UCMERCED

Incremental View Maintenance over Array Data

Weijie Zhao[1], Florin Rusu[1,2],
Bin Dong[2], Kesheng Wu[2], Peter Nugent[2]

University of California, Merced[1]
Lawrence Berkeley National Laboratory[2]

May 16, 2017

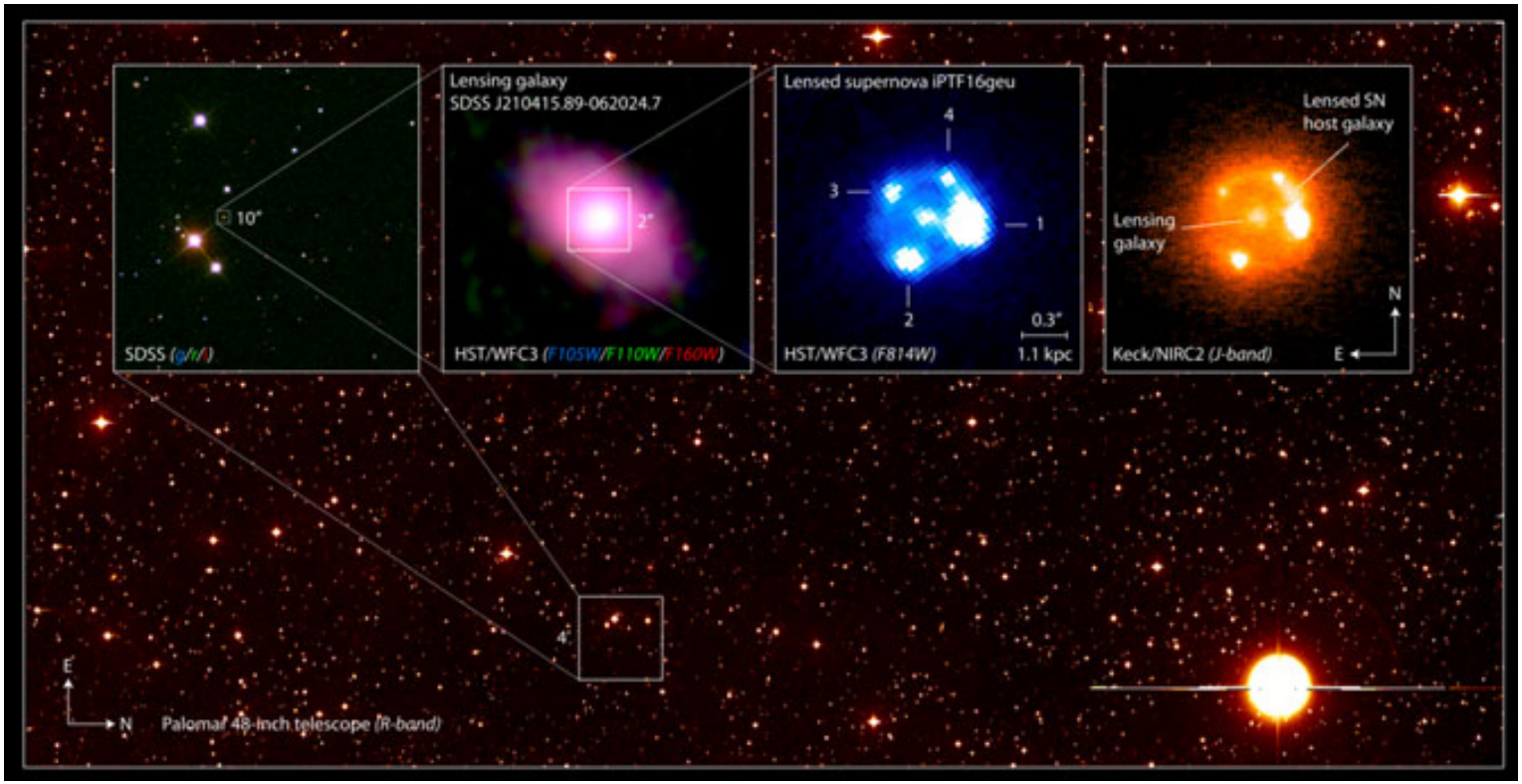
Outline

UCMERCED

Astronomical Catalogs

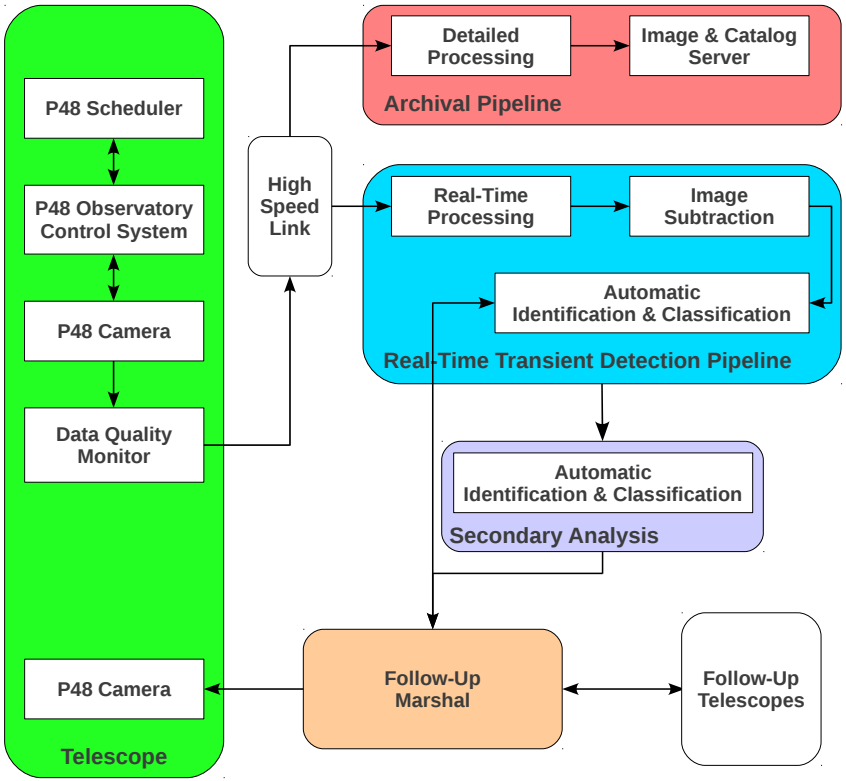
UCMERCED

- Sloan Digital Sky Survey (SDSS), Palomar Transient Factory (PTF), etc.
- Transient identification: supernovae detection, exoplanet search



Array Construction

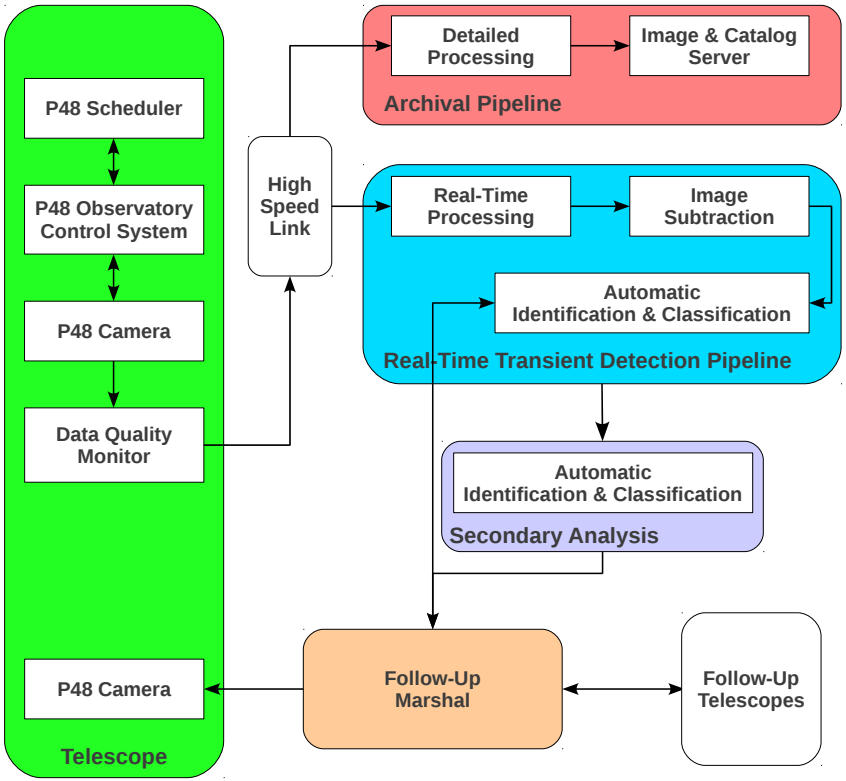
UCMERCED



- Image $\xrightarrow{\textit{Image processing}}$ Objects $\rightarrow$ 3-D array [`ra, dec, time`]

Array Construction

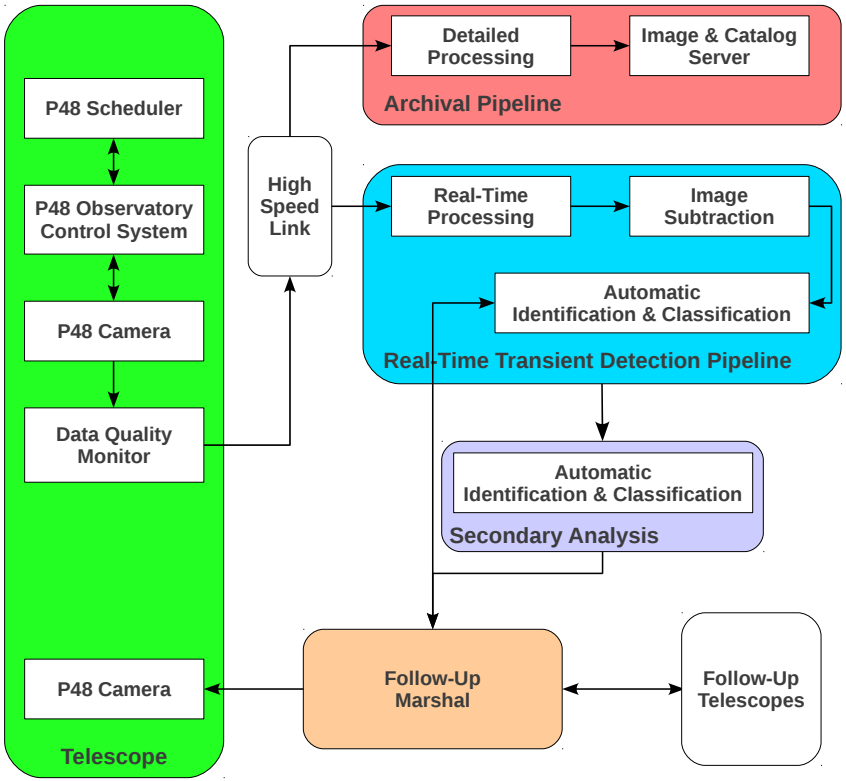
UCMERCED



- Image $\xrightarrow{\textit{Image processing}}$ Objects $\rightarrow$ 3-D array [`ra, dec, time`]
- Array is chunked and stored in a shared-nothing array database

Array Construction

UCMERCED



- Image $\xrightarrow{\textit{Image processing}}$ Objects $\rightarrow$ 3-D array [`ra, dec, time`]
- Array is chunked and stored in a shared-nothing array database
- Array is sparse and skewed

Astronomical Data Products

UCMERCED

- Gamma Ray Burst: extremely energetic explosions in distant galaxies that destroy everything around
  - For each dectection, all its neighbor detections are within 12 hours with increasing magnitude
- M-Dwarf Flares: periodic flares with period larger than a month
  - No two neighbor dectections within a month

Astronomical Data Products

UCMERCED

- Neighborhood matching $\rightarrow$ Array similarity join (expensive)
- Astronomical data products $\rightarrow$ Materialized query result

Astronomical Data Products

UCMERCED

- Neighborhood matching → Array similarity join (expensive)
- Astronomical data products → Materialized query result
- New detections every night → Base array updates
- Maintained data products → Incremental view maintenance over arrays

News Headline

UCMERCED

Rare Supernova Discovery Ushers in New Era for Cosmology

Berkeley Lab astrophysicists develop novel method for finding gravitationally lensed Type 1a supernovae

News Release Lir

'Major breakthrough' as supernova four billion light years from Earth is captured from four different angles in a stunning world first

- **Strange event occurred because light from the supernova bent through a galaxy**
- **Galaxies bend light through an effect that is called 'gravitational lensing'**
- **The galaxy magnified the supernova 50 times to give astronomers a unique view**
- **Supernovae are important to study as they reveal how the universe expands**

By HARRY PETTIT FOR MAILONLINE
PUBLISHED: 14:00 EDT, 20 April 2017 | **UPDATED:** 14:01 EDT, 20 April 2017

Astronomers Get Rare View Of Type Ia Supernova Magnified 50 Times

21 April 2017, 9:01 am EDT **By Luan Chan Tech Times**

Contributions

UCMERCED

- Formal array view definition

Contributions

UCMERCED

- Formal array view definition
- Model incremental array view maintenance as an optimization formulation – including adaptive array and view reorganization

Contributions

UCMERCED

- Formal array view definition
- Model incremental array view maintenance as an optimization formulation – including adaptive array and view reorganization
- Propose an effective three-stage heuristics

Contributions

UCMERCED

- Formal array view definition
- Model incremental array view maintenance as an optimization formulation – including adaptive array and view reorganization
- Propose an effective three-stage heuristics
- Introduce an analytical cost model for query integration

Contributions

UCMERCED

- Formal array view definition
- Model incremental array view maintenance as an optimization formulation – including adaptive array and view reorganization
- Propose an effective three-stage heuristics
- Introduce an analytical cost model for query integration
- Experimentally compare against existing solutions on PTF catalog and LinkedGeoData

Incremental Array View Maintenance Axes

UCMERCED

- Views over array similarity join (Zhao 2016)
  - Encompasses a larger variety of join predicates
  - Composable

Incremental Array View Maintenance Axes

UCMERCED

- Views over array similarity join (Zhao 2016)
  - Encompasses a larger variety of join predicates
  - Composable
- Batch updates – (He 2005, Katsis 2015)
  - Not only to reduce the per-transaction overhead
  - Share delta computation inside batch

Incremental Array View Maintenance Axes

UCMERCED

- Views over array similarity join (Zhao 2016)
  - Encompasses a larger variety of join predicates
  - Composable
- Batch updates – (He 2005, Katsis 2015)
  - Not only to reduce the per-transaction overhead
  - Share delta computation inside batch
- Update granularity – chunk (Brown 2010, Duggan 2014)

Incremental Array View Maintenance Axes

UCMERCED

- Views over array similarity join (Zhao 2016)
  - Encompasses a larger variety of join predicates
  - Composable
- Batch updates – (He 2005, Katsis 2015)
  - Not only to reduce the per-transaction overhead
  - Share delta computation inside batch
- Update granularity – chunk (Brown 2010, Duggan 2014)
- Aggregates – standard SQL aggregate functions, e.g., SUM, COUNT, AVG (Yang 2001, Palpanas 2002)

Incremental Array View Maintenance Axes

UCMERCED

- Views over array similarity join (Zhao 2016)
  - Encompasses a larger variety of join predicates
  - Composable
- Batch updates – (He 2005, Katsis 2015)
  - Not only to reduce the per-transaction overhead
  - Share delta computation inside batch
- Update granularity – chunk (Brown 2010, Duggan 2014)
- Aggregates – standard SQL aggregate functions, e.g., SUM, COUNT, AVG (Yang 2001, Palpanas 2002)
- Distributed processing (Luo 2003, Liu 2005)
  - Potential join pairs for each chunk are magnified

Incremental Array View Maintenance Axes

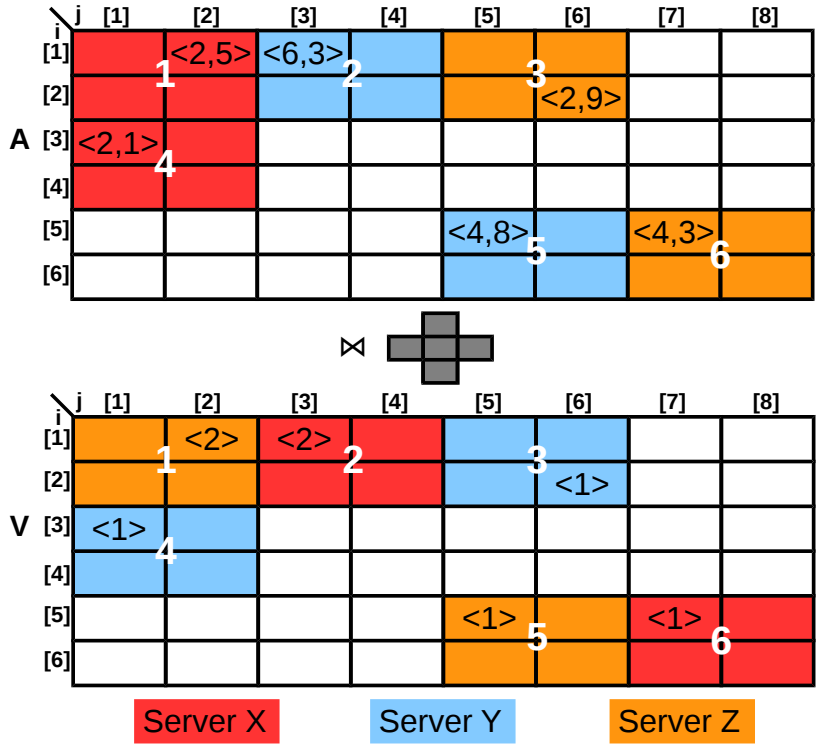
UCMERCED

- Views over array similarity join (Zhao 2016)
  - Encompasses a larger variety of join predicates
  - Composable
- Batch updates – (He 2005, Katsis 2015)
  - Not only to reduce the per-transaction overhead
  - Share delta computation inside batch
- Update granularity – chunk (Brown 2010, Duggan 2014)
- Aggregates – standard SQL aggregate functions, e.g., SUM, COUNT, AVG (Yang 2001, Palpanas 2002)
- Distributed processing (Luo 2003, Liu 2005)
  - Potential join pairs for each chunk are magnified
- Recursive maintenance (Ahmad 2012, Nikolic 2016)

Outline

UCMERCED

Array Data Model




```
CREATE ARRAY VIEW V AS
  SELECT COUNT(*) AS cnt
  FROM A A1 SIMILARITY JOIN A A2
       ON (A1.i = A2.i) AND (A1.j = A2.j)
       WITH SHAPE L1(1)
  GROUP BY A1.i, A1.j
```

Array View Definition

UCMERCED

n multi-dimensional arrays $\alpha_1, \ldots, \alpha_n$
k unary array operators $\oplus_1, \ldots, \oplus_k$
$$V \leftarrow\!\shortmid \oplus_1 \left(\cdots \oplus_k \left(\alpha_1 \bowtie^{\mathcal{M}_1}_{\sigma_1, f_1} \alpha_2 \bowtie^{\mathcal{M}_2}_{\sigma_2, f_2} \cdots \bowtie^{\mathcal{M}_{n-1}}_{\sigma_{n-1}, f_{n-1}} \alpha_n \right) \cdots \right)$$

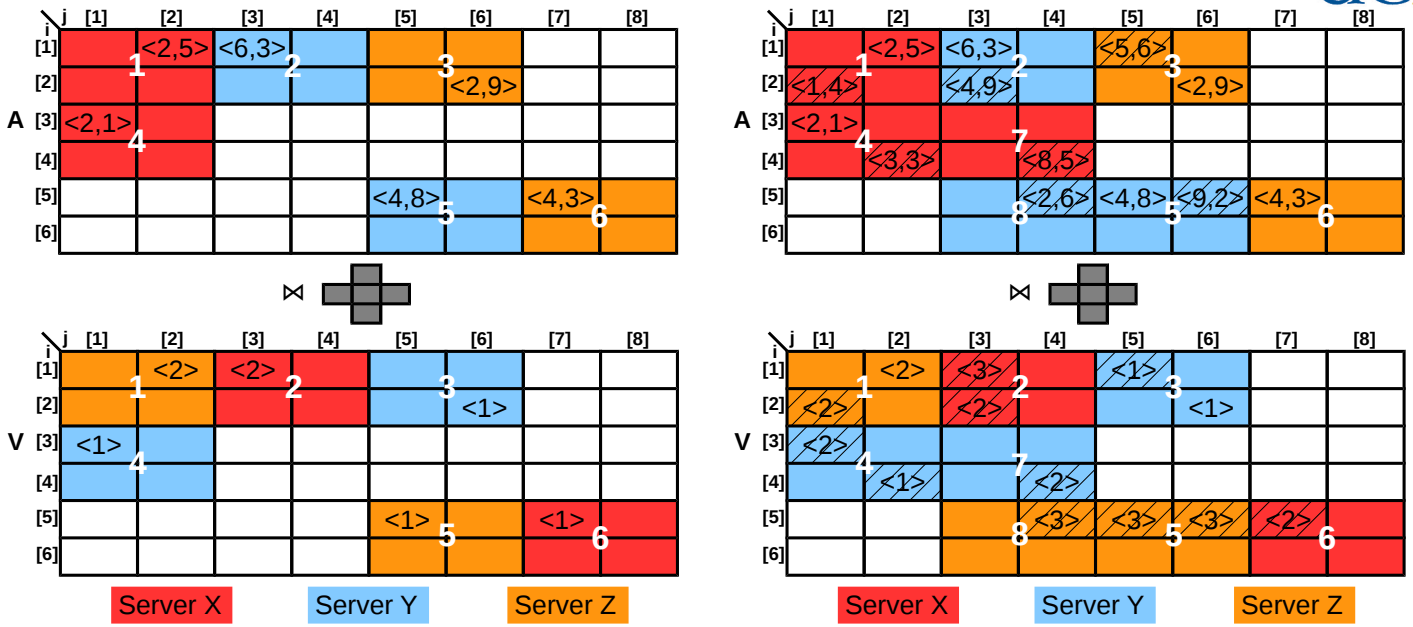
For example:

```
CREATE ARRAY VIEW V AS
  SELECT COUNT(*) AS cnt
  FROM A A1 SIMILARITY JOIN A A2
       ON (A1.i = A2.i) AND (A1.j = A2.j)
       WITH SHAPE L^1(1)
  GROUP BY A1.i, A1.j
```

Outline

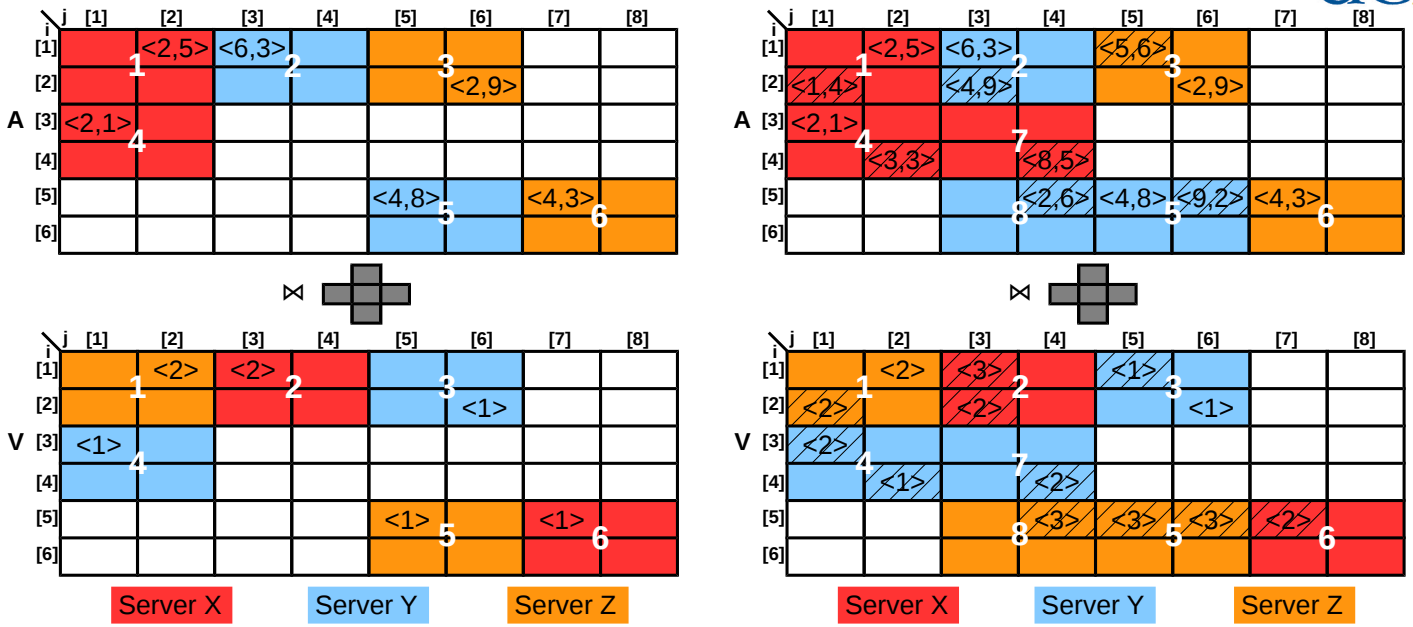
UCMERCED

Baseline Array View Maintenance



- For each update batch ΔD:
  - Assign ΔD to node according to chunking strategy
  - $Q(D, \Delta D)$ – differential view computation
  - $M(D) + \Delta Q(D, \Delta D)$ – view aggregation

Baseline Array View Maintenance



- For each update batch ΔD:
  - Assign ΔD to node according to chunking strategy
  - $Q(D, \Delta D)$ – differential view computation
  - $M(D) + \Delta Q(D, \Delta D)$ – view aggregation
- Excessive communication
- Load imbalance

Optimal Array View Maintenance

UCMERCED

- Array chunk (view chunk) reassignment – not stuck with an unfavorable static chunking strategy

Optimal Array View Maintenance

UCMERCED

- Array chunk (view chunk) reassignment – not stuck with an unfavorable static chunking strategy
- Incoming chunks are not first assigned to node based on pre-determined chunking strategy

Optimal Array View Maintenance

UCMERCED

- Array chunk (view chunk) reassignment – not stuck with an unfavorable static chunking strategy
- Incoming chunks are not first assigned to node based on pre-determined chunking strategy
- Piggyback on the chunk replication incurred by view maintenance when computing the reassignment

Optimal Array View Maintenance

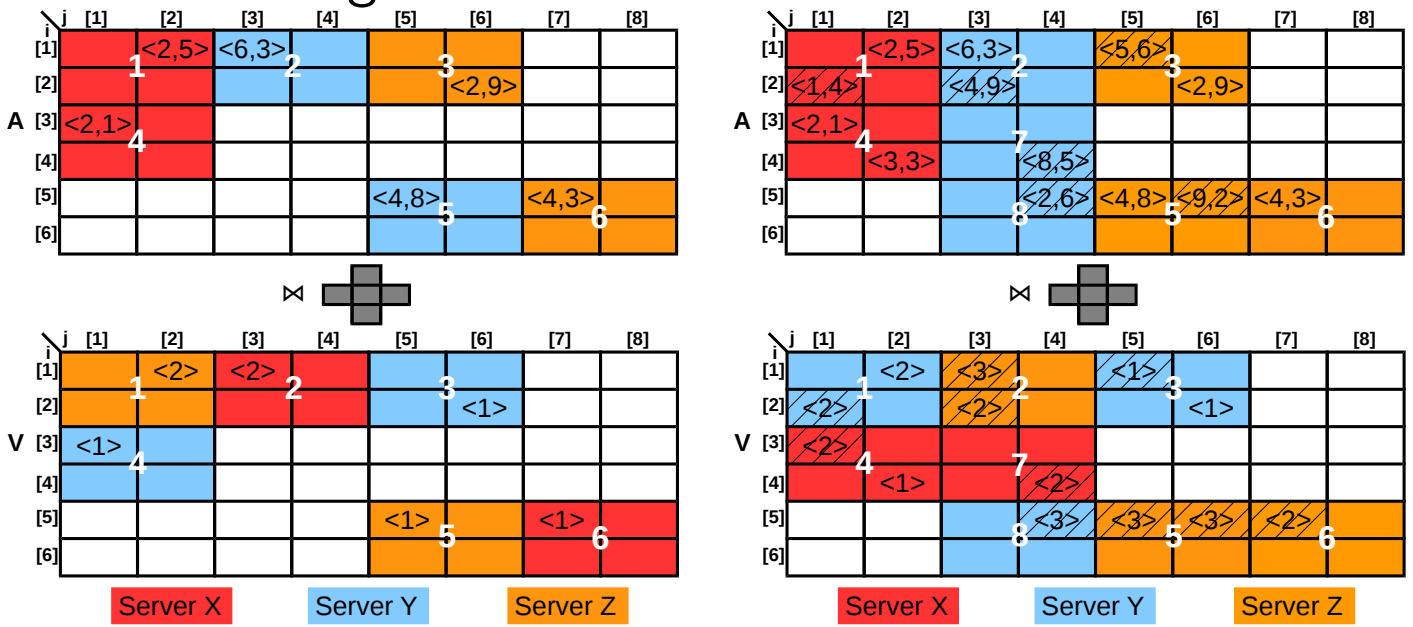
UCMERCED

- Array chunk (view chunk) reassignment – not stuck with an unfavorable static chunking strategy
- Incoming chunks are not first assigned to node based on pre-determined chunking strategy
- Piggyback on the chunk replication incurred by view maintenance when computing the reassignment
- A window of past batch updates is considered to avoid unstable reassignments

Optimal Array View Maintenance

UCMERCED

- Array chunk (view chunk) reassignment – not stuck with an unfavorable static chunking strategy
- Incoming chunks are not first assigned to node based on pre-determined chunking strategy
- Piggyback on the chunk replication incurred by view maintenance when computing the reassignment
- A window of past batch updates is considered to avoid unstable reassignments



Analytical Cost Model

UCMERCED

$$\min\left\{\lambda\cdot\max\left\{\max_k\left\{\sum_{i,j}^{k\neq j}x_{ikj}\cdot B_i T_{ntwk}+\sum_{p,q,v,j}^{k\neq j,(p,q,v)\in U_0}z_{pqk}\cdot y_{vj}\cdot B_{pq}T_{ntwk}\right\},\right.\right.$$
$$\left.\left.\max_k\left\{\sum_{p,q}z_{pqk}\cdot B_{pq}T_{cpu}\right\}\right\}$$
$$+(1-\lambda)\cdot\max\left\{\max_k\left\{\sum_{i,j}^{k\neq j}x'_{ikj}\cdot B_i T_{ntwk}+\sum_{p,q,v,j,l}^{k\neq j,(p,q,v)\in U_l}z'_{pqk}\cdot y_{vj}\cdot W_l B_{pq}T_{ntwk}\right\},\right.$$
$$\left.\left.\max_k\left\{\sum_{p,q,l}^{(p,q)\in U_l}z'_{pqk}\cdot W_l B_{pq}T_{cpu}\right\}\right\}\right\}$$

Network time to transfer chunks

Network time to transfer partial join results

CPU time to compute aggregations

Three-stage Heuristic

UCMERCED

Complicated formulation $\rightarrow$ decoupling

Three-stage Heuristic

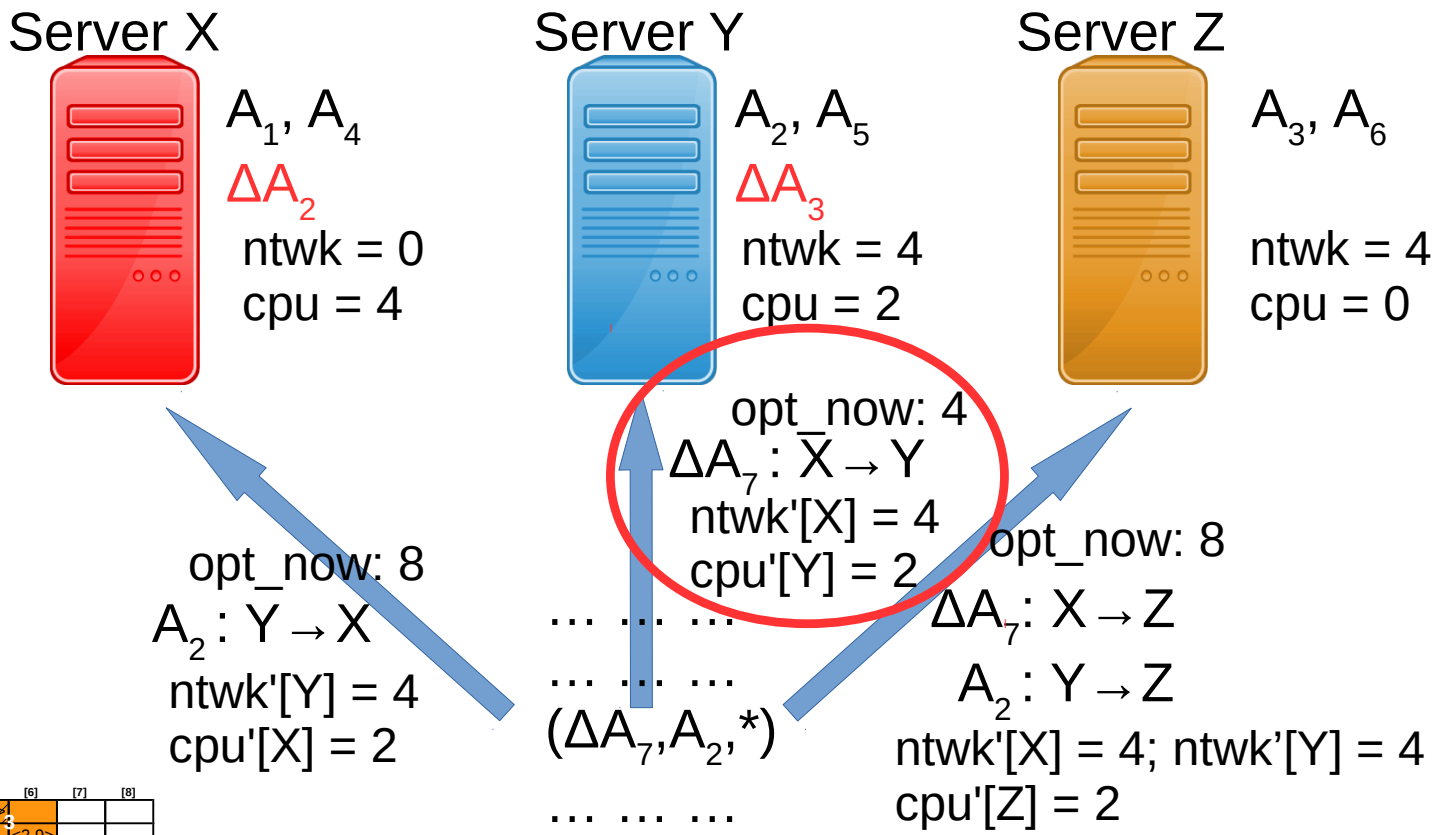
UCMERCED

Complicated formulation $\rightarrow$ decoupling

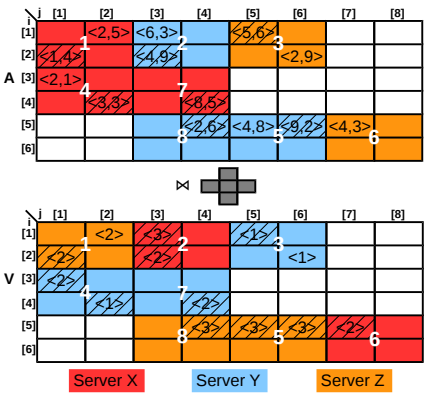
- Differential view computation $\rightarrow$ x_{ikj}, z_{pqk}
- View chunk reassignment $\rightarrow$ y_{vj}
- Array chunk reassignment $\rightarrow$ y_{aj}

Differential View Computation

UCMERCED



opt_now = max{ntwk + ntwk', cpu + cpu'}



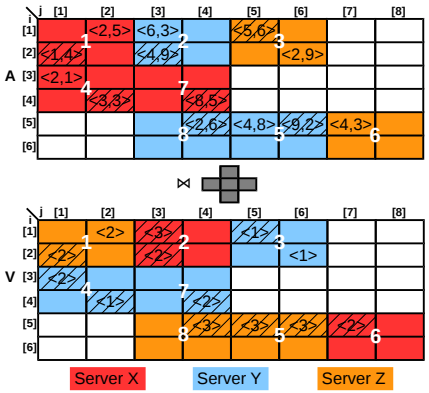
View Chunk Reassignment

UCMERCED

server	*ntwk*	*cpu*
X	32	36
Y	36	30
Z	30	35

join result	server
$J_1 : \Delta A_1 \bowtie A_1$	X
$J_2 : \Delta A_4 \bowtie A_1$	X
$J_3 : \Delta A_2 \bowtie A_1$	Y

$V_1 \rightarrow$	transfers	*ntwk′*	*cpu′*	*opt_now*
X	J_3	$Y = 4$	$X = 6$	42
Y	J_1, J_2	$X = 8$	$Y = 6$	**40**
Z	J_1, J_2, J_3	$X = 8,\ Y = 4$	$Z = 6$	41



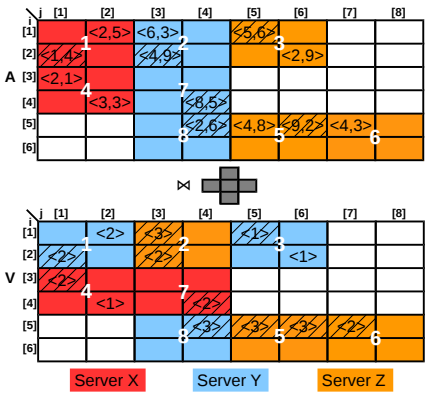
Array Chunk Reassignment

UCMERCED

score	
A_2, V_1	8
A_1, V_1	6
A_1, V_2	4
A_2, V_3	4
A_3, V_3	2
...	

server	views	cpu_thr
X	V_2, V_6	4
Y	V_1, V_4, V_7	3
Z	V_3, V_5, V_8	1

	A_1	A_2	A_3	...
size	1	1	1	...
replica	X, Z	Y, Z	Z, Y	



Outline

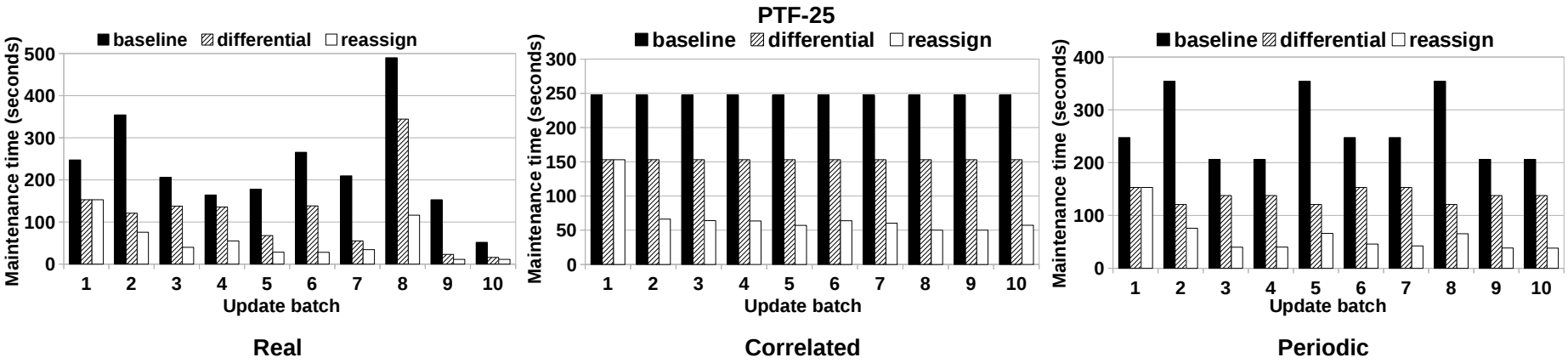
UCMERCED

View Maintenance Time

UCMERCED

`PTF[time=1,153064;ra=1,100000;dec=1,50000]`

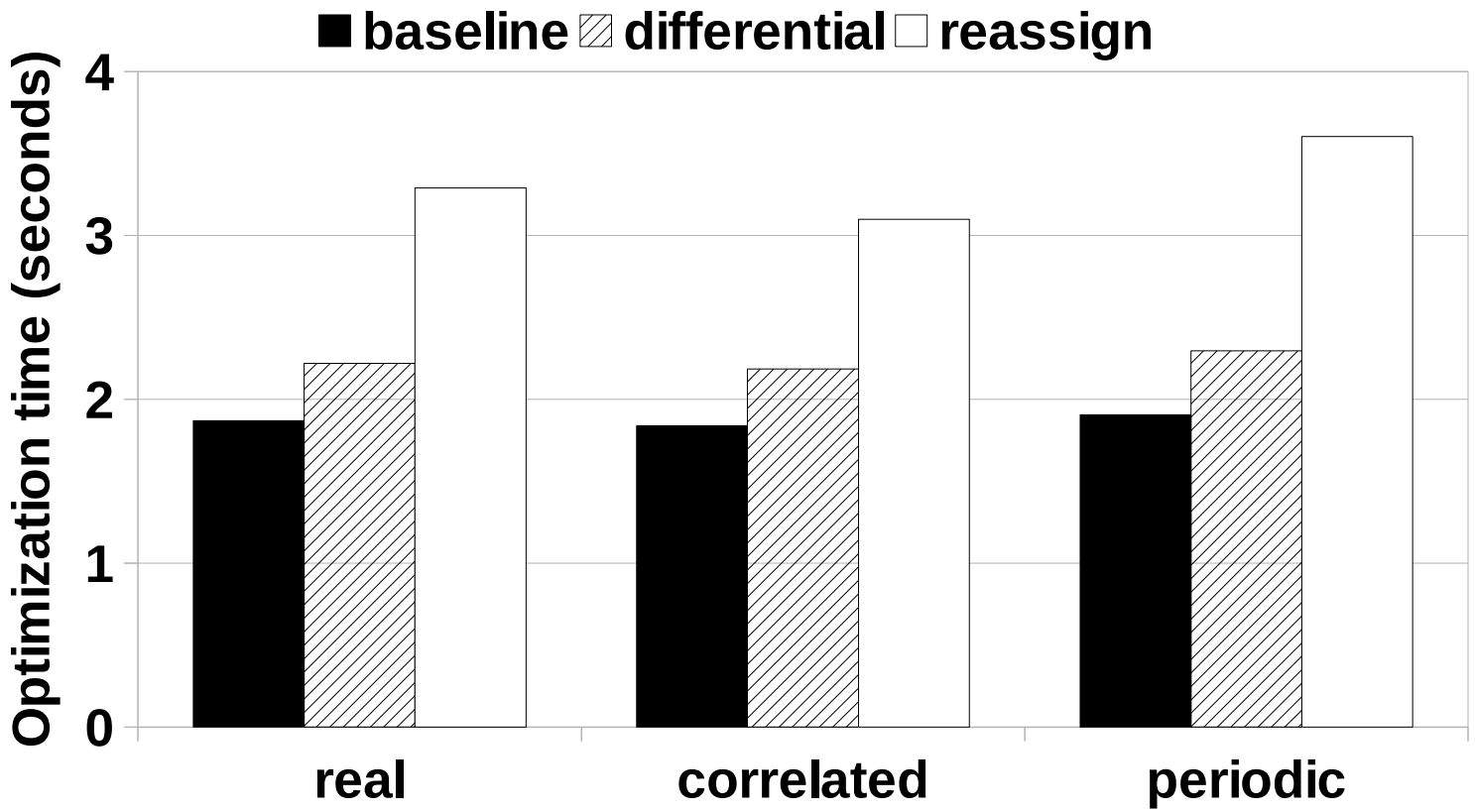
1 billion time-stamped objects, 343 GB



- Real: real updates from PTF pipeline
- Correlated: updates focusing on the same place
- Periodic: updates from several regions periodically – 1, 2, 3, 3, 2, 1, ...

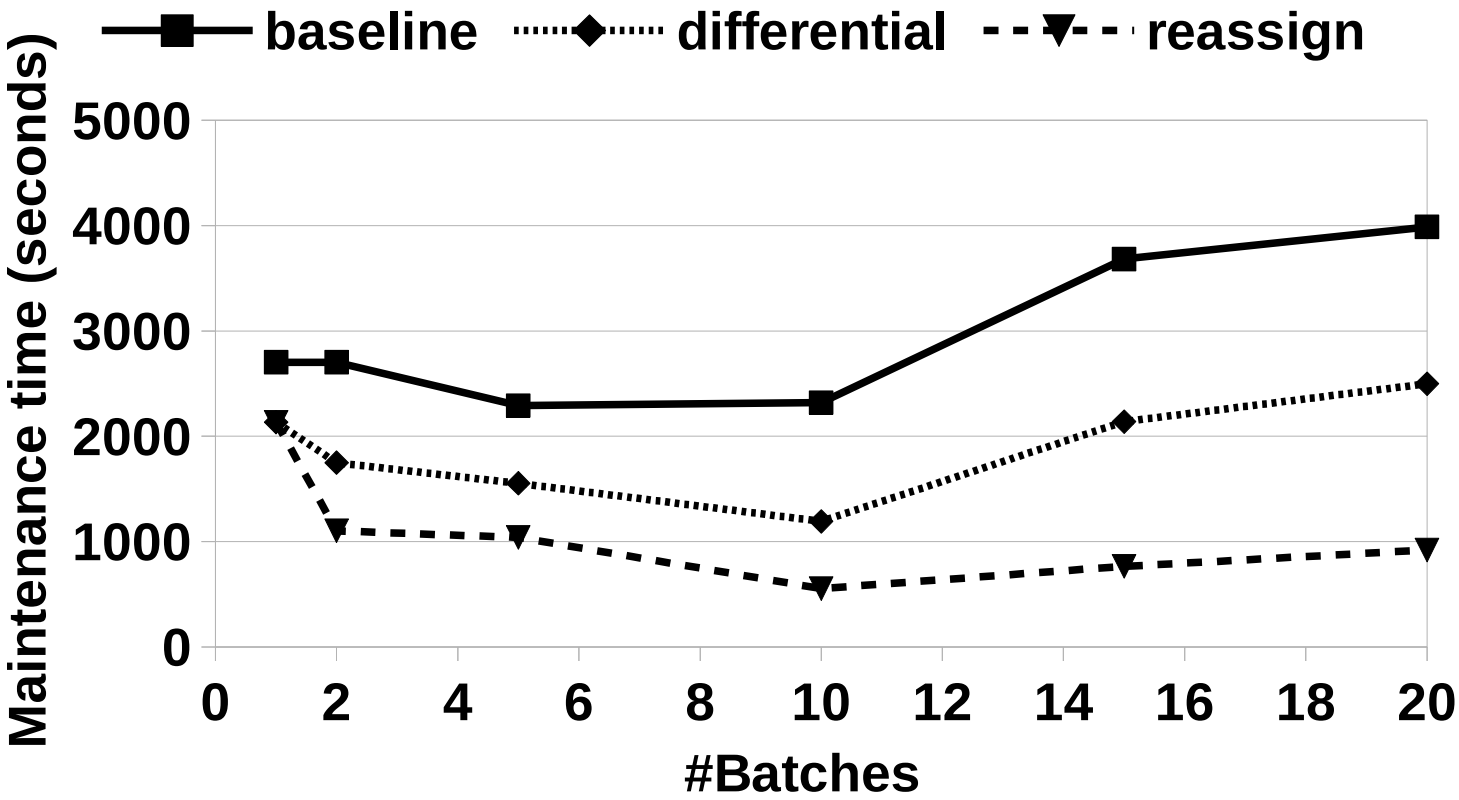
Average Optimization Time per Update Batch

UCMERCED



Sensitivity to #Batches

UCMERCED



Outline

UCMERCED

Conclusions

UCMERCED

- Introduce materialized array views as a database construct for derived data products in science

Conclusions

UCMERCED

- Introduce materialized array views as a database construct for derived data products in science
- Model incremental array view maintenance with batch updates as an MIP optimization and give a three-stage heuristic that finds effective update plans

Conclusions

UCMERCED

- Introduce materialized array views as a database construct for derived data products in science
- Model incremental array view maintenance with batch updates as an MIP optimization and give a three-stage heuristic that finds effective update plans
- Heuristic repartitions the array and the view continuously based on a window of past updates as a side-effect of view maintenance

Conclusions

UCMERCED

- Introduce materialized array views as a database construct for derived data products in science
- Model incremental array view maintenance with batch updates as an MIP optimization and give a three-stage heuristic that finds effective update plans
- Heuristic repartitions the array and the view continuously based on a window of past updates as a side-effect of view maintenance
- Design an analytical cost model for integrating materialized array views in queries

Conclusions

UCMERCED

- Introduce materialized array views as a database construct for derived data products in science
- Model incremental array view maintenance with batch updates as an MIP optimization and give a three-stage heuristic that finds effective update plans
- Heuristic repartitions the array and the view continuously based on a window of past updates as a side-effect of view maintenance
- Design an analytical cost model for integrating materialized array views in queries
- Experimental results confirm the effectiveness of the heuristics and the quality of the maintenance plan

Conclusions

UCMERCED

- Introduce materialized array views as a database construct for derived data products in science
- Model incremental array view maintenance with batch updates as an MIP optimization and give a three-stage heuristic that finds effective update plans
- Heuristic repartitions the array and the view continuously based on a window of past updates as a side-effect of view maintenance
- Design an analytical cost model for integrating materialized array views in queries
- Experimental results confirm the effectiveness of the heuristics and the quality of the maintenance plan
- Incrementally maintain the production PTF "association table" and help find rare supernovae at Berkeley Lab and CalTech

Thank you!
Questions?