(10 racing accesses/vertex)



Atomics (CAS) slightly faster than HTM

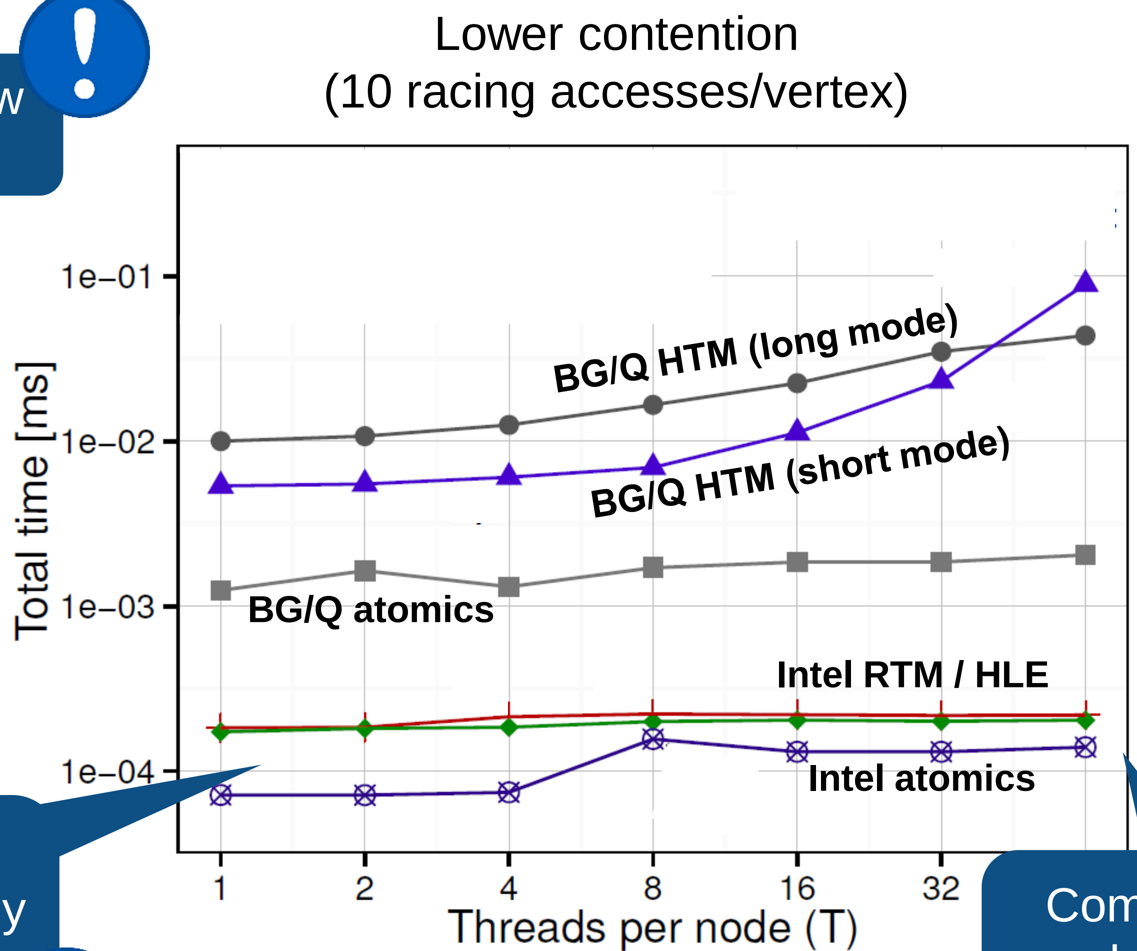
Commit overheads dominate

Single-Vertex Transactions
Marking a vertex as visited

Used in BFS, SSSP, ...

```
// start handler
if(!v.visited) {
 v.visited = 1;
}
// finish handler
```

Very few aborts



Atomics (CAS) slightly faster than HTM

Commit overheads dominate

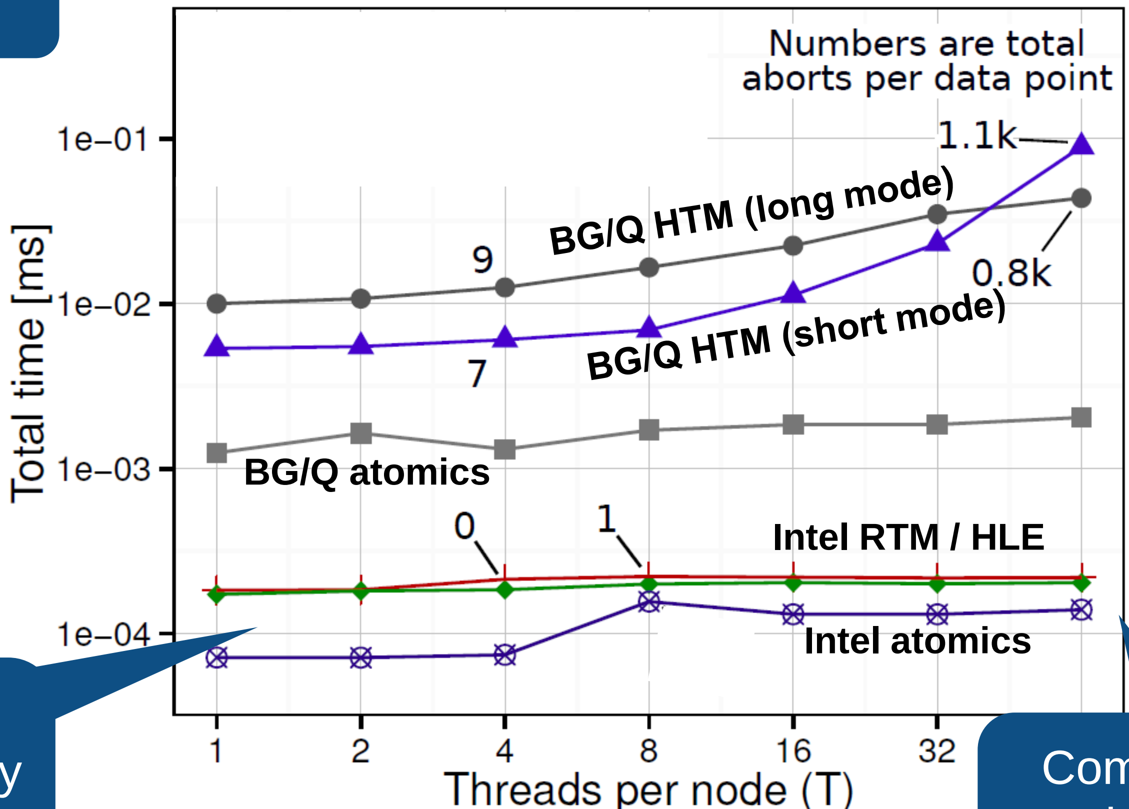
Single-Vertex Transactions
Marking a vertex as visited

Used in BFS, SSSP, ...

```
// start handler
if(!v.visited) {
 v.visited = 1;
}
// finish handler
```

Very few aborts

Lower contention
(10 racing accesses/vertex)



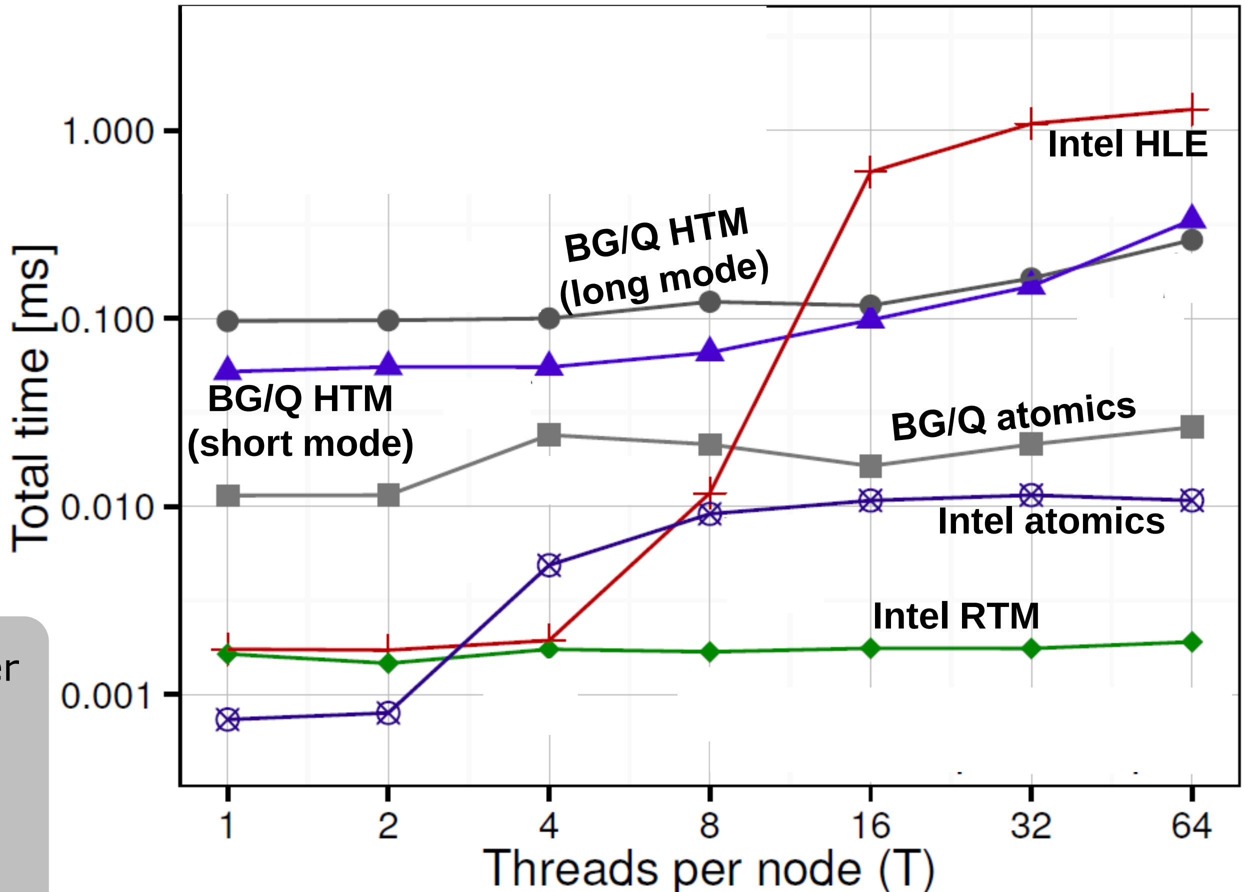
Atomics (CAS) slightly faster than HTM

Commit overheads dominate

SINGLE-VERTEX TRANSACTIONS
MARKING A VERTEX AS VISITED

Used in BFS, SSSP, ...

Higher contention
(100 racing accesses/vertex)

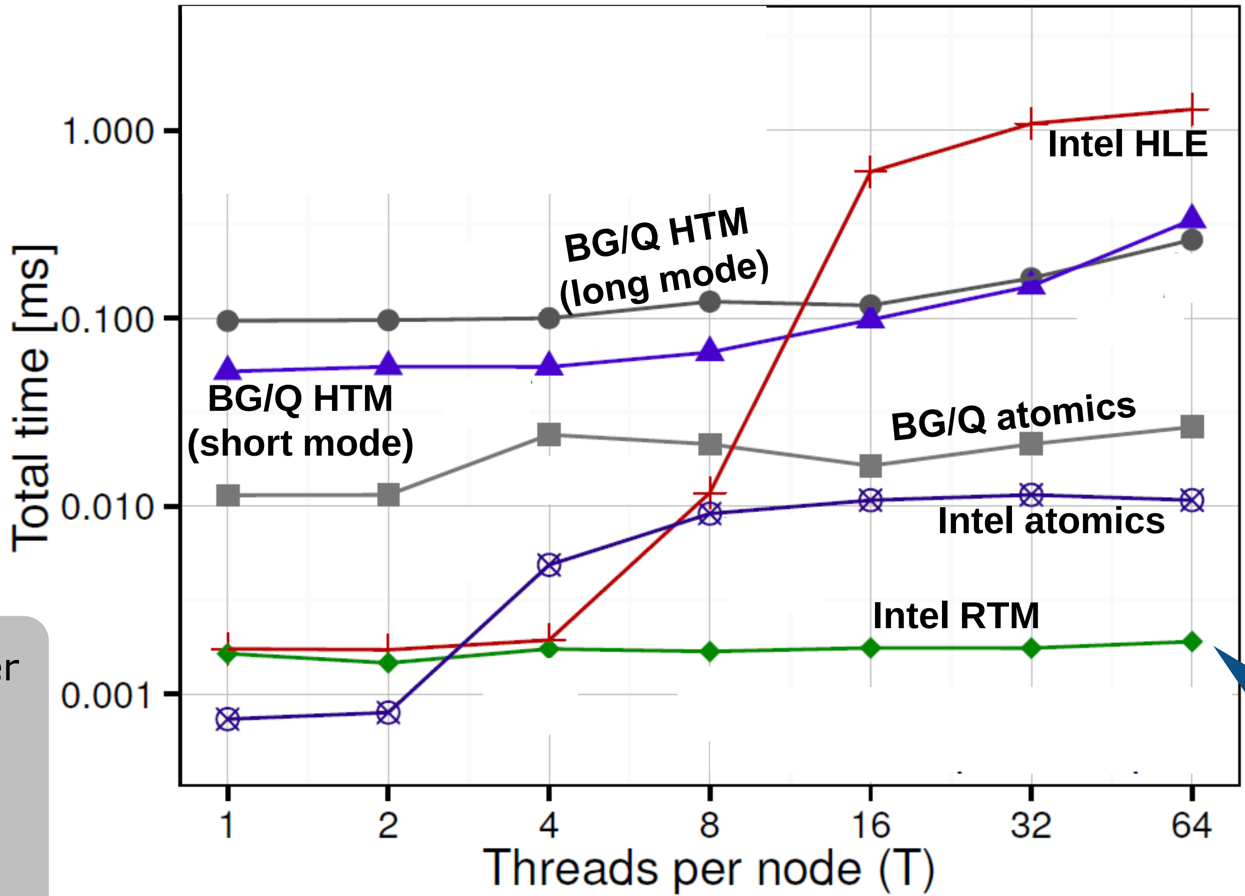



```
// start handler
if(!v.visited) {
 v.visited = 1;
}
// finish
handler
```

Single-Vertex Transactions
Marking a vertex as visited

Used in BFS, SSSP, ...

Higher contention
(100 racing accesses/vertex)



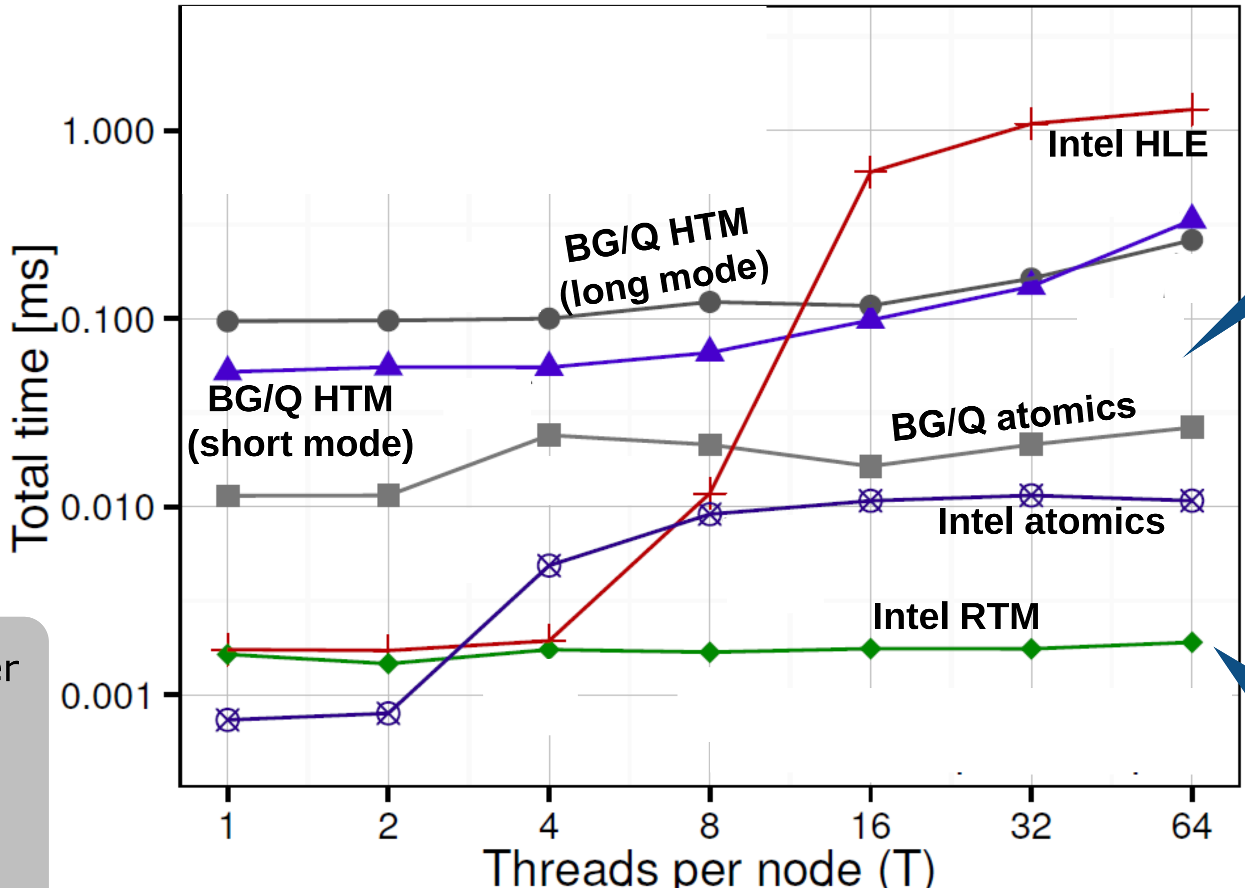

```
// start handler
if(!v.visited) {
 v.visited = 1;
}
// finish
handler
```

RTM better than atomics

Single-Vertex Transactions
Marking a vertex as visited

Used in BFS, SSSP, ...

Higher contention
(100 racing accesses/vertex)




```
// start handler
if(!v.visited) {
 v.visited = 1;
}
// finish
handler
```

BG/Q HTM still worse (L1 vs L2 matters!)

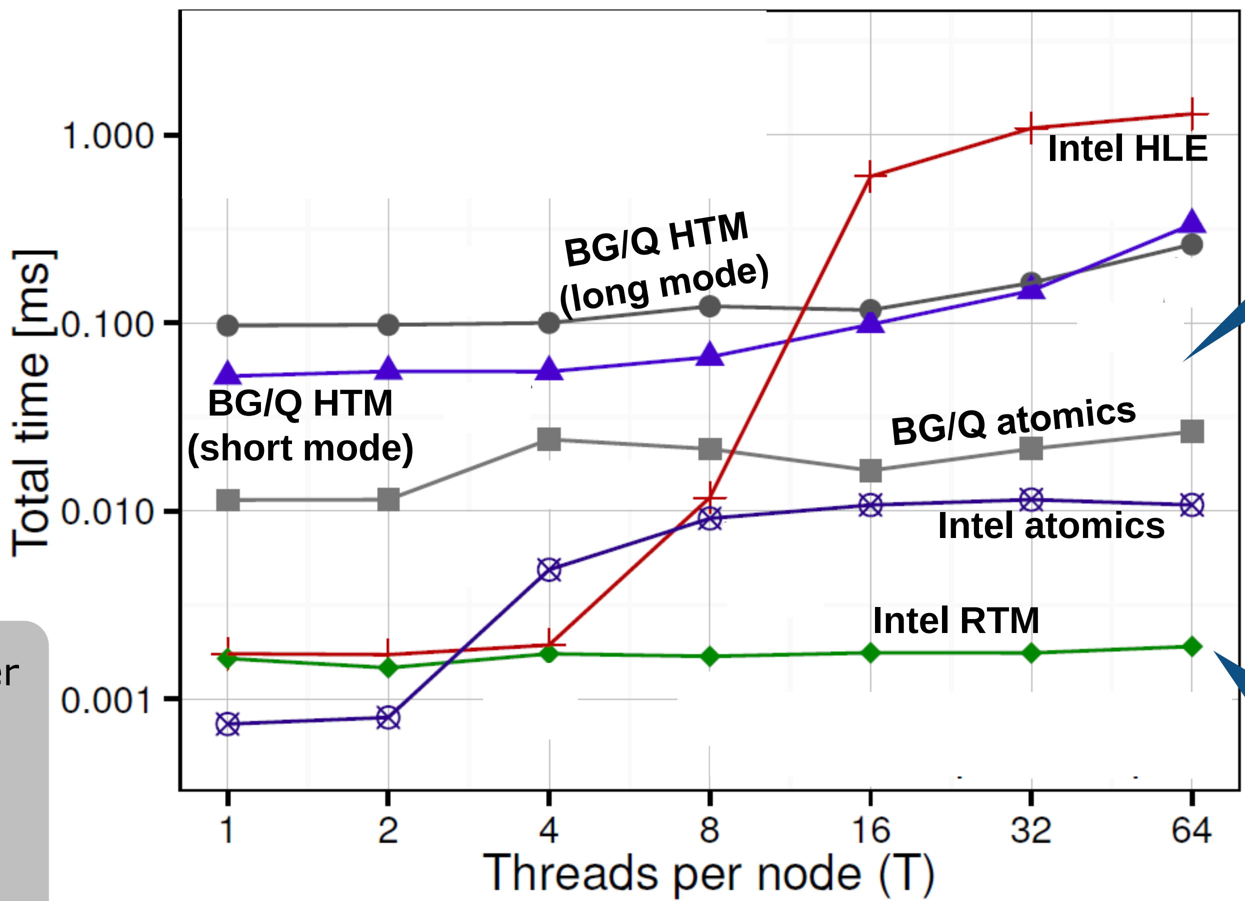
RTM better than atomics

Single-Vertex Transactions
Marking a vertex as visited

Used in BFS, SSSP, ...

Still very few aborts

Higher contention
(100 racing accesses/vertex)



BG/Q HTM still worse (L1 vs L2 matters!)

```
// start handler
if(!v.visited) {
 v.visited = 1;
}
// finish handler
```

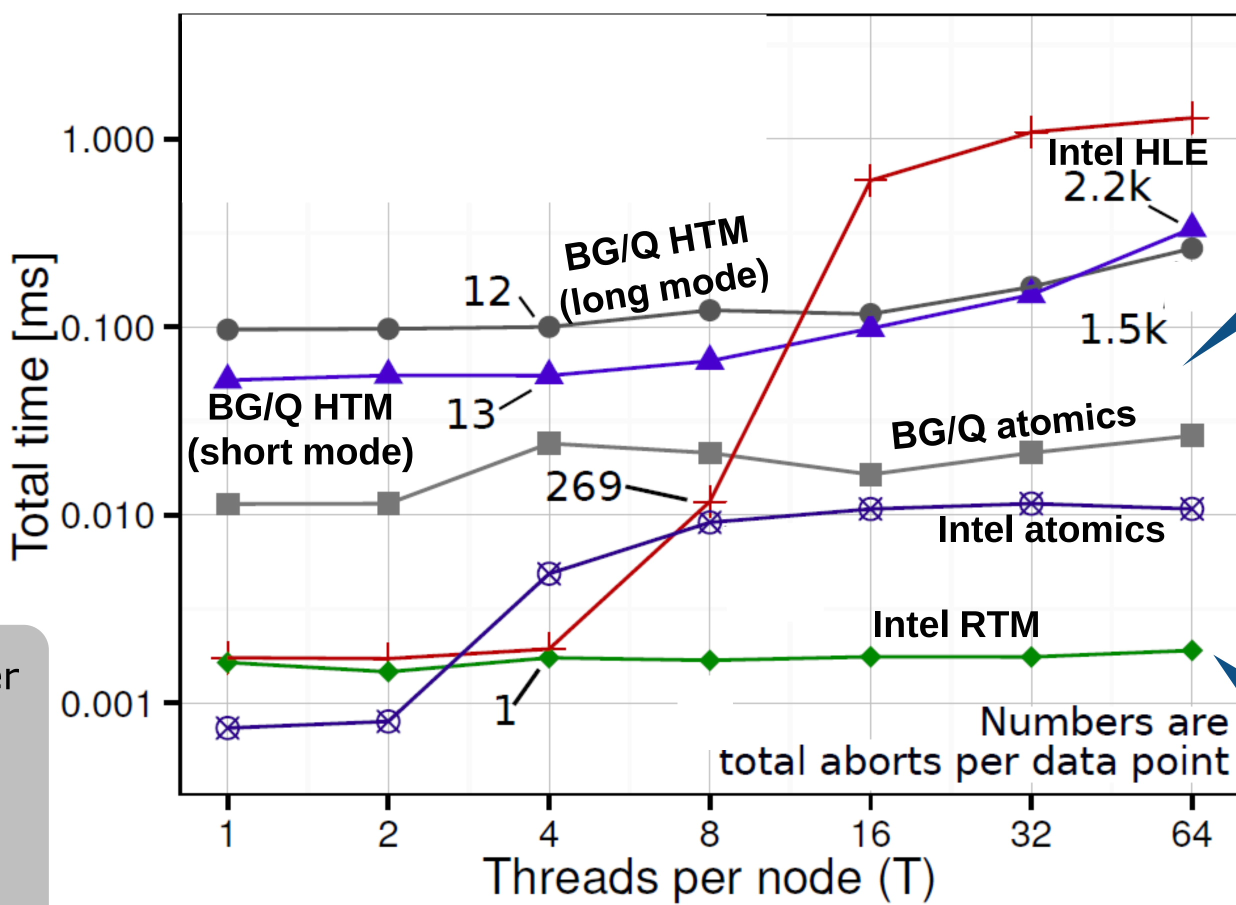
RTM better than atomics

Single-Vertex Transactions
Marking a vertex as visited

Used in BFS, SSSP, ...

Still very few aborts

Higher contention
(100 racing accesses/vertex)



BG/Q HTM still worse (L1 vs L2 matters!)

```
// start handler
if(!v.visited) {
 v.visited = 1;
}
// finish
handler
```

RTM better than atomics

SINGLE-VERTEX TRANSACTIONS

INCREMENTING VERTEX RANK

Used in PageRank

```
// start handler
v.rank++;
// finish handler
```

Single-Vertex Transactions
Incrementing vertex rank

Used in PageRank

```
// start handler
v.rank++;
// finish handler
```

Atomics always outperform HTM

SINGLE-VERTEX TRANSACTIONS

INCREMENTING VERTEX RANK

Used in PageRank

```
// start handler
v.rank++;
// finish handler
```

Atomics always outperform HTM

The reason: each transaction always modifies some memory cell, increasing the number of conflicts

Performance Model

Atomics vs transactions

Performance Model

Atomics vs transactions

Time to modify *N* vertices with atomics:

$$T_{AT}(N) = A_{AT}N + B_{AT}$$

Performance Model

Atomics vs transactions

Time to modify *N* vertices with atomics:

$$T_{AT}(N) = A_{AT}N + B_{AT}$$

Startup overheads

Performance Model

Atomics vs transactions

Time to modify *N* vertices with atomics:

$$T_{AT}(N) = A_{AT}N + B_{AT}$$

Overhead per vertex

Startup overheads

Performance Model
Atomics vs transactions

Time to modify *N* vertices with atomics:

$$T_{AT}(N) = A_{AT}N + B_{AT}$$

Overhead per vertex

Startup overheads

Time to modify *N* vertices with a transaction

$$T_{HTM}(N) = A_{HTM}N + B_{HTM}$$

Performance Model

Atomics vs transactions

Time to modify *N* vertices with atomics:

$$T_{AT}(N) = A_{AT}N + B_{AT}$$

Overhead per vertex

Startup overheads

Time to modify *N* vertices with a transaction

$$T_{HTM}(N) = A_{HTM}N + B_{HTM}$$

Overhead per vertex

Startup overheads

Performance Model
Atomics vs transactions

Time to modify *N* vertices with atomics:

$$T_{AT}(N) = A_{AT}N + B_{AT}$$

Overhead per vertex

Startup overheads

Time to modify *N* vertices with a transaction

$$T_{HTM}(N) = A_{HTM}N + B_{HTM}$$

Overhead per vertex

Startup overheads

We predict that:

$$B_{AT} < B_{HTM}$$

$$A_{AT} > A_{HTM}$$

Performance Model
Atomics vs transactions

Time to modify *N* vertices with atomics:

$$T_{AT}(N) = A_{AT}N + B_{AT}$$

Overhead per vertex

Startup overheads

Time to modify *N* vertices with a transaction

$$T_{HTM}(N) = A_{HTM}N + B_{HTM}$$

Overhead per vertex

Startup overheads

We predict that:

$$B_{AT} < B_{HTM}$$

$$A_{AT} > A_{HTM}$$

Transaction startup overheads dominate

PERFORMANCE MODEL
ATOMICS VS TRANSACTIONS

Time to modify *N* vertices with atomics:

$$T_{AT}(N) = A_{AT}N + B_{AT}$$

Overhead per vertex

Startup overheads

Time to modify *N* vertices with a transaction

$$T_{HTM}(N) = A_{HTM}N + B_{HTM}$$

Overhead per vertex

Startup overheads

We predict that:

$$B_{AT} < B_{HTM}$$

$$A_{AT} > A_{HTM}$$

Transactions' cost grows slower

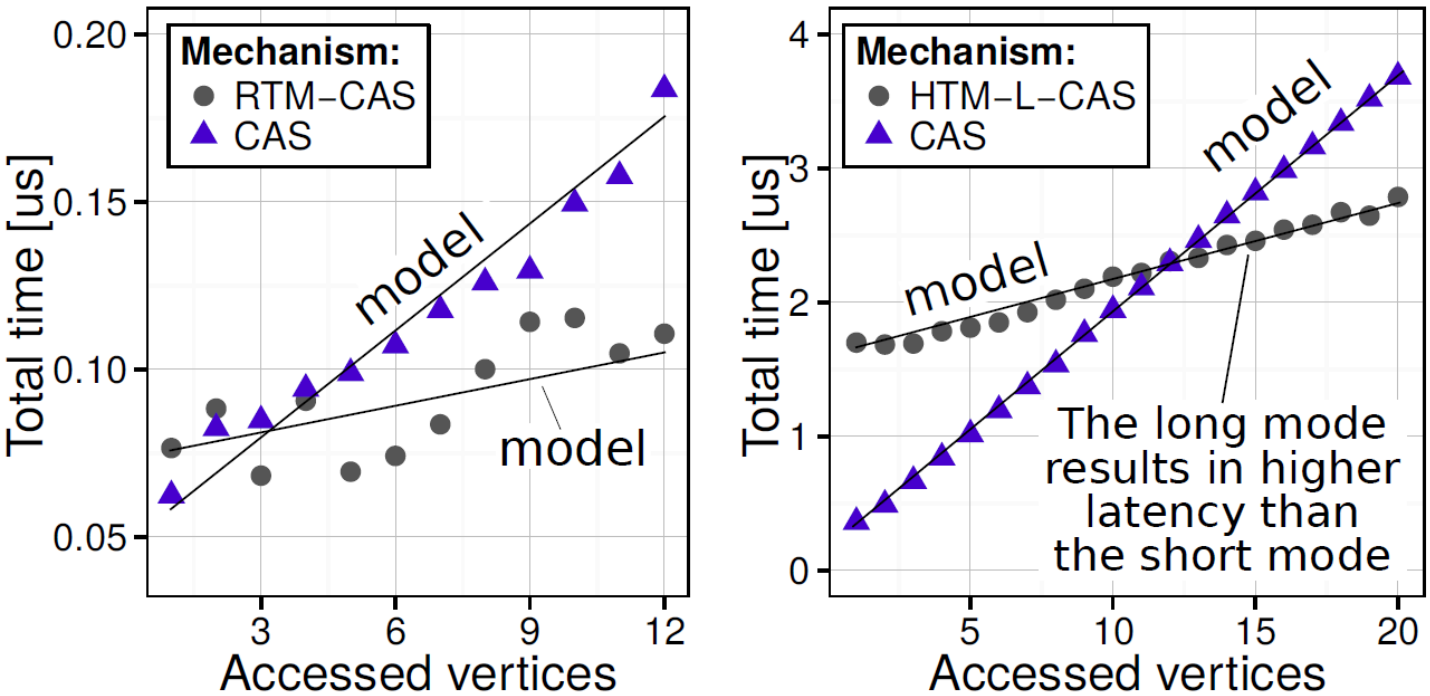
Transaction startup overheads dominate

Performance Model

Atomics vs transactions

Performance Model

Atomics vs transactions

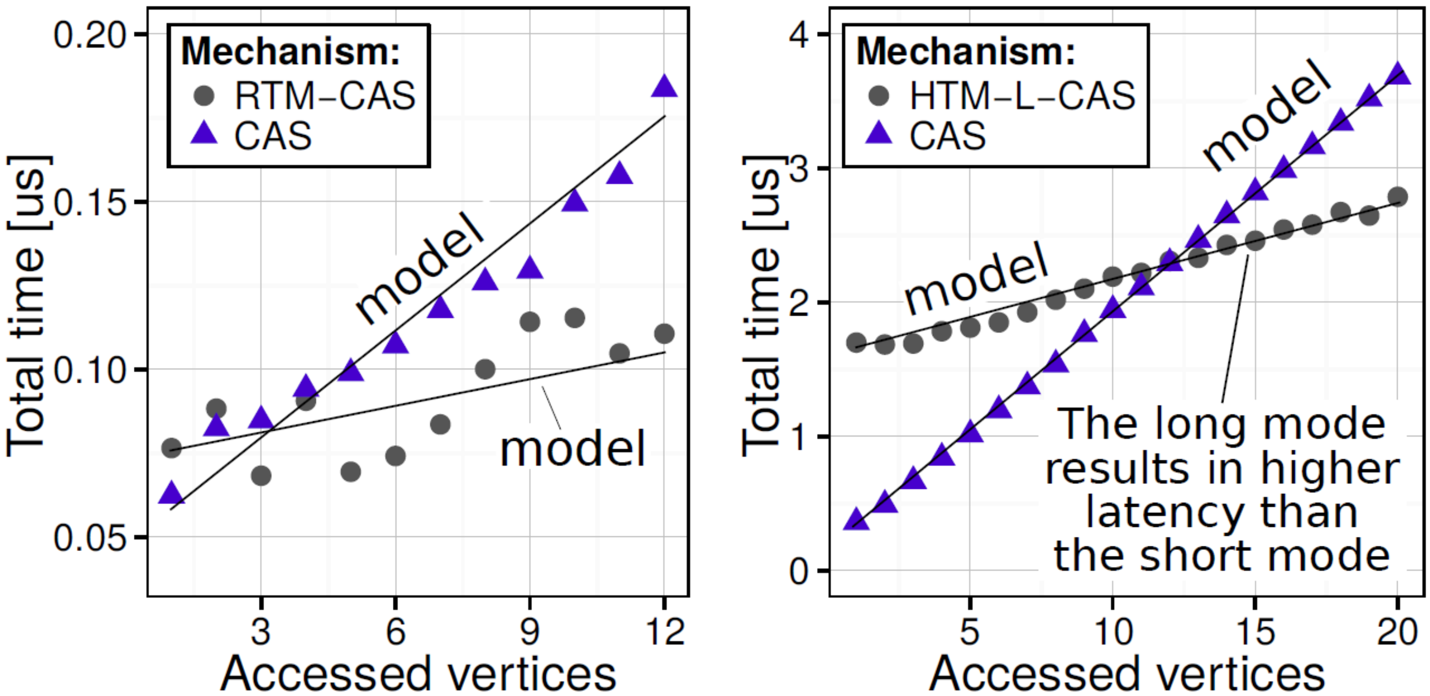


Performance Model
Atomics vs transactions

Indeed:

$$B_{AT} < B_{HTM}$$

$$A_{AT} > A_{HTM}$$



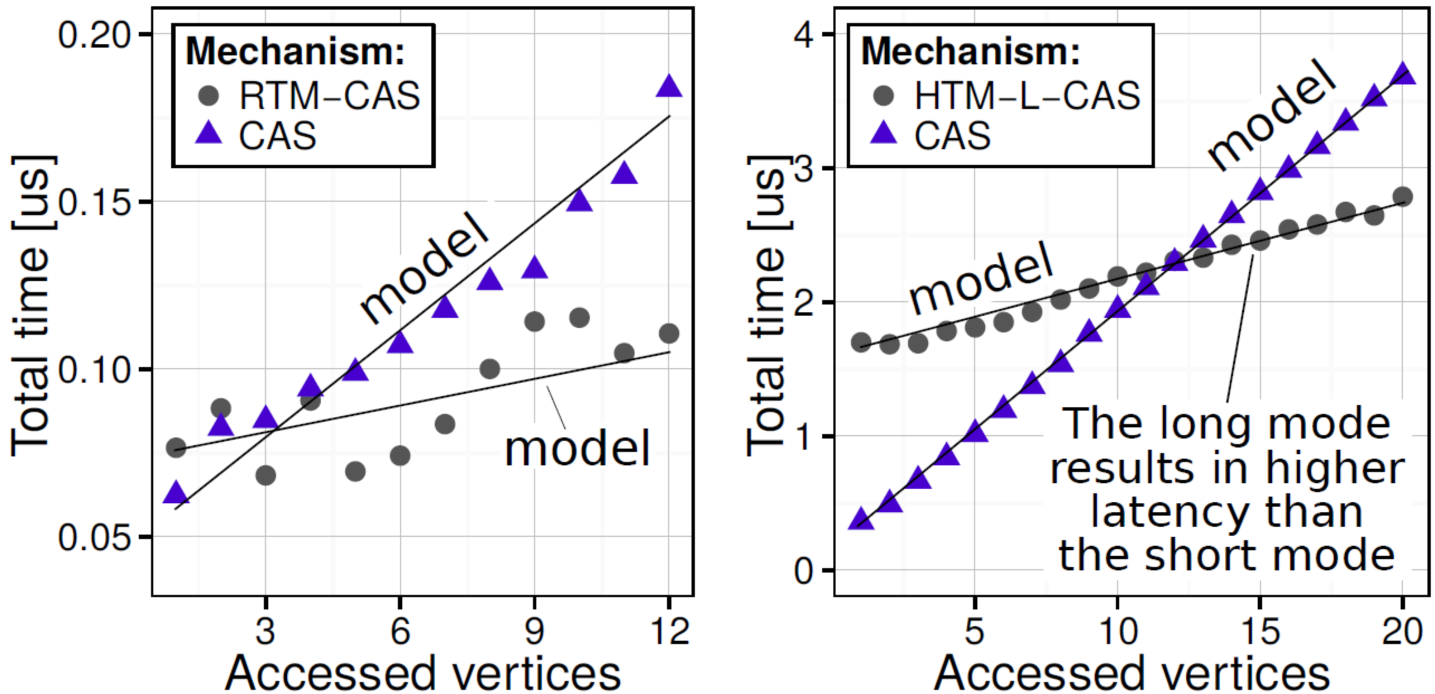
PERFORMANCE MODEL
ATOMICS VS TRANSACTIONS

- Can we amortize HTM startup/commit overheads with larger transaction sizes?

Indeed:

$$B_{AT} < B_{HTM}$$

$$A_{AT} > A_{HTM}$$



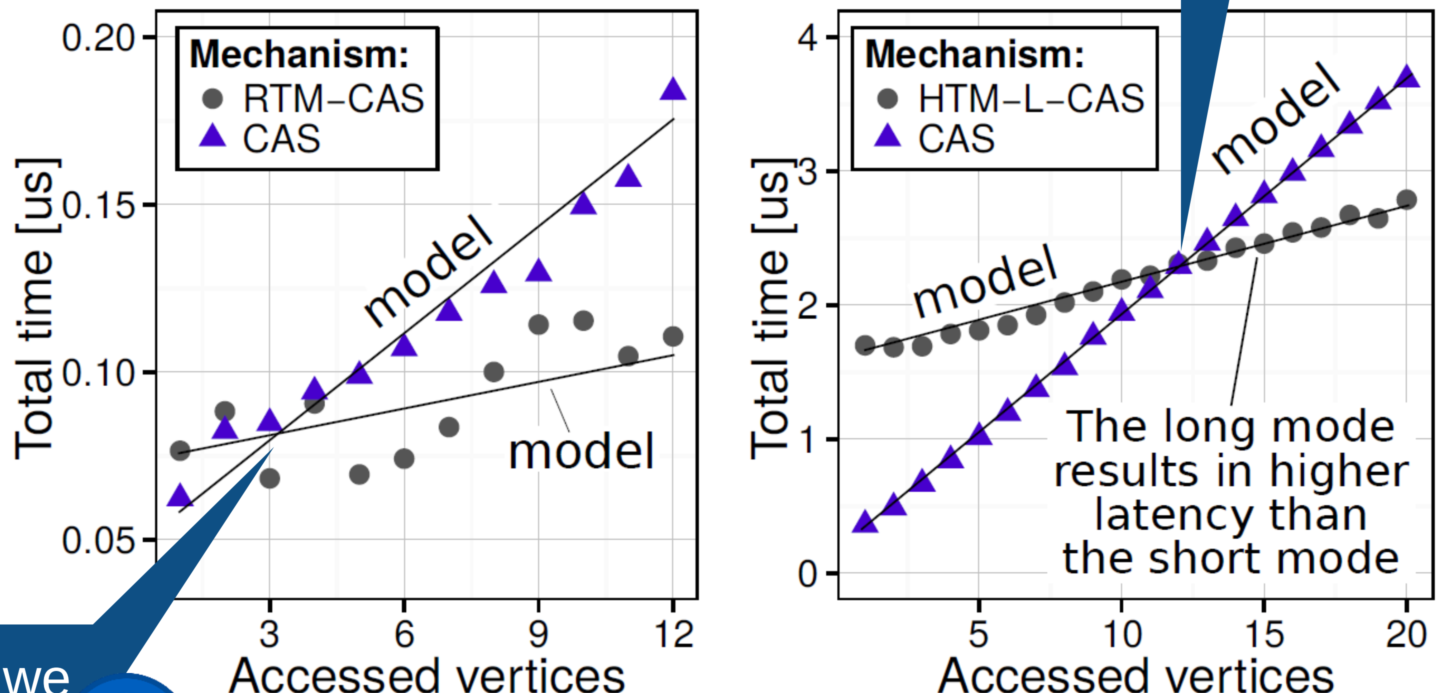
Performance Model
Atomics vs transactions

- Can we amortize HTM startup/commit overheads with larger transaction sizes?

Indeed:

$$B_{AT} < B_{HTM}$$
$$A_{AT} > A_{HTM}$$

Yes, we can!

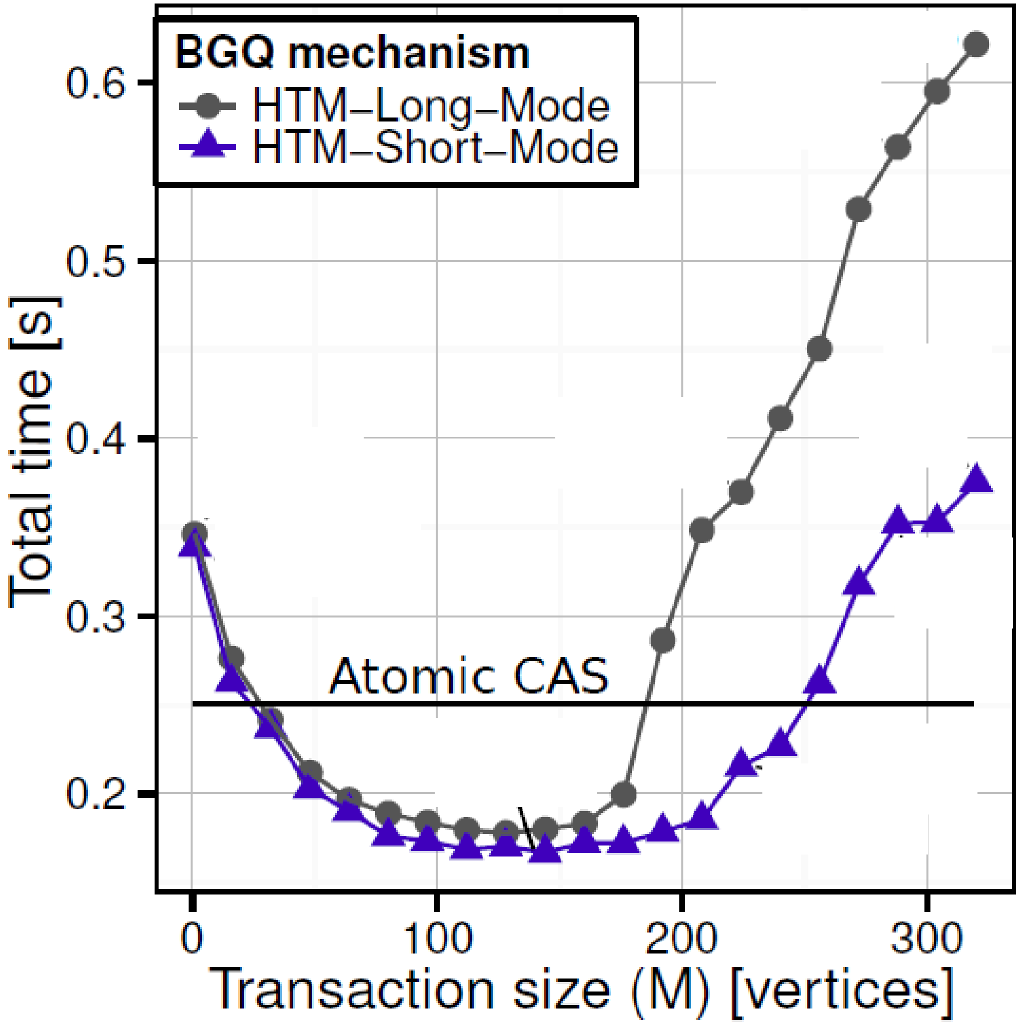


Yes, we can!

Multi-Vertex Transactions

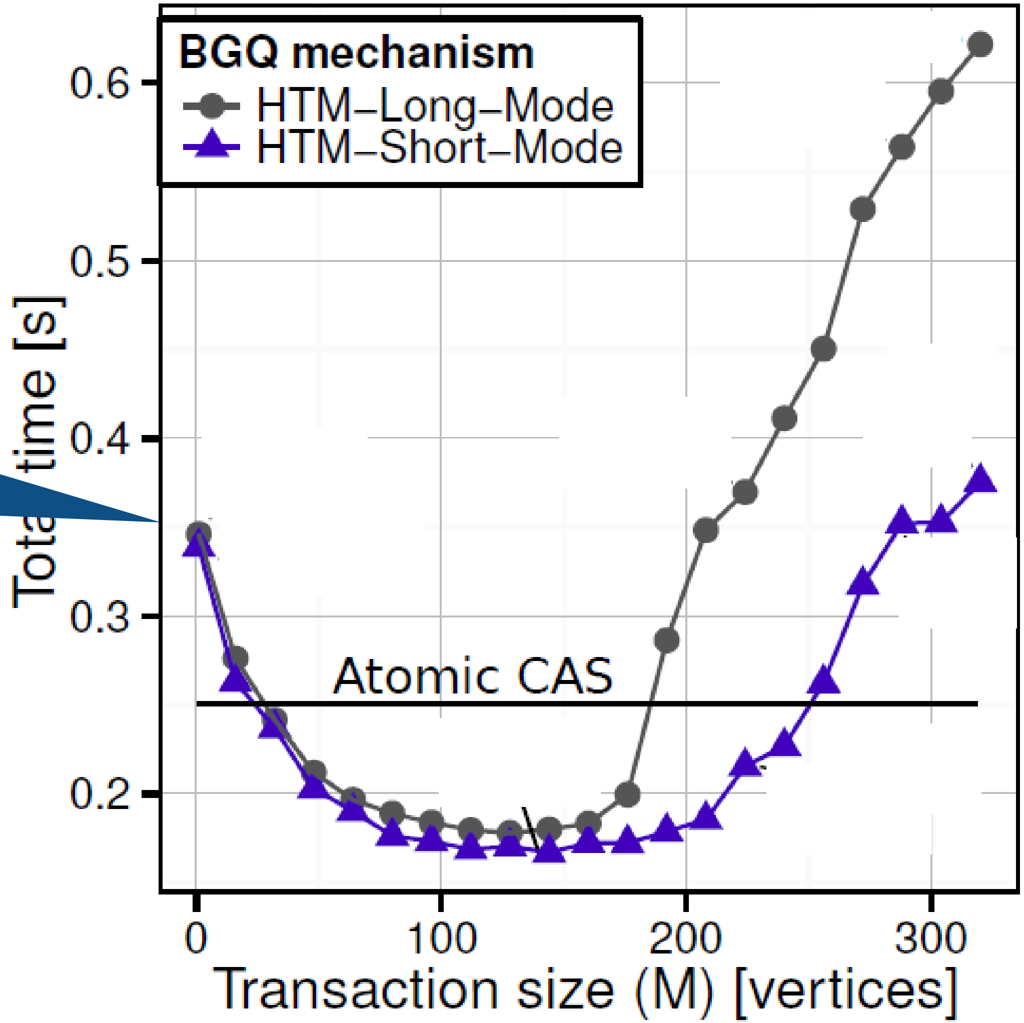
Marking vertices as visited

MULTI-VERTEX TRANSACTIONS
MARKING VERTICES AS VISITED



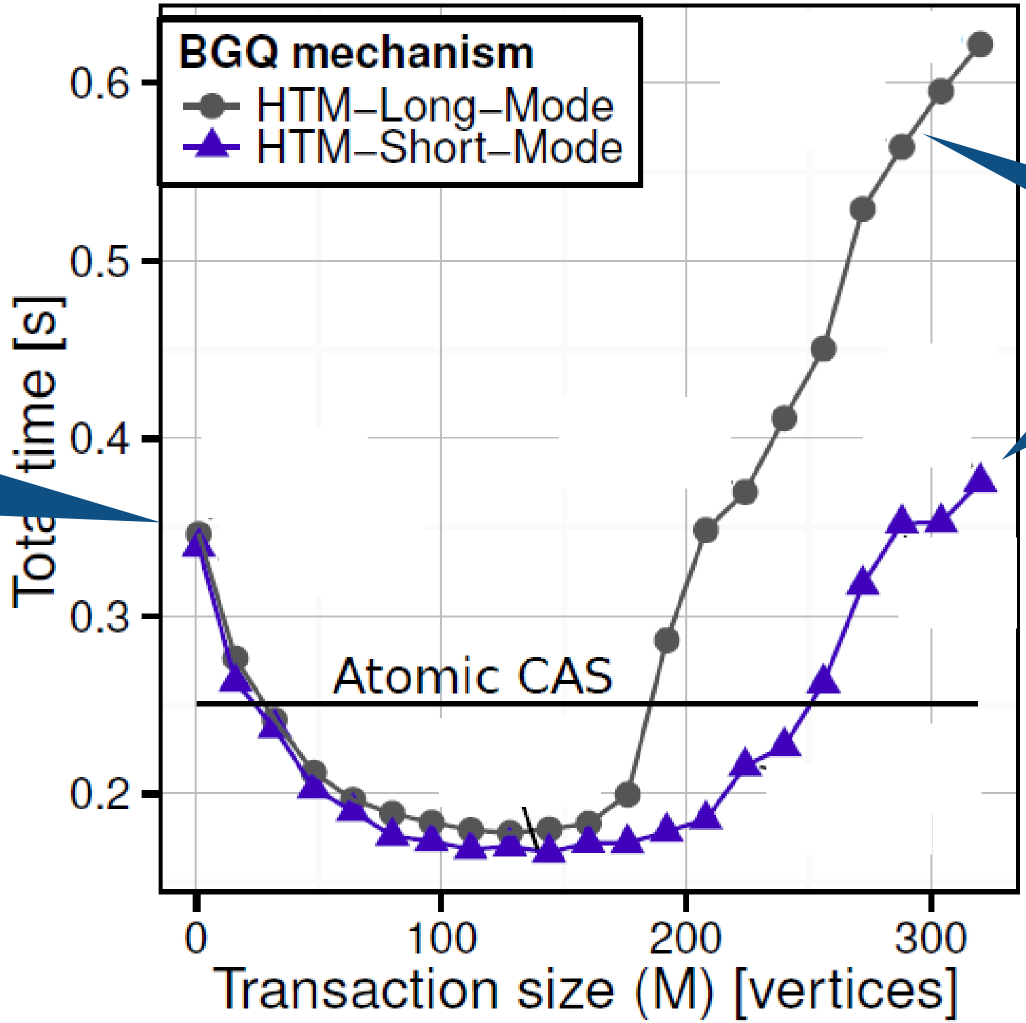
Multi-Vertex Transactions

Marking vertices as visited



Startup and commit overheads

MULTI-VERTEX TRANSACTIONS
MARKING VERTICES AS VISITED

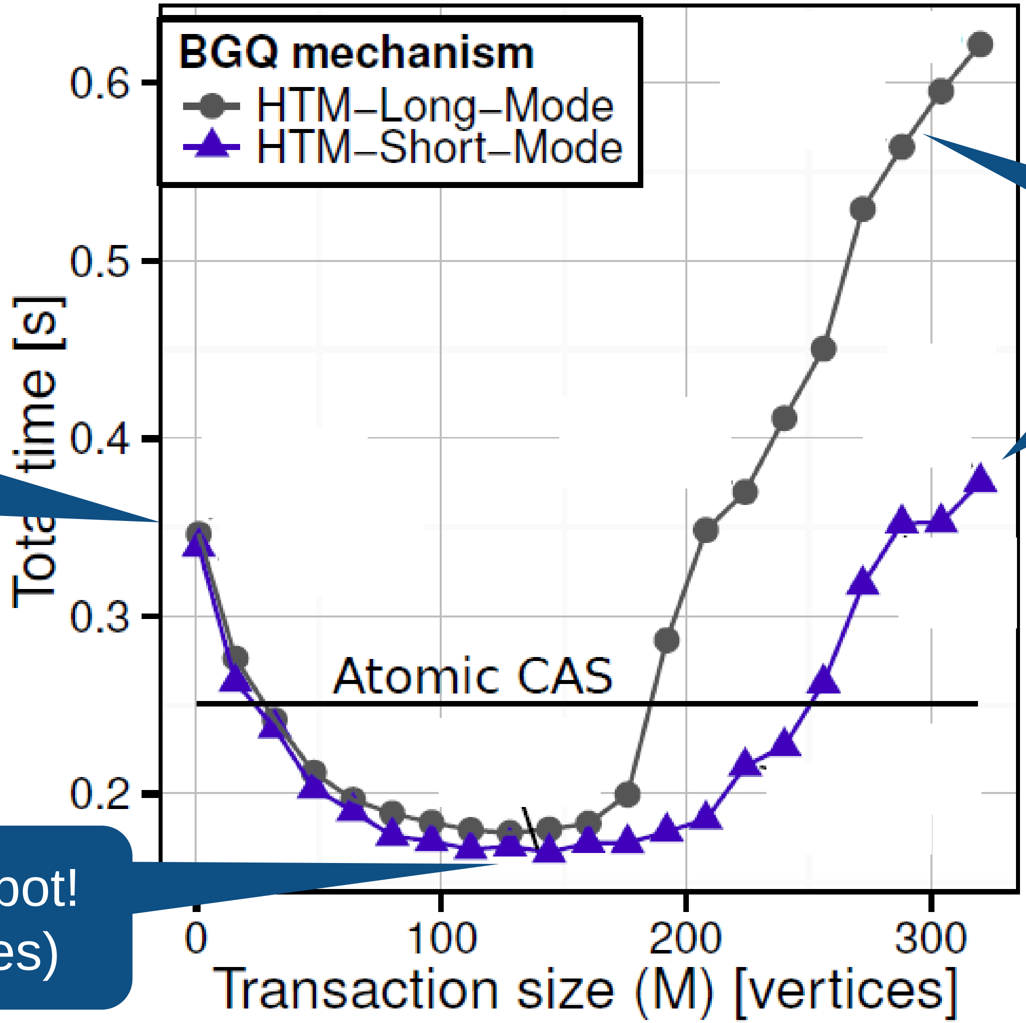


Startup and commit overheads

Abort and rollback overheads

Multi-Vertex Transactions

Marking vertices as visited

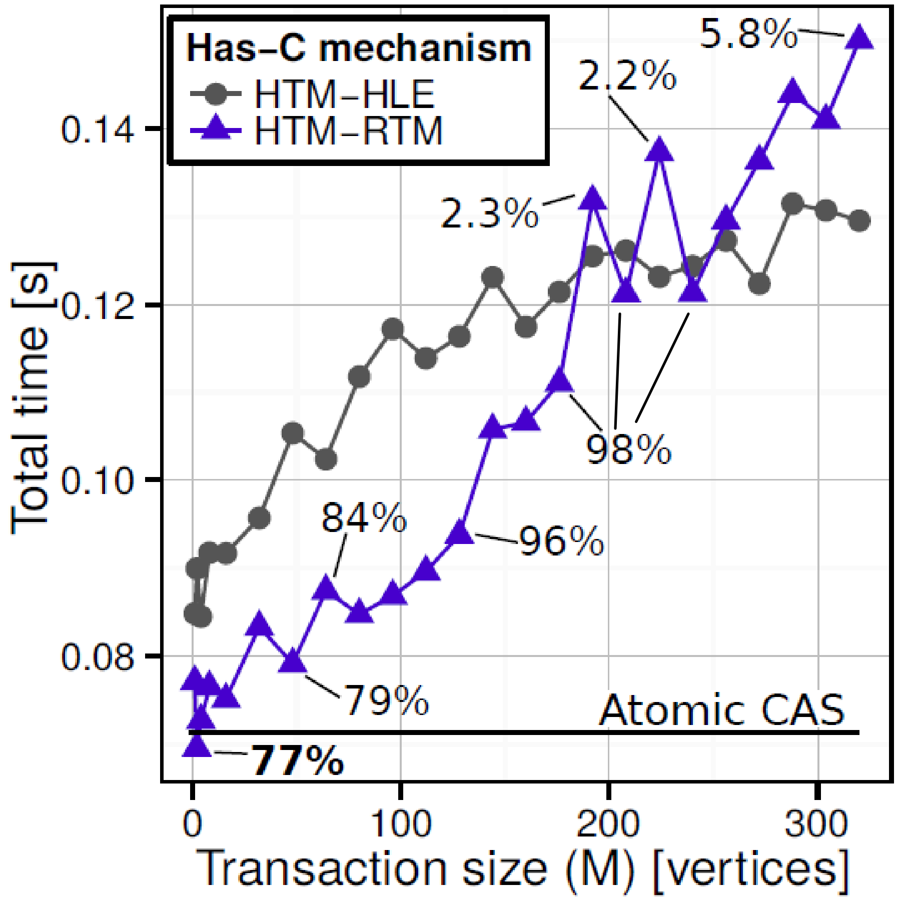


Startup and commit overheads

The sweetspot! (144 vertices)

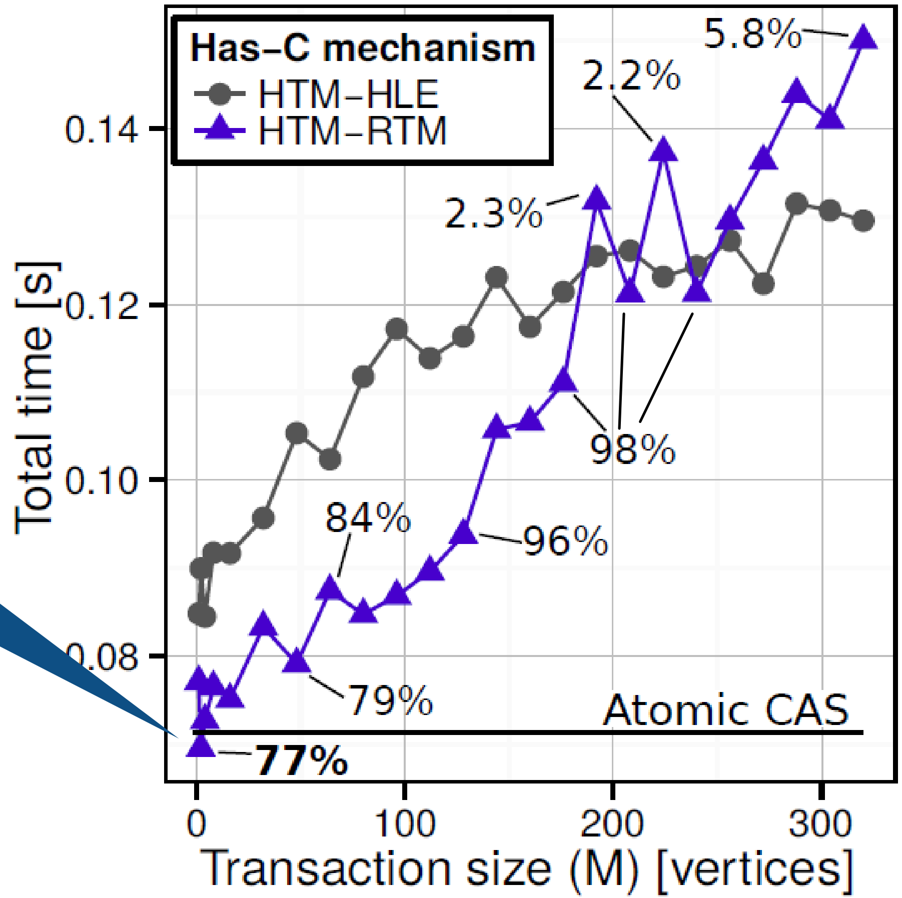
Abort and rollback overheads

Multi-Vertex Transactions
Marking vertices as visited



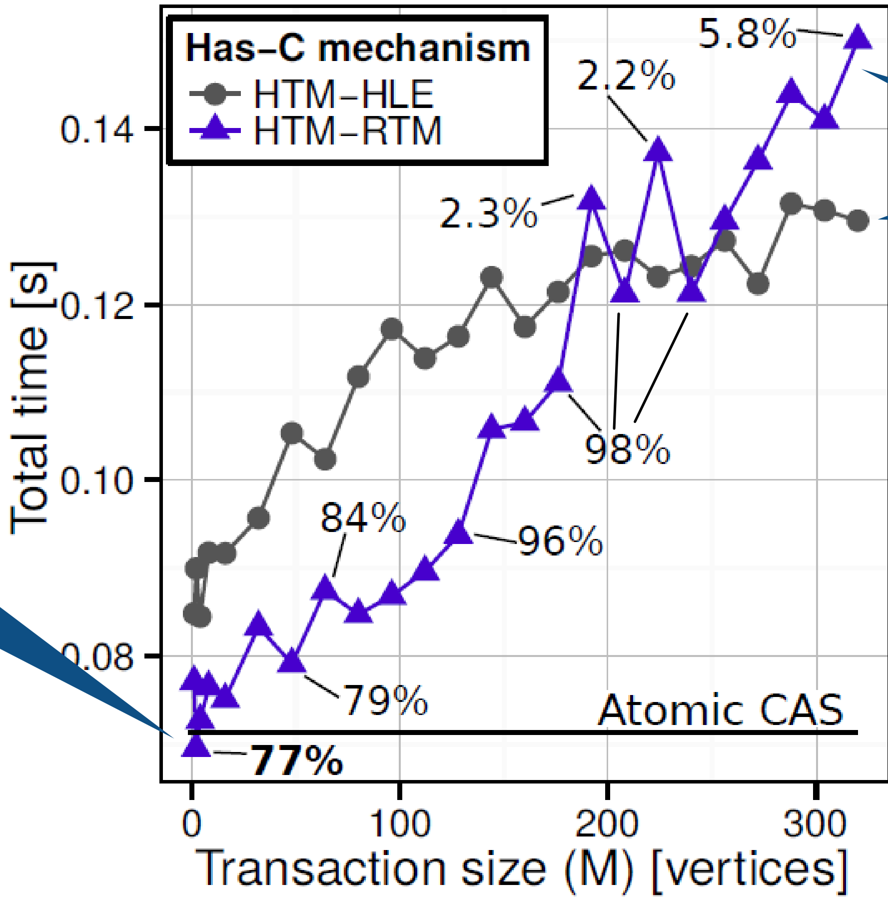
Multi-Vertex Transactions

Marking vertices as visited



Startup and commit overheads

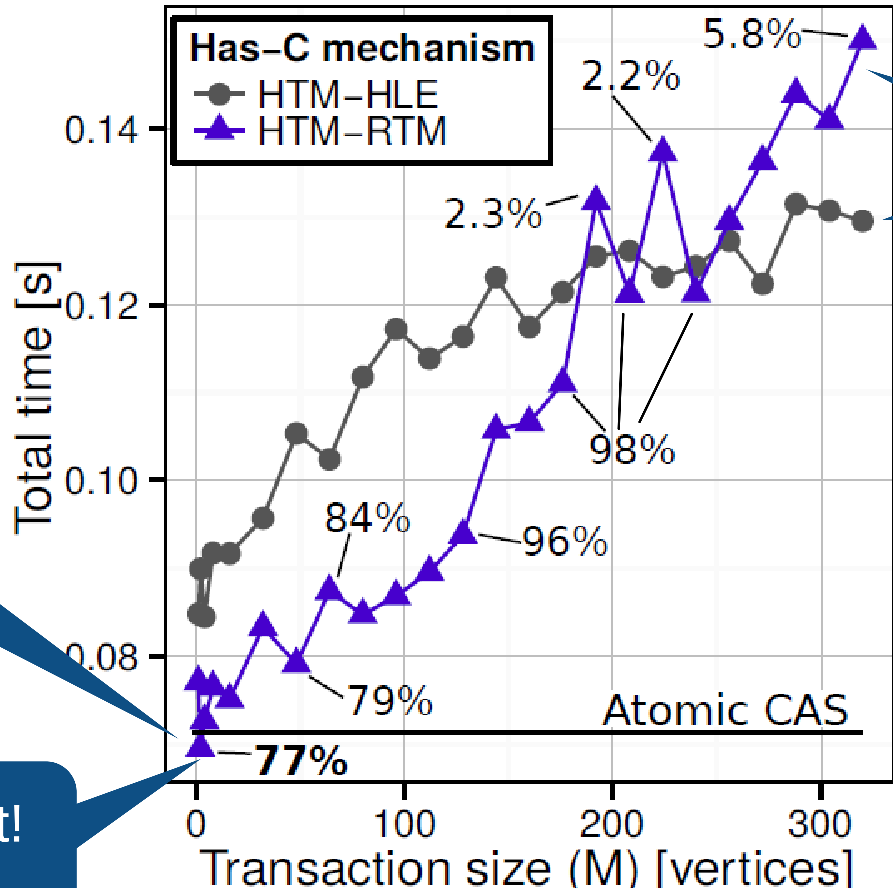
Multi-Vertex Transactions
Marking vertices as visited



Startup and commit overheads

Abort and rollback overheads

Multi-Vertex Transactions
Marking vertices as visited



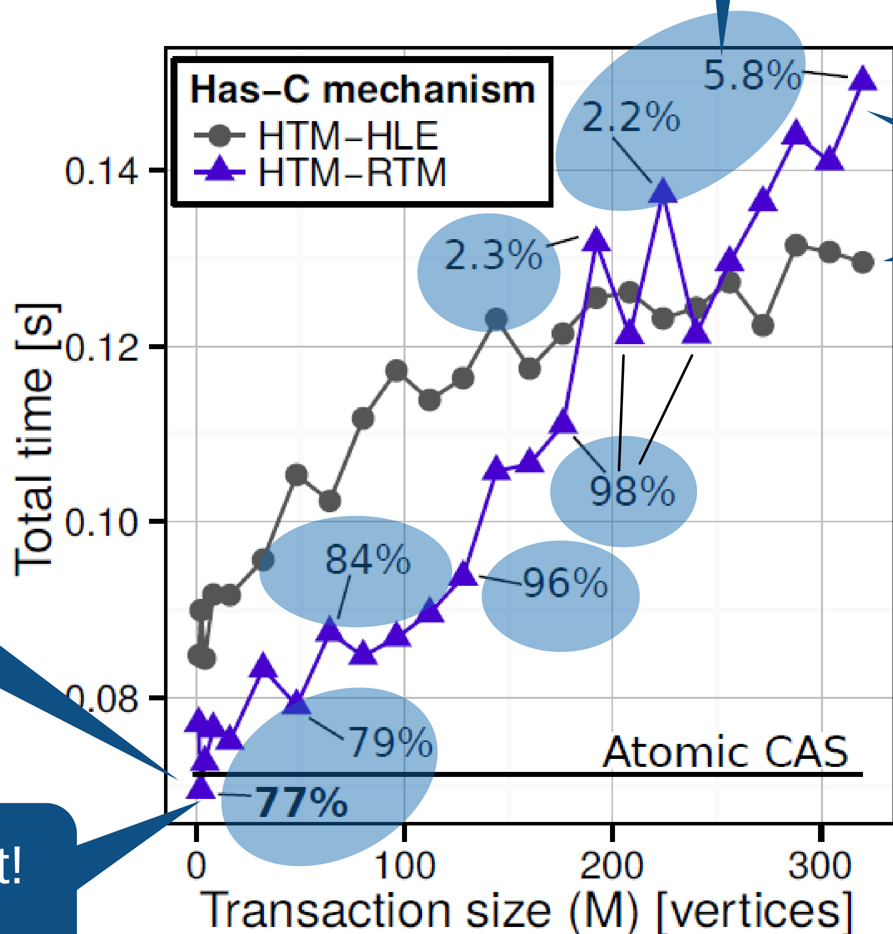
Startup and commit overheads

The sweetspot! (2 vertices)

Abort and rollback overheads

MULTI-VERTEX TRANSACTIONS
MARKING VERTICES AS VISITED

Numbers: % of aborts due to HTM capacity overflows



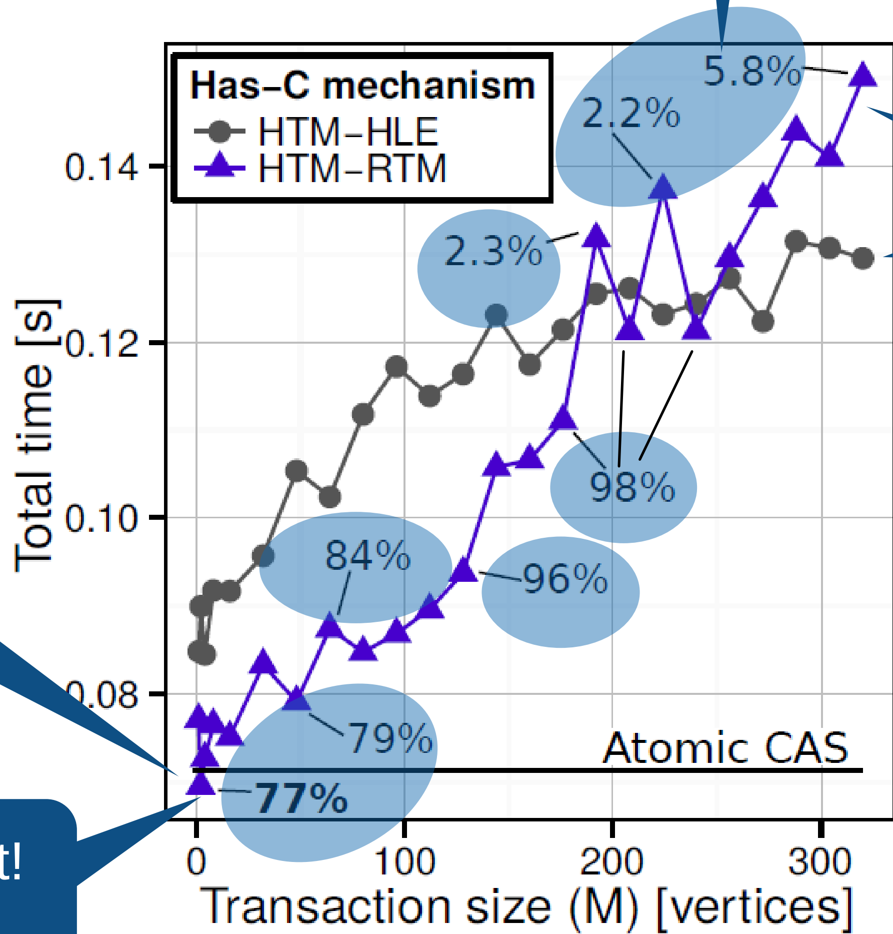
Abort and rollback overheads

Startup and commit overheads

The sweetspot! (2 vertices)

Multi-Vertex Transactions
Marking vertices as visited

Numbers: % of aborts due to HTM capacity overflows



Abort and rollback overheads

Startup and commit overheads

Majority of aborts are due to HTM capacity overflows (small cache size & associativity)

The sweetspot! (2 vertices)

Performance Analysis
Questions answered

How can we implement AAM handlers most effectively?

What are performance tradeoffs related to HTM?

What are advantages of HTM over atomics for AAM?

What are the optimal transaction sizes? Can we amortize transaction overheads?

PERFORMANCE ANALYSIS
QUESTIONS ANSWERED

„It really depends” ☺. But... There are some regularities

What are performance tradeoffs related to HTM?

What are advantages of HTM over atomics for AAM?

What are the optimal transaction sizes? Can we amortize transaction overheads?

PERFORMANCE ANALYSIS
QUESTIONS ANSWERED

What are performance tradeoffs related to HTM?

„It really depends” ☺. But... There are some regularities

What are the optimal transaction sizes? Can we amortize transaction overheads?

For some algorithms (BFS) HTM is better

PERFORMANCE ANALYSIS
QUESTIONS ANSWERED

„It really depends” ☺. But... There are some regularities

What are performance tradeoffs related to HTM?

For some algorithms (BFS) HTM is better

For others (PageRank) atomics give more performance

What are the optimal transaction sizes? Can we amortize transaction overheads?

Performance Analysis
Questions answered

What are performance tradeoffs related to HTM?

„It really depends" ☺. But... There are some regularities

„May fail"

For some algorithms (BFS) HTM is better

For others (PageRank) atomics give more performance

What are the optimal transaction sizes? Can we amortize transaction overheads?

PERFORMANCE ANALYSIS
QUESTIONS ANSWERED

„It really depends” ☺. But... There are some regularities

What are performance tradeoffs related to HTM?

For some algorithms (BFS) HTM is better

„May fail”

For others (PageRank) atomics give more performance

„Always succeed”

What are the optimal transaction sizes? Can we amortize transaction overheads?

PERFORMANCE ANALYSIS
QUESTIONS ANSWERED

What are performance tradeoffs related to HTM?

„It really depends" ☺. But... There are some regularities

What are the optimal transaction sizes? Can we amortize transaction overheads?

For some algorithms (BFS) HTM is better

„May fail"

For others (PageRank) atomics give more performance

„Always succeed"

AAM establishes a whole hierarchy of algorithms; check the paper ☺

Performance Analysis
Questions answered

What are performance tradeoffs related to HTM?

„It really depends” ☺. But... There are some regularities

For some algorithms (BFS) HTM is better

„May fail”

For others (PageRank) atomics give more performance

„Always succeed”

AAM establishes a whole hierarchy of algorithms; check the paper ☺

Size for BG/Q ~100 > Size for Haswell ~10

Yes, we can

PERFORMANCE ANALYSIS
QUESTIONS ANSWERED

What are performance tradeoffs related to HTM?

„It really depends" ☺. But... There are some regularities

For some algorithms (BFS) HTM is better

„May fail"

For others (PageRank) atomics give more performance

„Always succeed"

AAM establishes a whole hierarchy of algorithms; check the paper ☺

Same for other graphs

Size for BG/Q ~100 > Size for Haswell ~10

Yes, we can

PERFORMANCE ANALYSIS
QUESTIONS ANSWERED

Larger cache & associativity → fewer aborts & more coarsening

„It really depends" ☺. But... There are some regularities

For some algorithms (BFS) HTM is better

„May fail"

For others (PageRank) atomics give more performance

„Always succeed"

AAM establishes a whole hierarchy of algorithms; check the paper ☺

Same for other graphs

Size for BG/Q ~100 > Size for Haswell ~10

Yes, we can

PERFORMANCE ANALYSIS
QUESTIONS ANSWERED

„It really depends” ☺. But... There are some regularities

Larger cache & associativity → fewer aborts & more coarsening

Larger (L2) cache → higher latency

For some algorithms (BFS) HTM is better

„May fail”

For others (PageRank) atomics give more performance

„Always succeed”

AAM establishes a whole hierarchy of algorithms; check the paper ☺

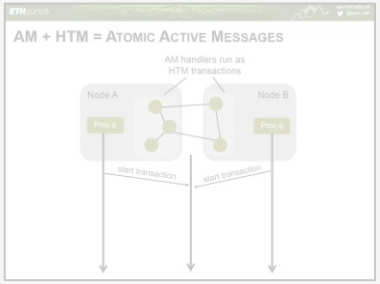
Same for other graphs

Size for BG/Q ~100 > Size for Haswell ~10

Yes, we can

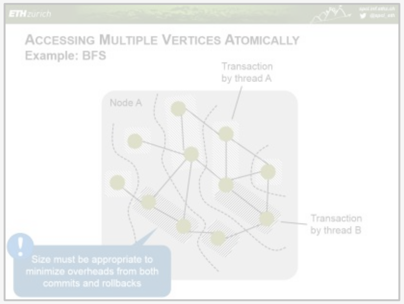
Overview of our Research

HTM for graphs in SM & DM environments



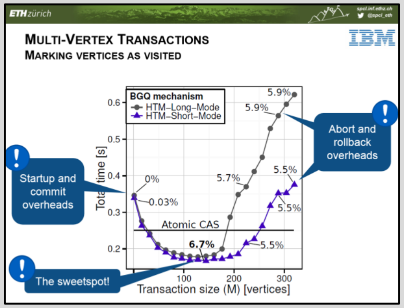
HTM + Active Messages
= Atomic Active Messages

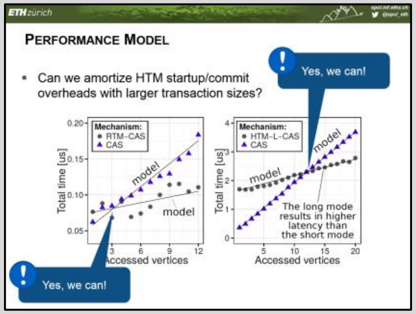
Coarsening & coalescing



Performance Modeling & Analysis

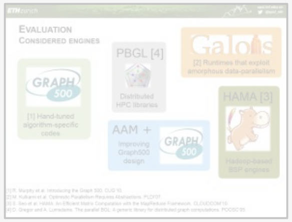
Haswell & BG/Q Analysis

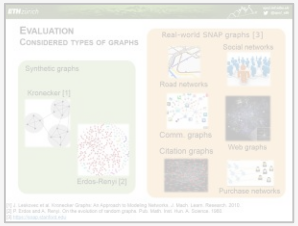




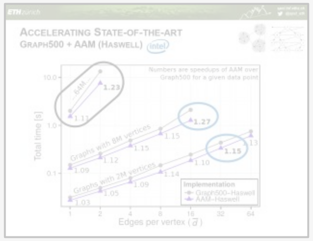
Performance model

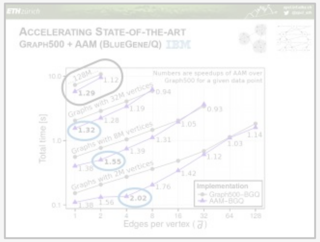
Evaluation



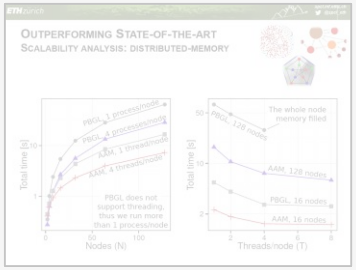


Considered engines and graphs





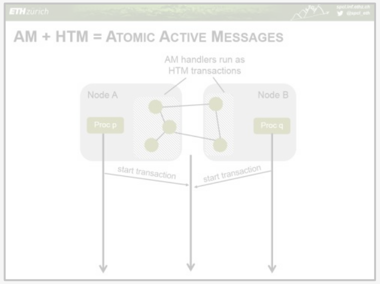
Accelerating state-of-the-art



Scalability

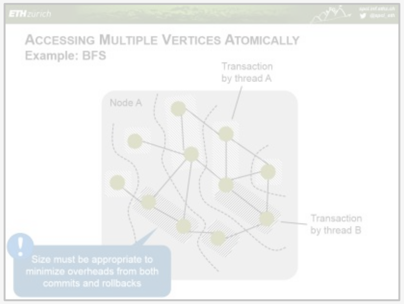
Overview of our Research

HTM for graphs in SM & DM environments



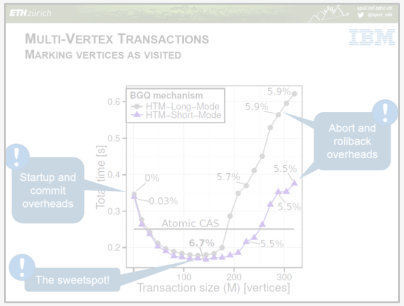
HTM + Active Messages
= Atomic Active Messages

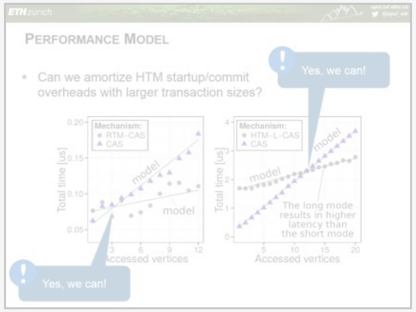
Coarsening & coalescing



Performance Modeling & Analysis

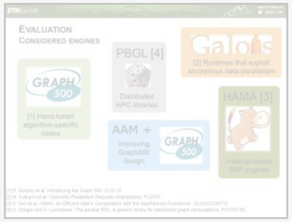
Haswell & BG/Q Analysis

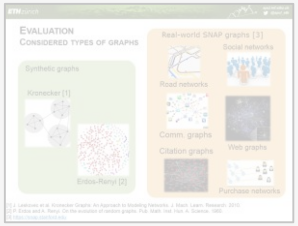




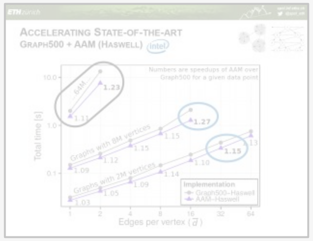
Performance model

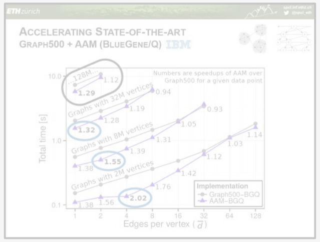
Evaluation



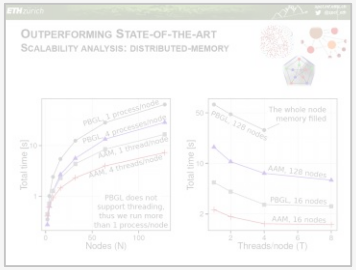


Considered engines and graphs





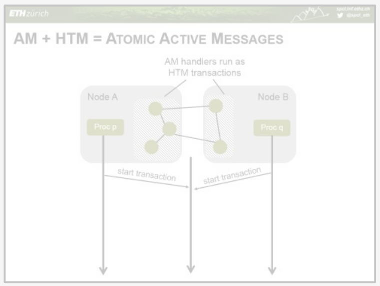
Accelerating state-of-the-art



Scalability

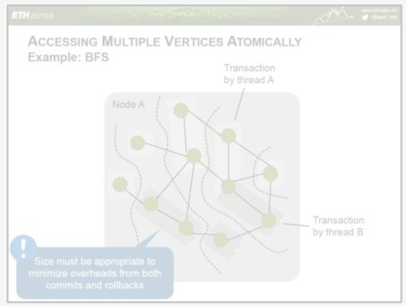
Overview of our Research

HTM for graphs in SM & DM environments



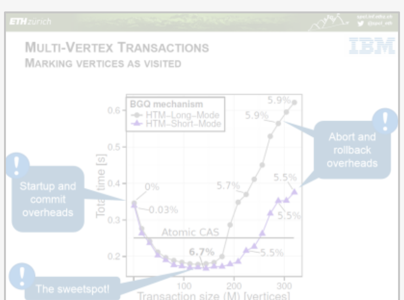
HTM + Active Messages
= Atomic Active Messages

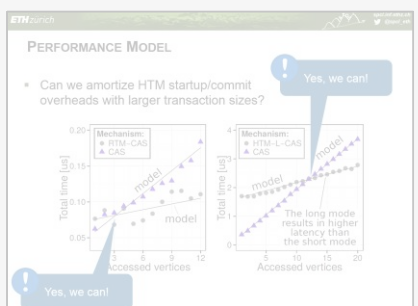
Coarsening & coalescing



Performance Modeling & Analysis

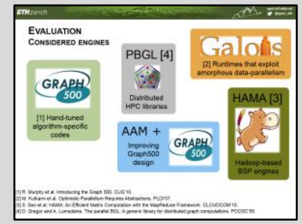
Haswell & BG/Q Analysis

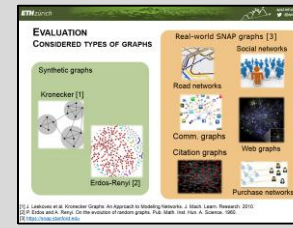




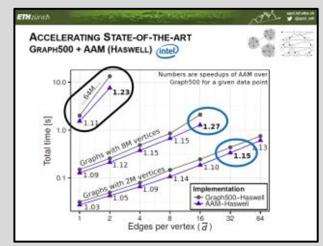
Performance model

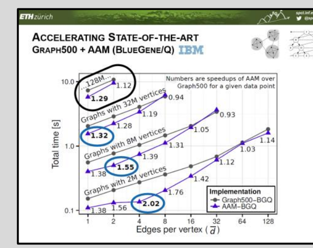
Evaluation



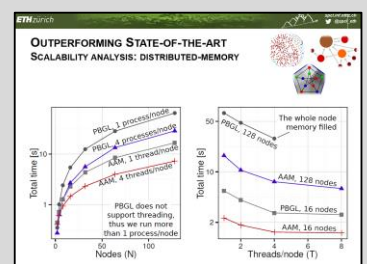


Considered engines and graphs





Accelerating state-of-the-art



Scalability

EVALUATION
CONSIDERED ENGINES

Evaluation

Considered engines

GRAPH 500

[1] Hand-tuned algorithm-specific codes

[1] R. Murphy et al. Introducing the Graph 500. CUG'10.

EVALUATION
CONSIDERED ENGINES

[1] Hand-tuned algorithm-specific codes

AAM +

Improving Graph500 design

[1] R. Murphy et al. Introducing the Graph 500. CUG'10.

EVALUATION
CONSIDERED ENGINES

Galois

[2] Runtimes that exploit amorphous data-parallelism

GRAPH 500

[1] Hand-tuned algorithm-specific codes

AAM + GRAPH 500

Improving Graph500 design

[1] R. Murphy et al. Introducing the Graph 500. CUG'10.
[2] M. Kulkarni et al. Optimistic Parallelism Requires Abstractions. PLDI'07.

EVALUATION
CONSIDERED ENGINES

[1] Hand-tuned algorithm-specific codes

[2] Runtimes that exploit amorphous data-parallelism

AAM +

Improving Graph500 design

HAMA [3]

Hadoop-based BSP engines

[1] R. Murphy et al. Introducing the Graph 500. CUG'10.
[2] M. Kulkarni et al. Optimistic Parallelism Requires Abstractions. PLDI'07.
[3] S. Seo et al. HAMA: An Efficient Matrix Computation with the MapReduce Framework. CLOUDCOM'10.

Evaluation
Considered engines

PBGL [4]

Distributed HPC libraries

Galois

[2] Runtimes that exploit amorphous data-parallelism

[1] Hand-tuned algorithm-specific codes

HAMA [3]

Hadoop-based BSP engines

AAM +

Improving Graph500 design

[1] R. Murphy et al. Introducing the Graph 500. CUG'10.
[2] M. Kulkarni et al. Optimistic Parallelism Requires Abstractions. PLDI'07.
[3] S. Seo et al. HAMA: An Efficient Matrix Computation with the MapReduce Framework. CLOUDCOM'10.
[4] D. Gregor and A. Lumsdaine. The parallel BGL: A generic library for distributed graph computations. POOSC'05.

EVALUATION

CONSIDERED TYPES OF GRAPHS

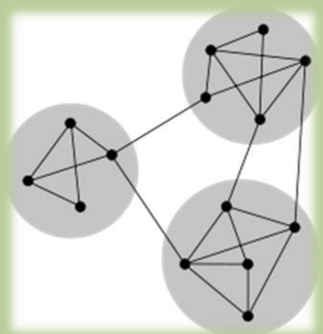
EVALUATION
CONSIDERED TYPES OF GRAPHS

Synthetic graphs

Evaluation
Considered types of graphs

Synthetic graphs

Kronecker [1]

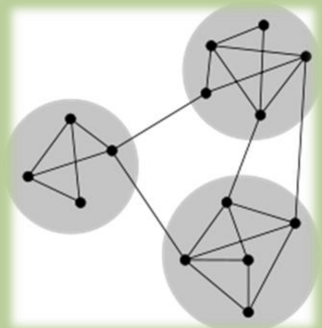


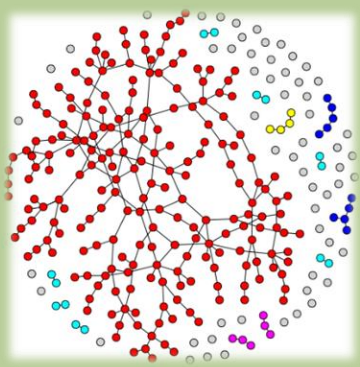
[1] J. Leskovec et al. Kronecker Graphs: An Approach to Modeling Networks. J. Mach. Learn. Research. 2010.

Evaluation
Considered types of graphs

Synthetic graphs

Kronecker [1]





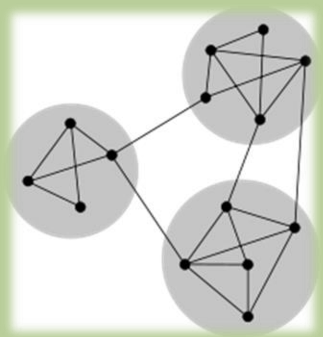
Erdös-Rényi [2]

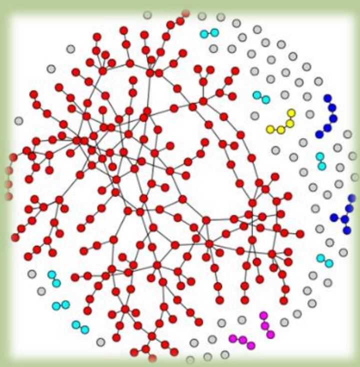
[1] J. Leskovec et al. Kronecker Graphs: An Approach to Modeling Networks. J. Mach. Learn. Research. 2010.
[2] P. Erdos and A. Renyi. On the evolution of random graphs. Pub. Math. Inst. Hun. A. Science. 1960.

Evaluation
Considered types of graphs

Synthetic graphs

Kronecker [1]





Erdös-Rényi [2]

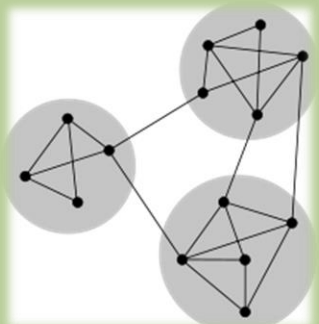
Real-world SNAP graphs [3]

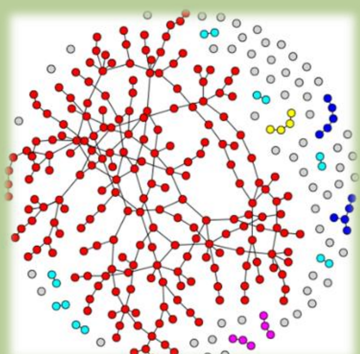
[1] J. Leskovec et al. Kronecker Graphs: An Approach to Modeling Networks. J. Mach. Learn. Research. 2010.
[2] P. Erdos and A. Renyi. On the evolution of random graphs. Pub. Math. Inst. Hun. A. Science. 1960.
[3] https://snap.stanford.edu

Evaluation
Considered types of graphs

Synthetic graphs

Kronecker [1]





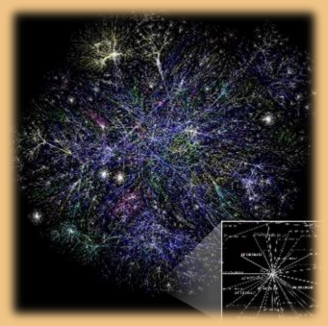
Erdös-Rényi [2]

Real-world SNAP graphs [3]

Road networks

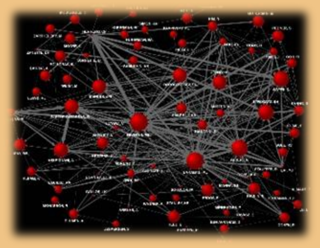
Social networks

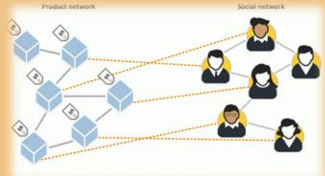
Comm. graphs



Web graphs

Citation graphs





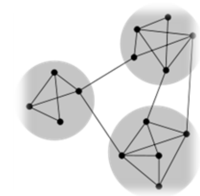
Purchase networks

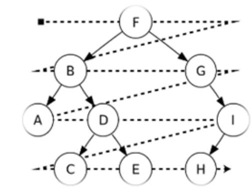
[1] J. Leskovec et al. Kronecker Graphs: An Approach to Modeling Networks. J. Mach. Learn. Research. 2010.
[2] P. Erdos and A. Renyi. On the evolution of random graphs. Pub. Math. Inst. Hun. A. Science. 1960.
[3] https://snap.stanford.edu

Accelerating State-of-the-art Graph500 + AAM (BlueGene/Q)

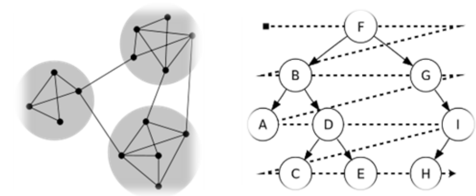
ACCELERATING STATE-OF-THE-ART

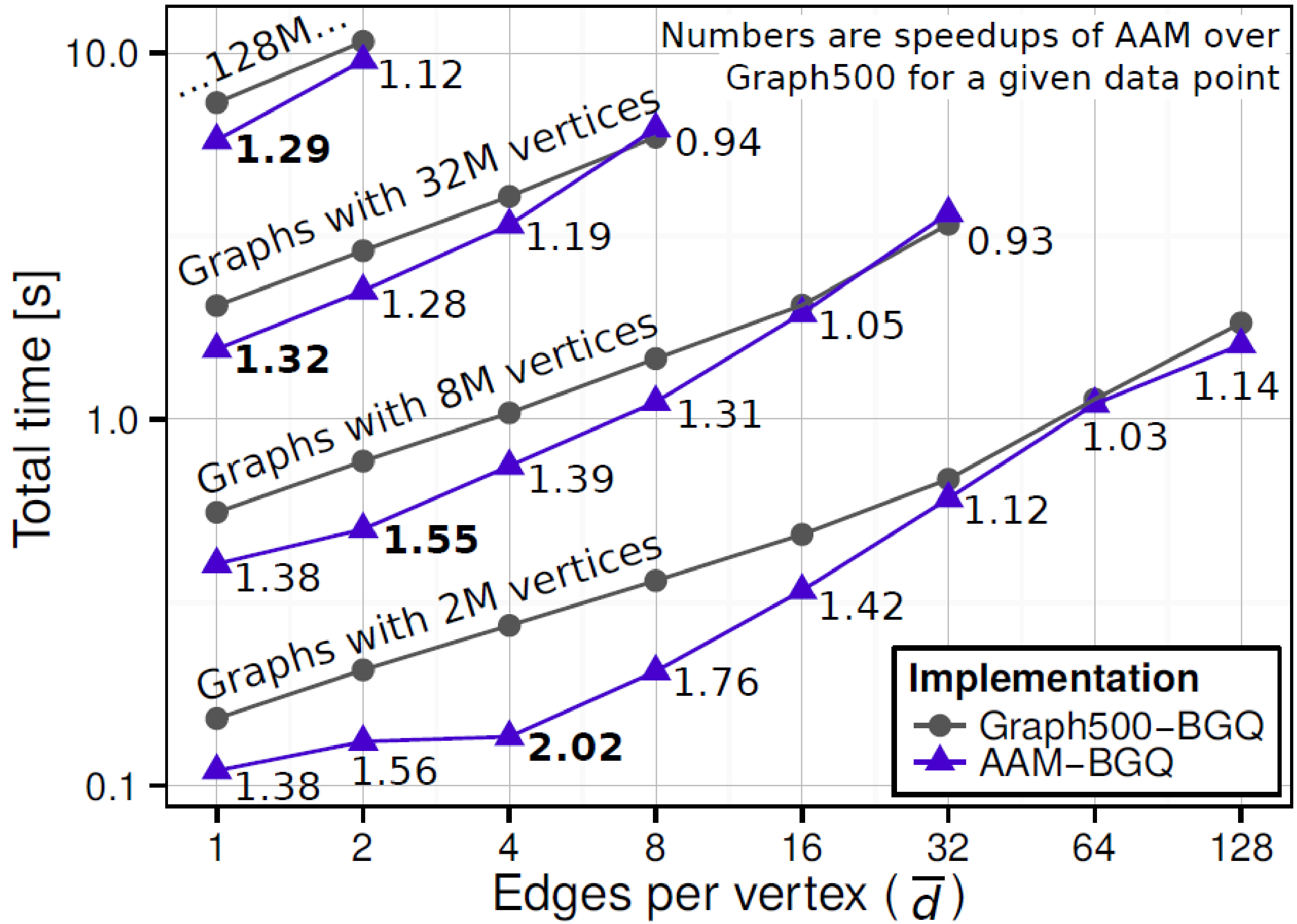
GRAPH500 + AAM (BLUEGENE/Q) IBM



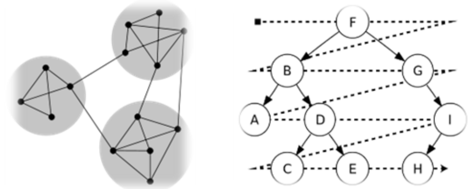


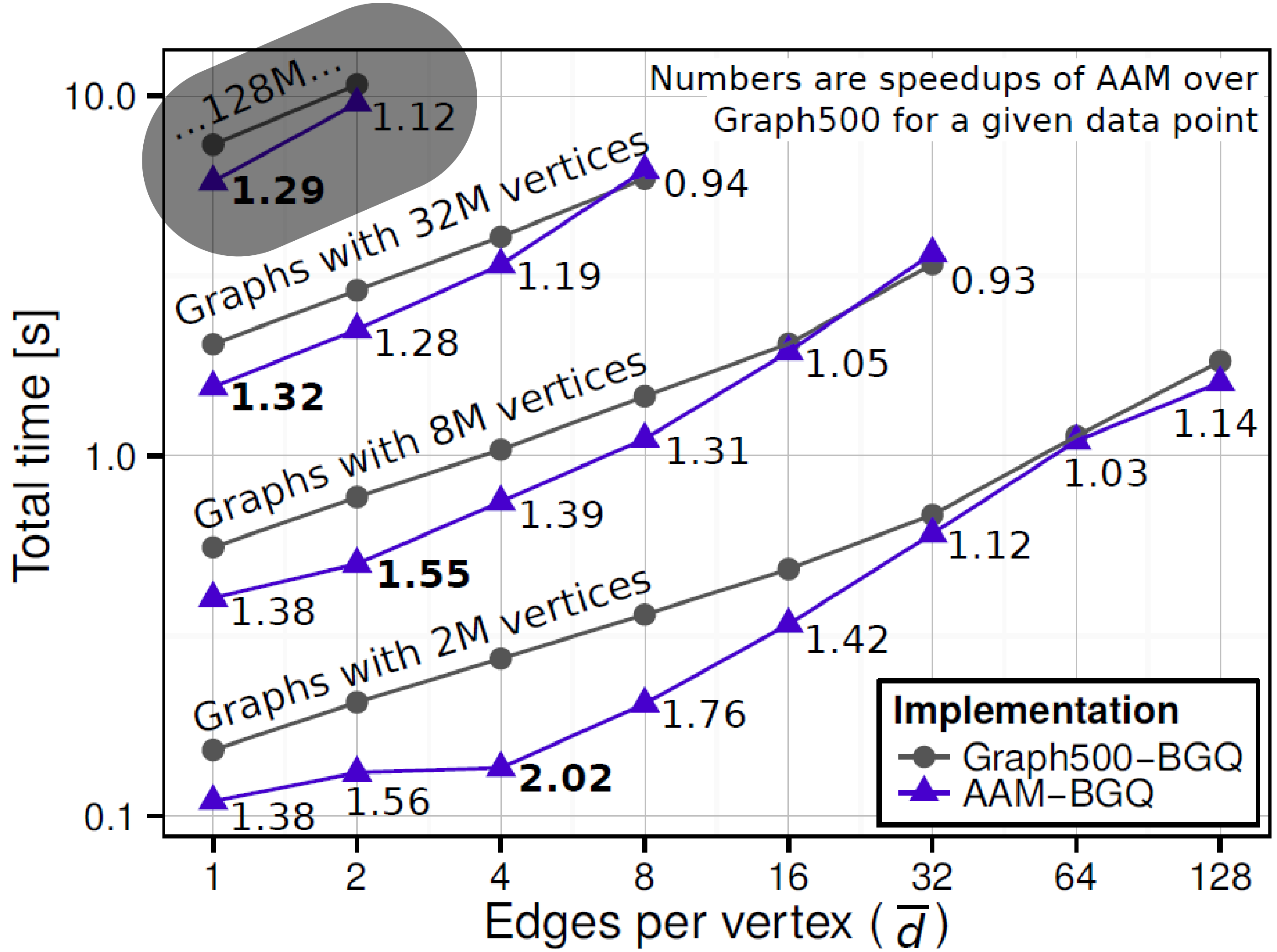
ACCELERATING STATE-OF-THE-ART GRAPH500 + AAM (BLUEGENE/Q) IBM



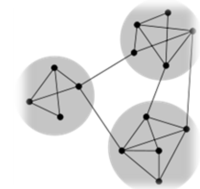


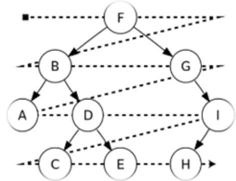
ACCELERATING STATE-OF-THE-ART GRAPH500 + AAM (BLUEGENE/Q)



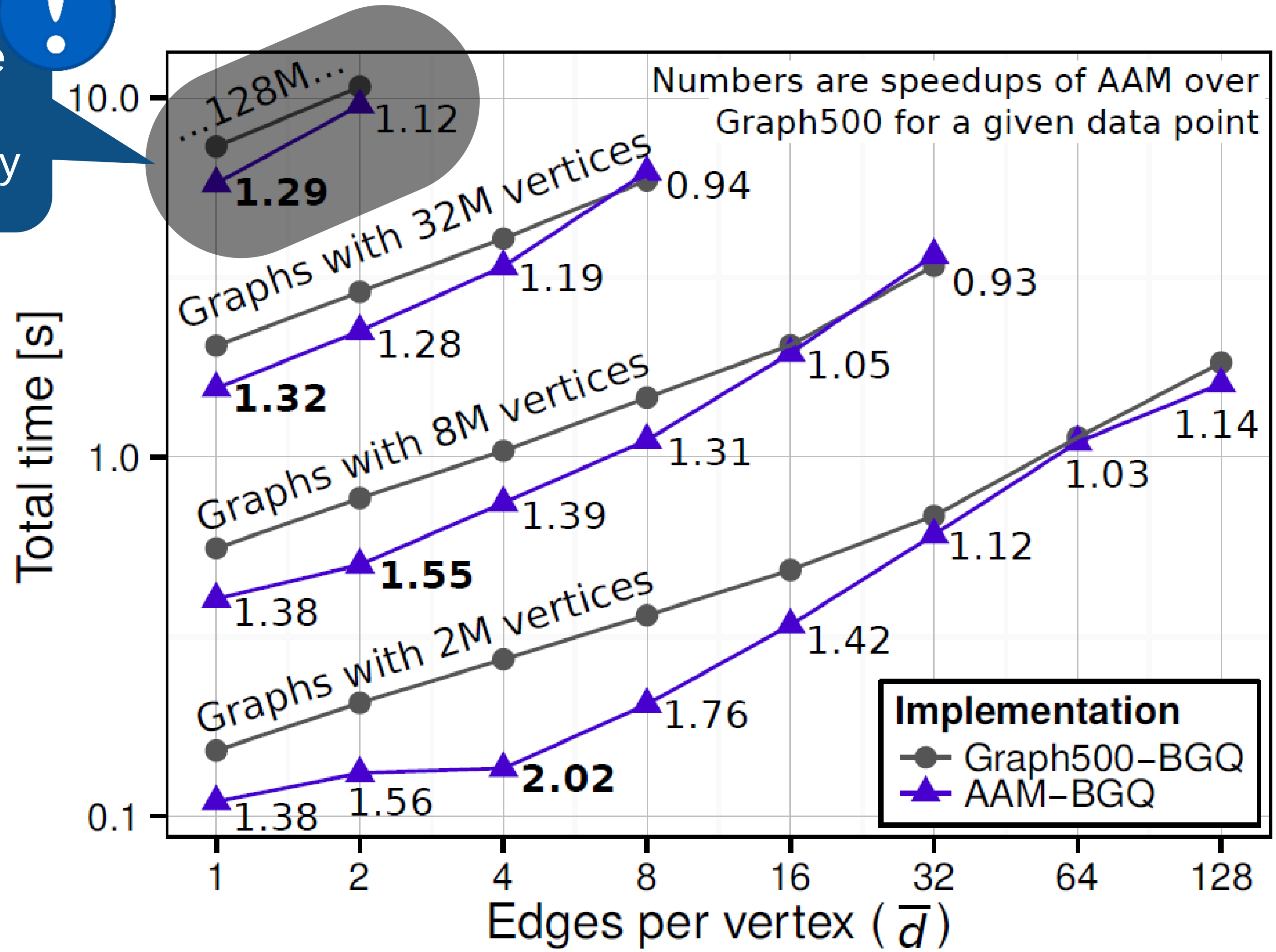


ACCELERATING STATE-OF-THE-ART GRAPH500 + AAM (BLUEGENE/Q)

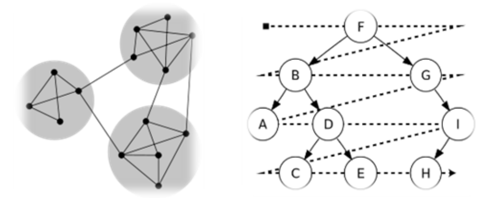




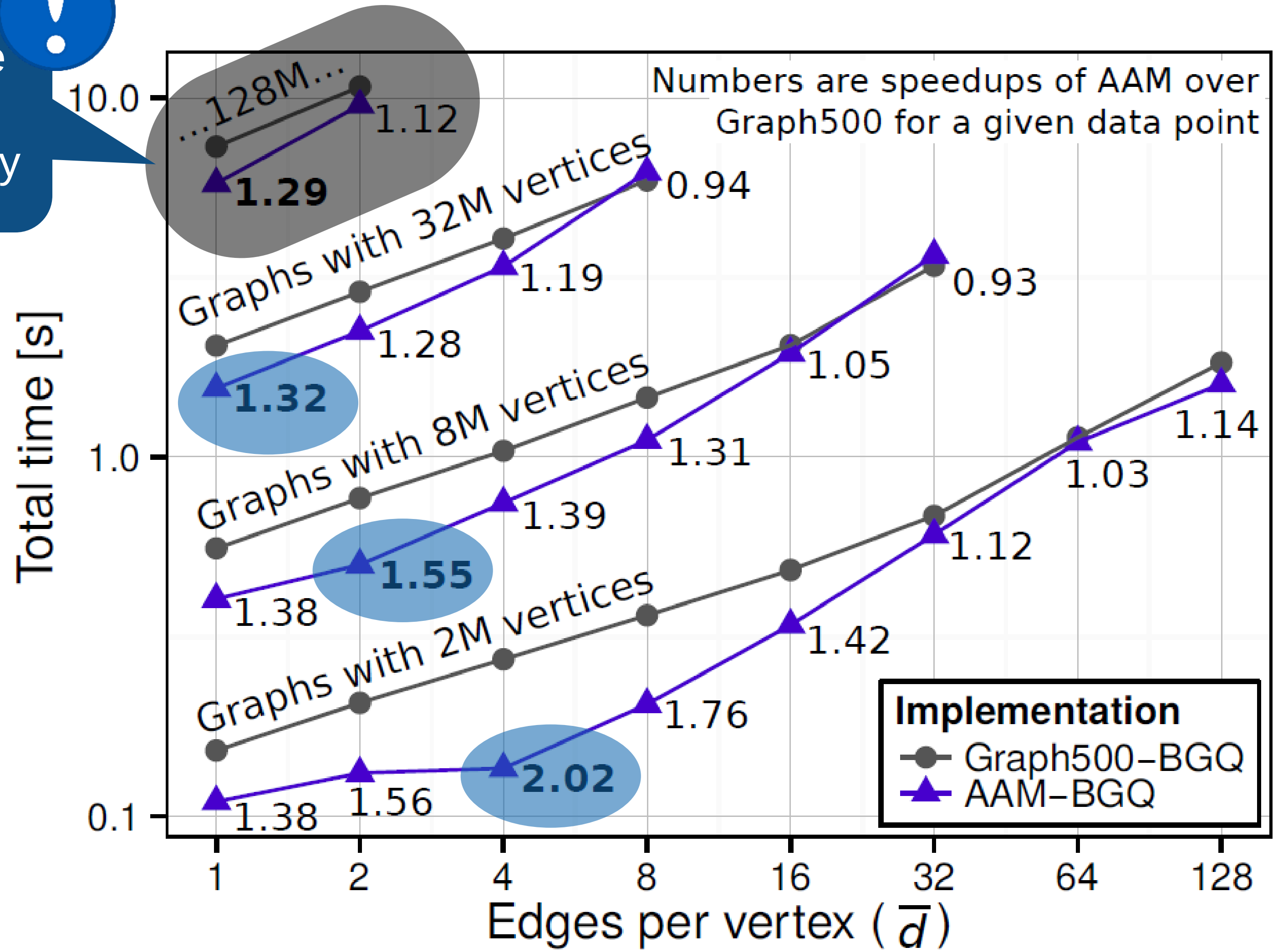
Fill the whole memory



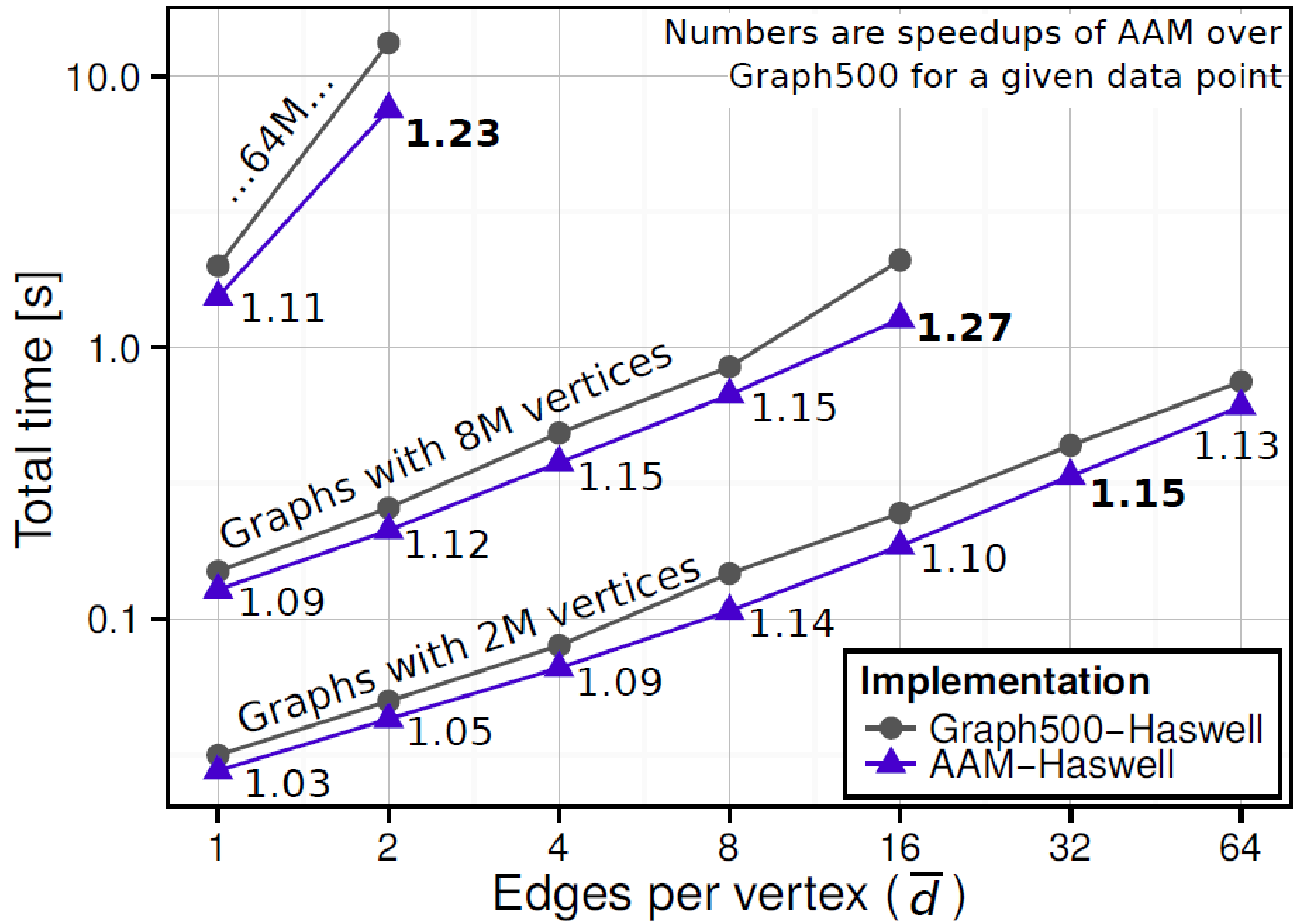
ACCELERATING STATE-OF-THE-ART GRAPH500 + AAM (BLUEGENE/Q) IBM



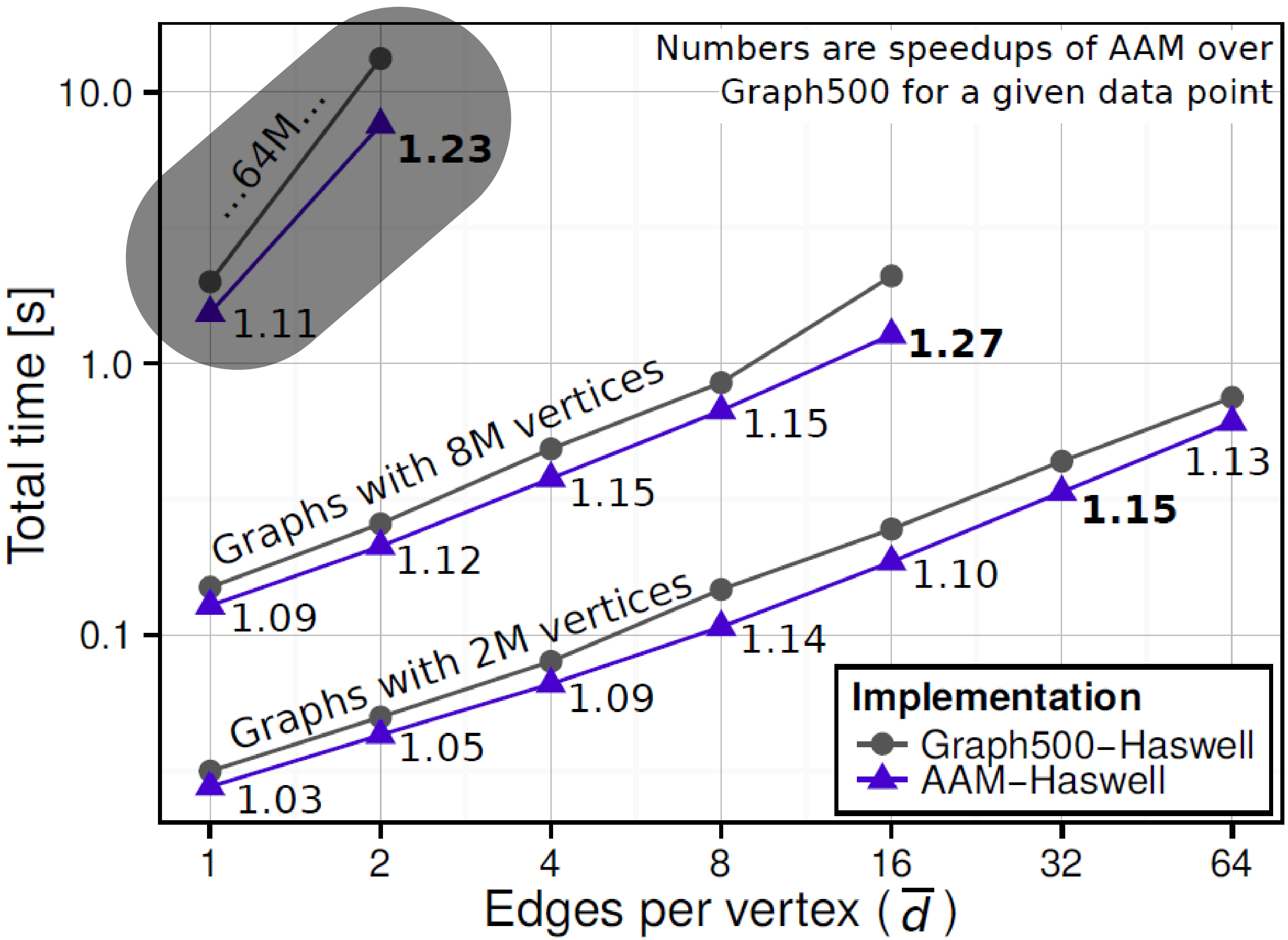
Fill the whole memory



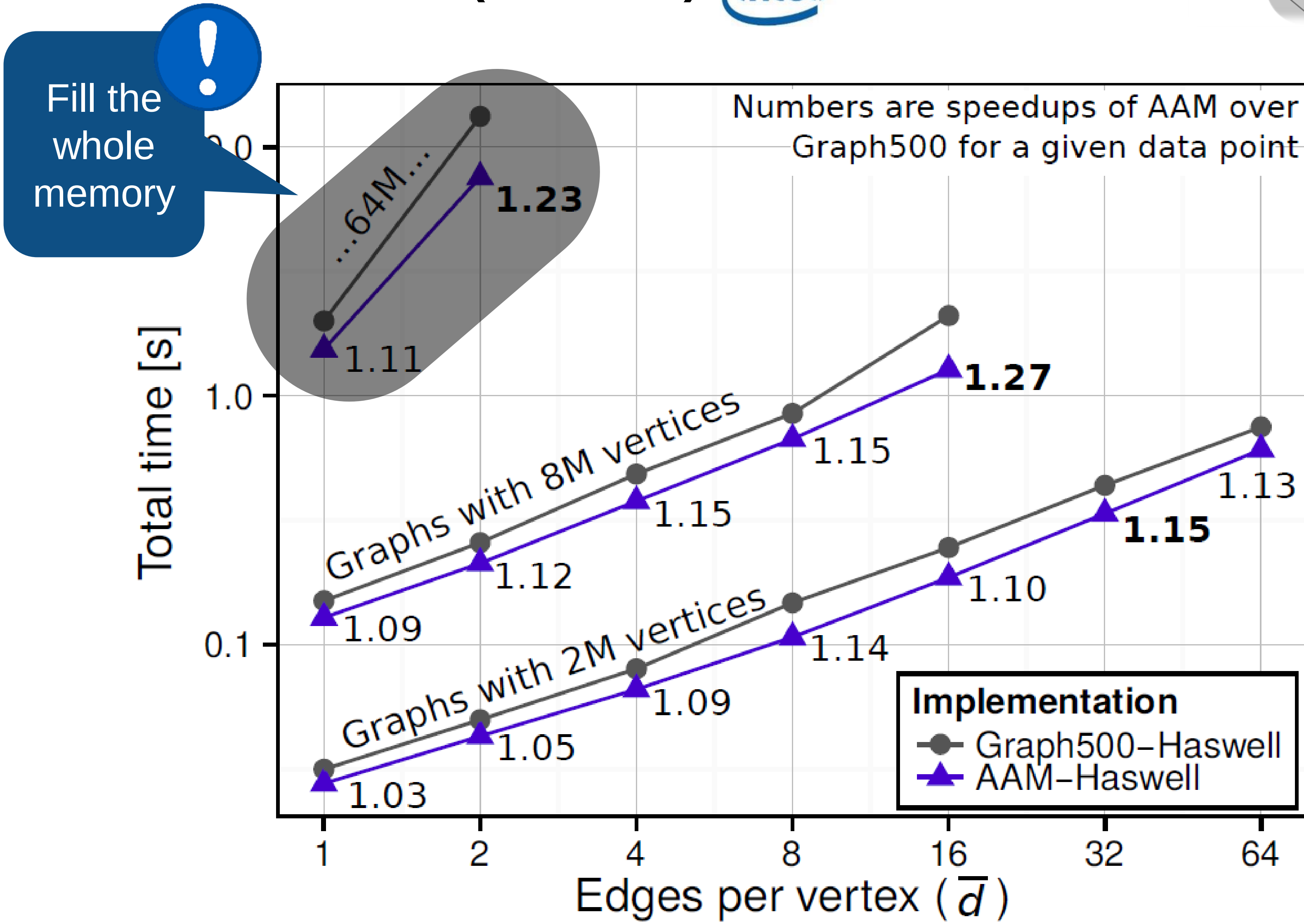
ACCELERATING STATE-OF-THE-ART GRAPH500 + AAM (HASWELL)



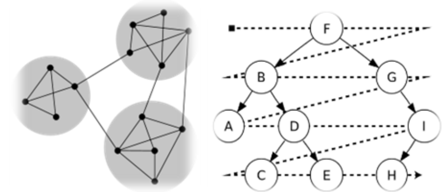
ACCELERATING STATE-OF-THE-ART GRAPH500 + AAM (HASWELL)

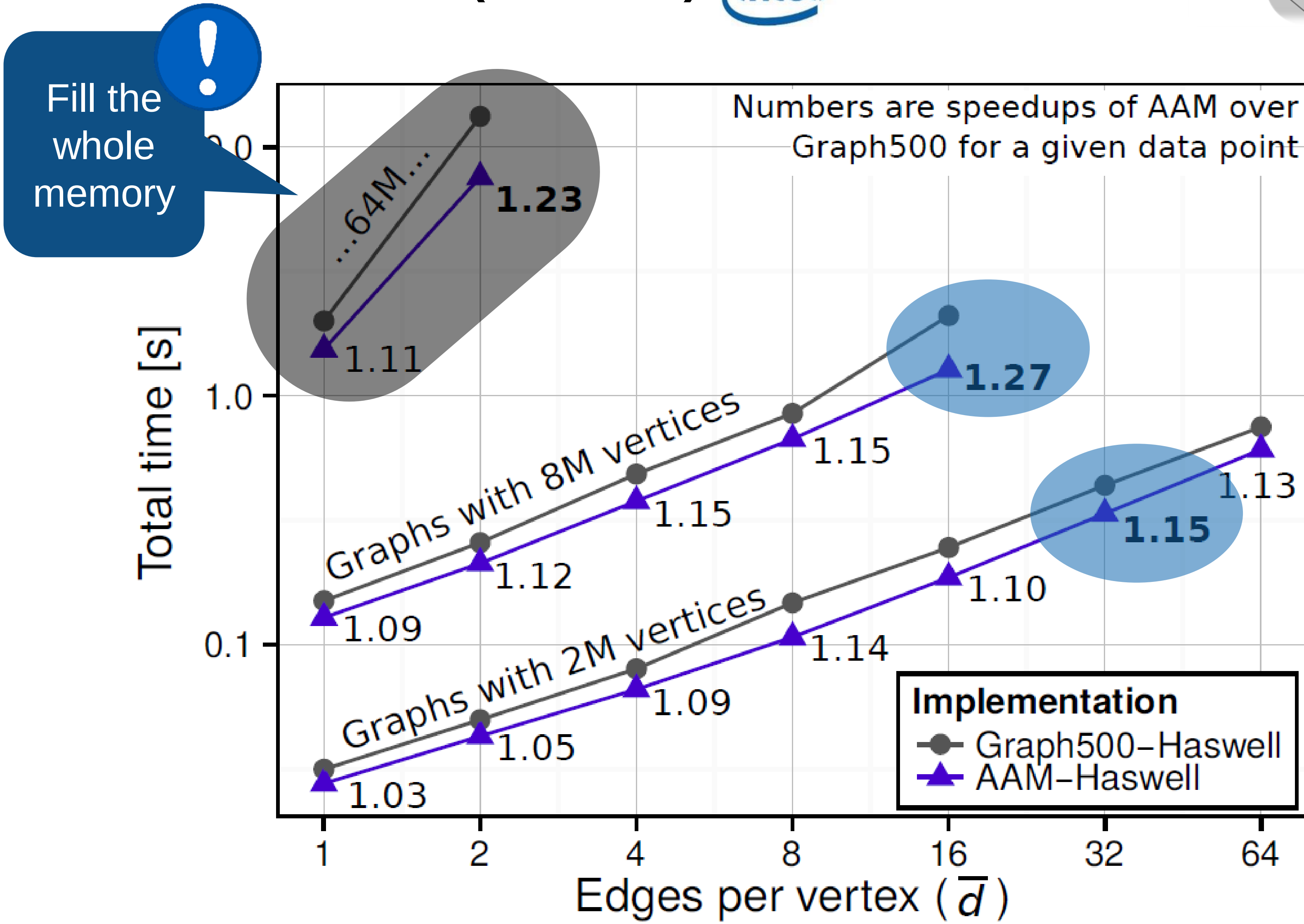


ACCELERATING STATE-OF-THE-ART GRAPH500 + AAM (HASWELL)



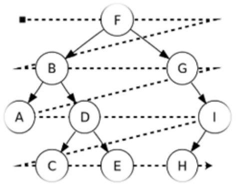
ACCELERATING STATE-OF-THE-ART GRAPH500 + AAM (HASWELL)



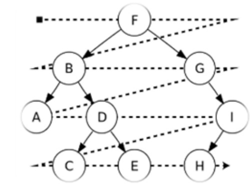


OUTPERFORMING STATE-OF-THE-ART

SNAP

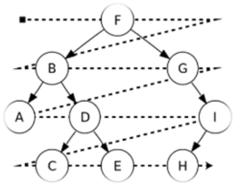


OUTPERFORMING STATE-OF-THE-ART



Input graph properties					BG/Q analysis			Haswell analysis					
Type	ID	Name	$\|V\|$	$\|E\|$	S over g500 ($M = 24$)	M	S over g500	S over g500 ($M = 2$)	S over Galois ($M = 2$)	M	S over g500	S over Galois	S over HAMA
Comm. networks (CNs)	cWT	wiki-Talk	2.4M	5M	2.82	48	3.35	0.91	1.22	6	0.96	1.28	344
	cEU	email-EuAll	265k	420k	3.67	32	4.36	0.76	0.88	4	0.97	1.12	1448
Social networks (SNs)	sLV	soc-LiveJ.	4.8M	69M	1.44	12	1.56	1.05	1.1	3	1.07	1.12	$> 10^4$
	sOR	com-orkut	3M	117M	1.22	20	1.27	1.06	0.69	4	1.13	0.74	$> 10^4$
	sLJ	com-lj	4M	34M	1.44	12	1.54	1.03	1.03	4	1.04	1.04	603
	sYT	com-youtube	1.1M	2.9M	1.67	8	1.84	0.96	1.1	5	0.98	1.11	670
	sDB	com-dblp	317k	1M	1.33	8	1.80	≈ 1	2.5	2	≈ 1	2.53	2160
	sAM	com-amazon	334k	925k	1.14	8	1.62	1.04	1.64	2	1.04	1.64	1426
Purchase network (PNs)	pAM	amazon0601	403k	3.3M	1.45	8	1.91	≈ 1	1.25	3	1.03	1.30	618
Road networks (RNs)	rCA	roadNet-CA	1.9M	5.5M	≈ 1	2	1.59	1.33	1.74	8	1.38	1.80	$> 10^4$
	rTX	roadNet-TX	1.3M	3.8M	≈ 1	2	1.53	1.29	1.89	6	1.42	2.08	$> 10^4$
	rPA	roadNet-PA	1M	3M	≈ 1	2	1.52	≈ 1	2.00	9	1.07	2.16	$> 10^4$
Citation graphs (CGs)	ciP	cit-Patents	3.7M	16.5M	1.16	8	1.57	1.01	1.26	2	1.01	1.26	1875
Web graphs (WGs)	wGL	web-Google	875k	5.1M	1.78	12	2.08	0.98	1.26	6	1.06	1.35	365
	wBS	web-BerkStan	685k	7.6M	1.91	24	1.91	0.93	1.31	5	1.07	1.40	755
	wSF	web-Stanford	281k	2.3M	1.89	24	1.89	0.98	1.54	5	1.07	1.58	1077

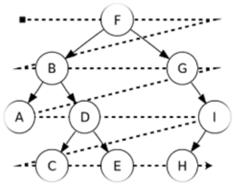
OUTPERFORMING STATE-OF-THE-ART



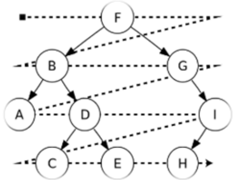
	Input graph properties				BG/Q analysis			Haswell analysis					
Type	ID	Name	$\lvert V\rvert$	$\lvert E\rvert$	S over g500 ($M = 24$)	M	S over g500	S over g500 ($M = 2$)	S over Galois ($M = 2$)	M	S over g500	S over Galois	S over HAMA
Comm. networks (CNs)	cWT	wiki-Talk	2.4M	5M	2.82	48	3.35	0.91	1.22	6	0.96	1.28	344
	cEU	email-EuAll	265k	420k	3.67	32	4.36	0.76	0.88	4	0.97	1.12	1448
Social networks (SNs)	sLV	soc-LiveJ.	4.8M	69M	1.44	12	1.56	1.05	1.1	3	1.07	1.12	$> 10^4$
	sOR	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	1.06	[illegible]	[illegible]	1.12	[illegible]	$> 10^4$
	sLJ	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
	sYT	com-youtube	1.1M	2.9M	1.67	8	1.54	0.96	1.1	5	0.98	1.11	670
	sDB	com-dblp	317k	1M	1.33	8	1.80	≈1	2.5	2	≈1	2.53	2160
	sAM	[illegible]	[illegible]	[illegible]	1.14	8	[illegible]	1.04	1.64	2	1.04	1.64	1426
Purchase network (PNs)	pAM	amazon0601	[illegible]	[illegible]	1.45	8	[illegible]	≈1	1.25	3	1.03	1.50	618
Road networks (RNs)	rCA	roadNet-CA	1.9M	5.5M	≈1	2	1.59	1.33	1.74	8	1.38	1.80	$> 10^4$
	rTX	roadNet-TX	1.3M	3.8M	≈1	2	1.53	1.29	1.89	6	1.42	2.08	$> 10^4$
	rPA	roadNet-PA	1M	3M	≈1	2	1.52	≈1	2.00	9	1.07	2.16	$> 10^4$
Citation graphs (CGs)	ciP	cit-Patents	3.7M	16.5M	1.16	8	1.57	1.01	1.26	2	1.01	1.26	1875
Web graphs (WGs)	wGL	web-Google	875k	5.1M	1.78	12	2.08	0.98	1.26	6	1.06	1.35	365
	wBS	web-BerkStan	685k	7.6M	1.91	24	1.91	0.93	1.31	5	1.07	1.40	755
	wSF	web-Stanford	281k	2.3M	1.89	24	1.89	0.98	1.54	5	1.07	1.58	1077

☺ No, you don't have to read it. All details are in the paper. Here: just a summary.

OUTPERFORMING STATE-OF-THE-ART
HASWELL



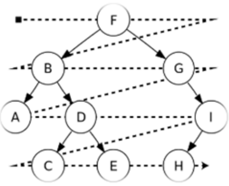
OUTPERFORMING STATE-OF-THE-ART
HASWELL



Average overall speedup (geometric mean) over Graph500: 1.07, Galois: 1.40, HAMA ~1000

Outperforming State-of-the-Art
Haswell (intel)

SNAP



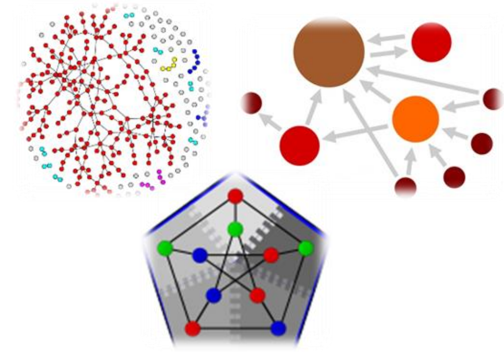
Average overall speedup (geometric mean) over Graph500: 1.07, Galois: 1.40, HAMA ~1000

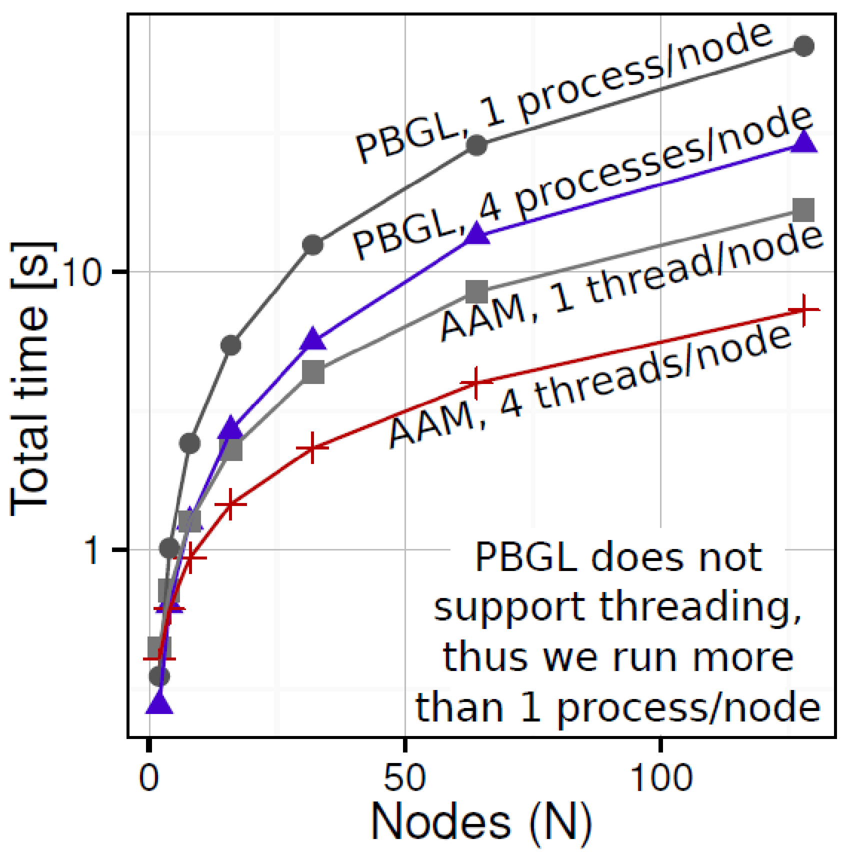
1.85x on average, up to 4.3x

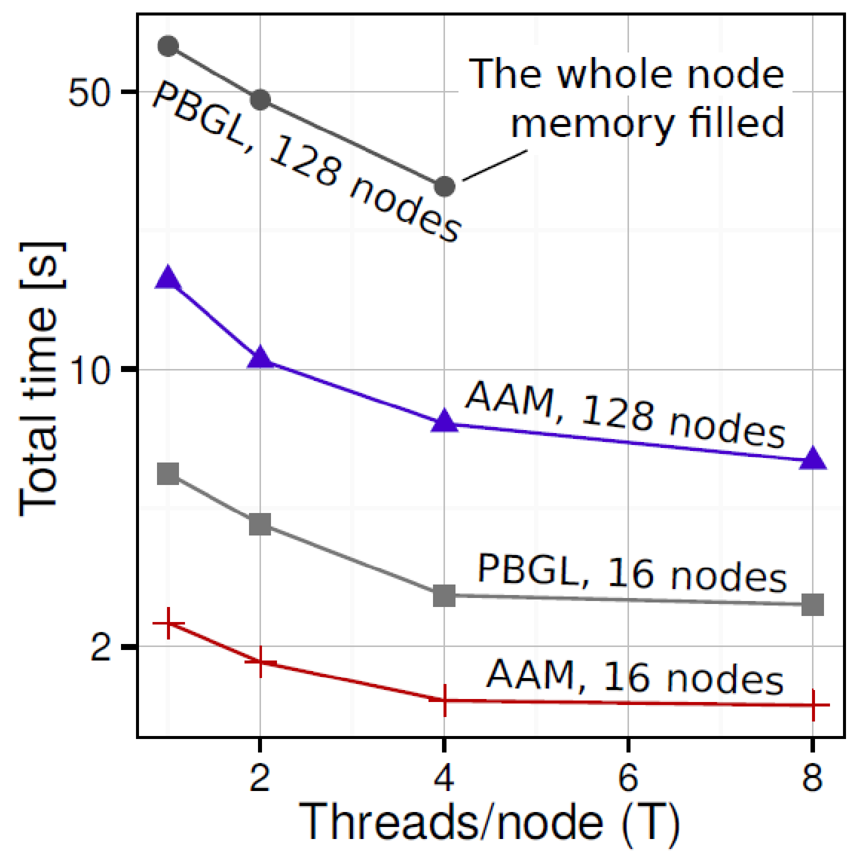
IBM

GRAPH 500

OUTPERFORMING STATE-OF-THE-ART
SCALABILITY ANALYSIS: DISTRIBUTED-MEMORY

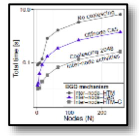


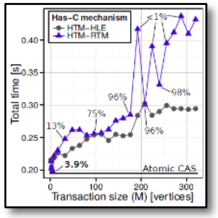


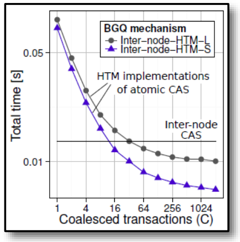


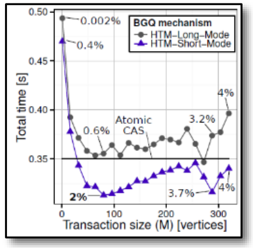
OTHER ANALYSES

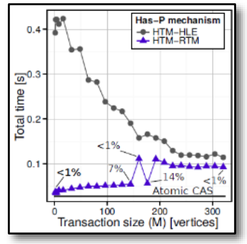
OTHER ANALYSES

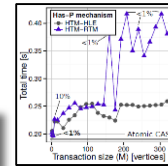


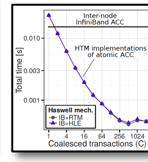


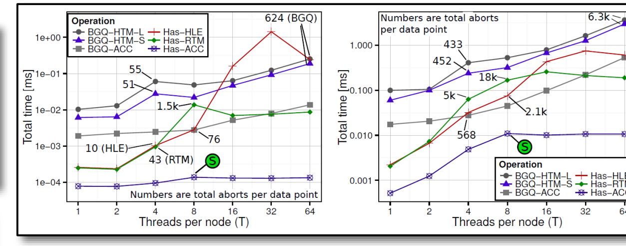


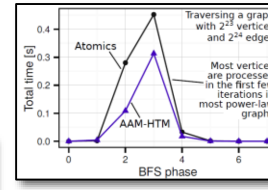


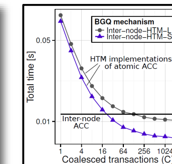


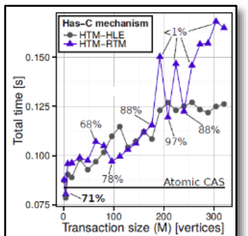


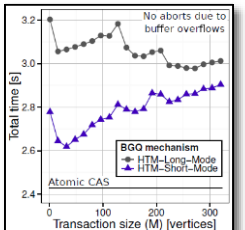


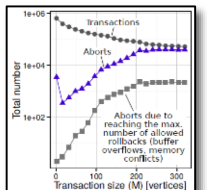


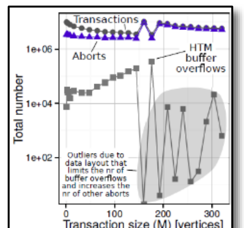


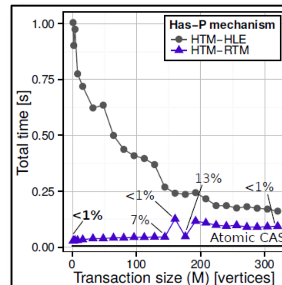






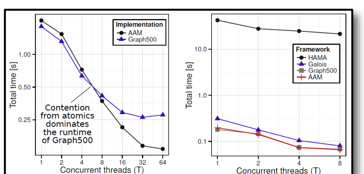


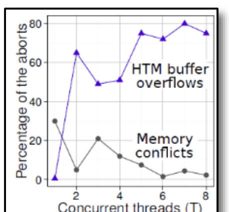


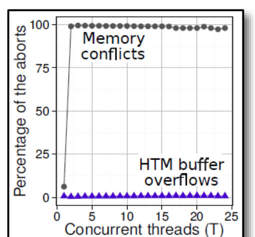


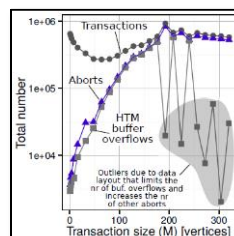
		Memory conflicts	Buffer overflows	Other reasons
10 ops	Has-RTM	1,520	1	0
	BGQ-HTM-L	624	62	614
	BGQ-HTM-S	623	62	613
100 ops	Has-RTM	18,952	33	0
	BGQ-HTM-L	6,374	637	6,360
	BGQ-HTM-S	6,392	639	6,380

Aborts due to:



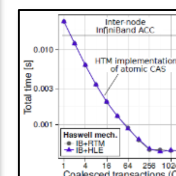


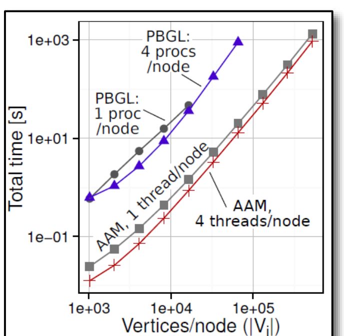


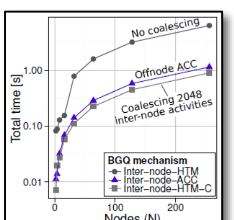


		Memory conflicts	Buffer overflows	Other reasons
10 ops	Has-RTM	2	2	0
	BGQ-HTM-L	802	3	1
	BGQ-HTM-S	1,118	46	180
100 ops	Has-RTM	2	2	0
	BGQ-HTM-L	1,539	5	1
	BGQ-HTM-S	2,242	13	2

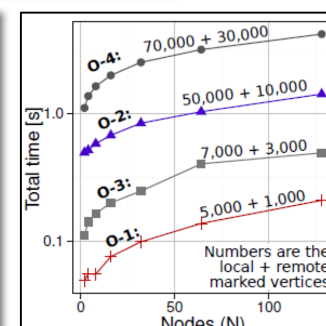
Aborts due to:







Input graph properties					BG/Q analysis			Haswell analysis					
Type	ID	Name	$\|V\|$	$\|E\|$	S over g500 ($M = 24$)	M	S over g500	S over g500 ($M = 2$)	S over Galois ($M = 2$)	M	S over g500	S over Galois	S over HAMA
Comm. networks (CNs)	cWT	wiki-Talk	2.4M	5M	2.82	48	3.35	0.91	1.22	6	0.96	1.28	344
	cEU	email-EuAll	265k	420k	3.67	32	4.36	0.76	0.88	4	0.97	1.12	1448
Social networks (SNs)	sLV	soc-LiveJ.	4.8M	69M	1.44	12	1.56	1.05	1.1	3	1.07	1.12	$> 10^4$
	sOR	com-orkut	3M	117M	1.22	20	1.27	1.06	0.69	4	1.13	0.74	$> 10^4$
	sLJ	com-lj	4M	34M	1.44	12	1.54	1.03	1.03	4	1.04	1.04	603
	sYT	com-youtube	1.1M	2.9M	1.67	8	1.84	0.96	1.1	5	0.98	1.11	670
	sDB	com-dblp	317k	1M	1.33	8	1.80	≈1	2.5	2	≈1	2.53	2160
	sAM	com-amazon	334k	925k	1.14	8	1.62	1.04	1.64	2	1.04	1.64	1426
Purchase network (PNs)	pAM	amazon0601	403k	3.3M	1.45	8	1.91	≈1	1.25	3	1.03	1.30	618
Road networks (RNs)	rCA	roadNet-CA	1.9M	5.5M	≈1	2	1.59	1.33	1.74	8	1.38	1.80	$> 10^4$
	rTX	roadNet-TX	1.3M	3.8M	≈1	2	1.53	1.29	1.89	6	1.42	2.08	$> 10^4$
	rPA	roadNet-PA	1M	3M	≈1	2	1.52	≈1	2.00	9	1.07	2.16	$> 10^4$
Citation graphs (CGs)	ciP	cit-Patents	3.7M	16.5M	1.16	8	1.57	1.01	1.26	2	1.01	1.26	1875
Web graphs (WGs)	wGL	web-Google	875k	5.1M	1.78	12	2.08	0.98	1.26	6	1.06	1.35	365
	wBS	web-BerkStan	685k	7.6M	1.91	24	1.91	0.93	1.31	5	1.07	1.40	755
	wSF	web-Stanford	281k	2.3M	1.89	24	1.89	0.98	1.54	5	1.07	1.58	1077



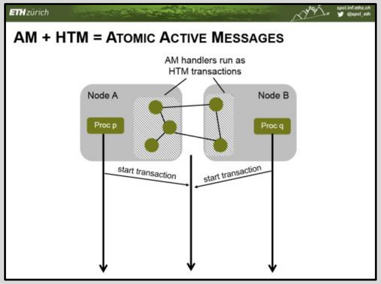
CONCLUSIONS

CONCLUSIONS

HTM for graphs in SM & DM environments

CONCLUSIONS

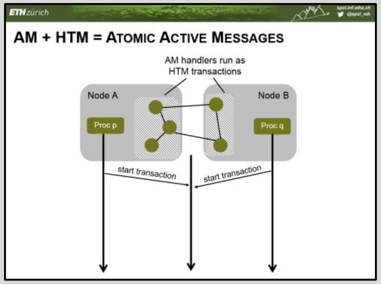
HTM for graphs in SM & DM environments



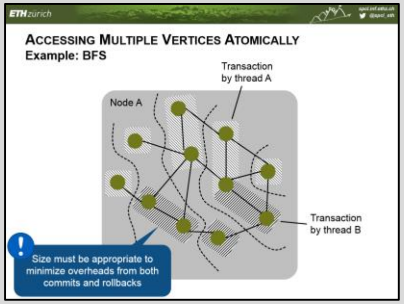
AAM: Combine the advantages of HTM and Active Messages

CONCLUSIONS

HTM for graphs in SM & DM environments



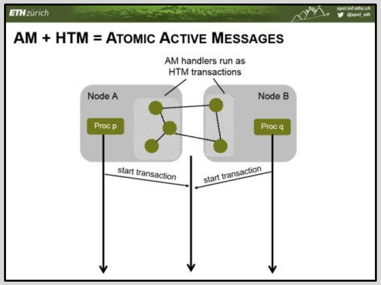
AAM: Combine the advantages of HTM and Active Messages



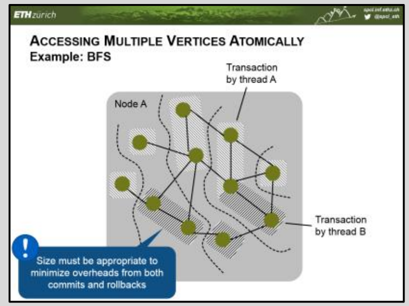
Illustrate HTM's advantages in performance, next to programmability

CONCLUSIONS

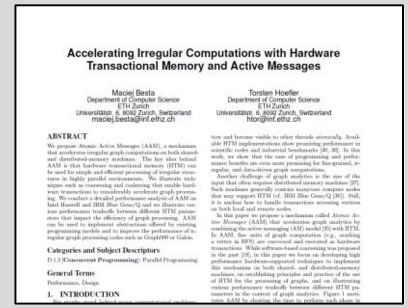
HTM for graphs in SM & DM environments



AAM: Combine the advantages of HTM and Active Messages



Illustrate HTM's advantages in performance, next to programmability


Accelerating Irregular Computations with Hardware Transactional Memory and Active Messages

Maciej Besta
Department of Computer Science
ETH Zurich

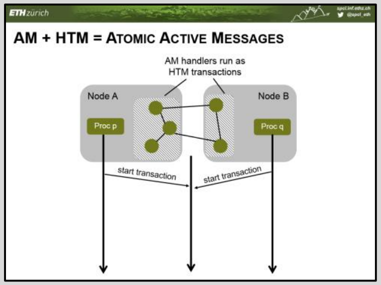
Torsten Hoefler
Department of Computer Science
ETH Zurich

ABSTRACT

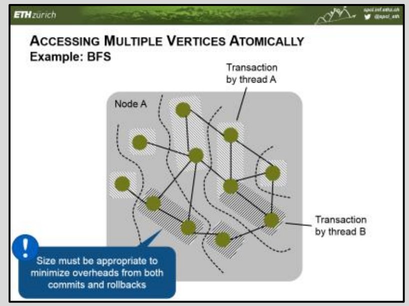
Deliver the of hierarchy of atomic messages that covers various graph algorithms

CONCLUSIONS

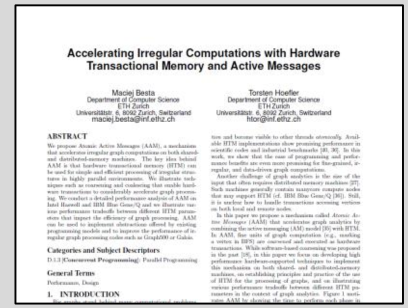
HTM for graphs in SM & DM environments



AAM: Combine the advantages of HTM and Active Messages



Illustrate HTM's advantages in performance, next to programmability


Accelerating Irregular Computations with Hardware Transactional Memory and Active Messages

Maciej Besta
Department of Computer Science
ETH Zurich

Torsten Hoefler
Department of Computer Science
ETH Zurich

ABSTRACT

Categories and Subject Descriptors

General Terms

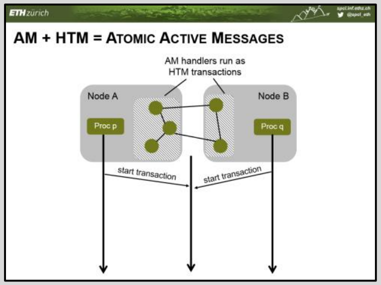
1. INTRODUCTION

Deliver the of hierarchy of atomic messages that covers various graph algorithms

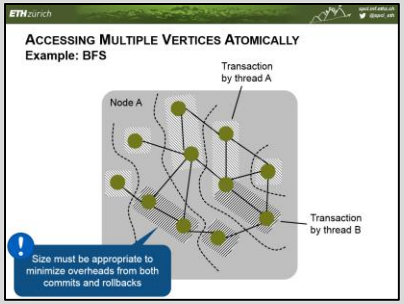
Detailed performance analysis

CONCLUSIONS

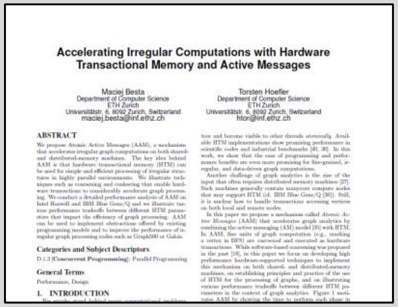
HTM for graphs in SM & DM environments



AAM: Combine the advantages of HTM and Active Messages



Illustrate HTM's advantages in performance, next to programmability


Accelerating Irregular Computations with Hardware Transactional Memory and Active Messages

Maciej Besta
Department of Computer Science
ETH Zurich
Universitätstr. 6, 8092 Zurich, Switzerland
maciej.besta@inf.ethz.ch

Torsten Hoefler
Department of Computer Science
ETH Zurich
Universitätstr. 6, 8092 Zurich, Switzerland
htor@inf.ethz.ch

ABSTRACT

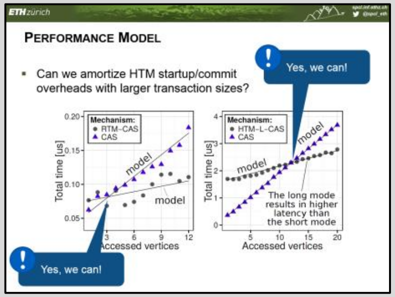
Categories and Subject Descriptors

General Terms

1. INTRODUCTION

Deliver the of hierarchy of atomic messages that covers various graph algorithms

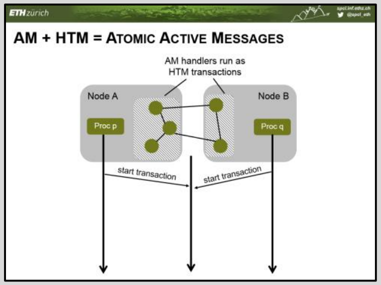
Detailed performance analysis



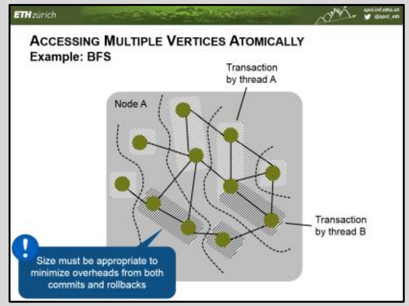
Model & analyze performance tradeoffs

Conclusions

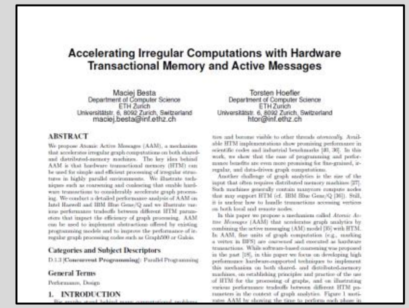
HTM for graphs in SM & DM environments



AAM: Combine the advantages of HTM and Active Messages

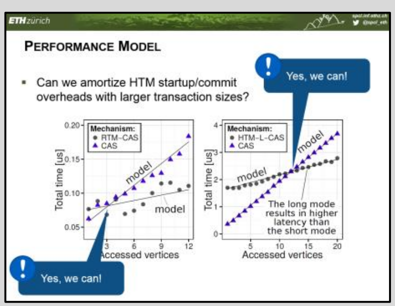


Illustrate HTM's advantages in performance, next to programmability



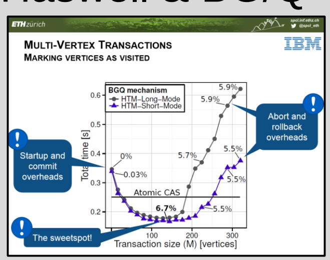
Deliver the of hierarchy of atomic messages that covers various graph algorithms

Detailed performance analysis



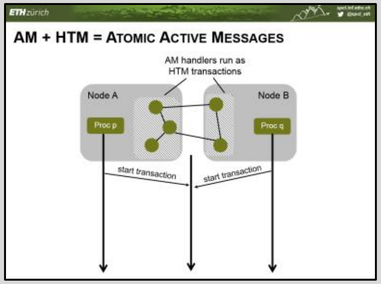
Model & analyze performance tradeoffs

Derive close-to-optimal transaction sizes for Haswell & BG/Q

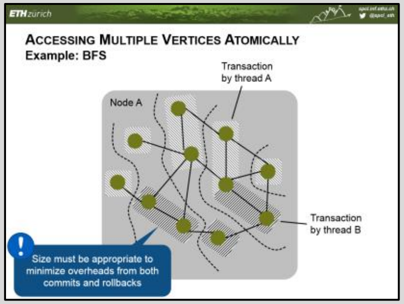


CONCLUSIONS

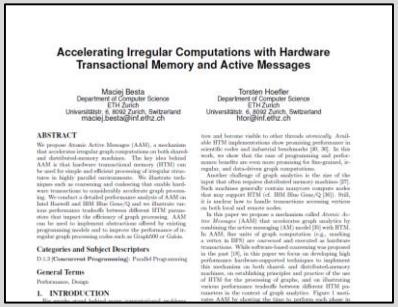
HTM for graphs in SM & DM environments



AAM: Combine the advantages of HTM and Active Messages



Illustrate HTM's advantages in performance, next to programmability


Accelerating Irregular Computations with Hardware Transactional Memory and Active Messages

Maciej Besta
Department of Computer Science
ETH Zurich
maciej.besta@inf.ethz.ch

Torsten Hoefler
Department of Computer Science
ETH Zurich
htor@inf.ethz.ch

ABSTRACT

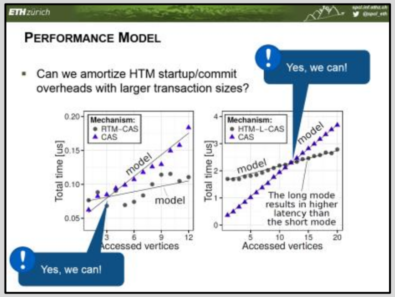
Categories and Subject Descriptors

General Terms

1. INTRODUCTION

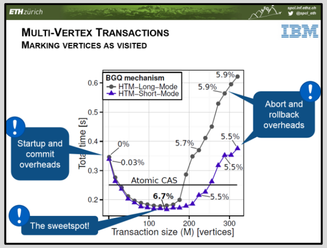
Deliver the of hierarchy of atomic messages that covers various graph algorithms

Detailed performance analysis

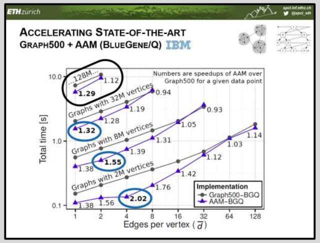


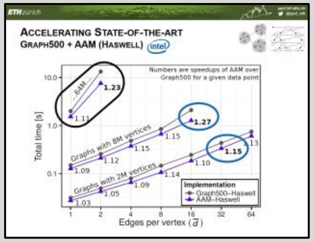
Model & analyze performance tradeoffs

Derive close-to-optimal transaction sizes for Haswell & BG/Q



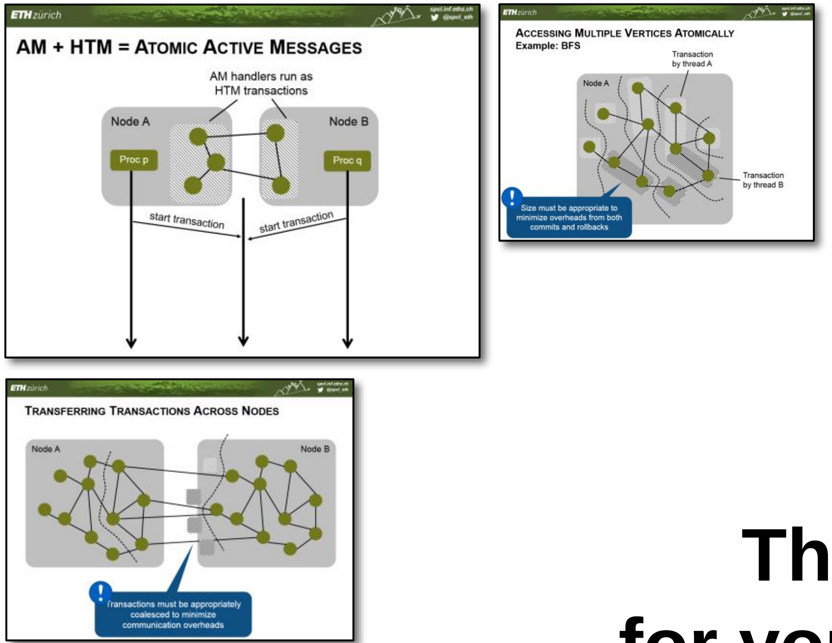
Accelerating state-of-the-art

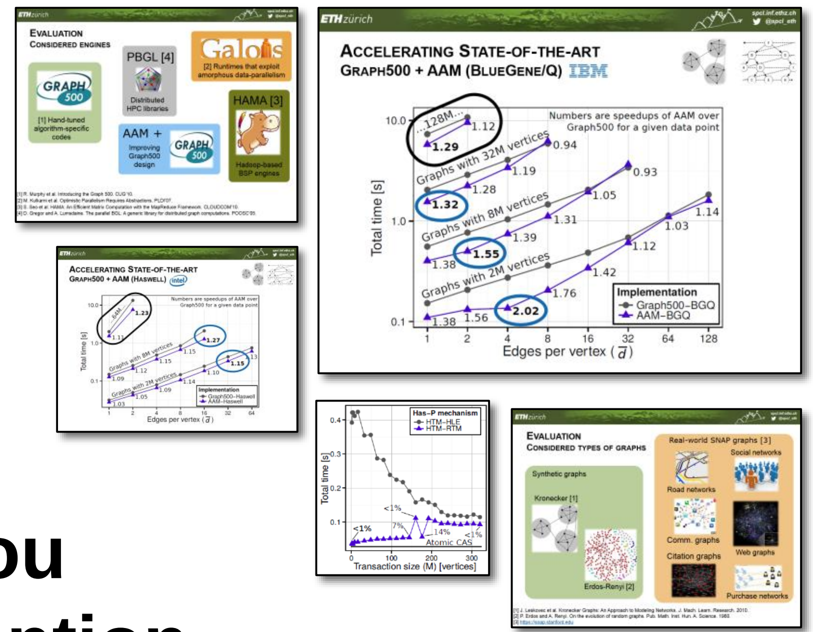




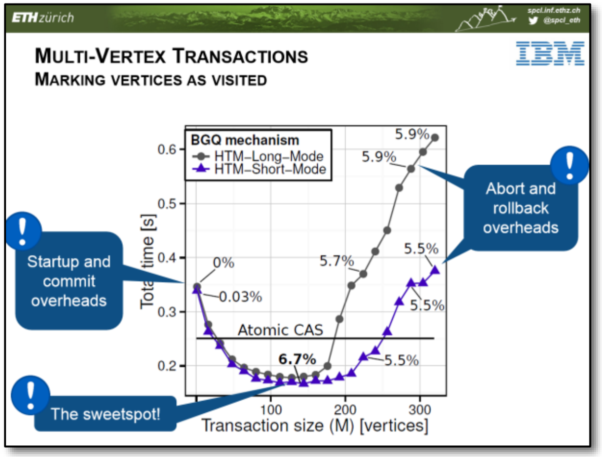
Average speedup 1.85x

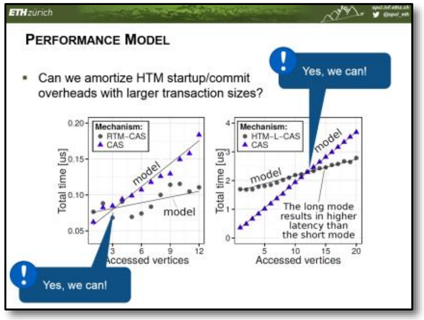
Up to 4x

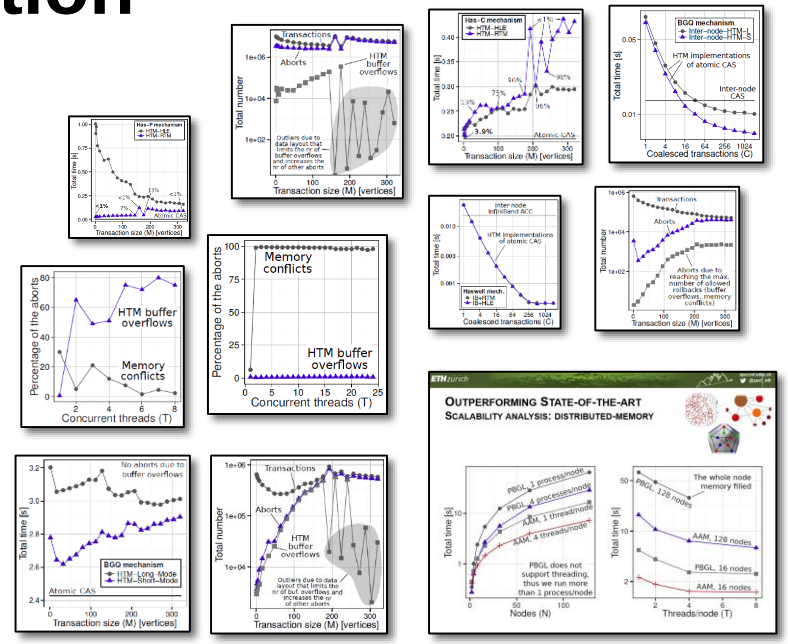




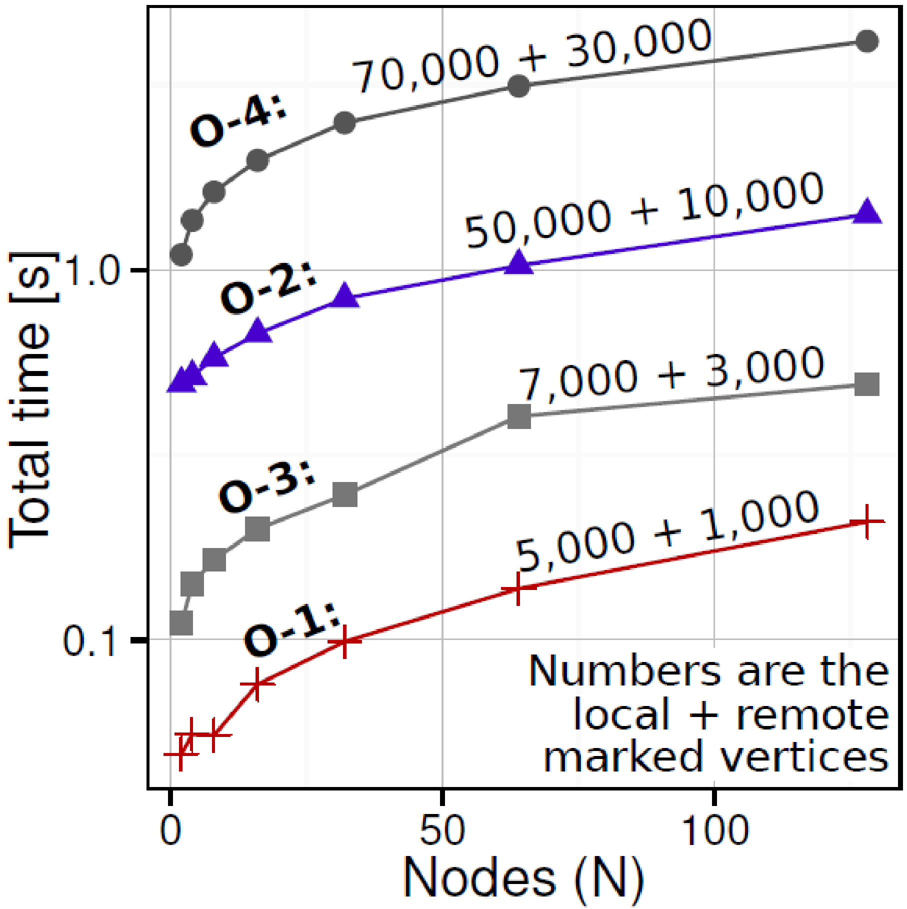
Thank you for your attention



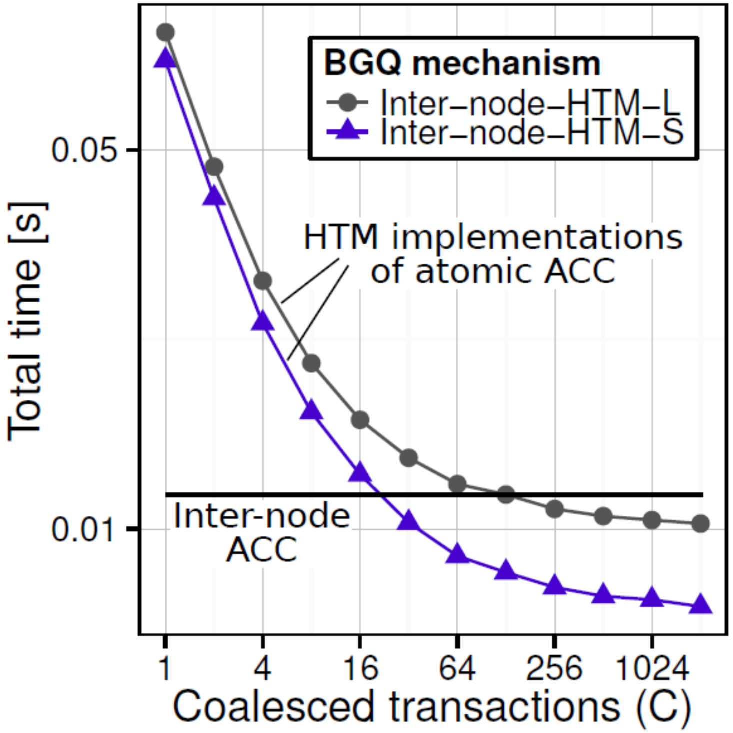


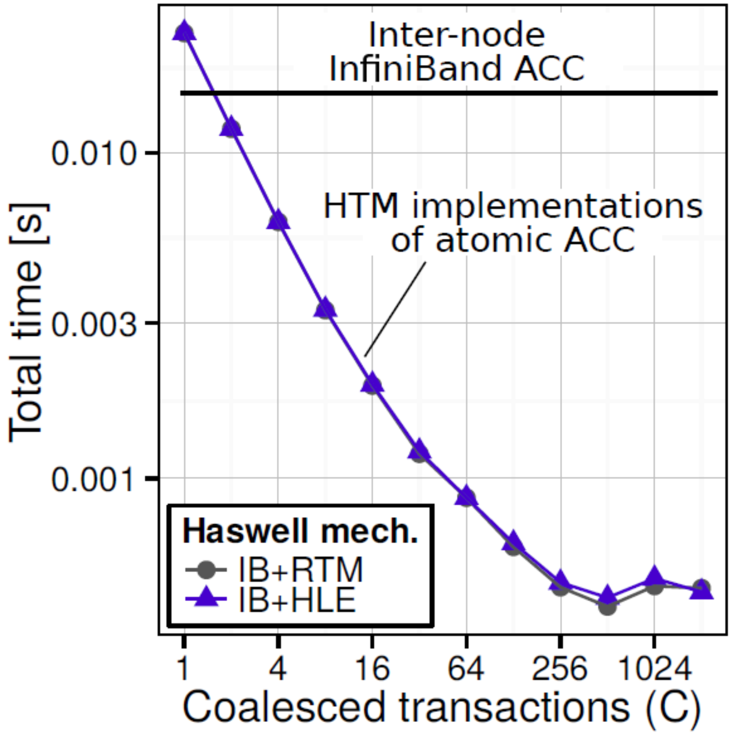


DISTRIBUTED HTM TRANSACTIONS



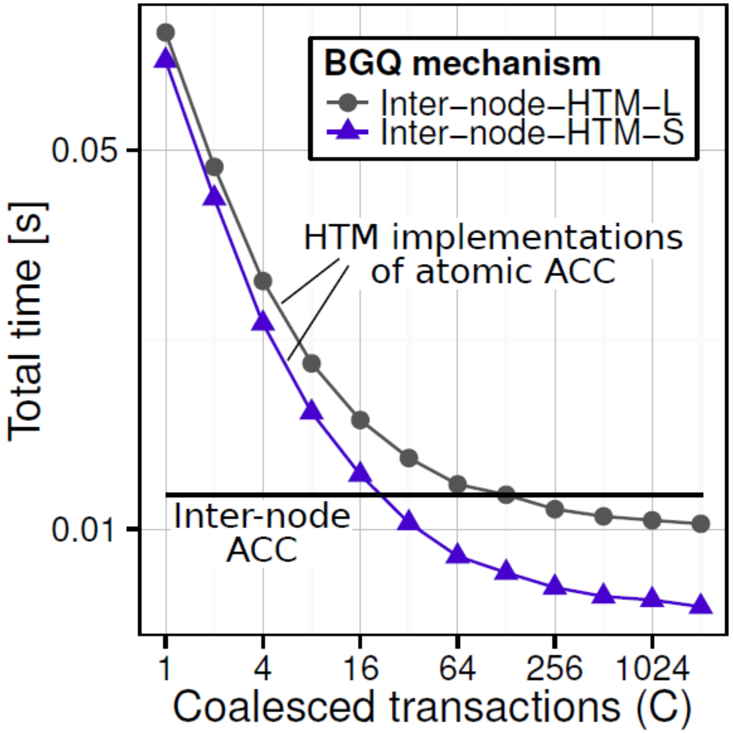
Transferring Transactions
Incrementing ranks of vertices

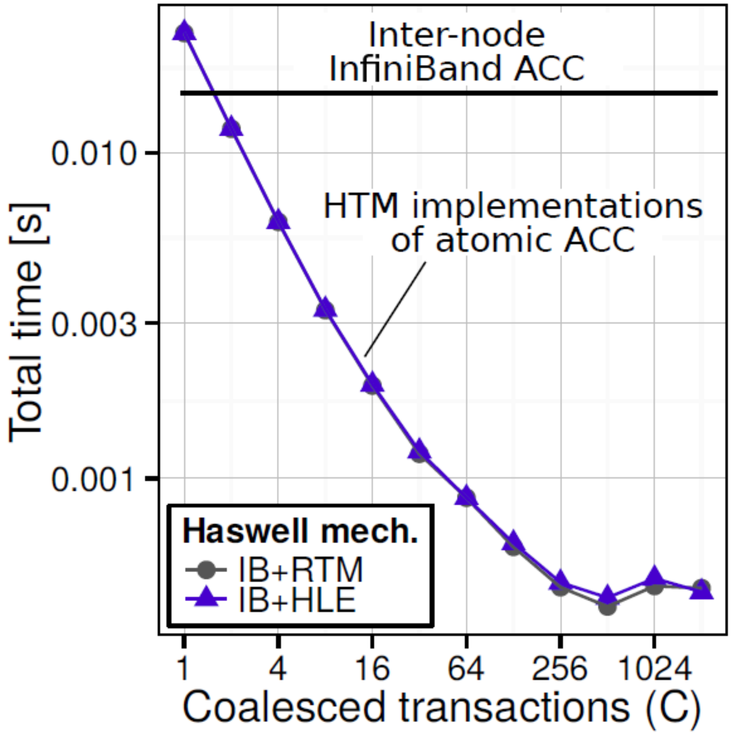




TRANSFERRING TRANSACTIONS
INCREMENTING RANKS OF VERTICES

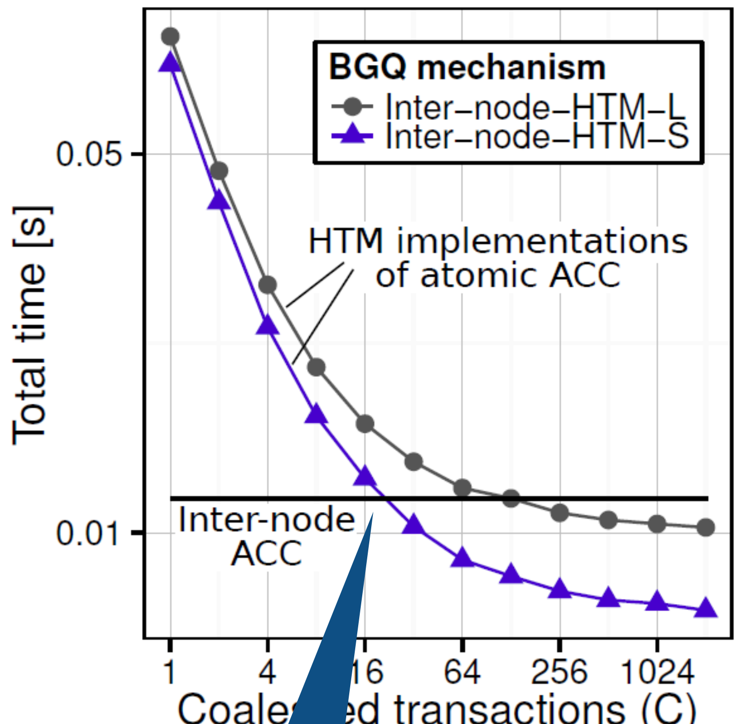
- Can we amortize HTM transactions' transfer overheads with coalescing?

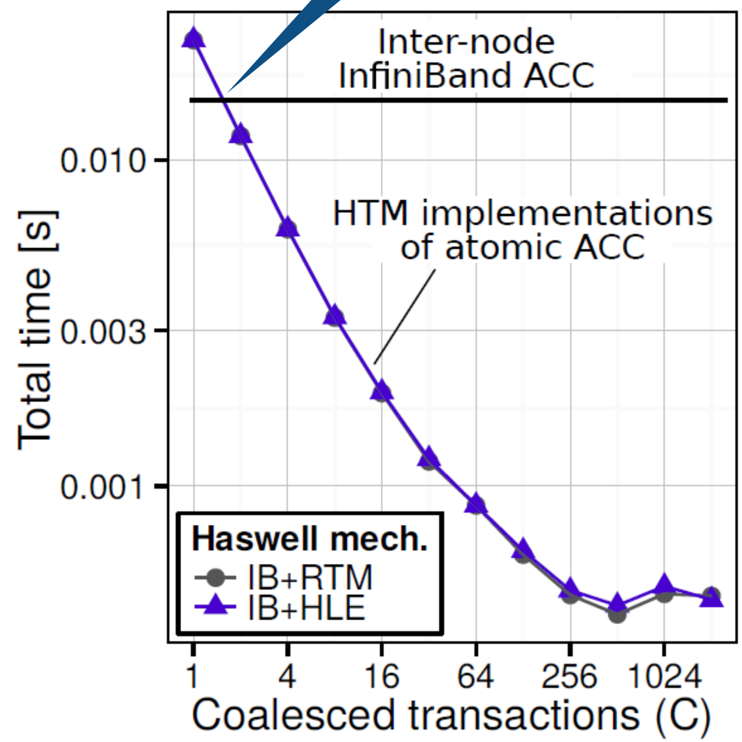




Transferring Transactions
Incrementing ranks of vertices

- Can we amortize HTM transactions' transfer overheads with coalescing?





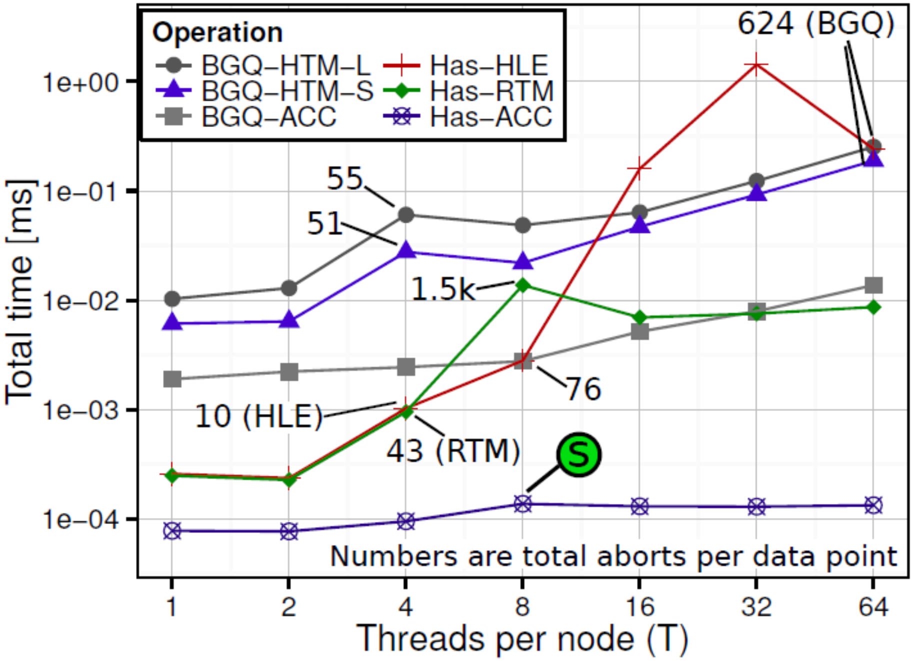
Yes, we can!

Yes, we can!

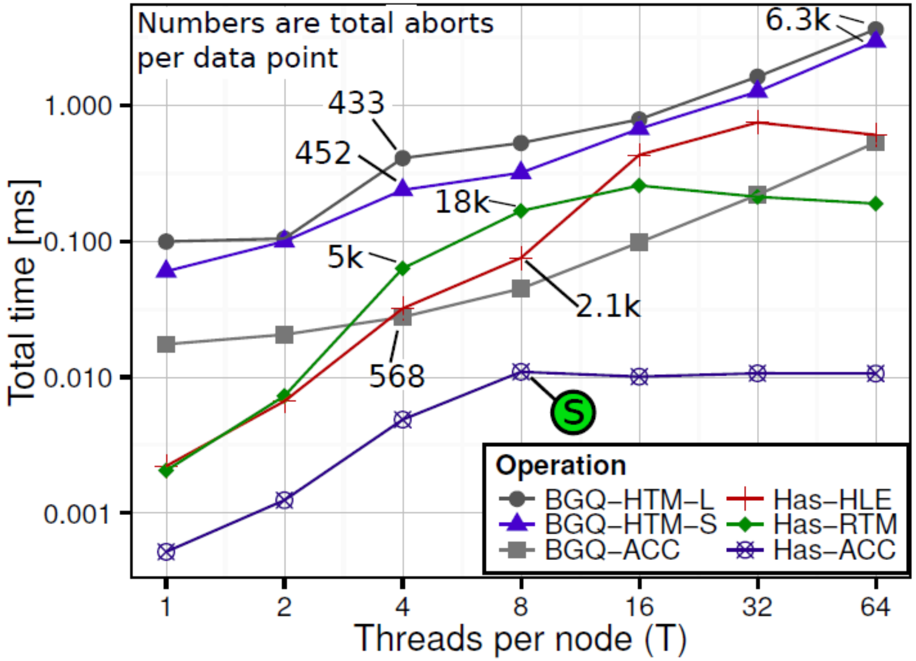
SINGLE-VERTEX TRANSACTIONS
INCREMENTING VERTEX RANK

Used in PageRank

Lower contention
(10 accesses/vertex)



Higher contention
(100 accesses/vertex)



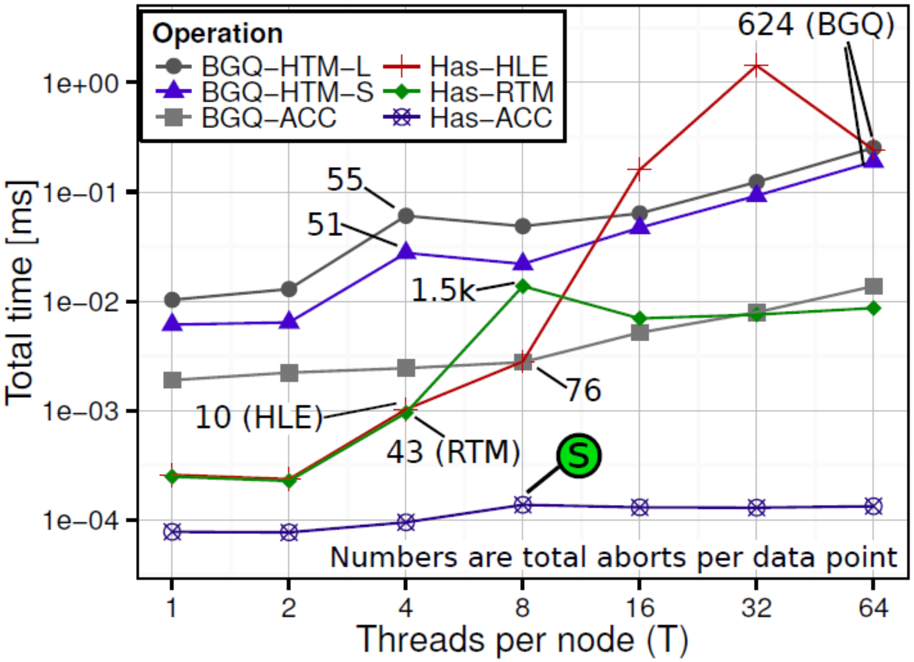
SINGLE-VERTEX TRANSACTIONS
INCREMENTING VERTEX RANK

Used in PageRank

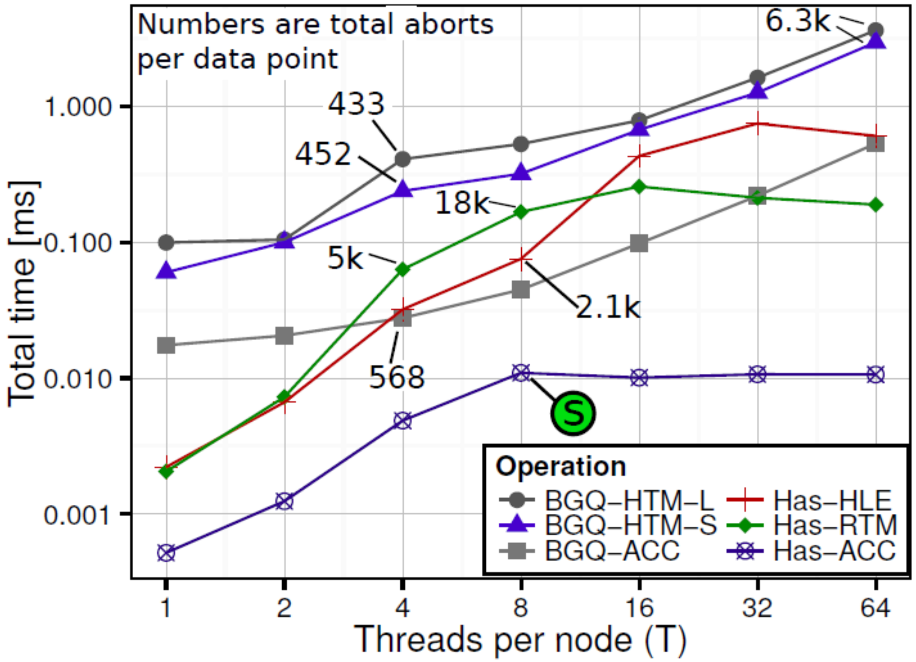
IBM intel

Atomics always outperform HTM

Lower contention (10 accesses/vertex)



Higher contention (100 accesses/vertex)

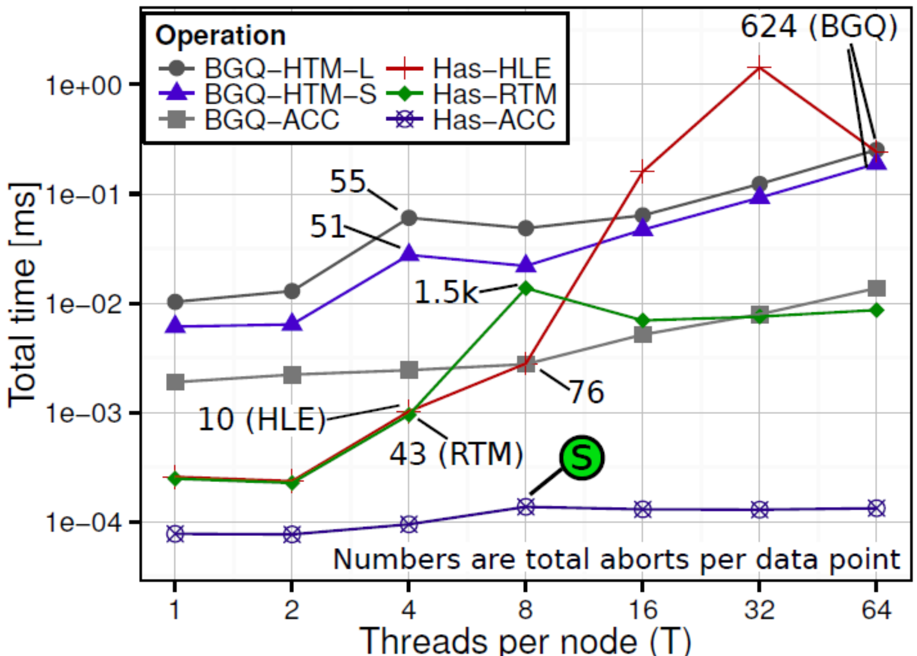


SINGLE-VERTEX TRANSACTIONS
INCREMENTING VERTEX RANK

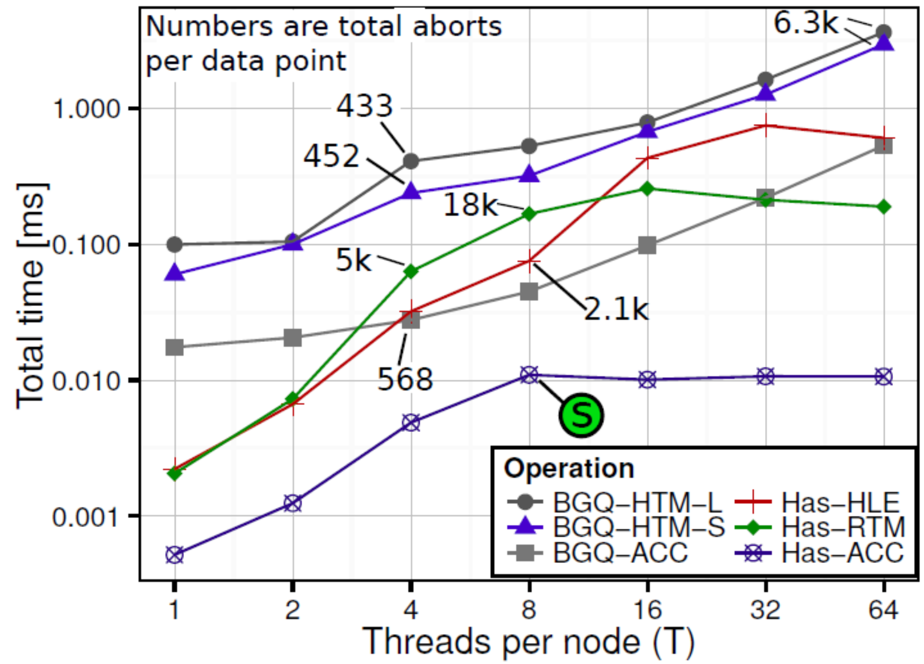
Used in PageRank

Atomics always outperform HTM

Lower contention (10 accesses/vertex)



Higher contention (100 accesses/vertex)



The reason: each transaction always modifies some memory cell, increasing the number of conflicts

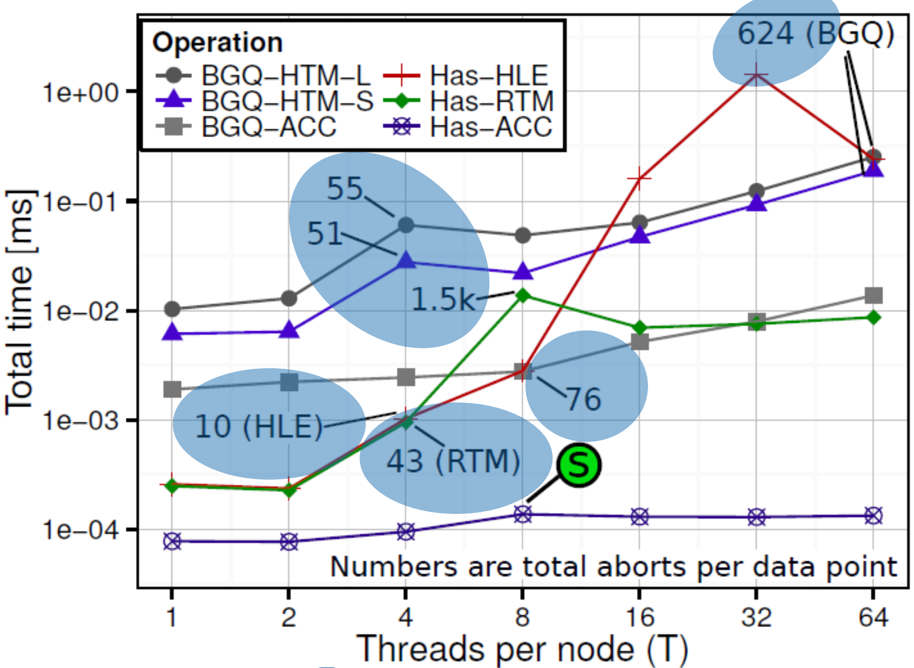
Single-Vertex Transactions
Incrementing vertex rank

Used in PageRank

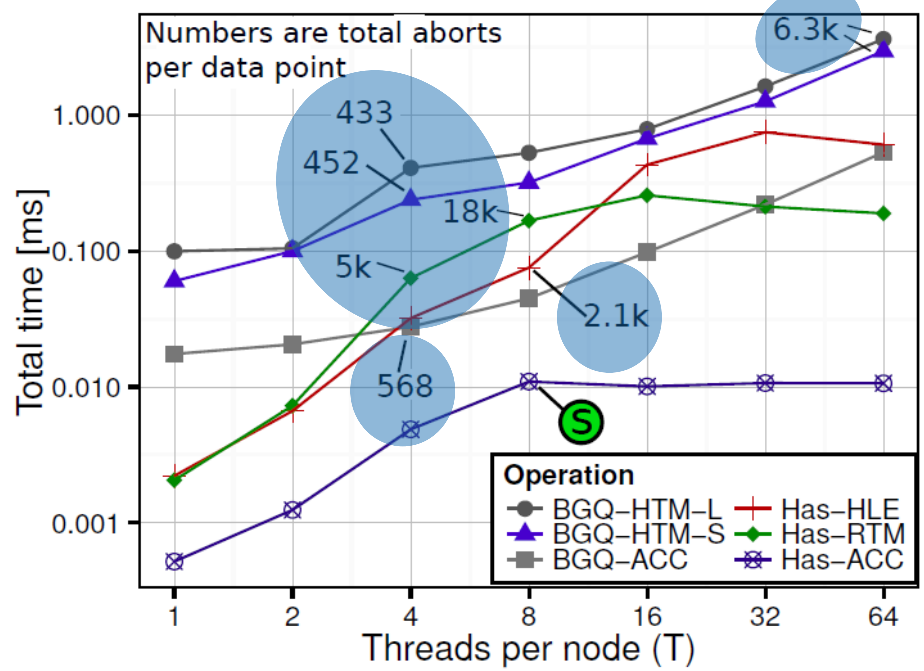
More aborts

Atomics always outperform HTM

Lower contention (10 accesses/vertex)



Higher contention (100 accesses/vertex)

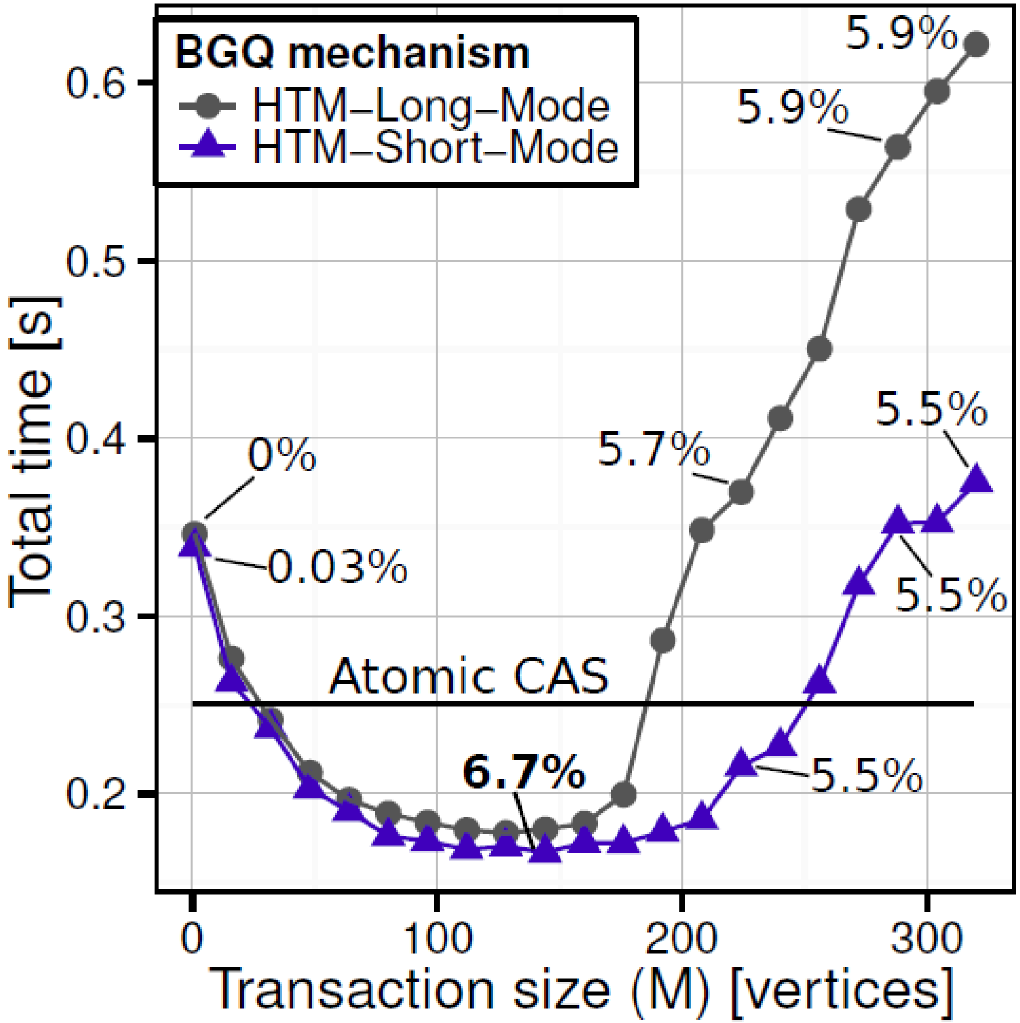


The reason: each transaction always modifies some memory cell, increasing the number of conflicts

MULTI-VERTEX TRANSACTIONS

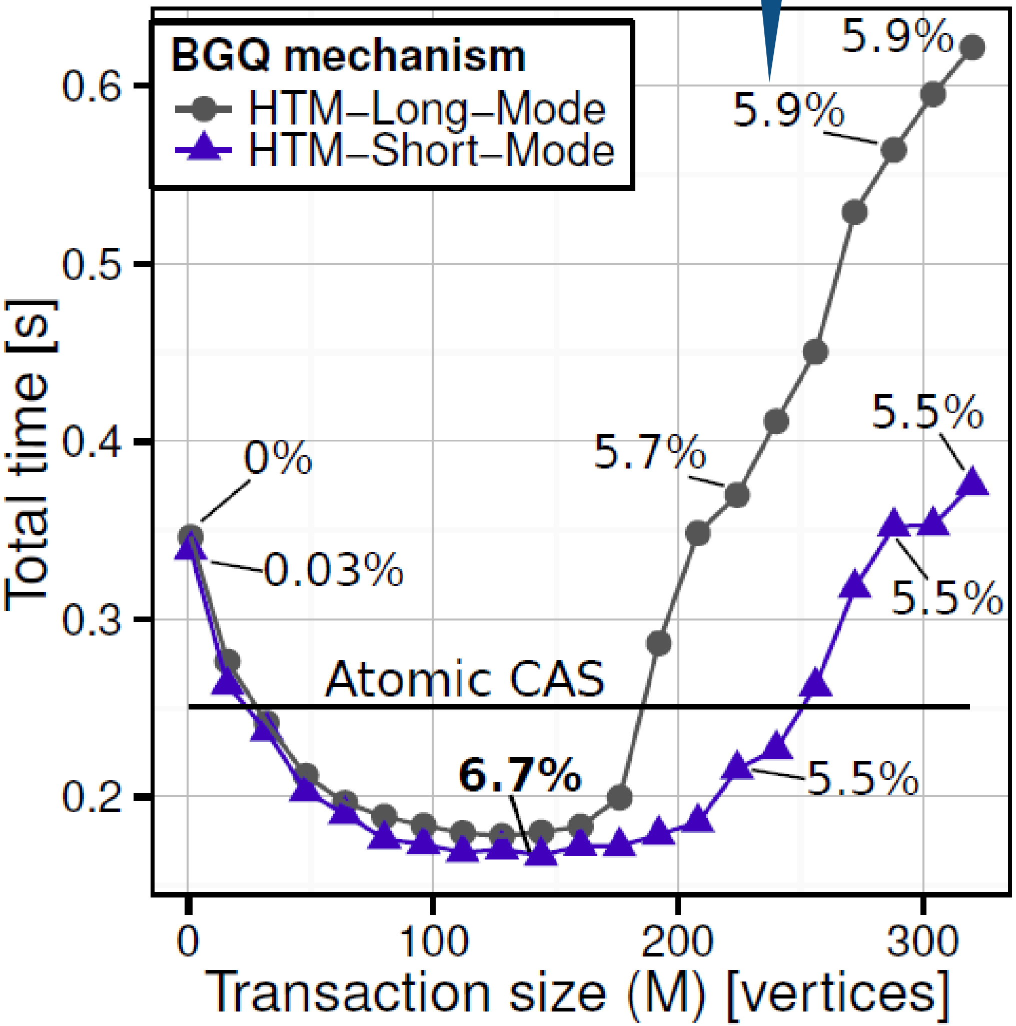
MARKING VERTICES AS VISITED

MULTI-VERTEX TRANSACTIONS
MARKING VERTICES AS VISITED



Multi-Vertex Transactions
Marking vertices as visited

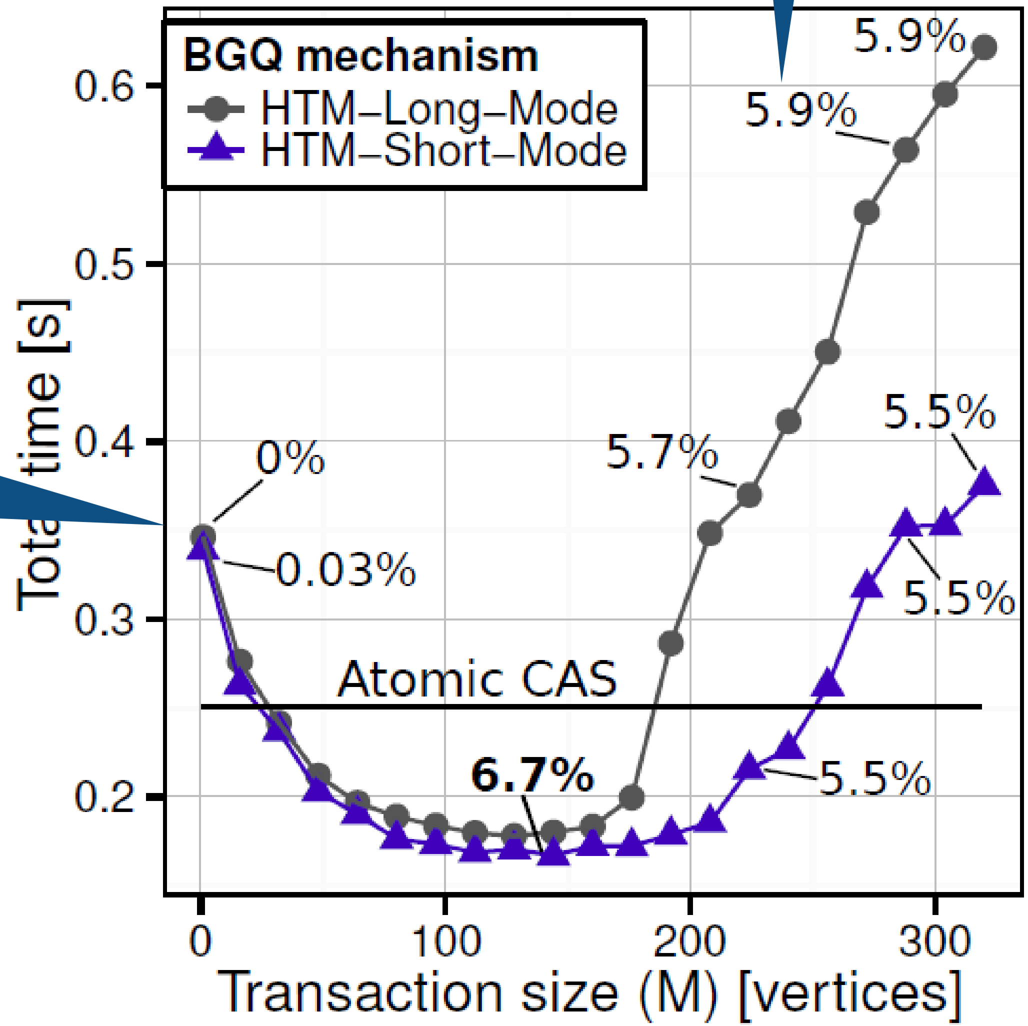
Numbers: % of aborts due to the lack of HTM resources + memory conflicts



MULTI-VERTEX TRANSACTIONS
MARKING VERTICES AS VISITED

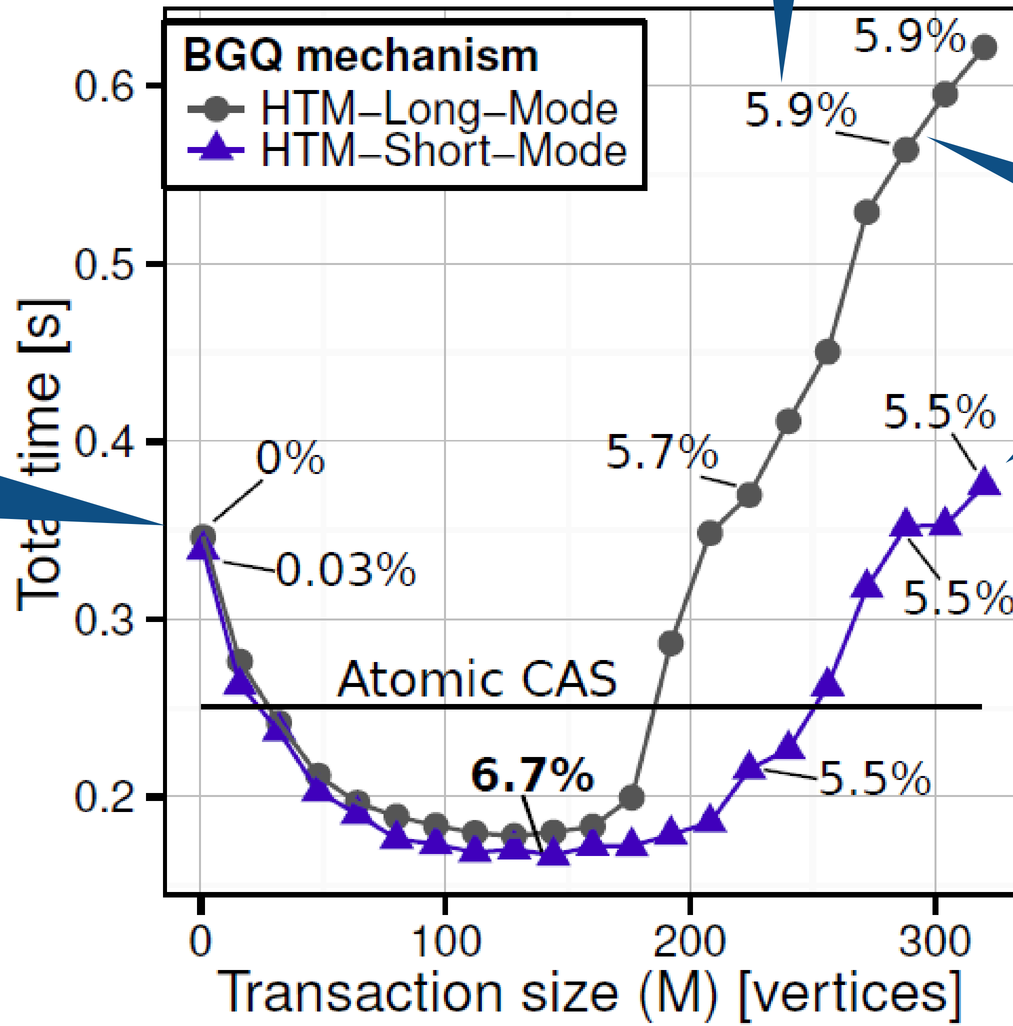
Numbers: % of aborts due to the lack of HTM resources + memory conflicts

Startup and commit overheads



MULTI-VERTEX TRANSACTIONS
MARKING VERTICES AS VISITED

Numbers: % of aborts due to the lack of HTM resources + memory conflicts

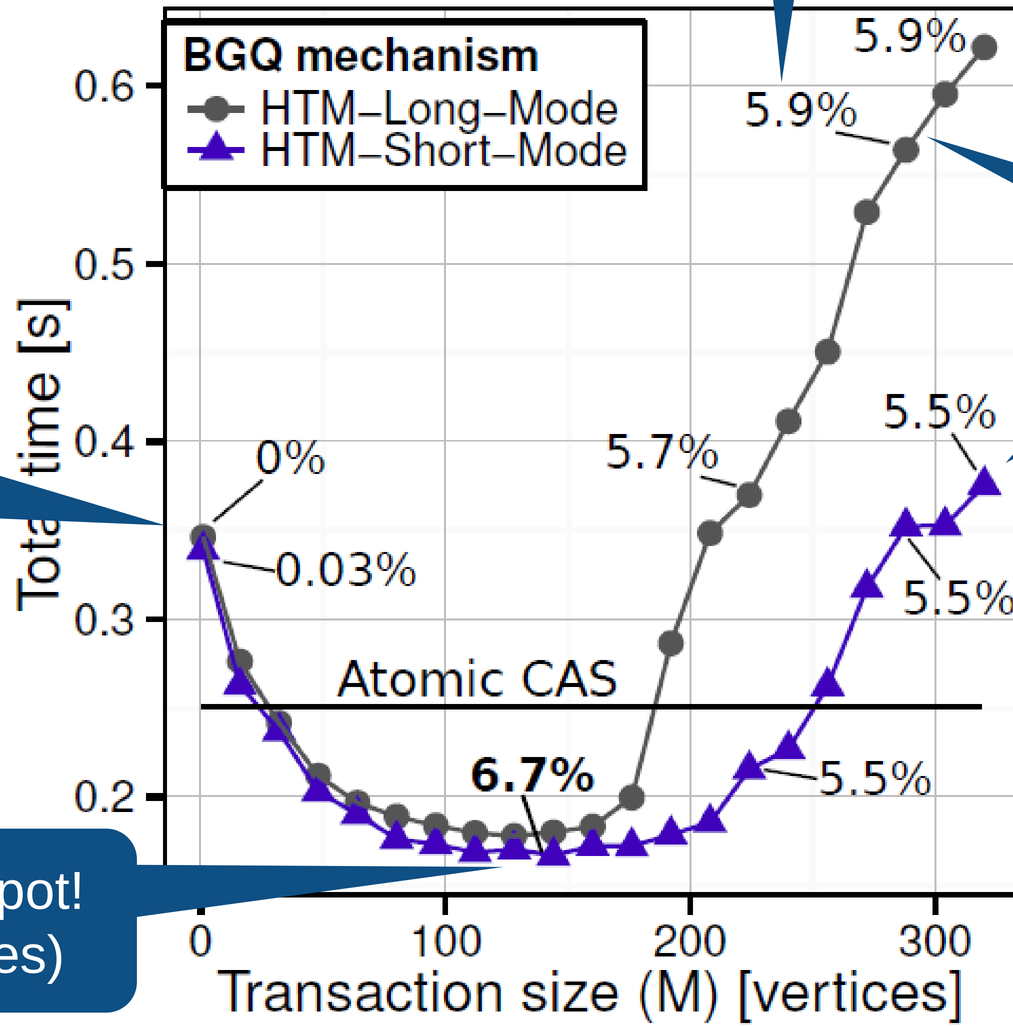


Startup and commit overheads

Abort and rollback overheads

MULTI-VERTEX TRANSACTIONS
MARKING VERTICES AS VISITED

Numbers: % of aborts due to the lack of HTM resources + memory conflicts



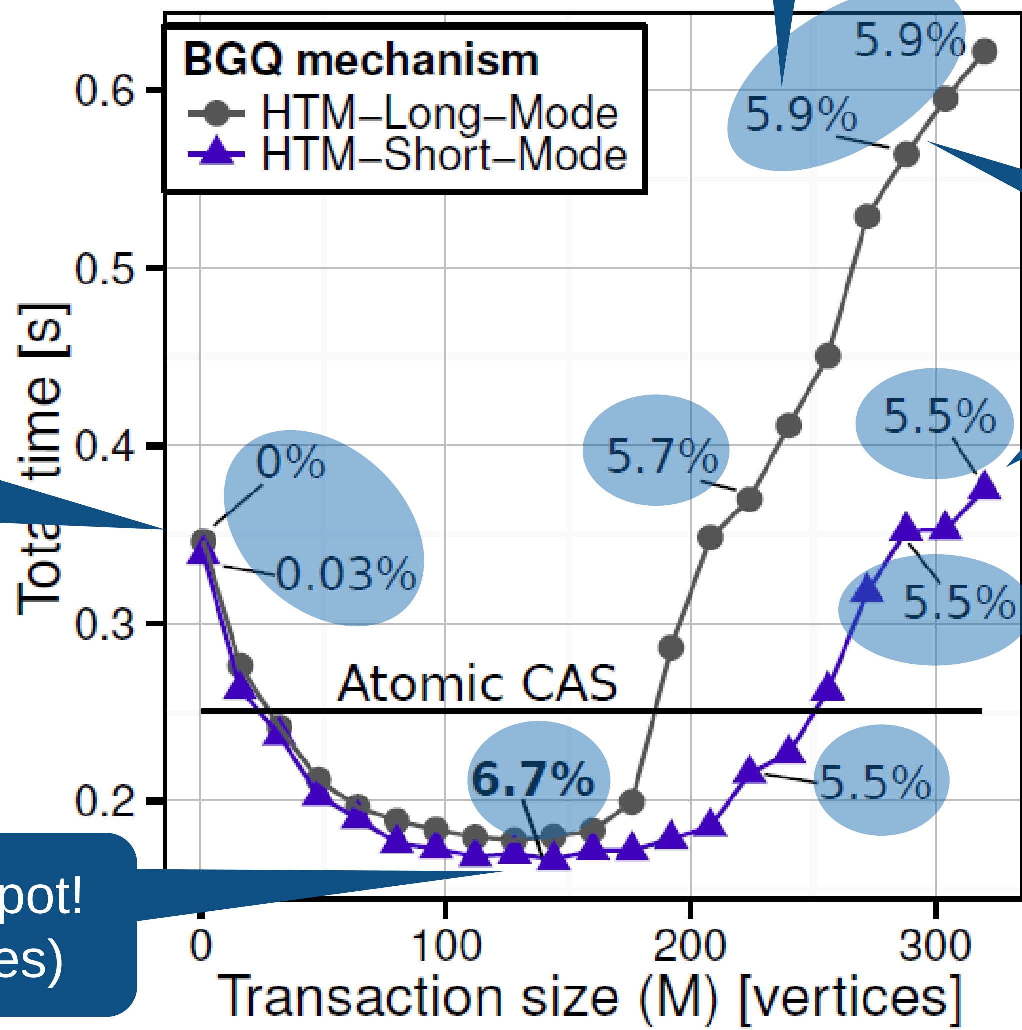
Abort and rollback overheads

Startup and commit overheads

The sweetspot! (144 vertices)

MULTI-VERTEX TRANSACTIONS
MARKING VERTICES AS VISITED

Numbers: % of aborts due to the lack of HTM resources + memory conflicts



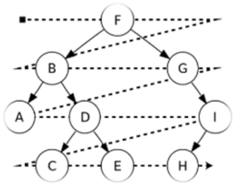
Abort and rollback overheads

Startup and commit overheads

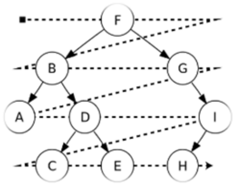
Not too many aborts due to the lack of HW resources (large cache size & associativity)

The sweetspot! (144 vertices)

Outperforming State-of-the-art BlueGene/Q



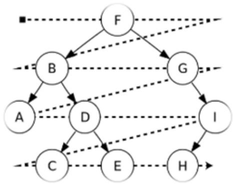
OUTPERFORMING STATE-OF-THE-ART
BLUEGENE/Q IBM



Average overall speedup over Graph500 (geometric mean): 1.51 (1.85)

Outperforming State-of-the-Art

BlueGene/Q IBM





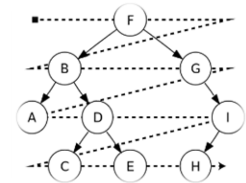


Average overall speedup over Graph500 (geometric mean): 1.51 (1.85)

OUTPERFORMING STATE-OF-THE-ART

BLUEGENE/Q IBM

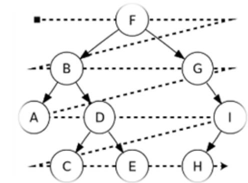
SNAP



Average overall speedup over Graph500 (geometric mean): 1.51 (1.85)

The same transaction size for all graphs

Outperforming State-of-the-Art
BlueGene/Q IBM

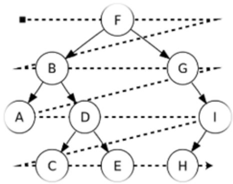


Average overall speedup over Graph500 (geometric mean): 1.51 (1.85)

The same transaction size for all graphs

The same transaction sizes for each graph separately

OUTPERFORMING STATE-OF-THE-ART
BLUEGENE/Q IBM



Average speedup: 1

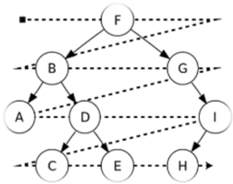
Average overall speedup over Graph500 (geometric mean): 1.51 (1.85)

The same transaction size for all graphs

The same transaction sizes for each graph separately

Outperforming State-of-the-Art

BlueGene/Q IBM



Average speedup: 1

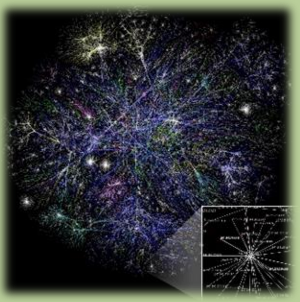




Average overall speedup over Graph500 (geometric mean): 1.51 (1.85)

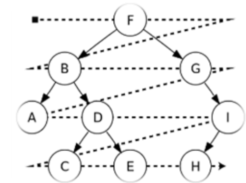
The same transaction size for all graphs

The same transaction sizes for each graph separately

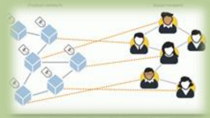


Average speedup: 1.85

Outperforming State-of-the-Art
BlueGene/Q IBM



Average speedup: 1

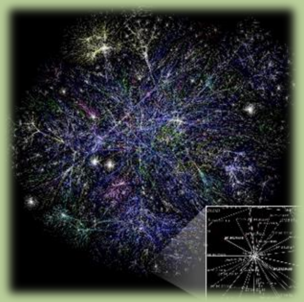


Average overall speedup over Graph500 (geometric mean): 1.51 (1.85)

The same transaction size for all graphs

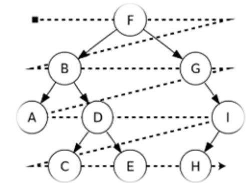
The same transaction sizes for each graph separately

Average speedup: 3.20



Average speedup: 1.85

Outperforming State-of-the-Art
BlueGene/Q IBM

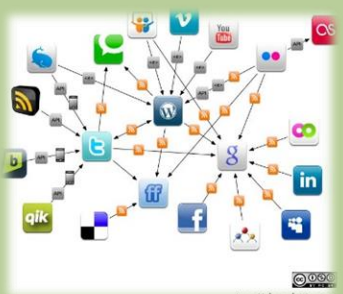


Average speedup: 1

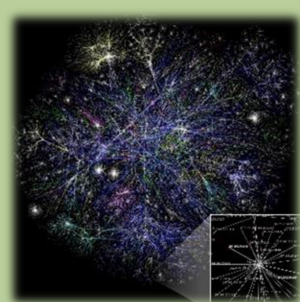
Average overall speedup over Graph500 (geometric mean): 1.51 (1.85)

The same transaction size for all graphs

The same transaction sizes for each graph separately



Average speedup: 3.20



Average speedup: 1.85

!

Best transaction size: ~24-100 vertices accessed

SINGLE-VERTEX TRANSACTIONS

MARKING A VERTEX AS VISITED

Single-Vertex Transactions

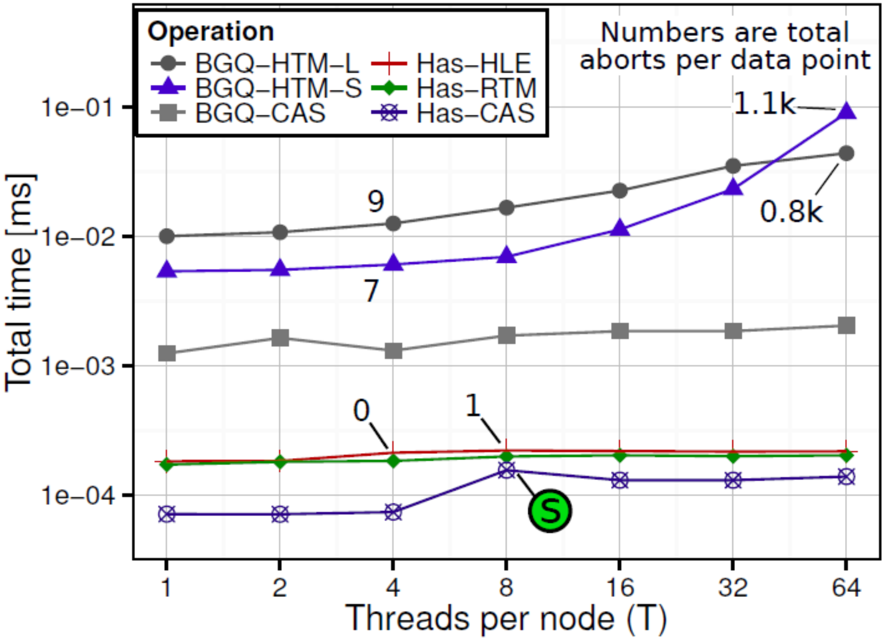
Marking a vertex as visited

Used in BFS, SSSP, ...

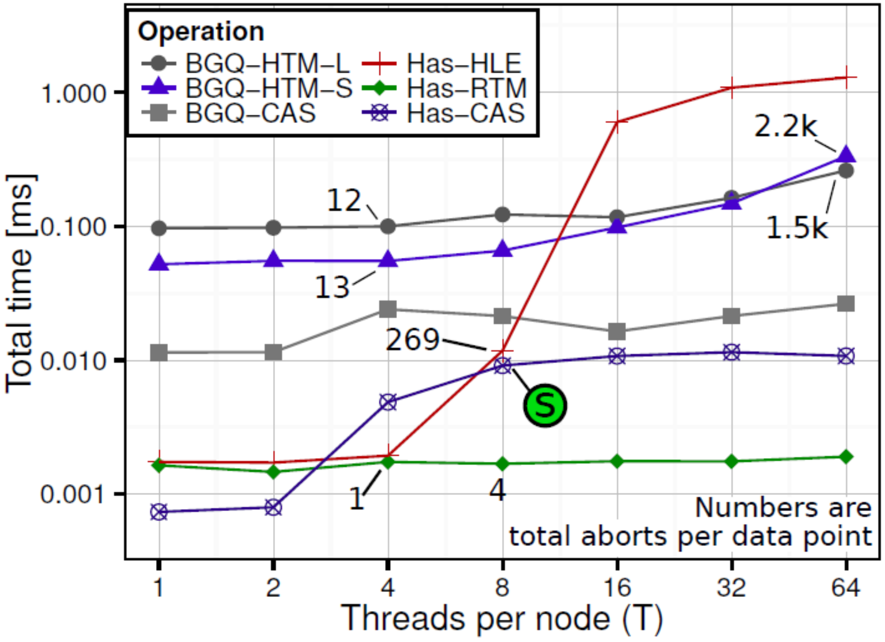
SINGLE-VERTEX TRANSACTIONS
MARKING A VERTEX AS VISITED

Used in BFS, SSSP, ...

Lower contention
(10 accesses/vertex)



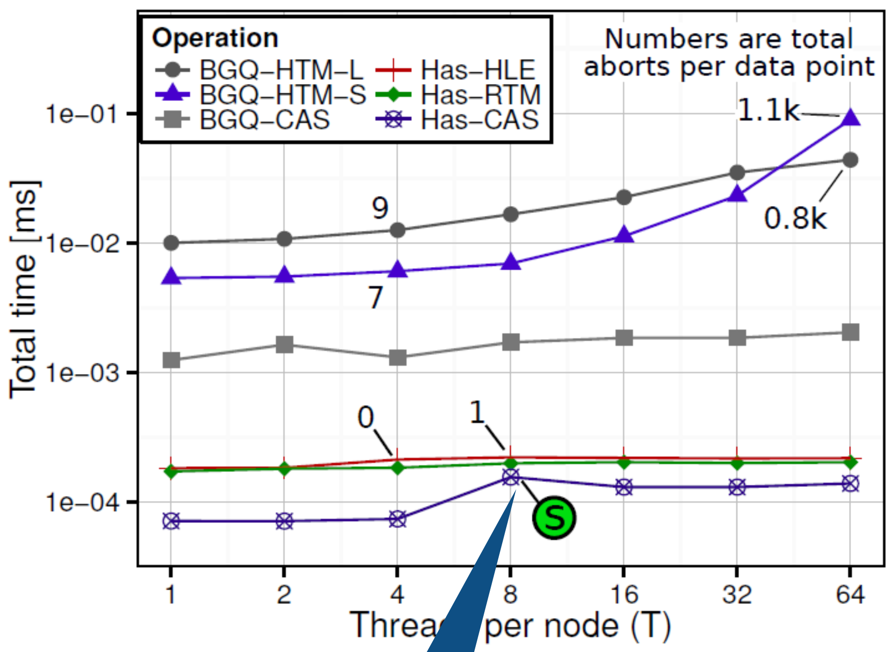
Higher contention
(100 accesses/vertex)



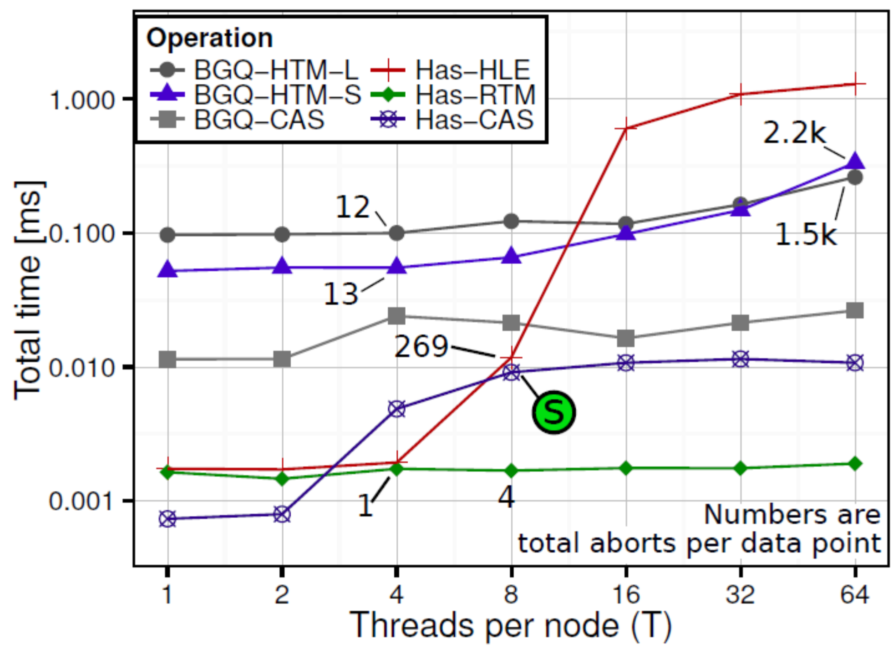
Single-Vertex Transactions
Marking a vertex as visited

Used in BFS, SSSP, ...

Lower contention (10 accesses/vertex)



Higher contention (100 accesses/vertex)

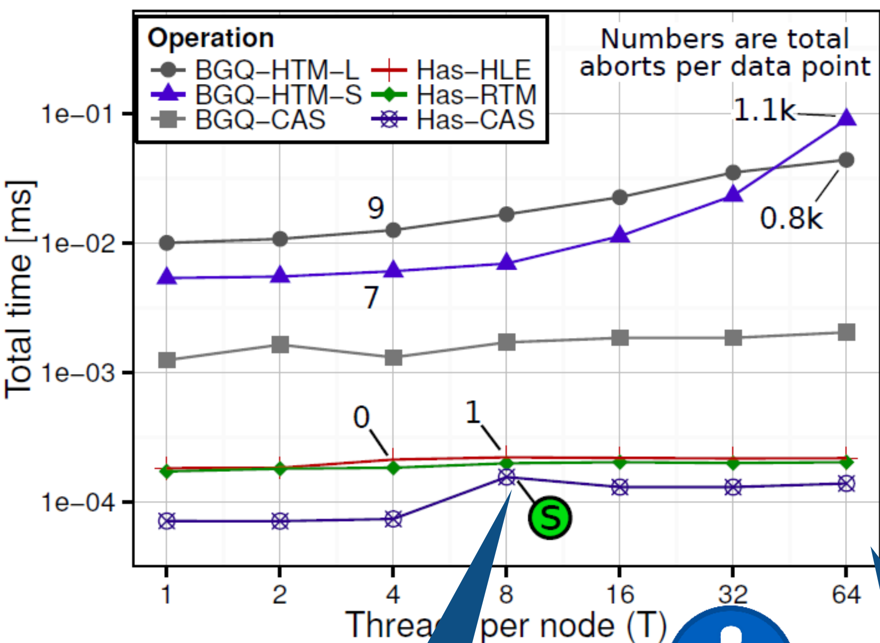


Atomics (CAS) slightly faster than HTM

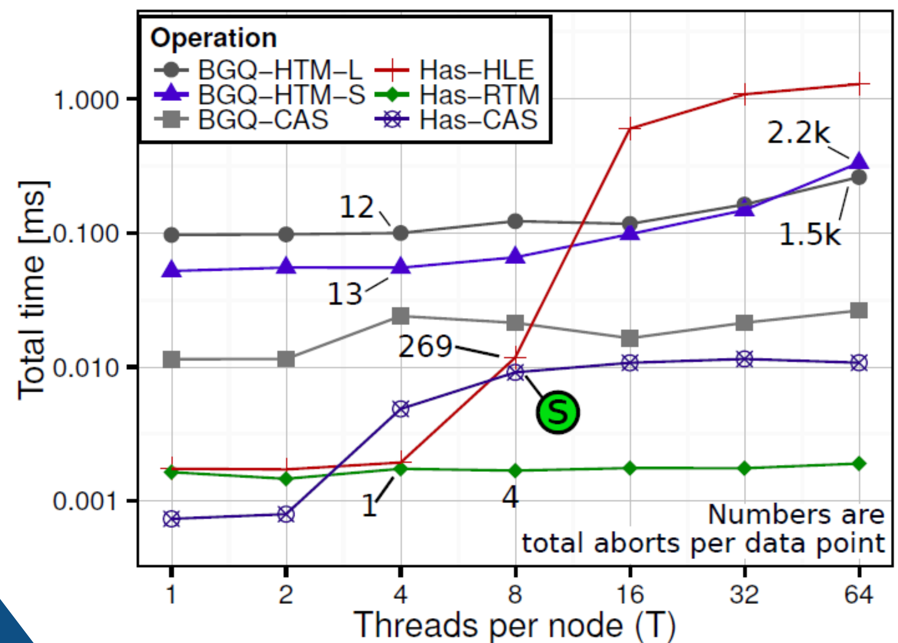
SINGLE-VERTEX TRANSACTIONS
MARKING A VERTEX AS VISITED

Used in BFS, SSSP, ...

Lower contention
(10 accesses/vertex)



Higher contention
(100 accesses/vertex)



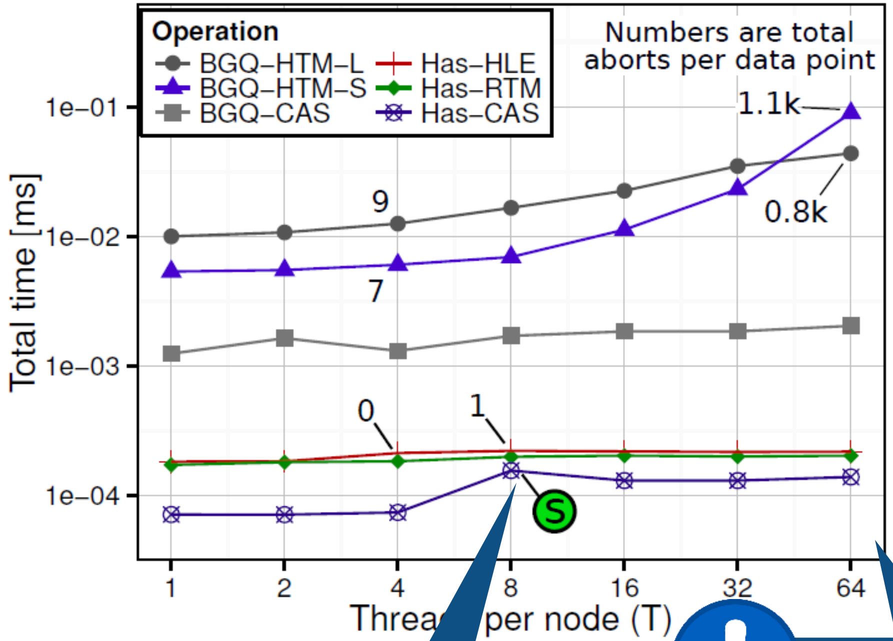
Atomics (CAS) slightly faster than HTM

Commit overheads dominate

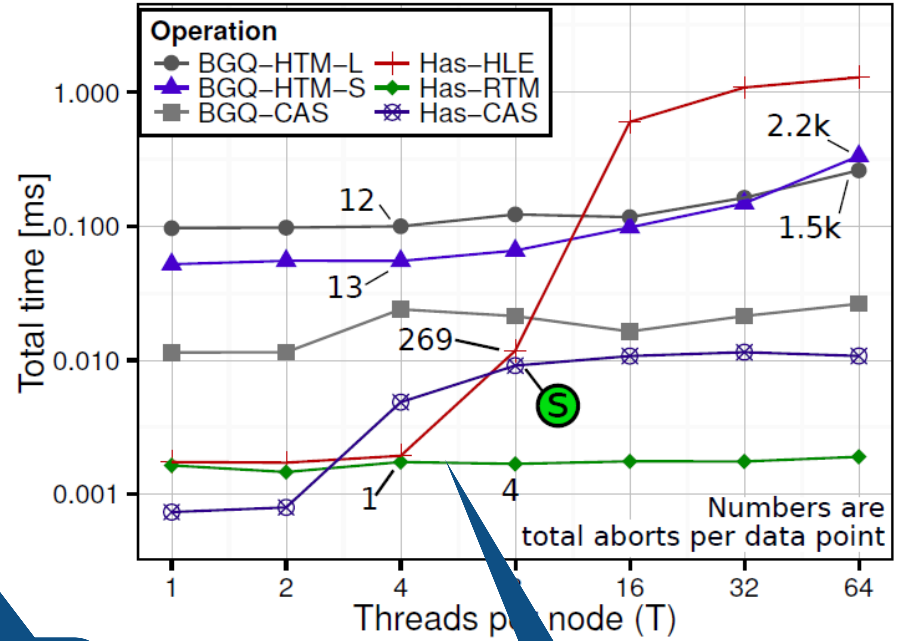
SINGLE-VERTEX TRANSACTIONS
MARKING A VERTEX AS VISITED

Used in BFS, SSSP, ...

Lower contention
(10 accesses/vertex)



Higher contention
(100 accesses/vertex)



Atomics (CAS) slightly faster than HTM

Commit overheads dominate

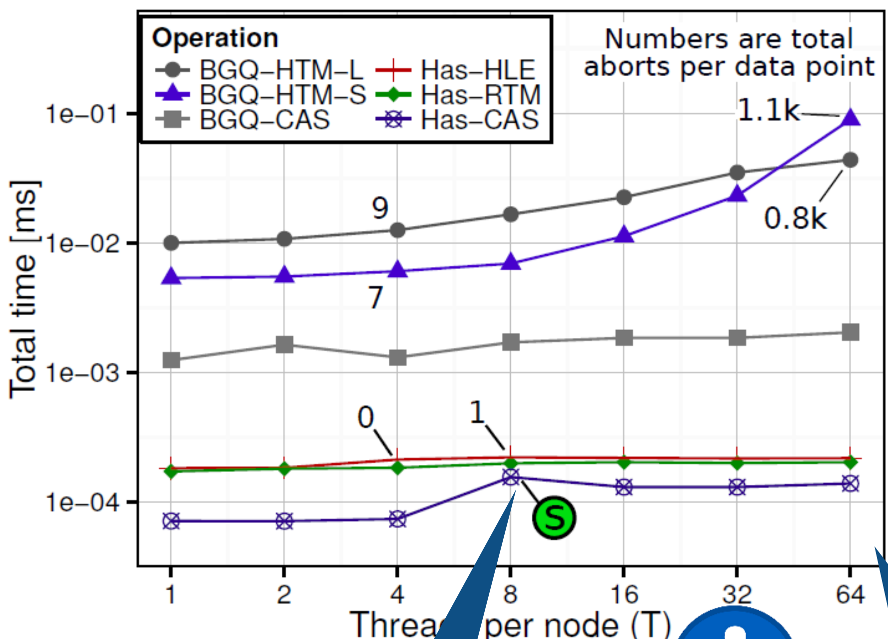
RTM outperforms other (overcontended) targets

SINGLE-VERTEX TRANSACTIONS
MARKING A VERTEX AS VISITED

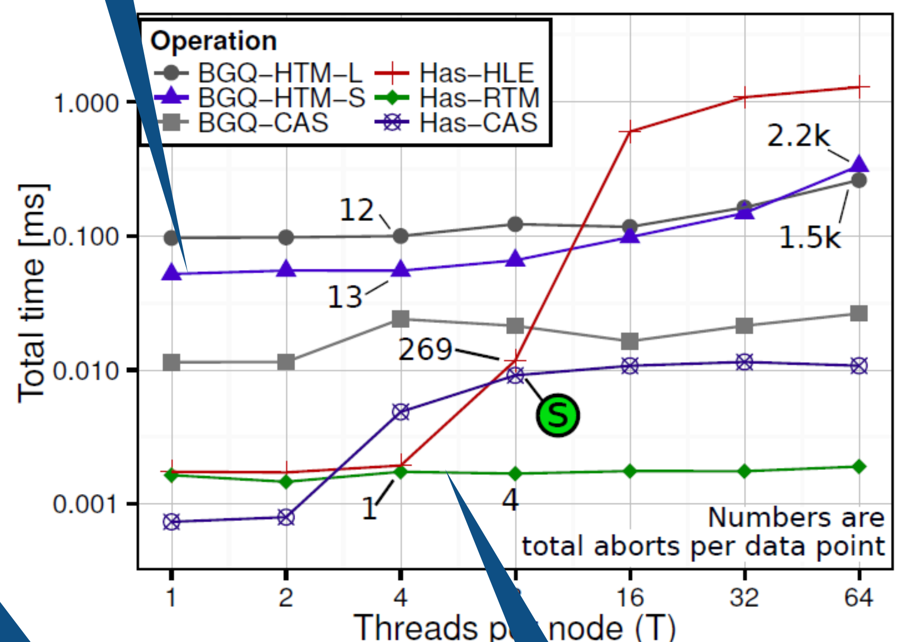
Used in BFS, SSSP, ...

BG/Q HTM still worse (L1 vs L2 matters!)

Lower contention (10 accesses/vertex)



Higher contention (100 accesses/vertex)



Atomics (CAS) slightly faster than HTM

Commit overheads dominate

RTM outperforms other (overcontended) targets

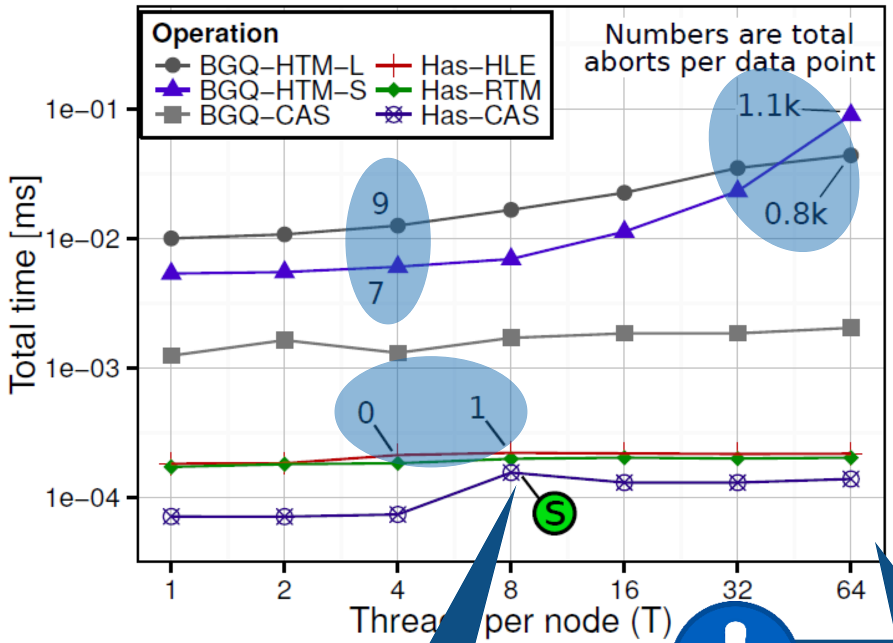
Single-Vertex Transactions
Marking a vertex as visited

Used in BFS, SSSP, ...

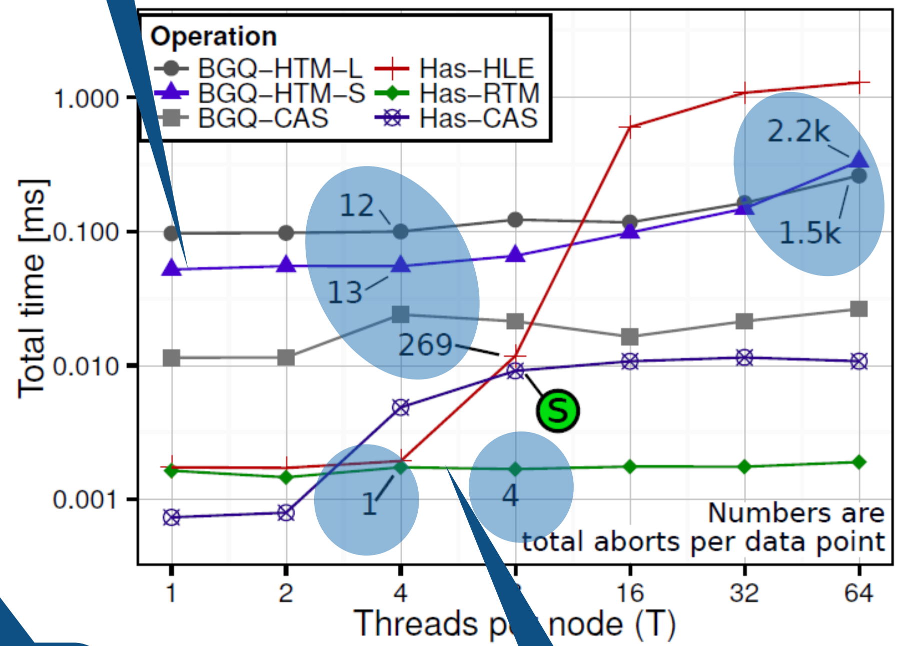
Very few aborts

BG/Q HTM still worse (L1 vs L2 matters!)

Lower contention (10 accesses/vertex)



Higher contention (100 accesses/vertex)

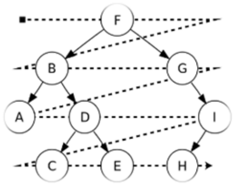


Atomics (CAS) slightly faster than HTM

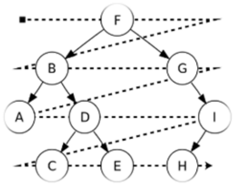
Commit overheads dominate

RTM outperforms other (overcontended) targets

OUTPERFORMING STATE-OF-THE-ART
HASWELL



Outperforming State-of-the-art
Haswell

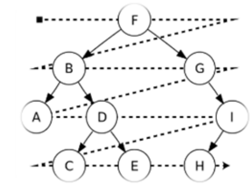


Average overall speedup (geometric mean) over Graph500: 1.07, Galois: 1.40, HAMA ~1000

OUTPERFORMING STATE-OF-THE-ART

HASWELL

SNAP

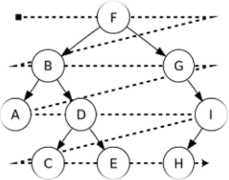


Average overall speedup (geometric mean) over Graph500: 1.07, Galois: 1.40, HAMA ~1000

IBM

Outperforming State-of-the-Art

Haswell



Average speedup: 1

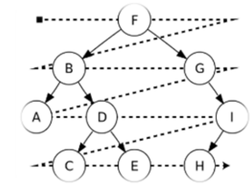




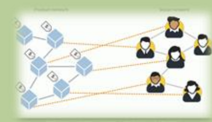
Average overall speedup (geometric mean) over Graph500: 1.07, Galois: 1.40, HAMA ~1000

OUTPERFORMING STATE-OF-THE-ART

HASWELL

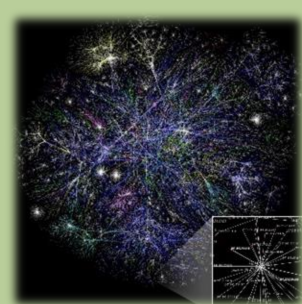


Average speedup: 1



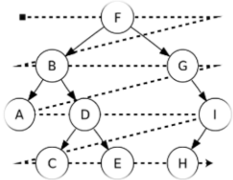
Average overall speedup (geometric mean) over Graph500: 1.07, Galois: 1.40, HAMA ~1000



Average speedup: 1.85

OUTPERFORMING STATE-OF-THE-ART

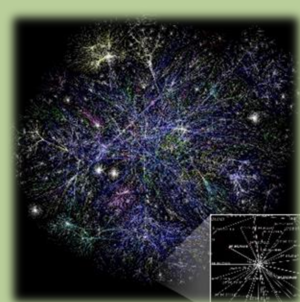
HASWELL



Average speedup: 1

Average overall speedup (geometric mean) over Graph500: 1.07, Galois: 1.40, HAMA ~1000

Average speedup: 3.20

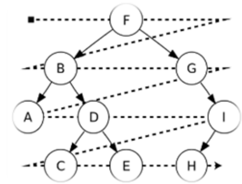


Average speedup: 1.85

OUTPERFORMING STATE-OF-THE-ART

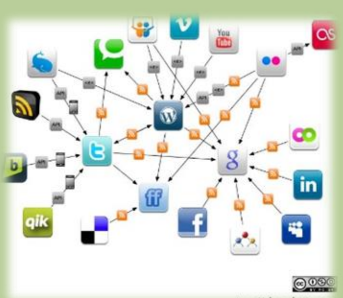
HASWELL

SNAP

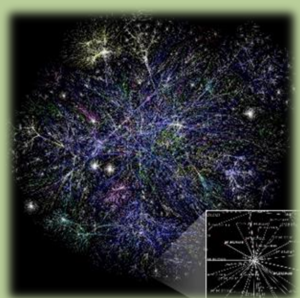


Average speedup: 1

Average overall speedup (geometric mean) over Graph500: 1.07, Galois: 1.40, HAMA ~1000



Average speedup: 3.20

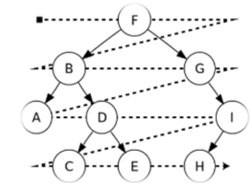


Average speedup: 1.85

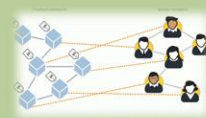
1.85x on average, up to 4.3x

Best transaction size: ~2-9 vertices accessed

OUTPERFORMING STATE-OF-THE-ART
HASWELL intel

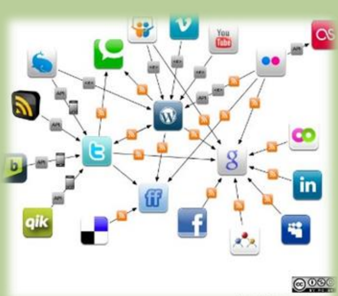


Average speedup: 1

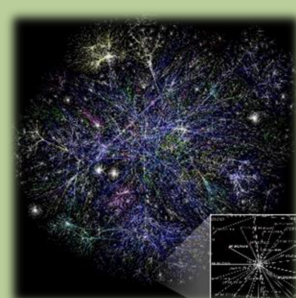





Average overall speedup (geometric mean) over Graph500: 1.07, Galois: 1.40, HAMA ~1000



Average speedup: 3.20



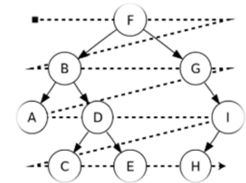
Average speedup: 1.85

! 1.85x on average, up to 4.3x

IBM

! Best transaction size: ~2-9 vertices accessed

Outperforming State-of-the-Art
Haswell



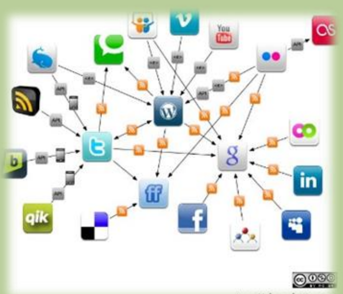
Average speedup: 1



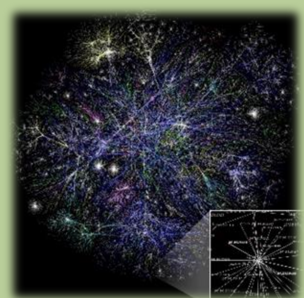



Average overall speedup (geometric mean) over Graph500: 1.07, Galois: 1.40, HAMA ~1000

1.85x on average, up to 4.3x



Average speedup: 3.20

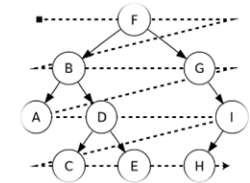


Average speedup: 1.85

Best transaction size: ~2-9 vertices accessed

OUTPERFORMING STATE-OF-THE-ART

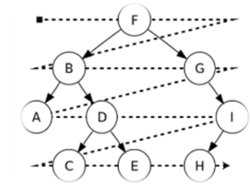
HASWELL

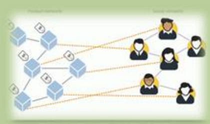


Best transaction size:
~4 vertices
~14 vertices

OUTPERFORMING STATE-OF-THE-ART

HASWELL



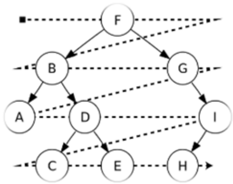


Average overall speedup (geometric mean) over Graph500: 1.07, Galois: 1.40, HAMA ~1000

!

Best transaction size:
~4 vertices
~14 vertices

OUTPERFORMING STATE-OF-THE-ART
HASWELL

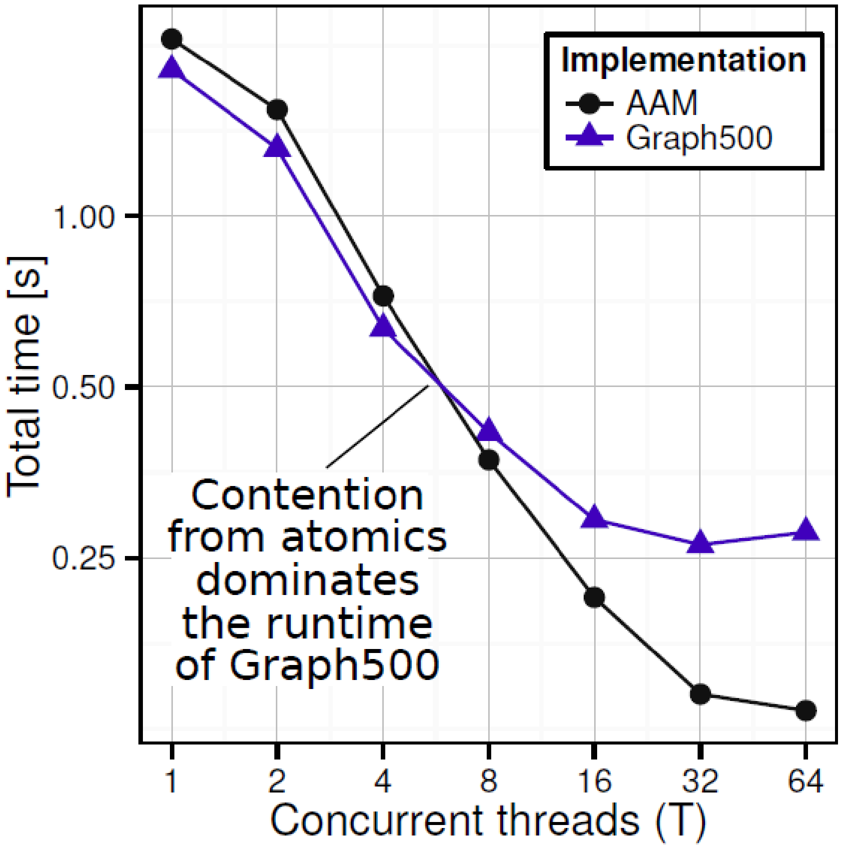


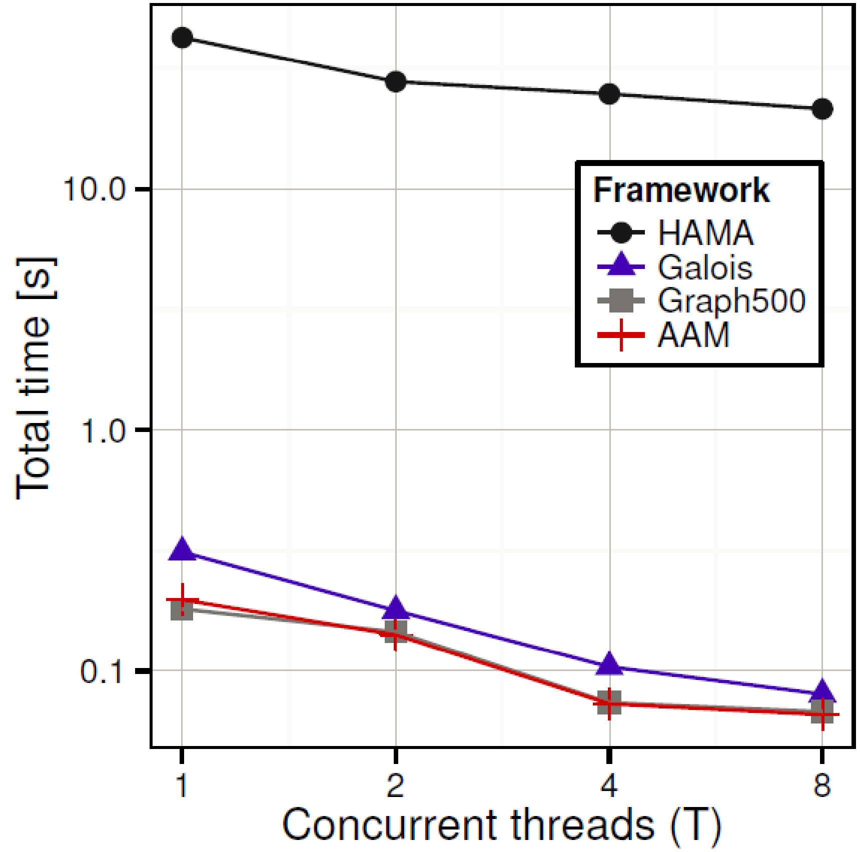
Average overall speedup (geometric mean) over Graph500: 1.07, Galois: 1.40, HAMA ~1000

1.85x on average, up to 4.3x

Best transaction size:
~4 vertices
~14 vertices

OUTPERFORMING STATE-OF-THE-ART
SCALABILITY ANALYSIS: SHARED-MEMORY





OUTPERFORMING STATE-OF-THE-ART
BLUEGENE/Q

IBM

Input graph properties					BG/Q analysis		
Type	ID	Name	$\|V\|$	$\|E\|$	S over g500 ($M = 24$)	M	S over g500
Comm. networks (CNs)	cWT	wiki-Talk	2.4M	5M	2.82	48	3.35
	cEU	email-EuAll	265k	420k	3.67	32	4.36
Social networks (SNs)	sLV	soc-LiveJ.	4.8M	69M	1.44	12	1.56
	sOR	com-orkut	3M	117M	1.22	20	1.27
	sLJ	com-lj	4M	34M	1.44	12	1.54
	sYT	com-youtube	1.1M	2.9M	1.67	8	1.84
	sDB	com-dblp	317k	1M	1.33	8	1.80
	sAM	com-amazon	334k	925k	1.14	8	1.62
Purchase network (PNs)	pAM	amazon0601	403k	3.3M	1.45	8	1.91
Road networks (RNs)	rCA	roadNet-CA	1.9M	5.5M	≈1	2	1.59
	rTX	roadNet-TX	1.3M	3.8M	≈1	2	1.53
	rPA	roadNet-PA	1M	3M	≈1	2	1.52
Citation graphs (CGs)	ciP	cit-Patents	3.7M	16.5M	1.16	8	1.57
Web graphs (WGs)	wGL	web-Google	875k	5.1M	1.78	12	2.08
	wBS	web-BerkStan	685k	7.6M	1.91	24	1.91
	wSF	web-Stanford	281k	2.3M	1.89	24	1.89

OUTPERFORMING STATE-OF-THE-ART
HASWELL

Input graph properties					Haswell analysis					
Type	ID	Name	$\|V\|$	$\|E\|$	S over g500 ($M=2$)	S over Galois ($M=2$)	M	S over g500	S over Galois	S over HAMA
Comm. networks (CNs)	cWT	wiki-Talk	2.4M	5M	0.91	1.22	6	0.96	1.28	344
	cEU	email-EuAll	265k	420k	0.76	0.88	4	0.97	1.12	1448
Social networks (SNs)	sLV	soc-LiveJ.	4.8M	69M	1.05	1.1	3	1.07	1.12	$> 10^4$
	sOR	com-orkut	3M	117M	1.06	0.69	4	1.13	0.74	$> 10^4$
	sLJ	com-lj	4M	34M	1.03	1.03	4	1.04	1.04	603
	sYT	com-youtube	1.1M	2.9M	0.96	1.1	5	0.98	1.11	670
	sDB	com-dblp	317k	1M	≈ 1	2.5	2	≈ 1	2.53	2160
	sAM	com-amazon	334k	925k	1.04	1.64	2	1.04	1.64	1426
Purchase network (PNs)	pAM	amazon0601	403k	3.3M	≈ 1	1.25	3	1.03	1.30	618
Road networks (RNs)	rCA	roadNet-CA	1.9M	5.5M	1.33	1.74	8	1.38	1.80	$> 10^4$
	rTX	roadNet-TX	1.3M	3.8M	1.29	1.89	6	1.42	2.08	$> 10^4$
	rPA	roadNet-PA	1M	3M	≈ 1	2.00	9	1.07	2.16	$> 10^4$
Citation graphs (CGs)	ciP	cit-Patents	3.7M	16.5M	1.01	1.26	2	1.01	1.26	1875
Web graphs (WGs)	wGL	web-Google	875k	5.1M	0.98	1.26	6	1.06	1.35	365
	wBS	web-BerkStan	685k	7.6M	0.93	1.31	5	1.07	1.40	755
	wSF	web-Stanford	281k	2.3M	0.98	1.54	5	1.07	1.58	1077