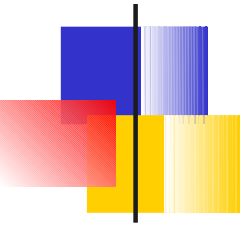
UCMERCED

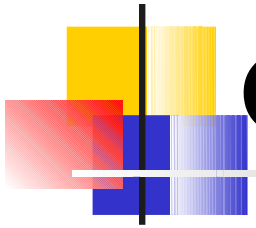
OLA-RAW: Scalable Exploration over Raw Data



Florin Rusu*, Yu Cheng+, Weijie Zhao*

+: Amobee, Inc.
*: University of California Merced

UCMERCED



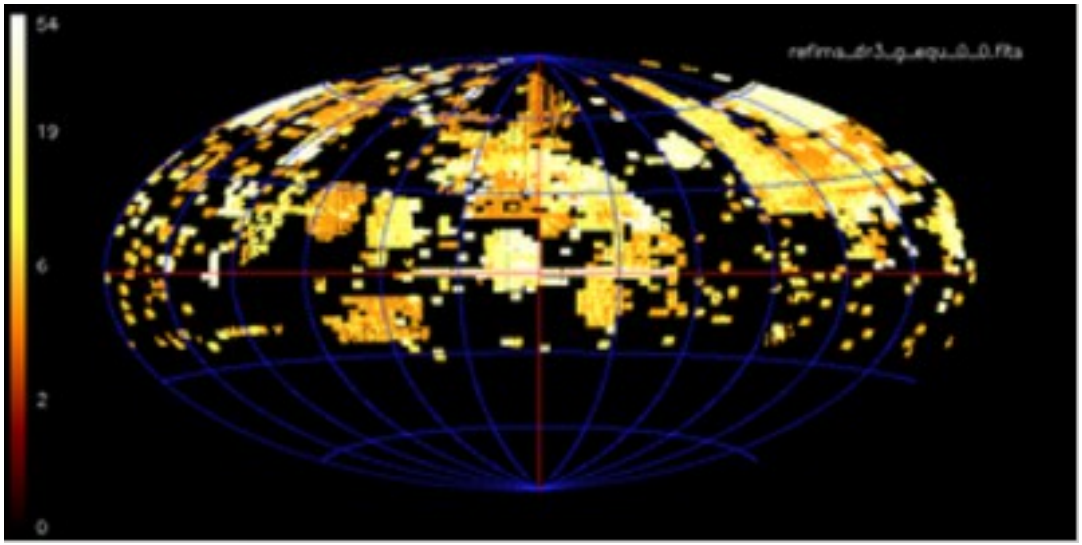
Outline

- Background
- Problem
- OLA-RAW
- Evaluation

UCMERCED

Palomar Transient Factory (PTF)

The Palomar Transient Factory (PTF) project aims to identify and automatically classify transient astrophysical objects such as variable stars and supernovae in real-time.



Merging Neutron Stars Create Heavy Elements

Science Home News Journals Topics Careers

Log in | My account | Contact Us

RESEARCH ARTICLE

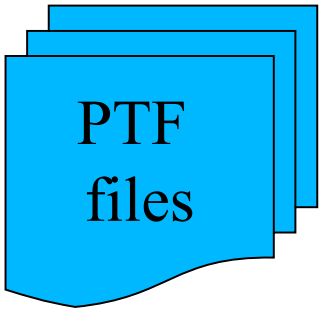
Illuminating gravitational waves: A concordant picture of photons from a neutron star merger

M. M. Kasliwal[1,*], E. Nakar[2], L. P. Singer[3,4], D. L. Kaplan[5], D. O. Cook[1], A. Van Sistine[5], R. M. Lau[1], C. Fremling[1], O. Gottlieb[2], J. E. Jencson[1], S. M. Adams[1], U. Feindt[6], K. Hotokezaka[7], S. Ghosh[5], D. A. Perley[8], P.-C. Yu[9], T. Piran[10], J. R. Allison[11,12], G. C. Anupama[13], A. Balasubramanian[14], K. W Bannister[15], J. Bally[16], J. Barnes[17], S. Barway[18], E. Bellm[19], V. Bhalerao[20], D. Bhattacharya[21], N. Blagorodnova[1], J. S. Bloom[22,23], P. R. Brady[5], C. Cannella[1], D. Chatterjee[5], S. B. Cenko[3,4], B. E. Cobb[24], C. Copperwheat[8], A. Corsi[25], K. De[1], D. Dobie[11,26,15], S. W. K. Emery[27], P. A. Evans[28], O. D. Fox[29], D. A. Frail[30], C. Frohmaier[31,32], A. Goobar[6], G. Hallinan[1], F. Harrison[1], G. Helou[33], T. Hinderer[34], A. Y. Q. Ho[1], A. Horesh[10], W.-H. Ip[9], R. Itoh[35], D. Kasen[22,36], H. Kim[37], N. P. M. Kuin[27], T. Kupfer[1], C. Lynch[11,26], K. Madsen[1], P. A. Mazzali[8,38], A. A. Miller[39,40], K. Mooley[41], T. Murphy[11,26], C.-C. Ngeow[9], D. Nichols[34], S. Nissanke[34], P. Nugent[22,23], E. O. Ofek[42], H. Qi[5], R. M. Quimby[43,44], S. Rosswog[45], F. Rusu[46], E. M. Sadler[11,26], P. Schmidt[34], J. Sollerman[45], I. Steele[8], A. R. Williamson[34], Y. Xu[1], L. Yan[1,33], Y. Yatsu[35], C. Zhang[5], W. Zhao[46]

UCMERCED

Illustrative Example

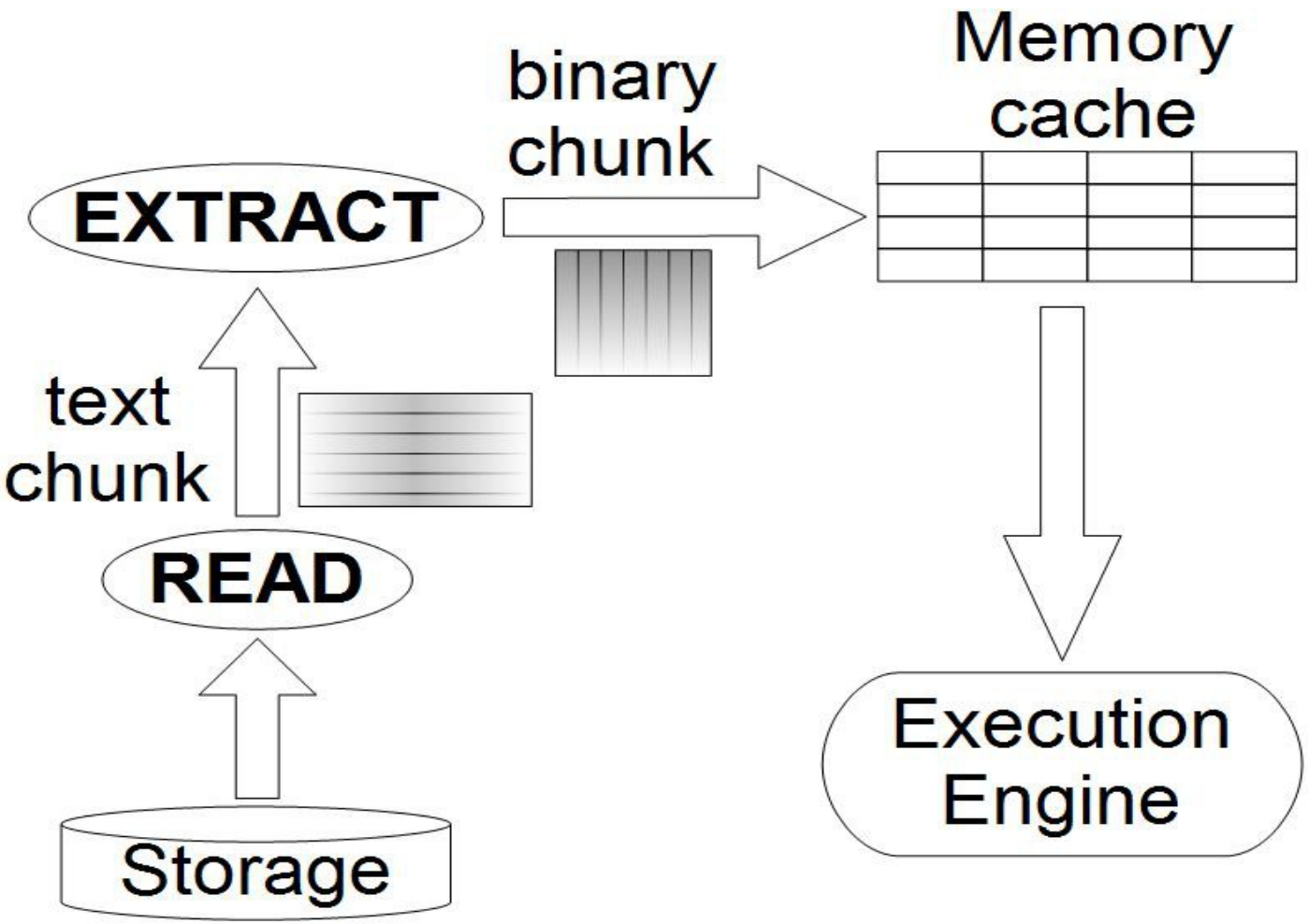
- Gamma ray burst identification




```
SELECT AGGREGATE(expression) AS agg
FROM candidate
WHERE predicate
HAVING agg < threshold
```

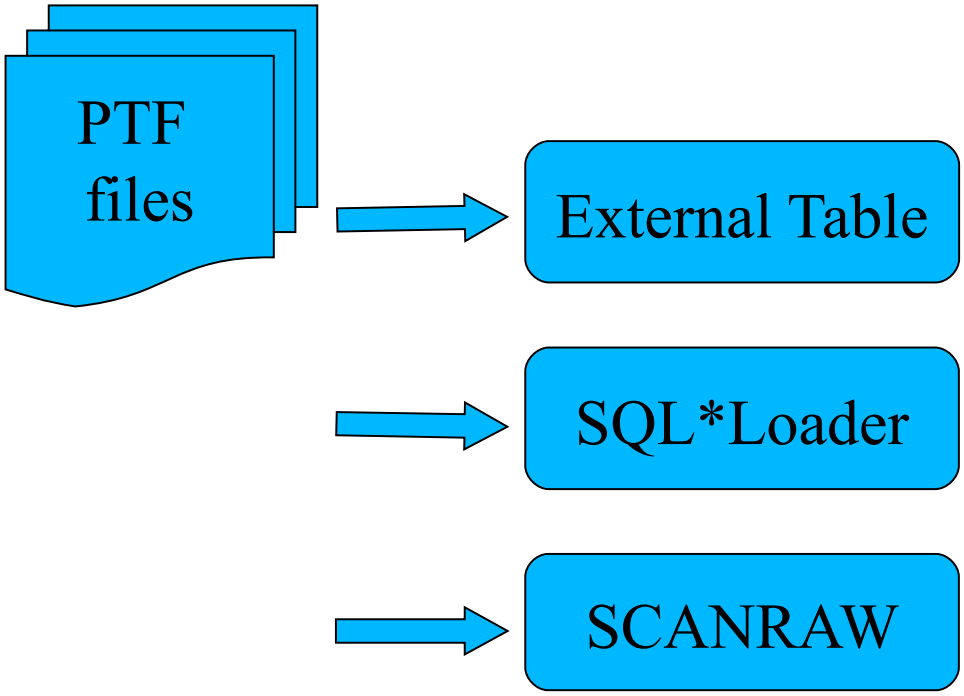
UCMERCED

Raw Data Processing



UCMERCED

Existing Solutions

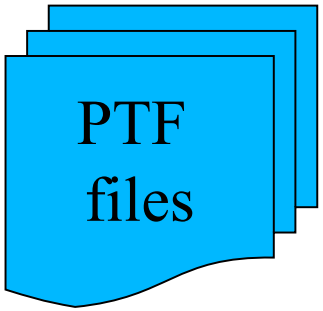


Time to query	Execution	Storage
instant	slow	zero
loading	fast	full replication
instant	fast	adaptive

UCMERCED

Illustrative Example

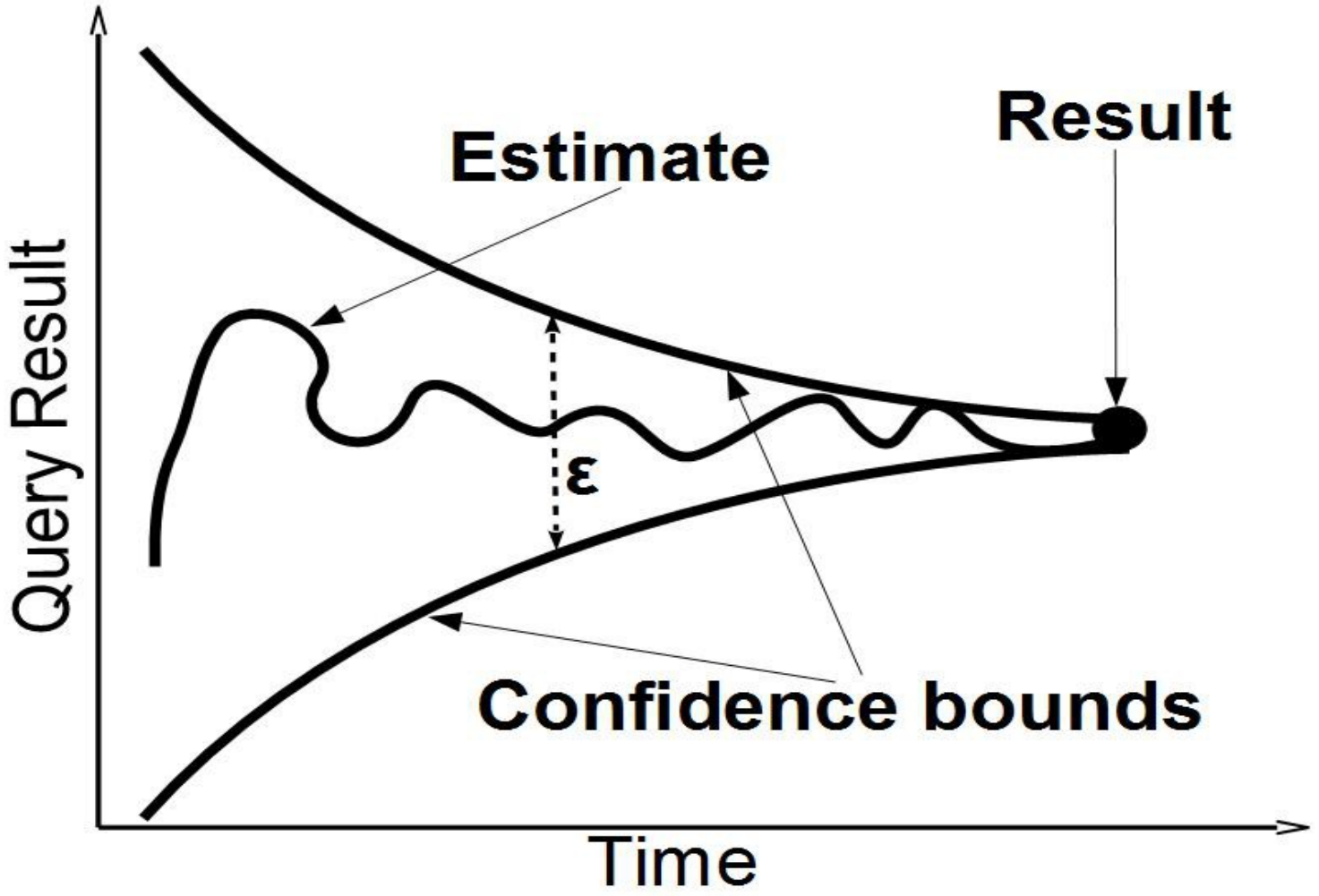
Gamma ray burst identification




```
SELECT AGGREGATE(expression) AS agg
FROM candidate
WHERE predicate
HAVING agg > threshold
WITH ACCURACY α
```

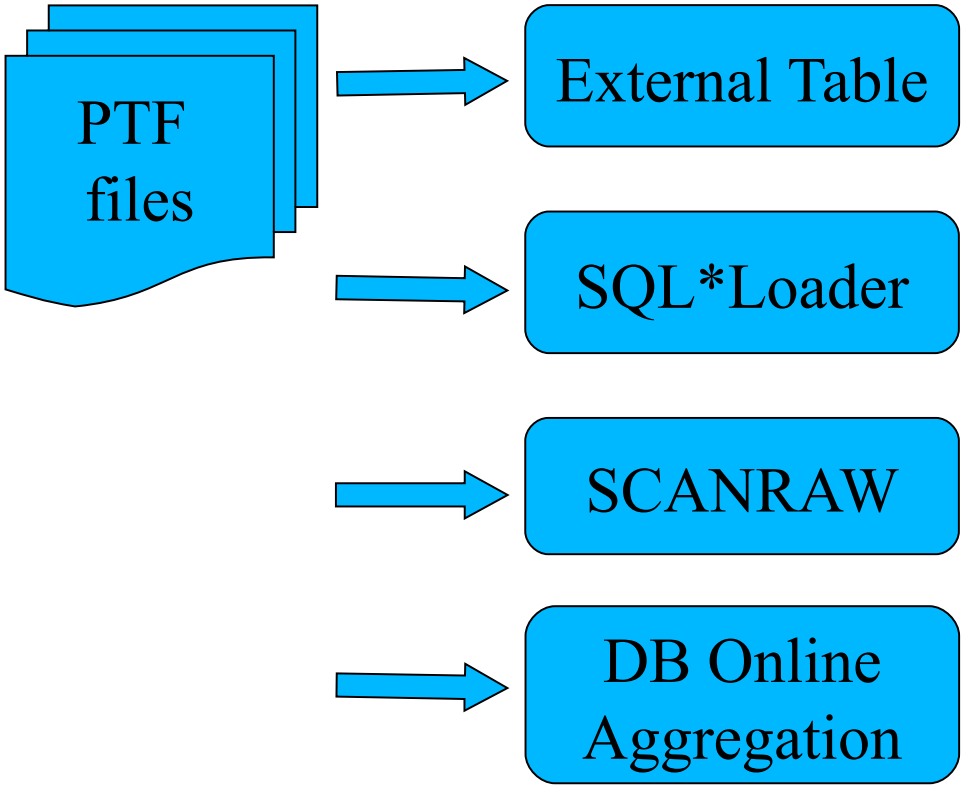
UCMERCED

Online Aggregation



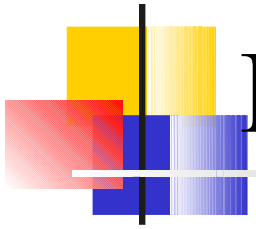
UCMERCED

Existing Solutions



Time to query	Execution	Storage
instant	slow	zero
loading	fast	full replication
instant	fast	adaptive
loading + shuffling	faster	full replication

UCMERCED

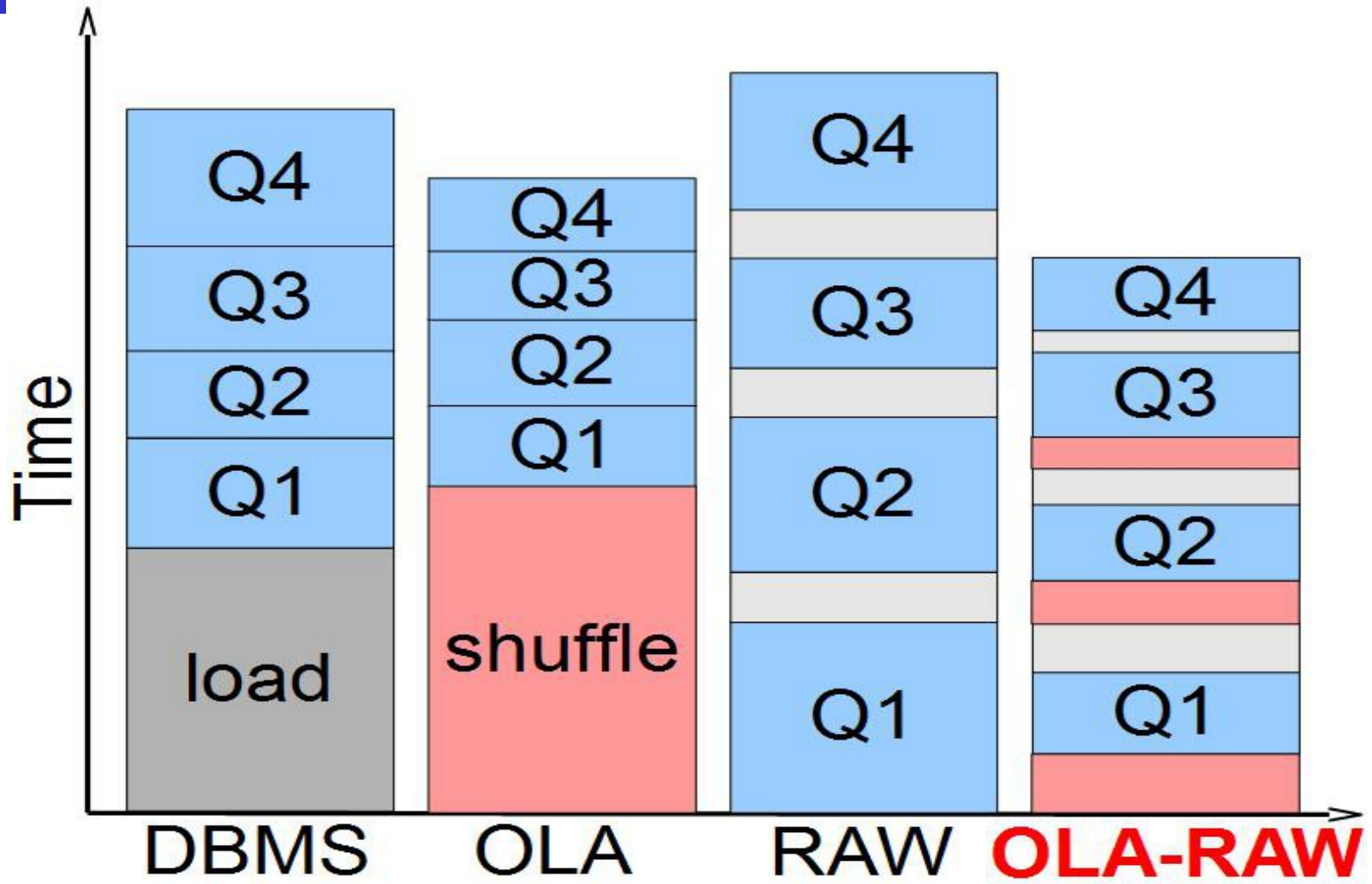


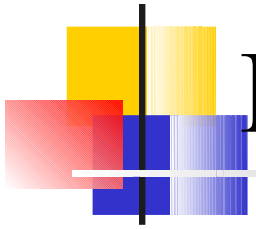
Research Problem

- Can we find an efficient parallel solution to execute approximate queries over raw files?
  - Instant access to data
    In-situ data processing
  - Generate results fast
    Online aggregation (OLA)
  - Minimize storage
    In-memory synopsis

UCMERCED

High Level Approach

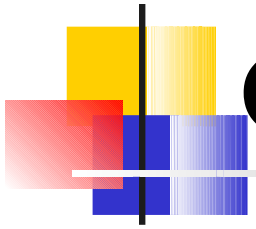




Related Work

- Adaptive partial loading [Idreos et al., CIDR 2011]
  Only load necessary attributes before query starts
- NoDB [Alagianis et al., SIGMOD 2012]
  Instead of loading, build index and cache necessary attributes in memory
- Invisible loading [Abouzied et al., EDBT/ICDT 2013]
  Portion of necessary data is loaded into database for every query
- Data vaults [Ivanova et al., SSDBM 2012]
  Memory cache for complex data in scientific repositories
- SCANRAW [Cheng and Rusu, SIGMOD 2014]
  Load data using spare system resources without affecting query processing

UCMERCED

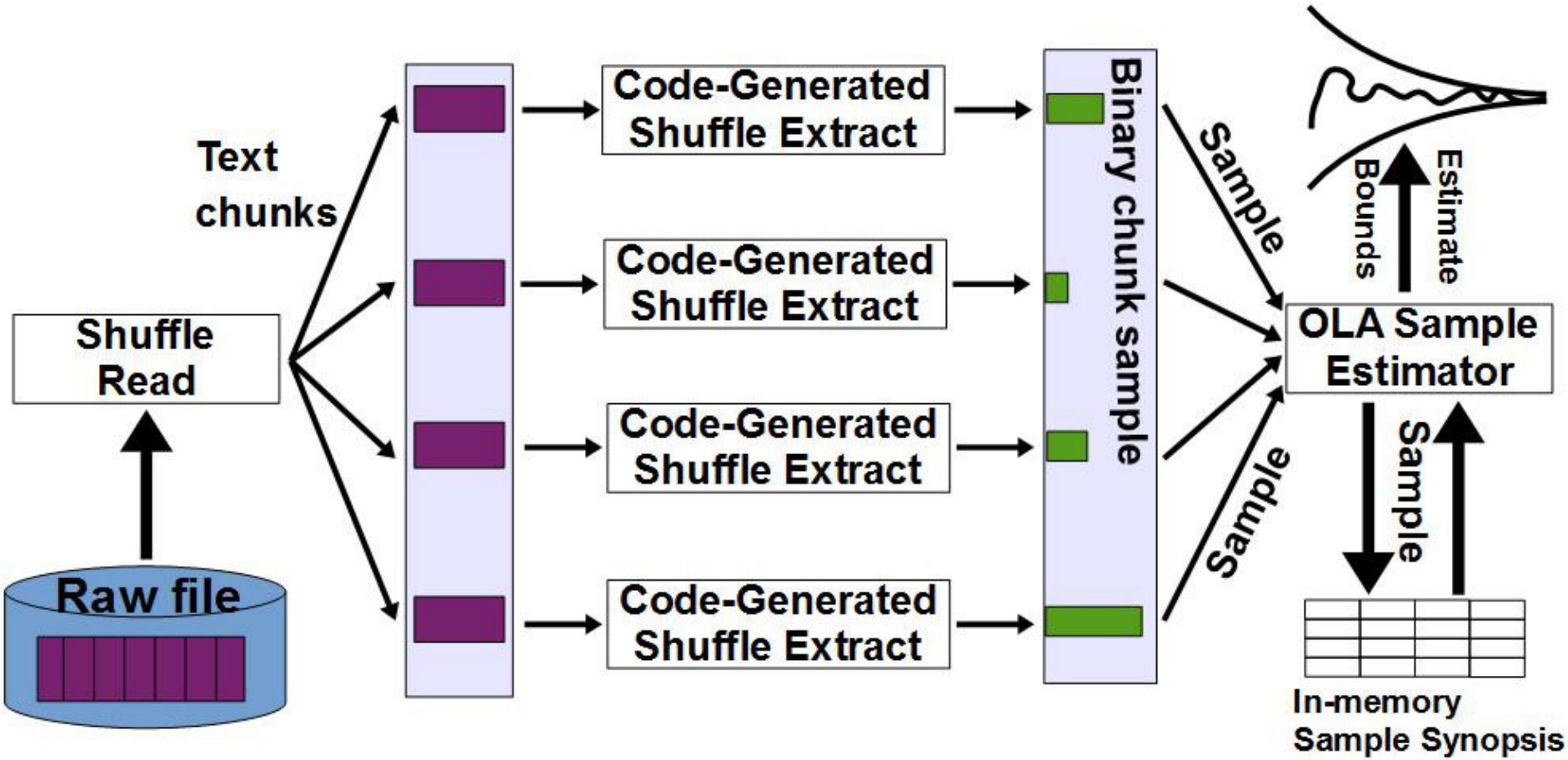


OLA-RAW

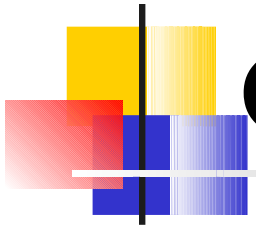
- **OnLine Aggregation for RAW data processing**
  - How to generate random samples in-situ in parallel?
  - How to efficiently reuse extracted samples?

UCMERCED

OLA-RAW Architecture



UCMERCED



OLA-RAW

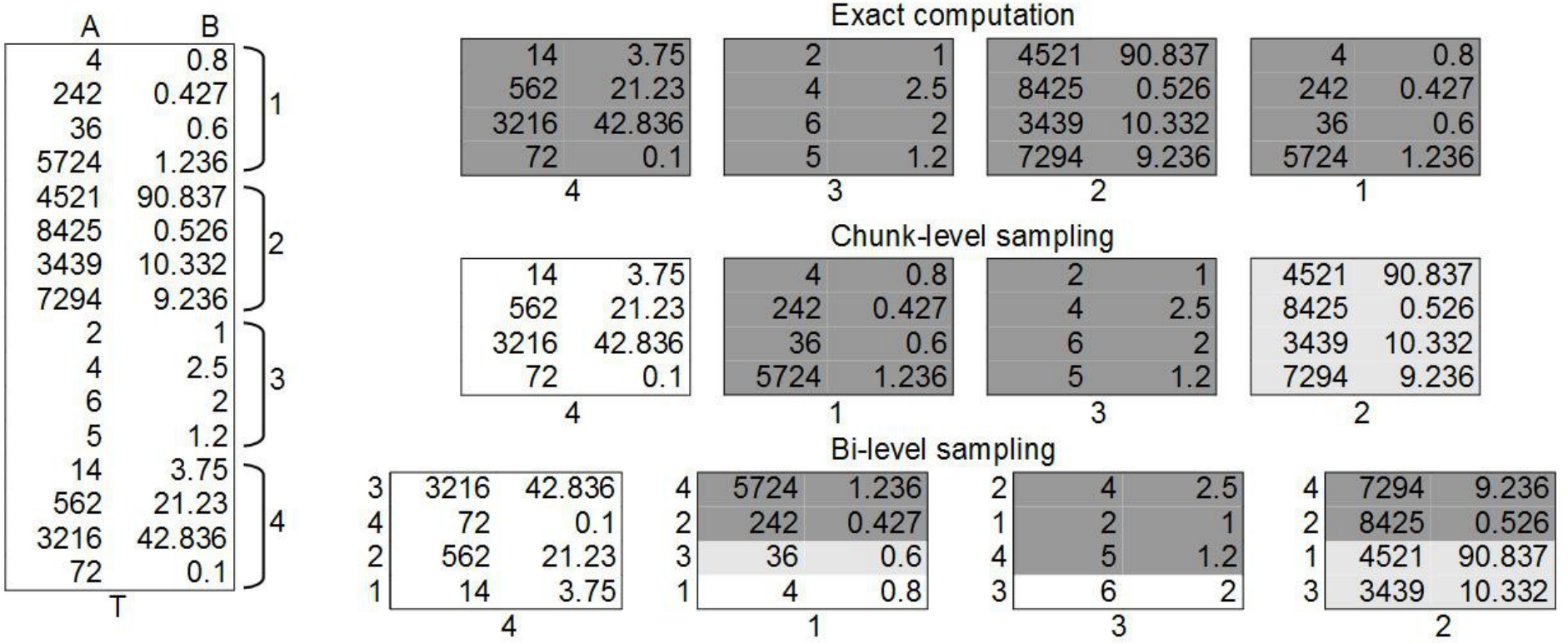
- **O**n**L**ine **A**ggregation for **RAW** data processing

- How to generate random samples in-situ in parallel?
  Resource-aware bi-level sampling

- How to efficiently reuse extracted samples?

UCMERCED

Parallel Bi-Level Sampling



Inspection paradox: Result order ≠ Random chunk order!!!

UCMERCED

Estimator and Variance

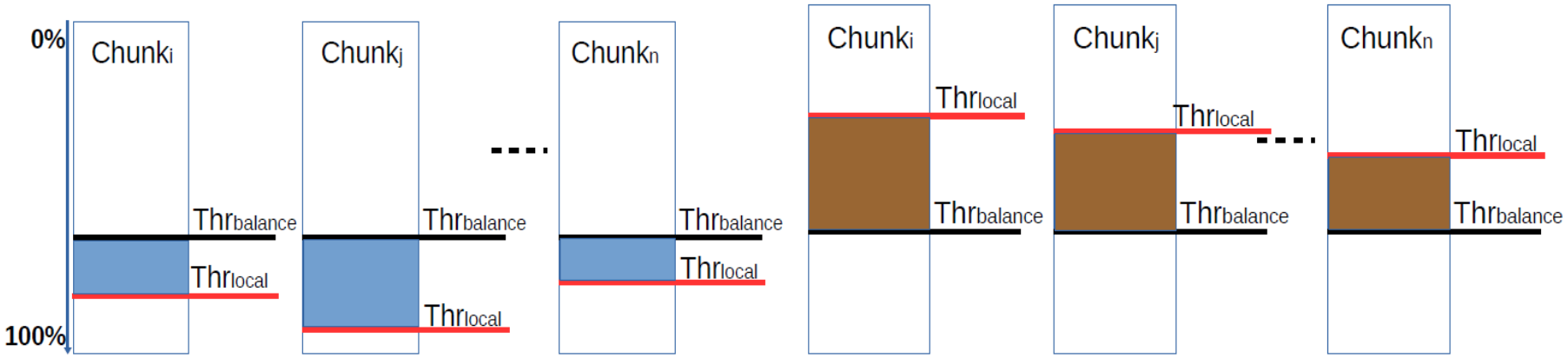
$$\widehat{\tau} = \frac{N}{n}\sum_{j=1}^{n}\widehat{y}_j = \frac{N}{n}\sum_{j=1}^{n}\frac{M_j}{m_j}\sum_{i\in C'_j} x_i$$

$$Var\left(\widehat{\tau}\right) = \frac{N}{N-1}\frac{N-n}{n}\sum_{j=1}^{N}\left(y_j - \frac{\sum_{i\in T} x_i}{N}\right)^2 + \frac{N}{n}\sum_{j=1}^{N}\left[\frac{M_j}{M_j-1}\frac{M_j-m_j}{m_j}\sum_{i\in C_j}\left(x_i - \frac{y_j}{M_j}\right)^2\right]$$

$$\widehat{Var\left(\widehat{\tau}\right)} = \frac{N}{n}\frac{N-n}{n-1}\sum_{j=1}^{n}\left(\frac{M_j}{m_j}y'_j - \frac{\sum_{j'=1}^{n}\frac{M_j}{m_j}y'_{j'}}{n}\right)^2 + \frac{N}{n}\sum_{j=1}^{n}\left[\frac{M_j}{m_j}\frac{M_j-m_j}{m_j-1}\sum_{i\in C'_j}\left(x_i - \frac{y'_j}{m_j}\right)^2\right]$$

UCMERCED

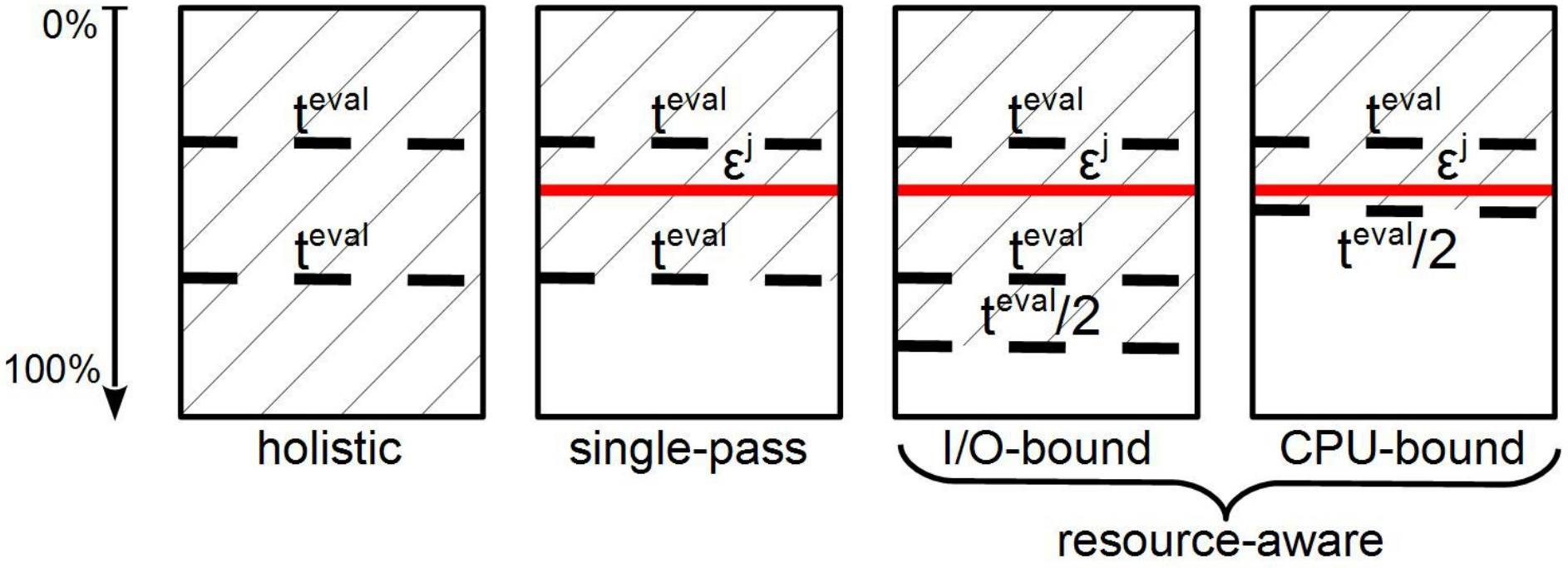
How many samples are enough?



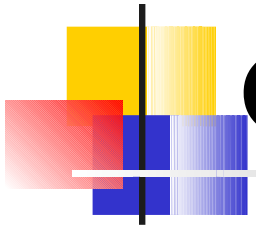
- Make sure to generate good-enough estimate in one pass over data
- Generate accurate estimate for each chunk

UCMERCED

Resource-Aware Bi-Level Sampling



UCMERCED

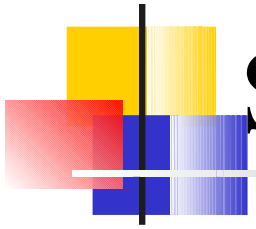


OLA-RAW

- **O**n**L**ine **A**ggregation for **RAW** data processing

- How to generate random samples in-situ in parallel?
  Resource-aware bi-level sampling

- How to efficiently reuse extracted samples?
  In-memory sample synopsis

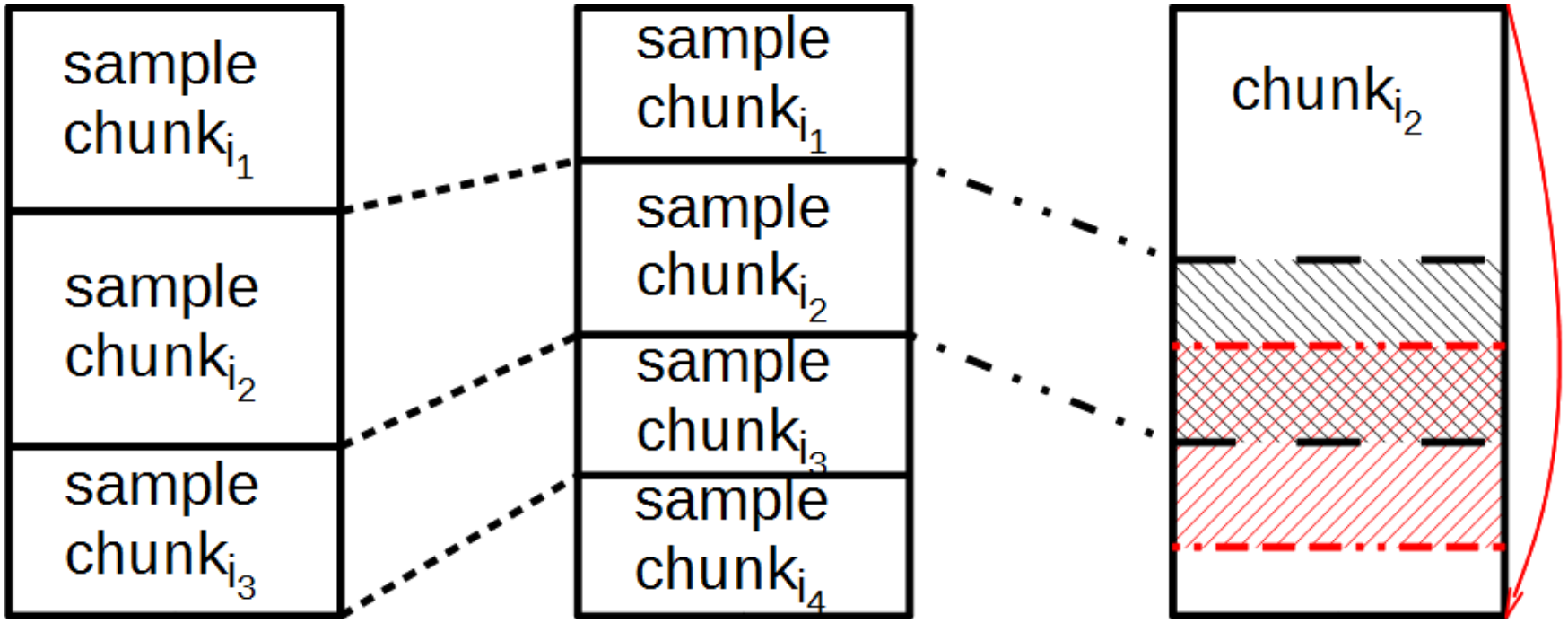


Sample Synopsis

- What kind of samples should be preserved?
  - Chunk-based variance-driven
- When to populate synopsis?
  - During query
- How to make sure the additional samples have not been selected before?
  - Permutation seeds + offset

UCMERCED

Sample Synopsis



UCMERCED

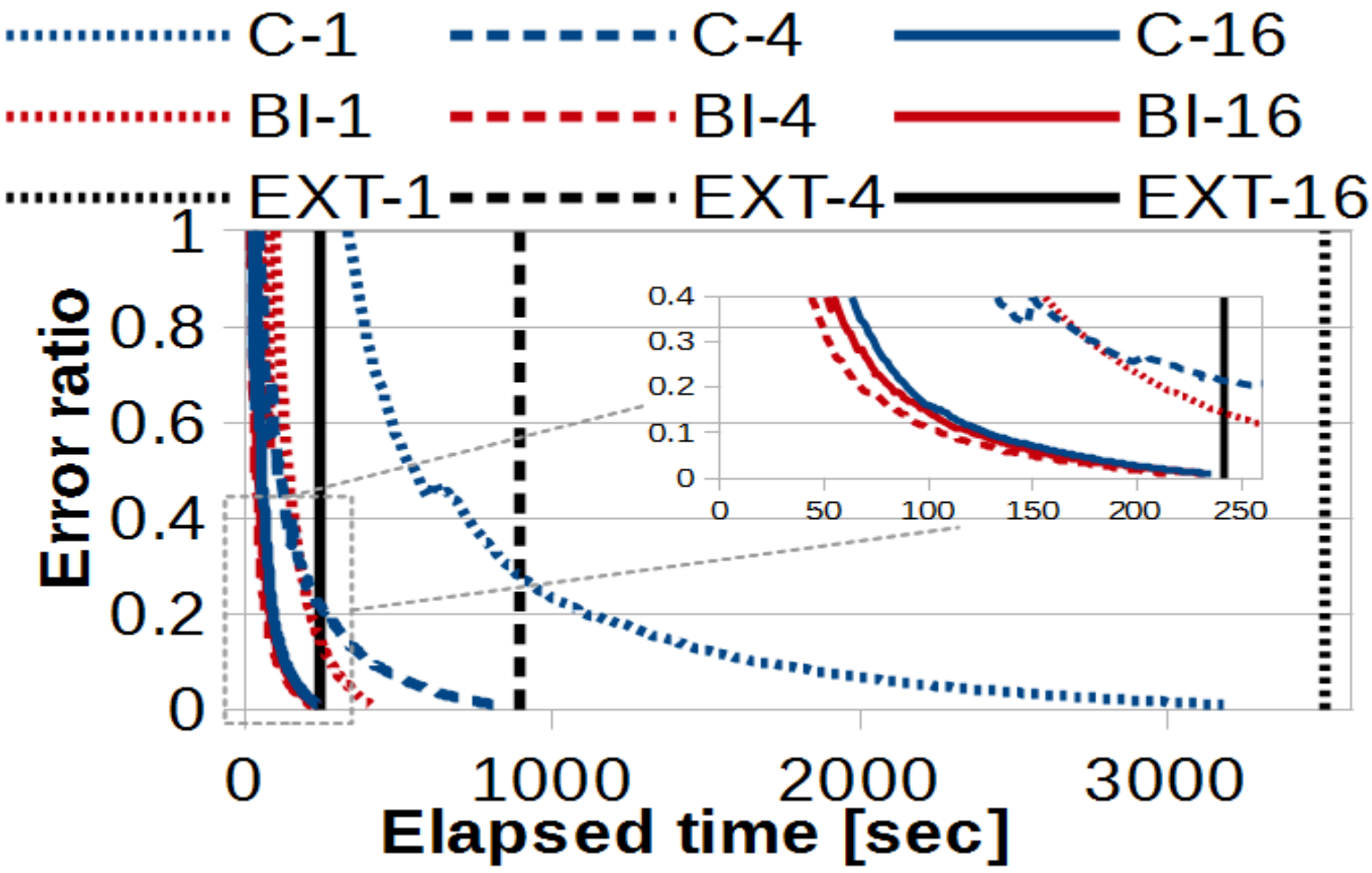
Evaluation

System: 16 cores, 64 GB of memory, 4 disks in RAID-0 with 565 MB/s I/O bandwidth

Dataset	# Tuples	# Chunks	# Columns	Size
ptf-csv	1B	1000	8	68 GB
ptf-fits	1B	1000	8	60 GB
wiki	1.8B	130	4	19 GB
synthetic	134M	512	16	20 GB

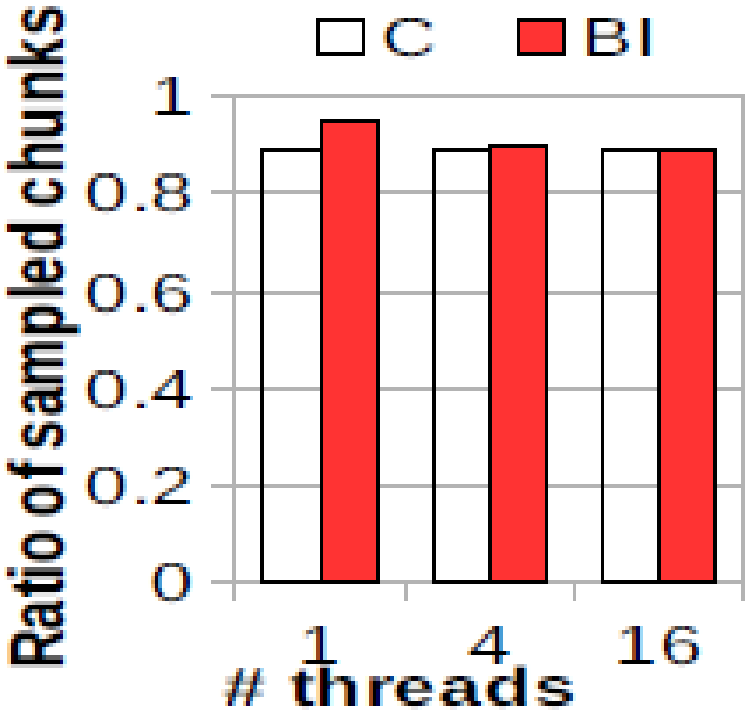
UCMERCED

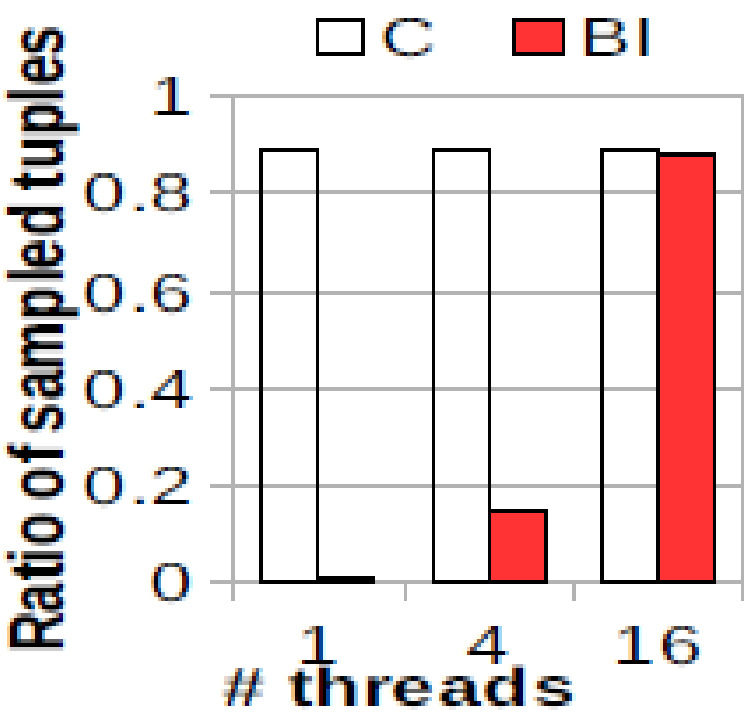
Query Execution Time: ptf-csv



UCMERCED

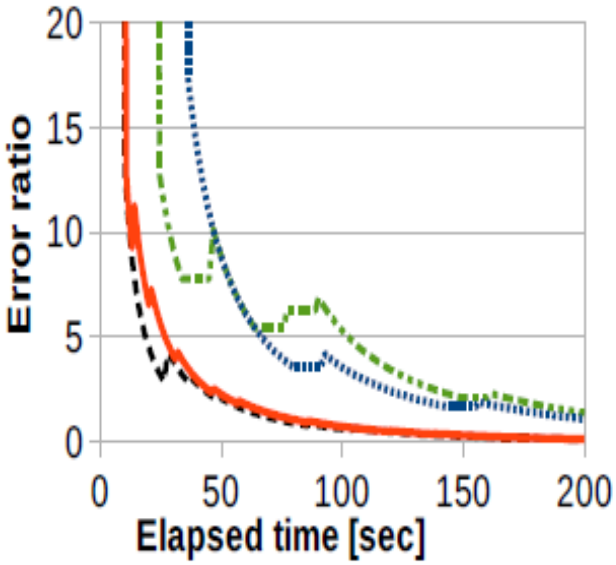
Sample Size: ptf-csv



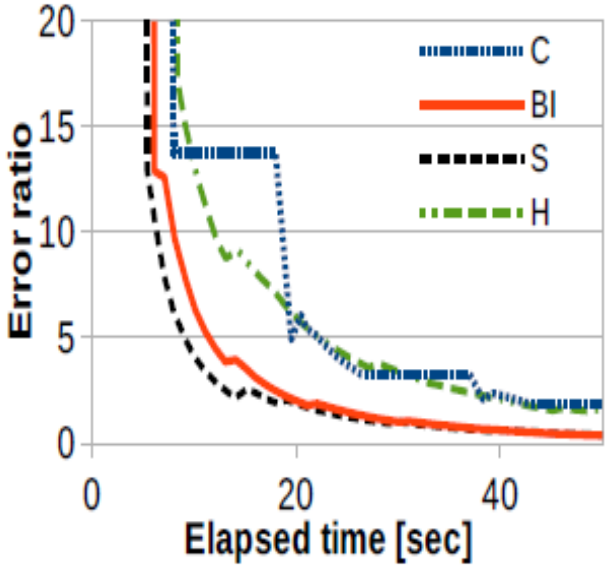


UCMERCED

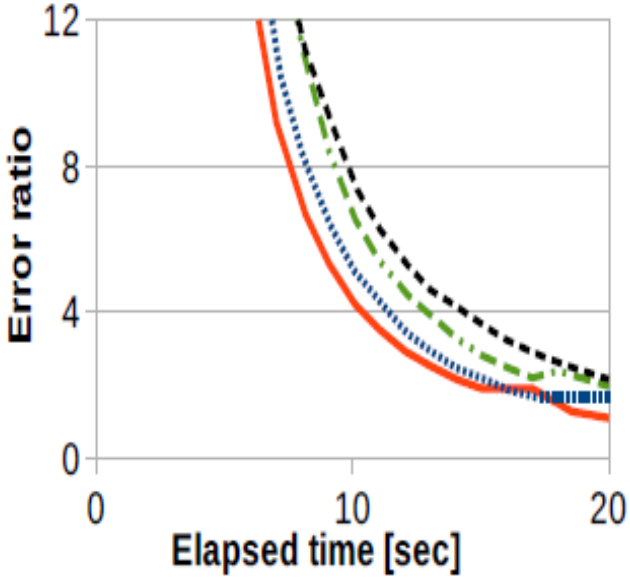
Resource-Aware Bi-Level Sampling



1 thread



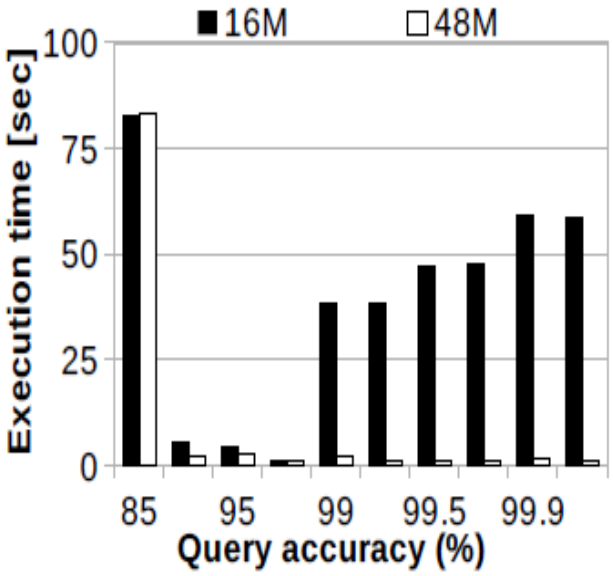
4 threads

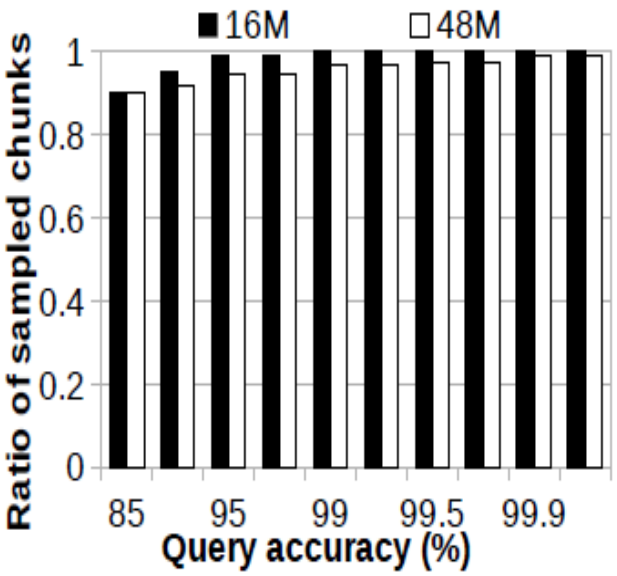


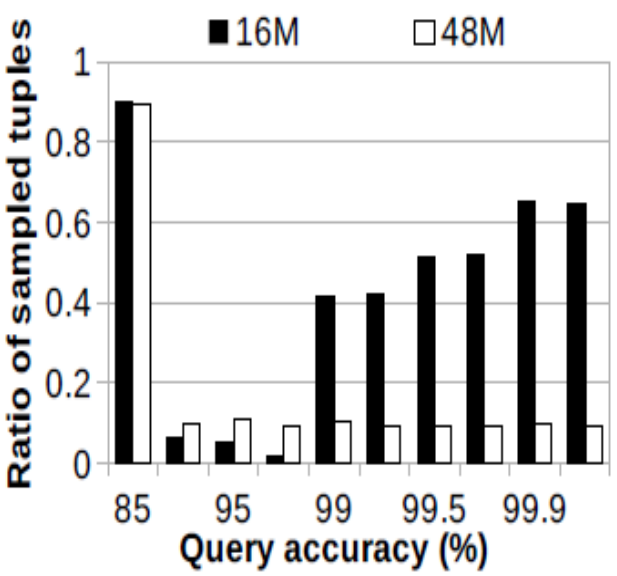
16 threads

UCMERCED

Sample Synopsis

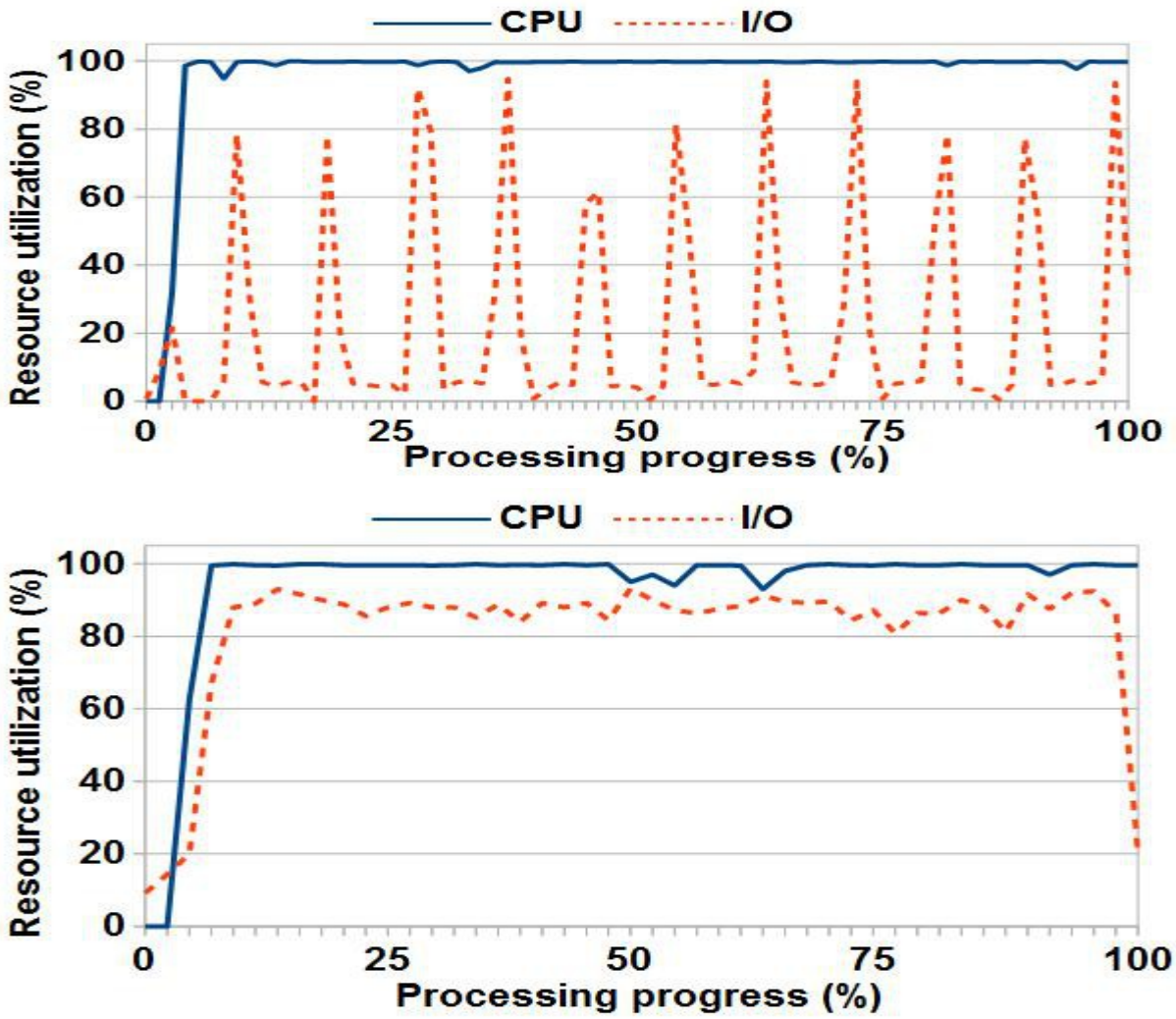




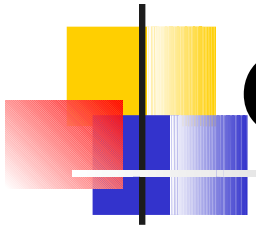


UCMERCED

Resource Utilization



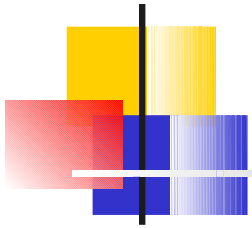
UCMERCED



Conclusions

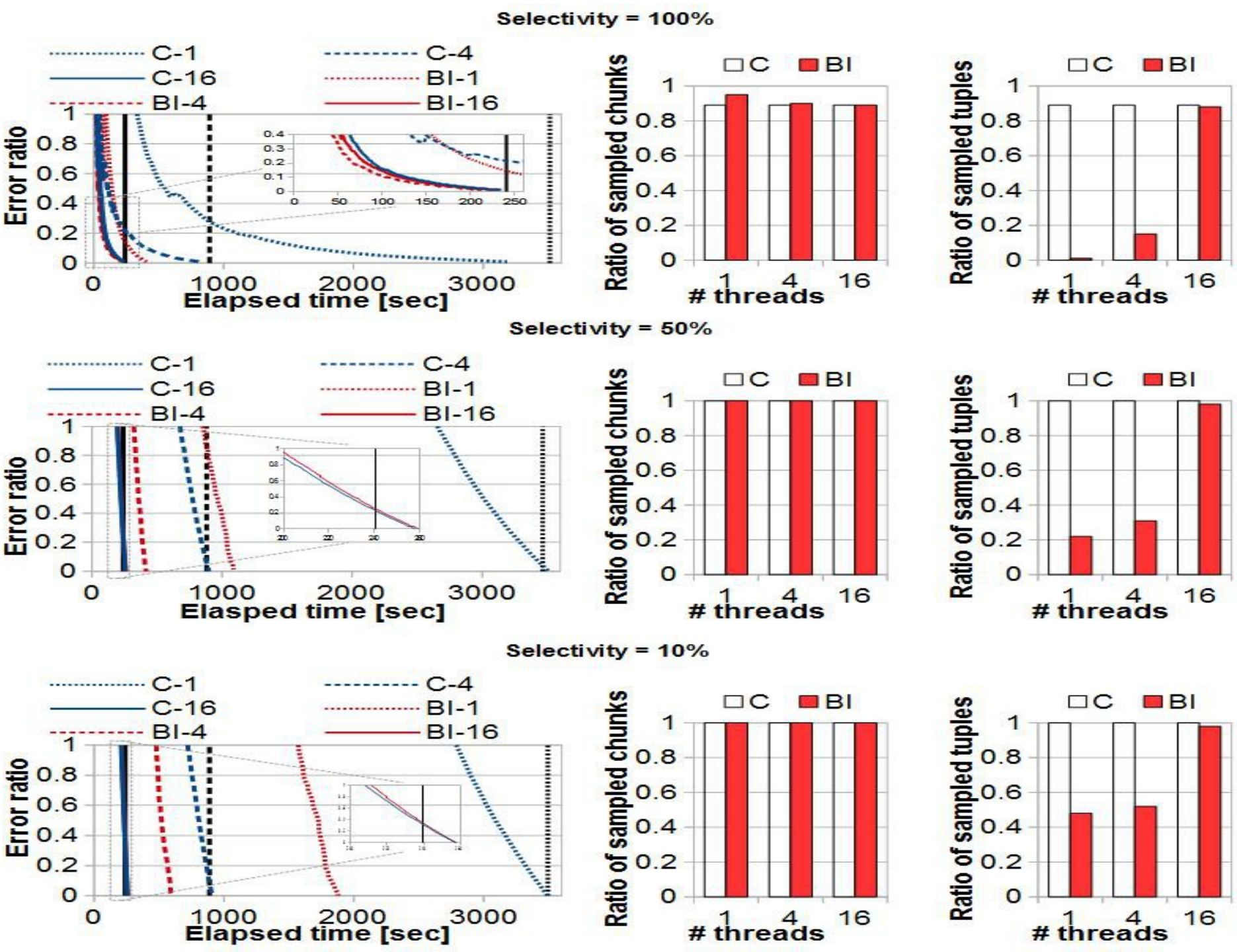
- OLA-RAW is a novel resource-aware bi-level sampling method for parallel on-line aggregation over raw data
- OLA-RAW is an efficient scheme for data exploration that avoids unnecessary work

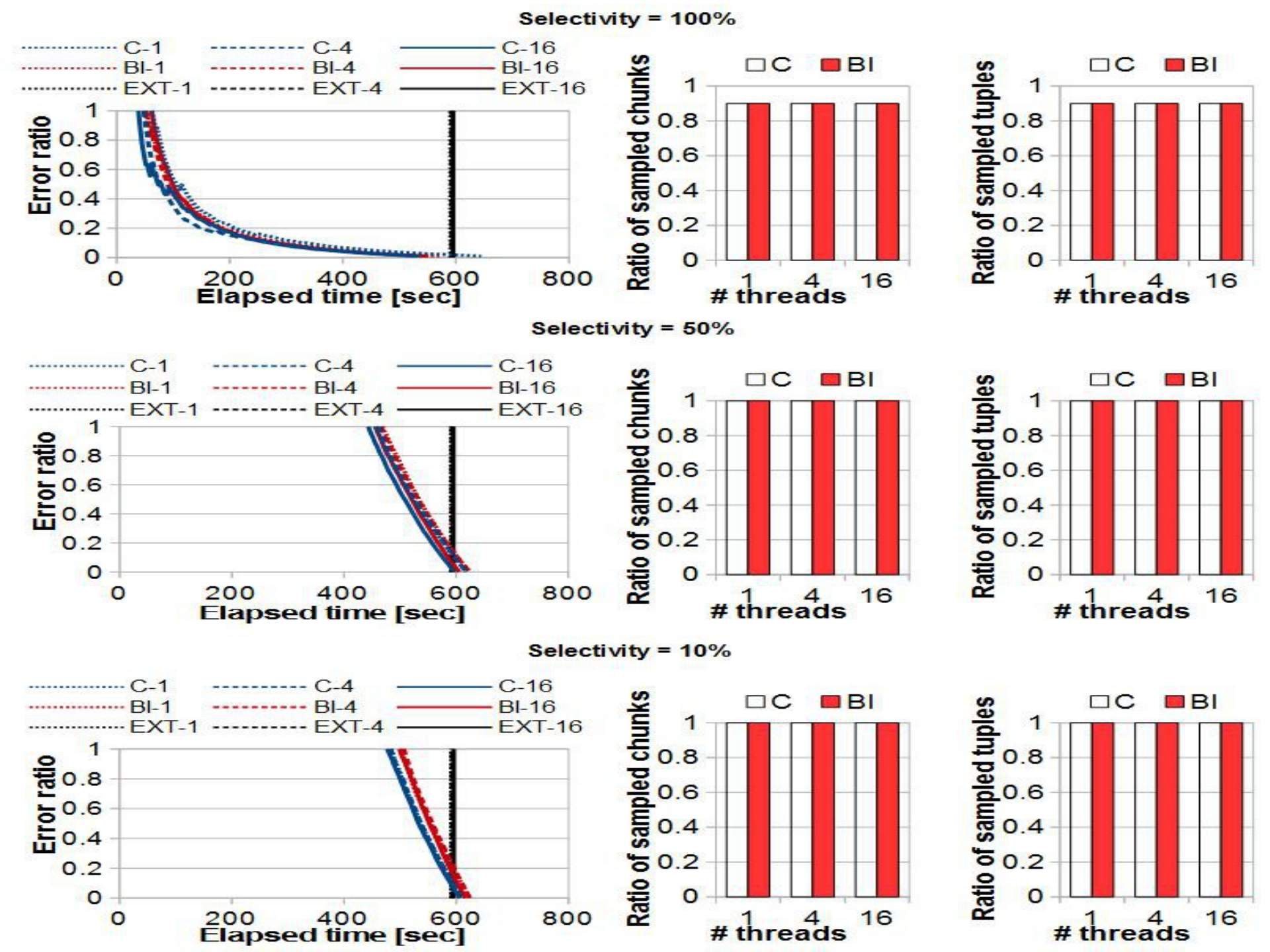
UCMERCED

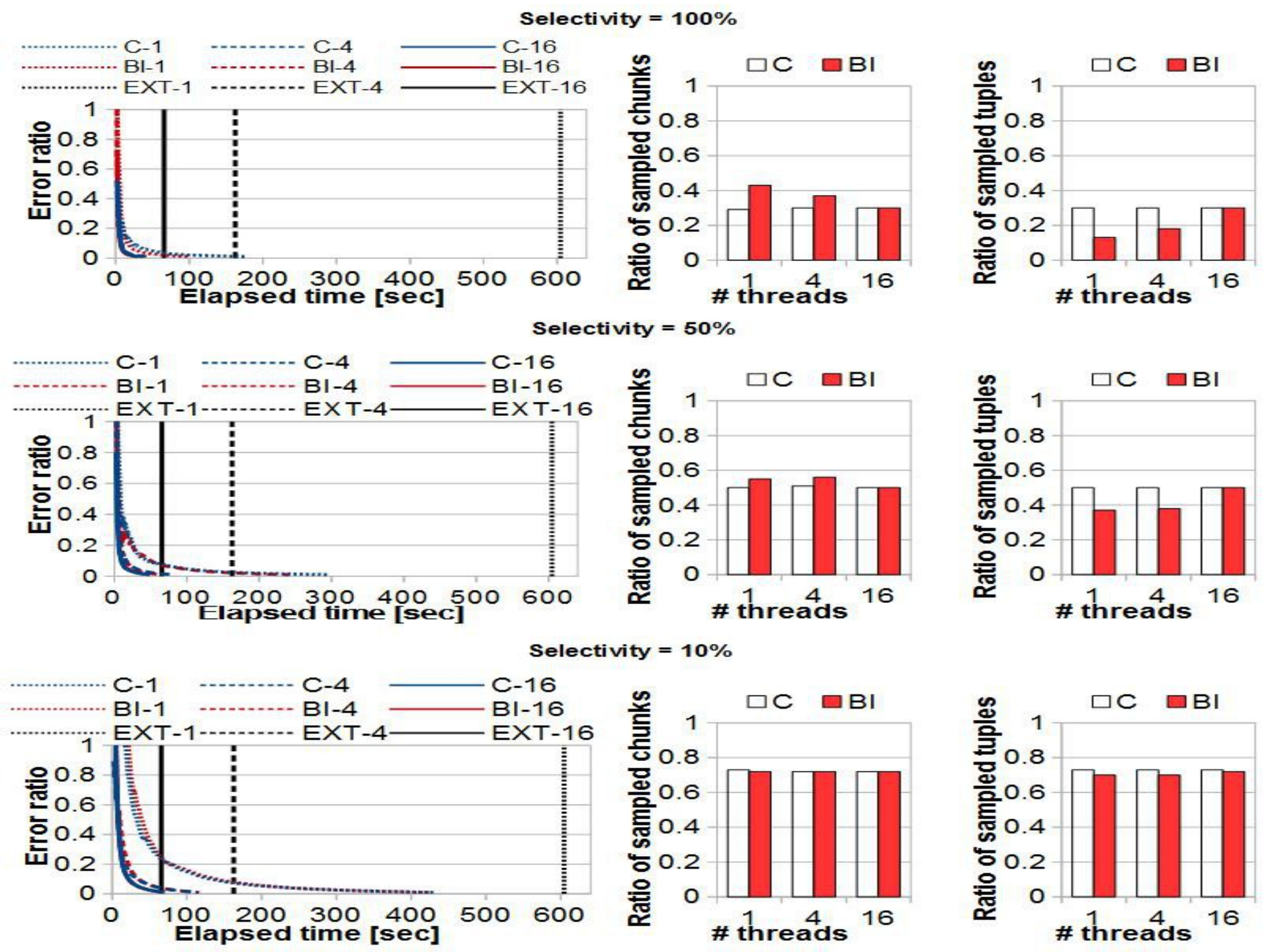


Thank you!

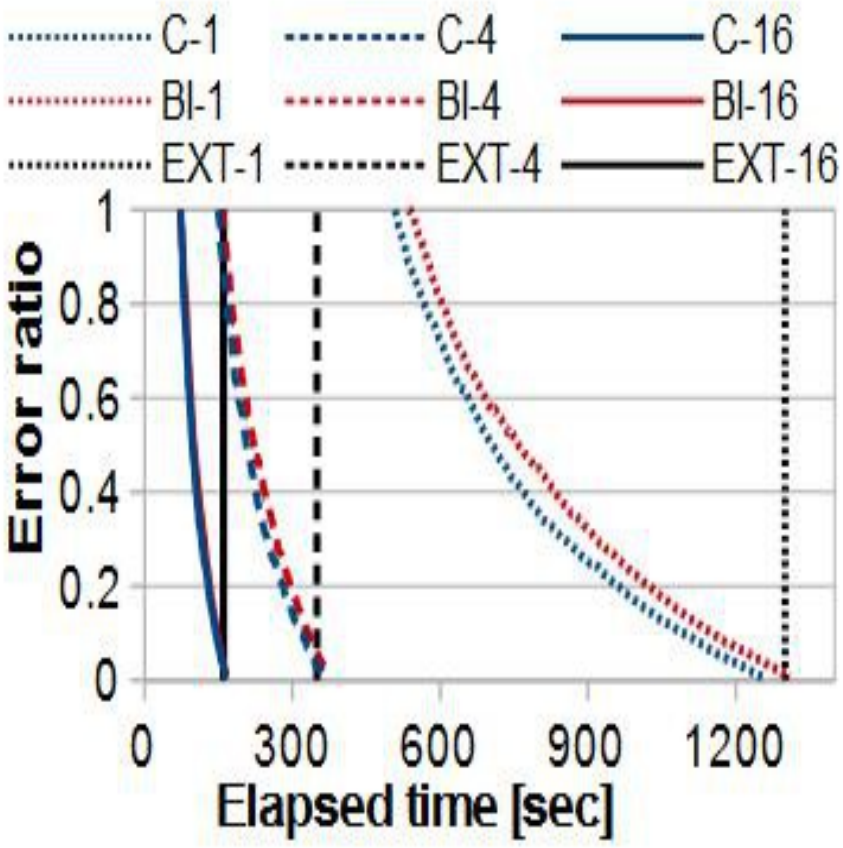
Questions?

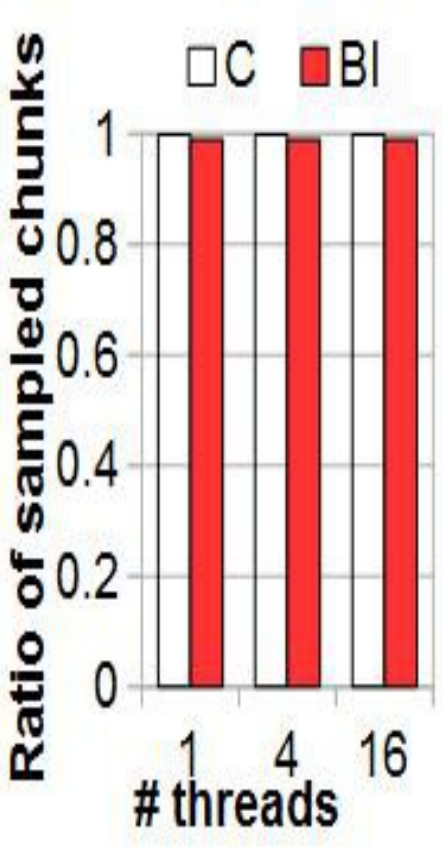

Selectivity = 100%
C-1
C-4
C-16
BI-1
BI-4
BI-16
Error ratio
Elapsed time [sec]
C
BI
Ratio of sampled chunks
threads
Ratio of sampled tuples
# threads
Selectivity = 50%
C-1
C-4
C-16
BI-1
BI-4
BI-16
Error ratio
Elasped time [sec]
C
BI
Ratio of sampled chunks
threads
Ratio of sampled tuples
# threads
Selectivity = 10%
C-1
C-4
C-16
BI-1
BI-4
BI-16
Error ratio
Elapsed time [sec]
C
BI
Ratio of sampled chunks
threads
Ratio of sampled tuples
threads

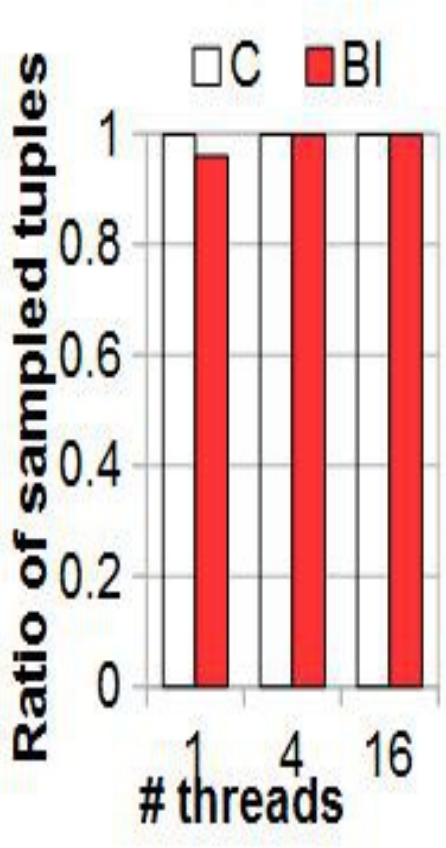

Selectivity = 100%
C-1
BI-1
EXT-1
C-4
BI-4
EXT-4
C-16
BI-16
EXT-16
Error ratio
Elapsed time [sec]
C
BI
Ratio of sampled chunks
threads
Ratio of sampled tuples
# threads
Selectivity = 50%
C-1
BI-1
EXT-1
C-4
BI-4
EXT-4
C-16
BI-16
EXT-16
Error ratio
Elapsed time [sec]
C
BI
Ratio of sampled chunks
threads
Ratio of sampled tuples
# threads
Selectivity = 10%
C-1
BI-1
EXT-1
C-4
BI-4
EXT-4
C-16
BI-16
EXT-16
Error ratio
Elapsed time [sec]
C
BI
Ratio of sampled chunks
threads
Ratio of sampled tuples
threads


Selectivity = 100%
C-1
C-4
C-16
BI-1
BI-4
BI-16
EXT-1
EXT-4
EXT-16
Error ratio
Elapsed time [sec]
C
BI
Ratio of sampled chunks
threads
Ratio of sampled tuples
# threads
Selectivity = 50%
C-1
C-4
C-16
BI-1
BI-4
BI-16
EXT-1
EXT-4
EXT-16
Error ratio
Elapsed time [sec]
C
BI
Ratio of sampled chunks
threads
Ratio of sampled tuples
# threads
Selectivity = 10%
C-1
C-4
C-16
BI-1
BI-4
BI-16
EXT-1
EXT-4
EXT-16
Error ratio
Elapsed time [sec]
C
BI
Ratio of sampled chunks
threads
Ratio of sampled tuples
threads

wiki


C-1
C-4
C-16
BI-1
BI-4
BI-16
EXT-1
EXT-4
EXT-16
Error ratio
1
0.8
0.6
0.4
0.2
0
0
300
600
900
1200
Elapsed time [sec]


C
BI
Ratio of sampled chunks
1
0.8
0.6
0.4
0.2
0
1
4
16
threads


C
BI
Ratio of sampled tuples
1
0.8
0.6
0.4
0.2
0
1
4
16
threads