YourSQL: A High-Performing Database System Leveraging In-Storage Computing

I. Jo et al, VLDB 2016
Presented by Nishad Phadke

April 18, 2017

Motivation

- YourSQL is designed for data-intensive database queries
- Intuition: $\downarrow$ data transferred $\Rightarrow$ $\uparrow$ query speed
- *Early filtering* is data-intensive but non-complex
- Number of I/O requests is largest contributor to cost of data-itensive operations

Motivation

- YourSQL is designed for data-intensive database queries
- Intuition: $\downarrow$ data transferred $\Rightarrow \uparrow$ query speed
- *Early filtering* is data-intensive but non-complex
- Number of I/O requests is largest contributor to cost of data-itensive operations

In-storage computing (ISC) in Solid State Drives (SSDs)

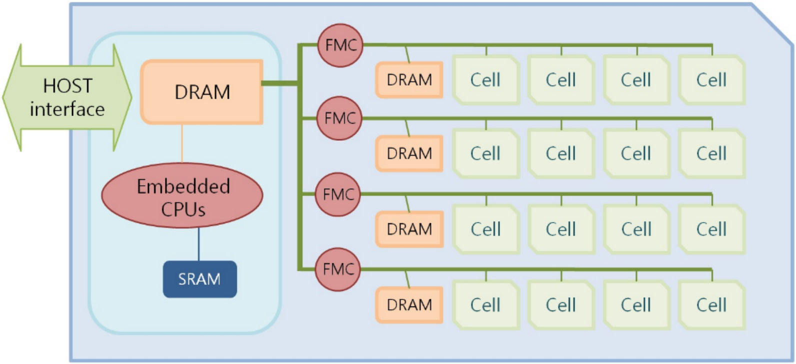


Figure: From Bae, et al.[1]

SSD photo from YourSQL

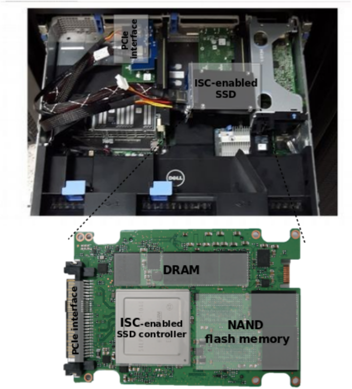
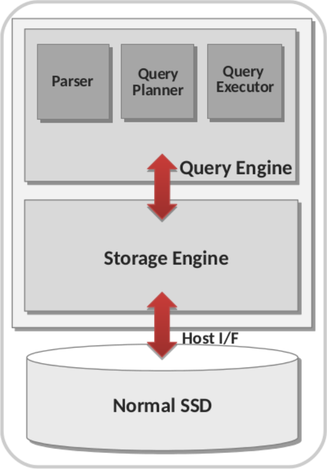


Figure 6: Dell R720 server with PM1725 SSD.

Database Architecture



Definitions

Query 1: A simple selection query.

```
SELECT p_partkey, p_mfgr FROM part
WHERE p_size = 15 AND p_type LIKE '%BRASS';
```

- *Selectivity:* fraction of relevant rows from row set
  - 0 is highest (most selective)
  - 1 is lowest
- $\uparrow$ selectivity $\not\Rightarrow$ $\uparrow$ query speed

- *Filtering ratio:* fraction of relevant pages from page set
- Filtering ratio is much better indicator of speed-up guarantees

Definitions

Query 1: A simple selection query.

```
SELECT p_partkey, p_mfgr FROM part
WHERE p_size = 15 AND p_type LIKE '%BRASS';
```

- *Selectivity:* fraction of relevant rows from row set
  - 0 is highest (most selective)
  - 1 is lowest
- $\uparrow$ selectivity $\not\Rightarrow$ $\uparrow$ query speed

- *Filtering ratio:* fraction of relevant pages from page set
- Filtering ratio is much better indicator of speed-up guarantees

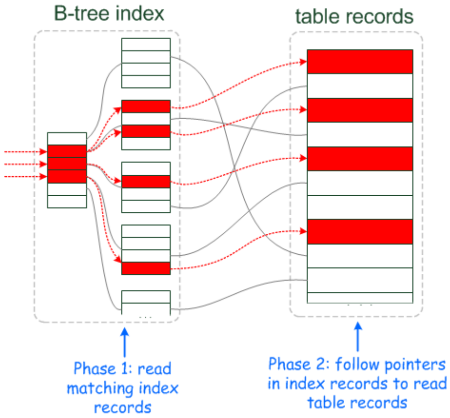
Query 1

Query 1: A simple selection query.

```
SELECT p_partkey, p_mfgr FROM part
WHERE p_size = 15 AND p_type LIKE '%BRASS';
```

What if the `WHERE` predicates are indices?

Index Condition Pushdown (ICP)[1]



[1] https://mariadb.com/kb/en/mariadb/index-condition-pushdown/

Query 1 (again)

Query 1: A simple selection query.

SELECT p_partkey, p_mfgr **FROM** part
WHERE p_size = 15 **AND** p_type **LIKE** '%BRASS';

What if the `WHERE` predicates are not indices?

Filtering Condition Pushdown (FCP)

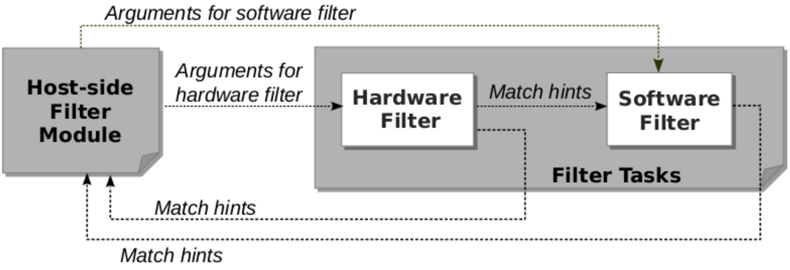


Figure 4: ISC filters.

Hardware filtering

$$\begin{aligned}
\mathcal{X} &= \text{lots of dates between 1990 and 2020} \\
M_{2016,9} &= \text{dates in 2016 September} \\
M_{2016,10} &= \text{dates in 2016 October} \\
M_{2016,11} &= \text{dates in 2016 November} \\
Y_{2016} &= \text{dates in 2016}
\end{aligned}$$

$$\left[\mathcal{X} \cap M_{2016,9}\right] \cup \left[\mathcal{X} \cap M_{2016,10}\right] \cup \left[\mathcal{X} \cap M_{2016,11}\right] \subseteq \left[\mathcal{X} \cap Y_{2016}\right]$$

Hardware filtering

$$\begin{aligned}
\mathcal{X} &= \text{lots of dates between 1990 and 2020} \\
M_{2016,9} &= \text{dates in 2016 September} \\
M_{2016,10} &= \text{dates in 2016 October} \\
M_{2016,11} &= \text{dates in 2016 November} \\
Y_{2016} &= \text{dates in 2016}
\end{aligned}$$

$$\left[\mathcal{X} \cap M_{2016,9}\right] \cup \left[\mathcal{X} \cap M_{2016,10}\right] \cup \left[\mathcal{X} \cap M_{2016,11}\right] \subseteq \left[\mathcal{X} \cap Y_{2016}\right]$$

Hardware filtering

Query 3: TPC-H Q14.

```
SELECT 100.00 * SUM(CASE WHEN p_type LIKE 'PROMO%'
  THEN l_extendedprice * (1-l_discount) ELSE 0 END) /
  SUM(l_extendedprice * (1-l_discount)) AS promo_revenue
FROM lineitem, part
WHERE l_partkey = p_partkey
  AND l_shipdate >= date '1995-09-01'
  AND l_shipdate < date '1995-09-01' + INTERVAL '1' MONTH;
```

```
date >= '1995-09-01' AND date < '1995-10-01'
date >= 0x8F9721 AND date < 0x8F9741
date.startswith(0x8F97)
```

Hardware filtering

Query 3: TPC-H Q14.

```
SELECT 100.00 * SUM(CASE WHEN p_type LIKE 'PROMO%'
  THEN l_extendedprice * (1-l_discount) ELSE 0 END) /
  SUM(l_extendedprice * (1-l_discount)) AS promo_revenue
FROM lineitem, part
WHERE l_partkey = p_partkey
  AND l_shipdate >= date '1995-09-01'
  AND l_shipdate < date '1995-09-01' + INTERVAL '1' MONTH;
```

```
date >= '1995-09-01' AND date < '1995-10-01'
date >= 0x8F9721 AND date < 0x8F9741
date.startswith(0x8F97)
```

Hardware filtering

Query 3: TPC-H Q14.

```
SELECT 100.00 * SUM(CASE WHEN p_type LIKE 'PROMO%'
  THEN l_extendedprice * (1-l_discount) ELSE 0 END) /
  SUM(l_extendedprice * (1-l_discount)) AS promo_revenue
FROM lineitem, part
WHERE l_partkey = p_partkey
  AND l_shipdate >= date '1995-09-01'
  AND l_shipdate < date '1995-09-01' + INTERVAL '1' MONTH;
```

```
date >= '1995-09-01' AND date < '1995-10-01'
date >= 0x8F9721 AND date < 0x8F9741
date.startswith(0x8F97)
```

Hardware filtering inherent limitations

$$\begin{aligned} \mathcal{X} &= \text{lots of dates between 1990 and 2020} \\ M_{2016,12} &= \text{dates in 2016 December} \\ M_{2017,1} &= \text{dates in 2017 January} \end{aligned}$$

$$\left[\mathcal{X} \cap M_{2016,12}\right] \cup \left[\mathcal{X} \cap M_{2017,1}\right] \subseteq \left[\mathcal{X} \cap Y_{??}\right]$$

Hardware filtering

$$\left[\mathcal{X} \cap M_{2016,9}\right] \cup \left[\mathcal{X} \cap M_{2016,10}\right] \cup \left[\mathcal{X} \cap M_{2016,11}\right] \subseteq \left[\mathcal{X} \cap Y_{2016}\right]$$
$$\left[\mathcal{X} \cap M_{2016,12}\right] \cup \left[\mathcal{X} \cap M_{2017,1}\right] \subseteq \left[\mathcal{X} \cap Y_{??}\right]$$

False positives? Possibly.
False negatives? No.

Filtering Condition Pushdown (FCP)

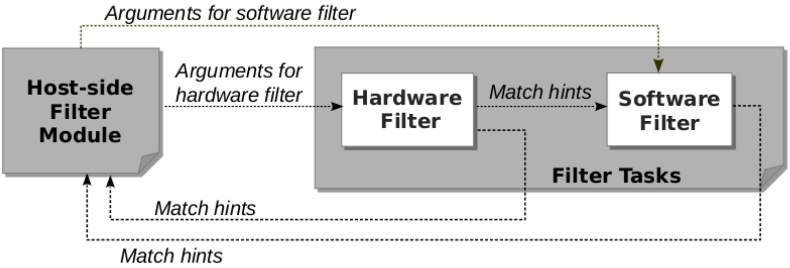


Figure 4: ISC filters.

Software filtering

- Only activated in some cases (depending on filter query)
- Removes (un-matches) pages that are 100% false positives

Code 1: Pseudo code of the software filter.

```
//IN: Match_Hint
//ARG: SW_KEY, OP_TYPE, BULK_MATCH_SIZE
//OUT: Match_Hint

DO Get Match_Hint from the hardware filter
FOR i = 0 TO i < BULK_MATCH_SIZE DO {
  IF Match_Hint[i] != 0 THEN {
    Do Match_Hint[i] to 0
    FOR j = 0 TO j < # of rows in Page i DO {
      FOR each filter column in row j DO {
        IF 'the column value OP_TYPE SW_KEY' is true THEN
          DO Set Match_Hint[i] to 1
          DO Break
        END IF
      }
      IF Match_Hint[i] == 1 THEN
        DO Break
      END IF
}}}

DO Put Match_Hint to output
```

YourSQL architecture

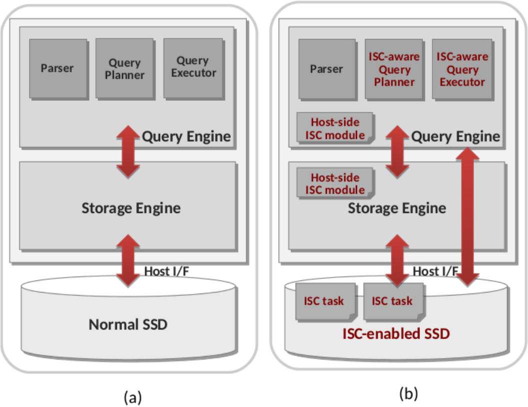


Figure 1: Two database system architectures. (a) Traditional system. (b) YourSQL.

ISC-enabled queries

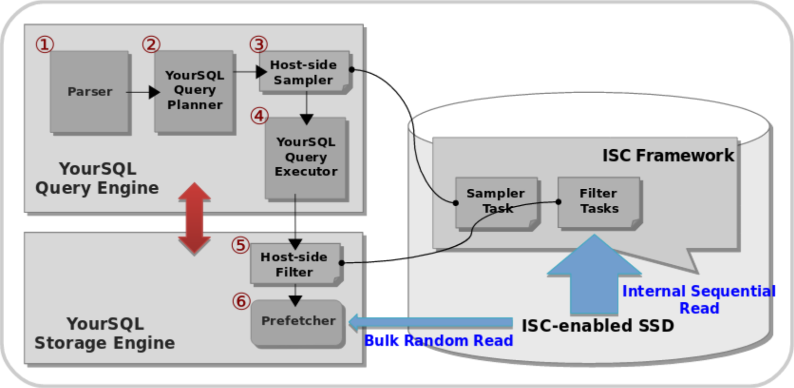


Figure 2: Early filtering of YourSQL.

Prefetcher

- Bulk read from match hints
- Choosing the size of `bulk_match_size` (number of match-hinted pages) is important
  - SSD can concurrently process requests up to a certain `bulk_match_size`

Complex queries

Query 1: A simple selection query.

```
SELECT p_partkey, p_mfgr FROM part
WHERE p_size = 15 AND p_type LIKE '%BRASS';
```

Query 2: TPC-H Q2.

```
SELECT s_acctbal, s_name, n_name, p_partkey, p_mfgr,
       s_address, s_phone, s_comment
FROM part, supplier, partsupp, nation, region
WHERE p_partkey = ps_partkey
      AND s_suppkey = ps_suppkey
      AND p_size = 15 AND p_type LIKE '%BRASS'
      AND s_nationkey = n_nationkey
      AND n_regionkey = r_regionkey
      AND r_name = 'EUROPE'
      AND ps_supplycost = (SELECT MIN(ps_supplycost)
            FROM partsupp, supplier, nation, region
            WHERE p_partkey = ps_partkey
              AND s_suppkey = ps_suppkey
              AND s_nationkey = n_nationkey
              AND n_regionkey = r_regionkey
              AND r_name = 'EUROPE')
ORDER BY s_acctbal DESC, n_name, s_name, p_partkey LIMIT 100;
```

Goal

Put the smallest table first in JOIN order

- Nested-loop algorithm
- Why smallest first?

Query 2: TPC-H Q2.

```
SELECT s_acctbal, s_name, n_name, p_partkey, p_mfgr,
       s_address, s_phone, s_comment
FROM part, supplier, partsupp, nation, region
WHERE p_partkey = ps_partkey
      AND s_suppkey = ps_suppkey
      AND p_size = 15 AND p_type LIKE '%BRASS'
      AND s_nationkey = n_nationkey
      AND n_regionkey = r_regionkey
      AND r_name = 'EUROPE'
      AND ps_supplycost = (SELECT MIN(ps_supplycost)
           FROM partsupp, supplier, nation, region
           WHERE p_partkey = ps_partkey
             AND s_suppkey = ps_suppkey
             AND s_nationkey = n_nationkey
             AND n_regionkey = r_regionkey
             AND r_name = 'EUROPE')
ORDER BY s_acctbal DESC, n_name, s_name, p_partkey LIMIT 100;
```

Estimating I/O reduction per table

Nominate by heuristic → accept/reject by estimate

- Limiting score (heuristic)
  - Threshold set presumably through experimentation
- Estimated filter ratio
  - Much more complex
  - **is** quantitatively correlated with actual filtering ratio
  - Requires FCP

Estimating I/O reduction per table

Nominate by heuristic → accept/reject by estimate

- Limiting score (heuristic)
  - Threshold set presumably through experimentation

- Estimated filter ratio
  - Much more complex
  - **is** quantitatively correlated with actual filtering ratio
  - Requires FCP

Estimating I/O reduction per table

Nominate by heuristic → accept/reject by estimate

- Limiting score (heuristic)
  - Threshold set presumably through experimentation

- Estimated filter ratio
  - Much more complex
  - **is** quantitatively correlated with actual filtering ratio
  - Requires FCP

Best-guessing smallest table

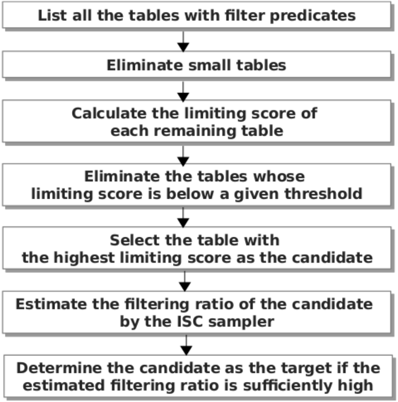


Figure 3: Selection of the early filtering target table.

ISC Sampler

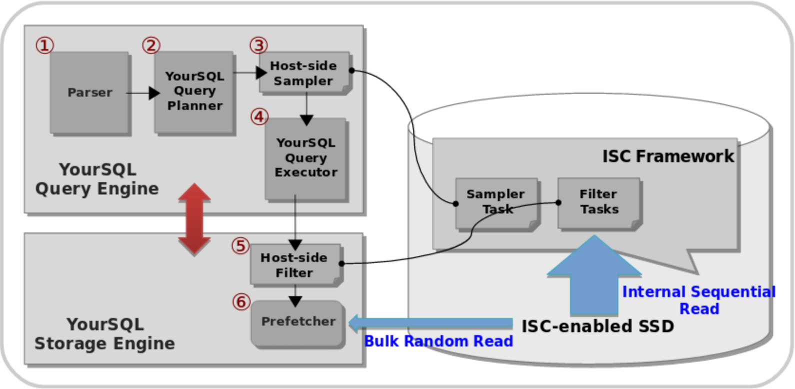


Figure 2: Early filtering of YourSQL.

Bandwidth comparisons

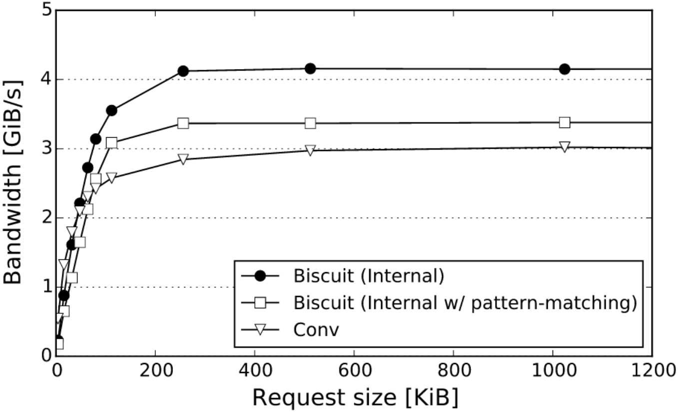


Figure: From Gu, et al.[2]

Speed of queries

Table 4: Execution time for Query 1.

	MySQL	YourSQL
Execution time [sec]	15.9	2.2

- 7× speed up
- 6.7% of pages needed to answer query

Table 5: Execution time for TPC-H Q2.

	MySQL	YourSQL
Execution time [sec]	1,104	25

- 44× speed up

Query 1: A simple selection query.

```
SELECT p_partkey, p_mfgr FROM part
WHERE p_size = 15 AND p_type LIKE '%BRASS';
```

Query 2: TPC-H Q2.

```
SELECT s_acctbal, s_name, n_name, p_partkey, p_mfgr,
       s_address, s_phone, s_comment
FROM part, supplier, partsupp, nation, region
WHERE p_partkey = ps_partkey
      AND s_suppkey = ps_suppkey
      AND p_size = 15 AND p_type LIKE '%BRASS'
      AND s_nationkey = n_nationkey
      AND n_regionkey = r_regionkey
      AND r_name = 'EUROPE'
      AND ps_supplycost = (SELECT MIN(ps_supplycost)
          FROM partsupp, supplier, nation, region
          WHERE p_partkey = ps_partkey
            AND s_suppkey = ps_suppkey
            AND s_nationkey = n_nationkey
            AND n_regionkey = r_regionkey
            AND r_name = 'EUROPE')
ORDER BY s_acctbal DESC, n_name, s_name, p_partkey LIMIT 100;
```

Power consumption

Table 5: Execution time for TPC-H Q2.

	MySQL	YourSQL
Execution time [sec]	1,104	25

- 44× speed up

Table 7: Overall energy consumption.

	MySQL	YourSQL
Total energy (kJ)	131.0	5.3

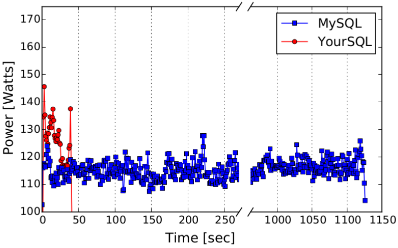


Figure 10: System power consumption during the execution of TPC-H Q2.

Effect of system memory size

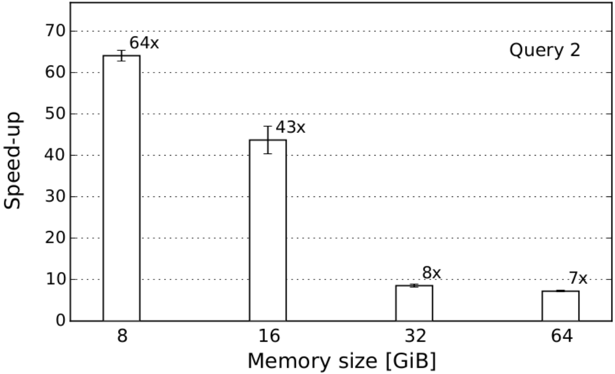


Figure 7: Speed-up with different memory size.

TCP-H queries

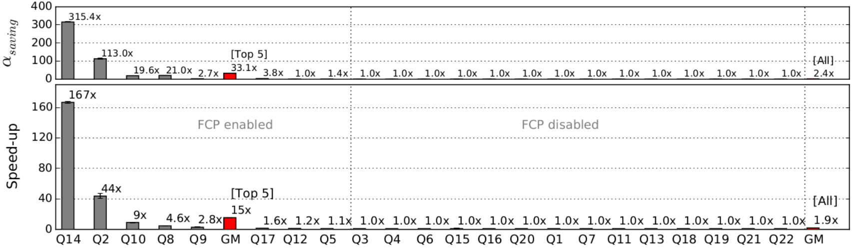


Figure 8: TPC-H results.

Impact of optimization

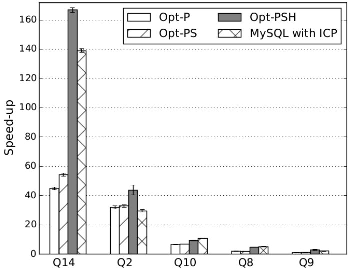


Figure 9: Speed-ups of the top five accelerated queries with different optimization schemes.

Table 6: Different levels of optimization.

Scheme	Configuration
Opt-P	Hardware filter
Opt-PS	Hardware filter + Software filter
Opt-PSH	Hardware filter + Software filter + HABP

HABP == prefetching

Column store

*Lower (*i.e.*, shorter bars) is better

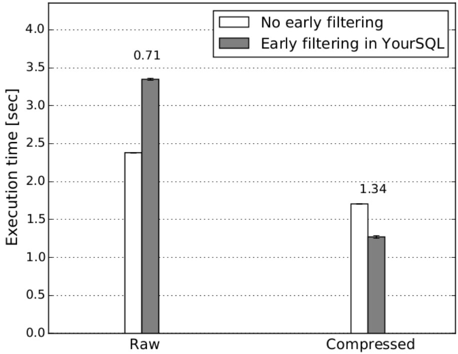


Figure 11: Execution time for the column stores.

Duck-Ho Bae et al. "Intelligent SSD: a turbo for big data mining". In: *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. CIKM '13. San Francisco, California, USA: ACM, 2013, pp. 1573–1576. ISBN: 978-1-4503-2263-8. DOI: 10.1145/2505515.2507847. URL: http://doi.acm.org/10.1145/2505515.2507847.

Boncheol Gu et al. "Biscuit: A Framework for Near-data Processing of Big Data Workloads". In: *SIGARCH Comput. Archit. News* 44.3 (June 2016), pp. 153–165. ISSN: 0163-5964. DOI: 10.1145/3007787.3001154. URL: http://doi.acm.org/10.1145/3007787.3001154.