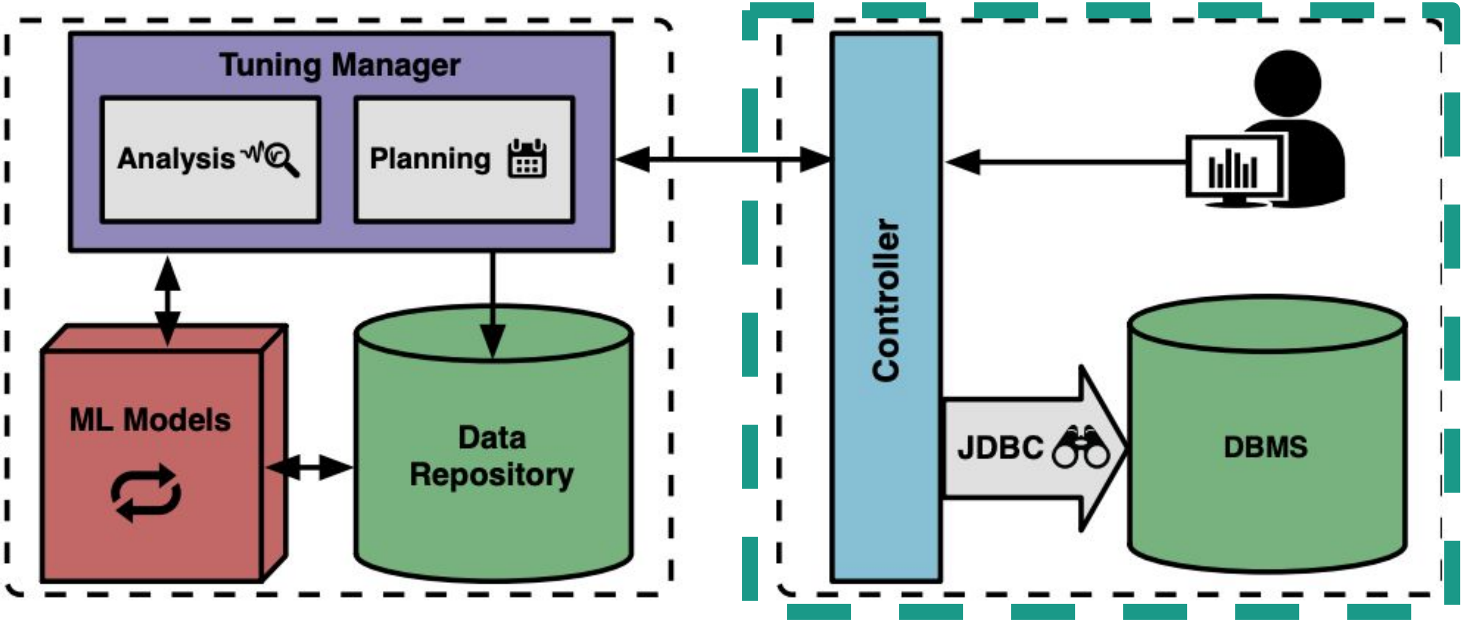
OLTP workload mapping for database tuning

CS310 - Valentin Kodderitzsch

Big picture

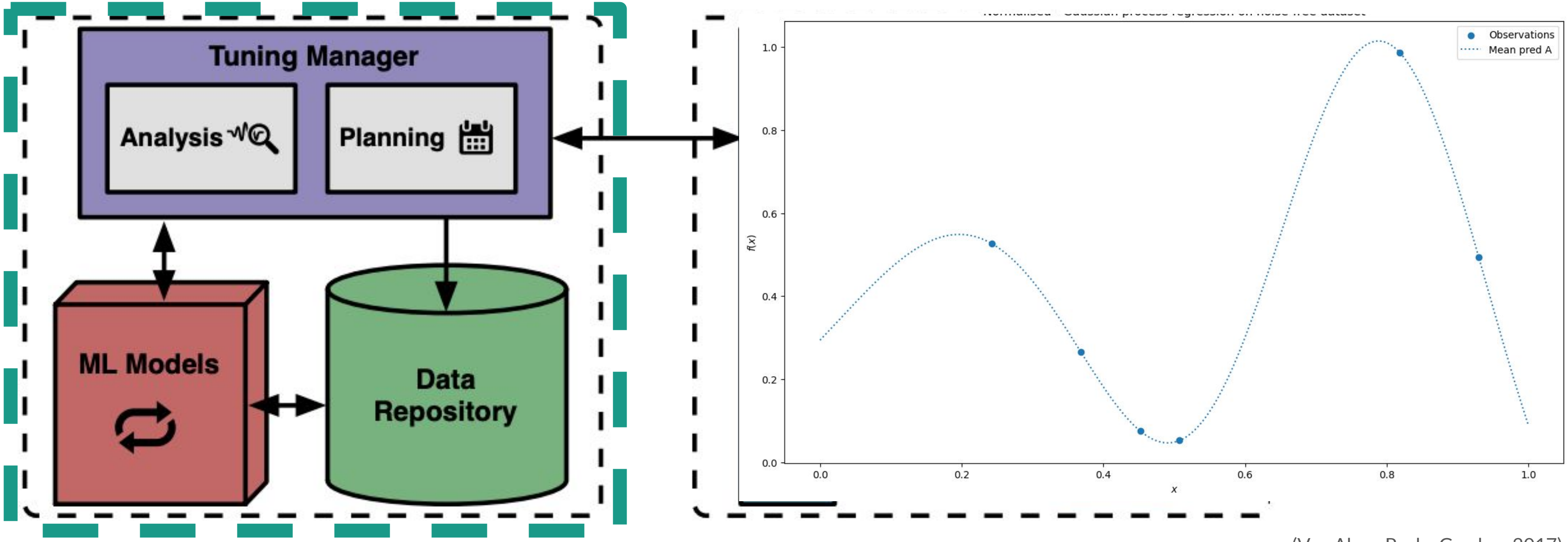
1. Improve database performance
2. Changing database settings
3. Reduce tuning time

Tuning pipeline



(Van Aken, Pavlo, Gordon; 2017)

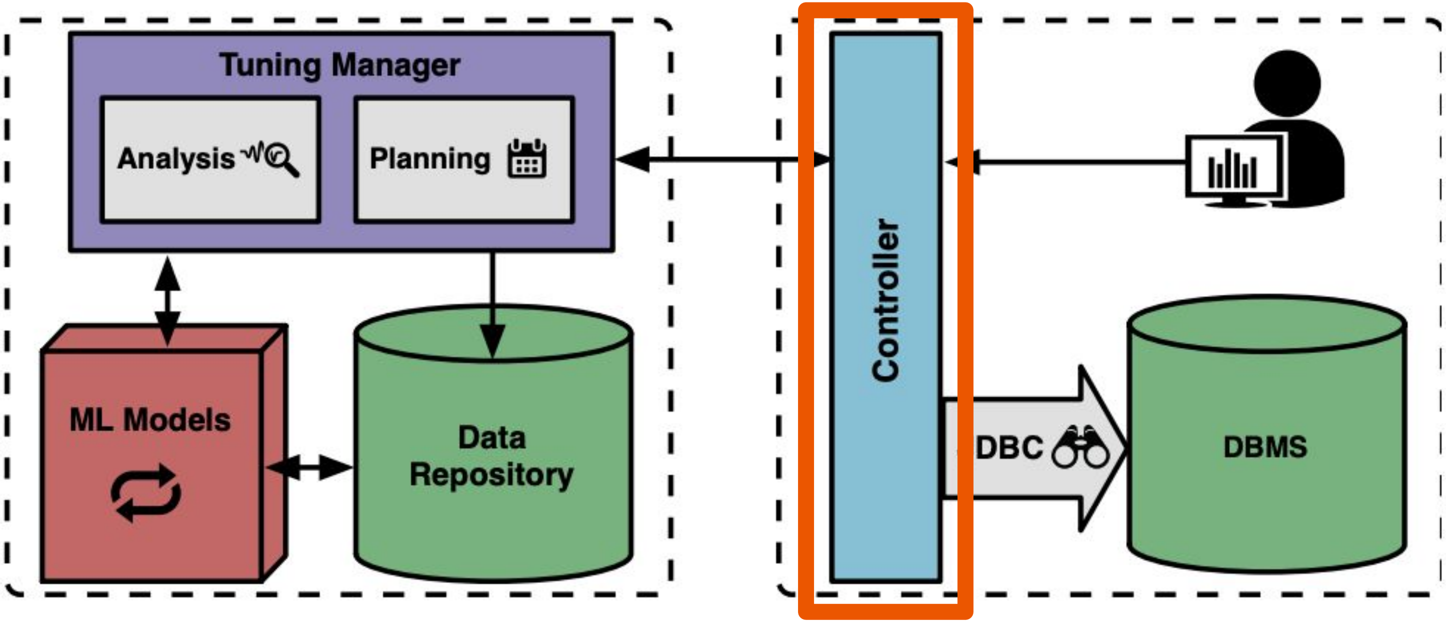
Tuning pipeline



(Van Aken, Pavlo, Gordon; 2017)

Tuning pipeline

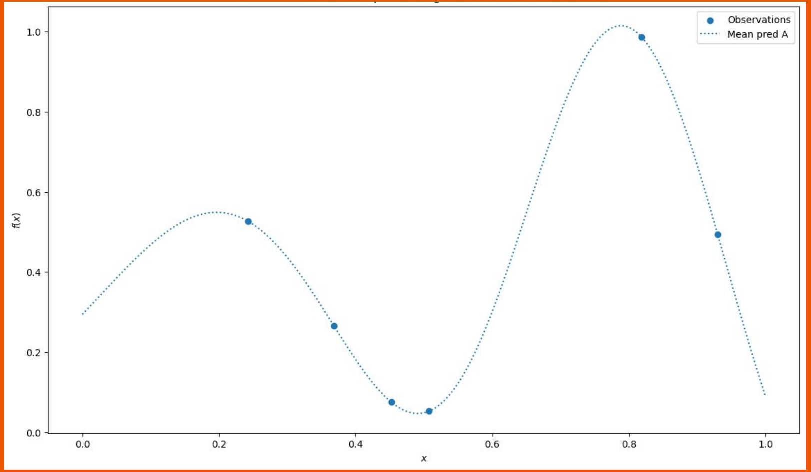
Sampling bottleneck

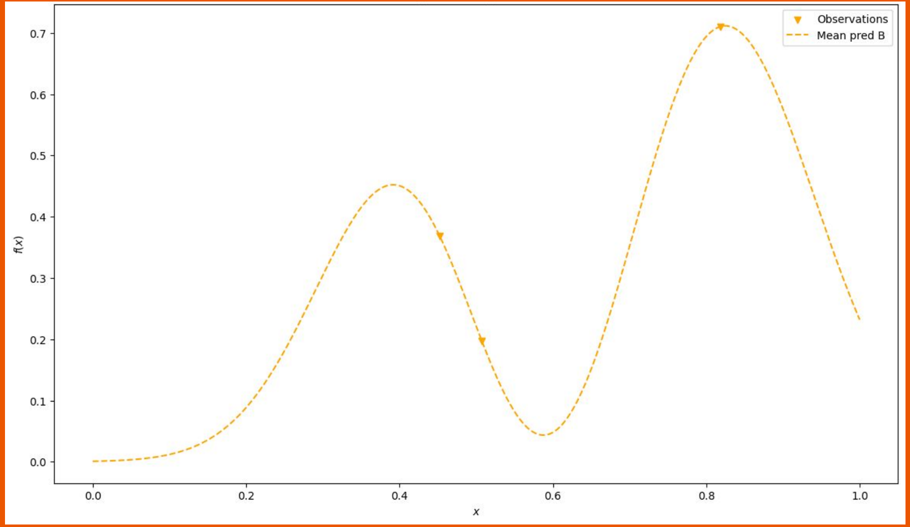


9 hours

(Van Aken, Pavlo, Gordon; 2017)

What if the workload changes?





9 hours

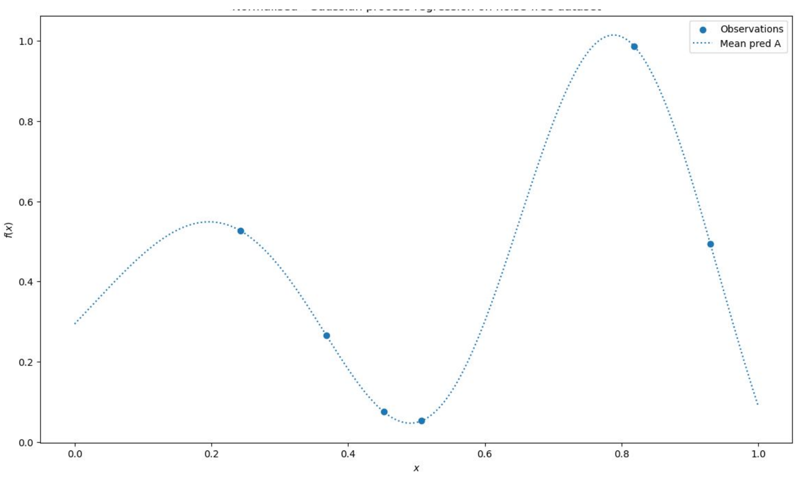
9 hours again??

Can we reduce the sampling bottleneck by mapping between workloads?

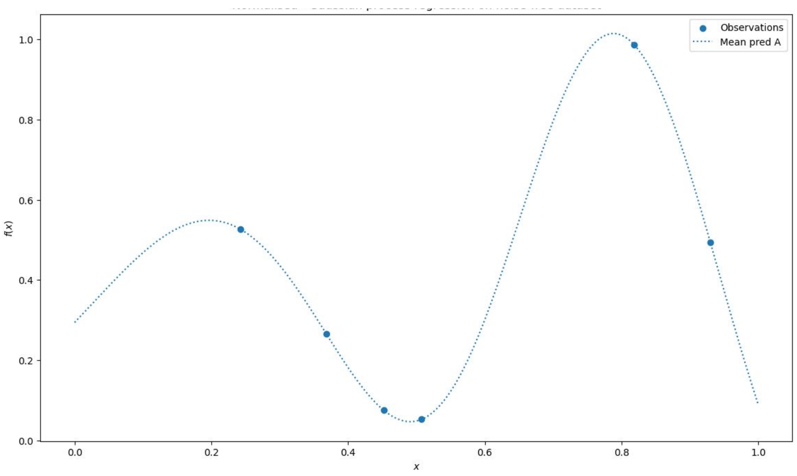
25% faster

In only ⅓ of the tuning time

Key intuition



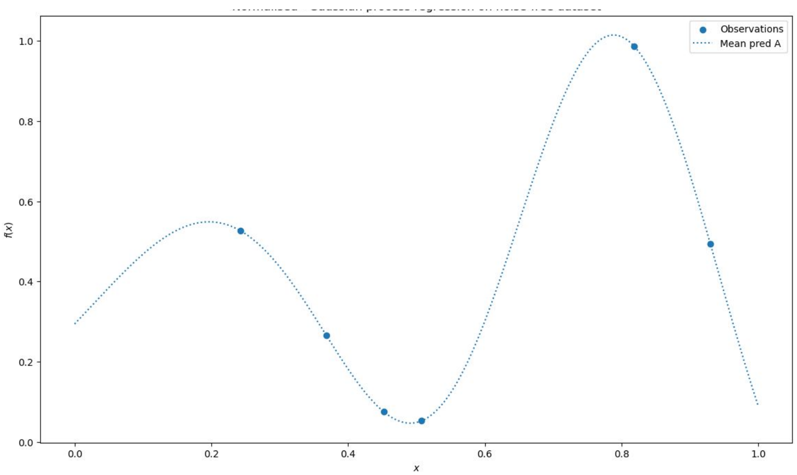
Old



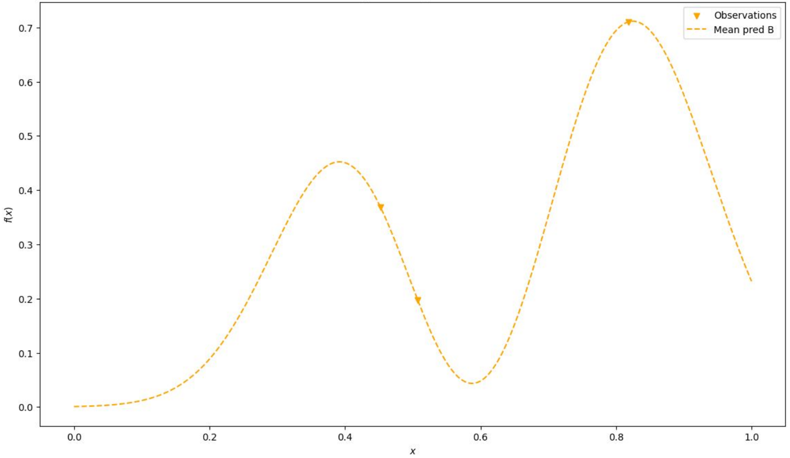
New

Reuse samples

Key intuition



Old



New

Forget old samples

New sampling function

$$n_{new} = f(n_{old}, \theta) = \theta * n_{old}$$

$0 \leq \theta \leq 1$

$n_{old} = 100$

Theta before sampling??

Jensen-Shannon divergence

Relative entropy between two probability distributions

Symmetric version of the Kullback–Leibler divergence (KLD or DKL)

Base 2 gives a score between 0 and 1

0 → Identical

1 → Completely different

Jensen-Shannon divergence

$$\mathrm{JSD}(P \parallel Q) = \frac{1}{2} D(P \parallel M) + \frac{1}{2} D(Q \parallel M),$$

Score between 0 and 1

→ Sampling function

→ Ensemble method

Prob. distributions P and Q

P and Q ??

$$M = \frac{1}{2}(P + Q)$$

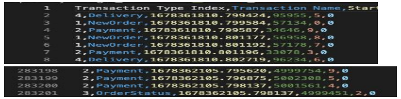
$$D_{\mathrm{KL}}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x) \log\left(\frac{P(x)}{Q(x)}\right).$$

https://en.wikipedia.org/wiki/Jensen%E2%80%93Shannon_divergence

Key assumption

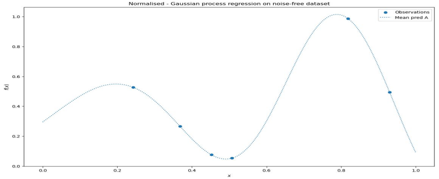
$$JSD(A_{default}||B_{default}) \approx JSD(A_{all\_samples}||B_{all\_samples})$$

A: PDF of Default logs



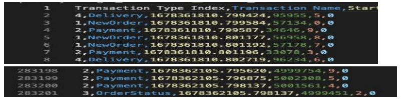
```
Transaction Type Index,Transaction Name,Start
4,Delivery,1678361810.799424,95955,5,0
1,NewOrder,1678361810.799584,57134,0,0
2,Payment,1678361810.799587,34646,9,0
1,NewOrder,1678361810.801177,56958,8,0
1,NewOrder,1678361810.801192,57178,7,0
2,Payment,1678361810.801196,33078,3,0
4,Delivery,1678361810.802719,96234,6,0
```

```
2,Payment,1678362105.795620,4999754,9,0
2,Payment,1678362105.796875,5002308,5,0
2,Payment,1678362105.798137,5001561,4,0
3,OrderStatus,1678362105.798137,4999451,2,0
```



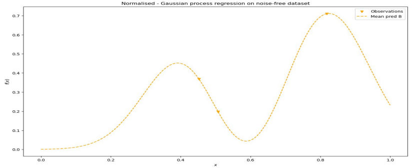
A: PDF all samples

B: PDF of Default logs



```
Transaction Type Index,Transaction Name,Start
4,Delivery,1678361810.799424,95955,5,0
1,NewOrder,1678361810.799584,57134,0,0
2,Payment,1678361810.799587,34646,9,0
1,NewOrder,1678361810.801177,56958,8,0
1,NewOrder,1678361810.801192,57178,7,0
2,Payment,1678361810.801196,33078,3,0
4,Delivery,1678361810.802719,96234,6,0
```

```
2,Payment,1678362105.795620,4999754,9,0
2,Payment,1678362105.796875,5002308,5,0
2,Payment,1678362105.798137,5001561,4,0
3,OrderStatus,1678362105.798137,4999451,2,0
```



B: PDF all samples

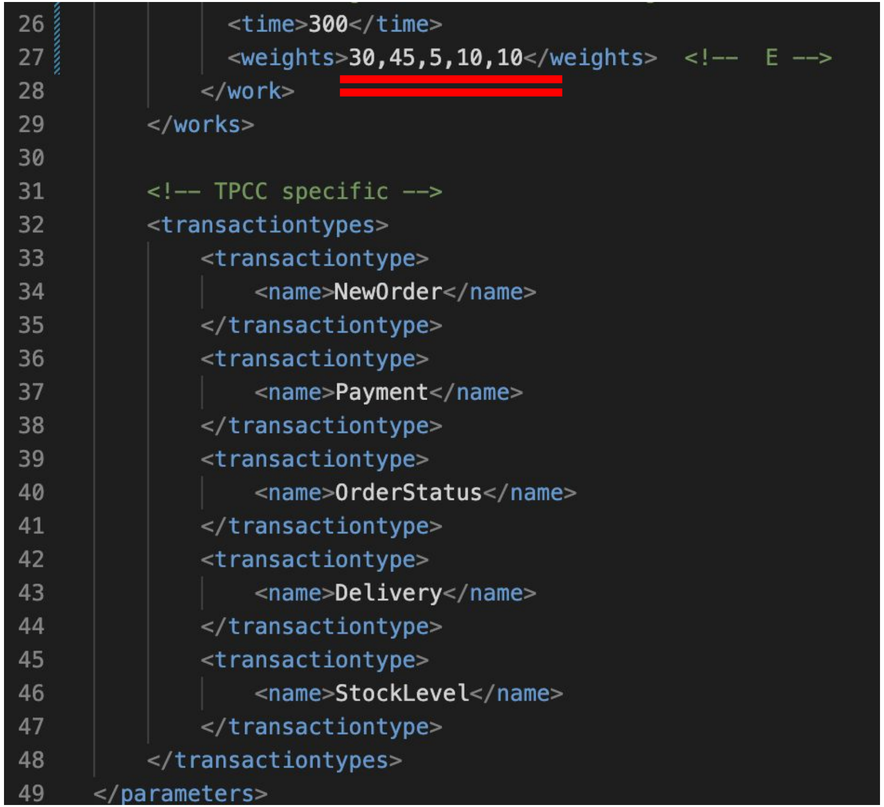
Additional statistics

Workload

Synthetic

ID = (45, 45, 3, 3, 4) or (20, 20, 20, 39, 1), etc.

⇒ 4.6 million

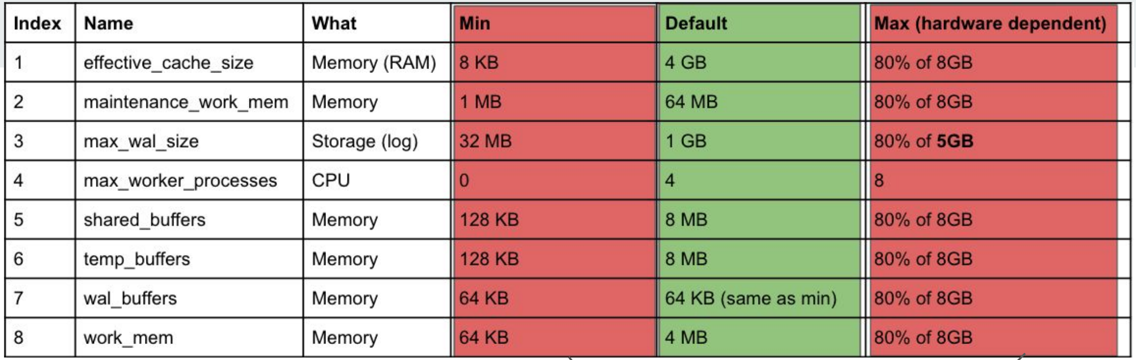


```
          <time>300</time>
          <weights>30,45,5,10,10</weights>  <!--  E -->
        </work>
    </works>

    <!-- TPCC specific -->
    <transactiontypes>
        <transactiontype>
            <name>NewOrder</name>
        </transactiontype>
        <transactiontype>
            <name>Payment</name>
        </transactiontype>
        <transactiontype>
            <name>OrderStatus</name>
        </transactiontype>
        <transactiontype>
            <name>Delivery</name>
        </transactiontype>
        <transactiontype>
            <name>StockLevel</name>
        </transactiontype>
    </transactiontypes>
</parameters>
```

(Difallah, Pavlo, Curino; 2013)

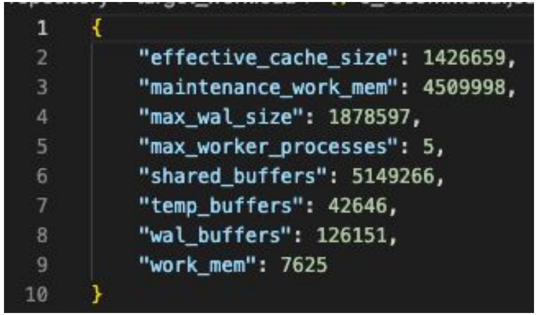
DB settings



Index	Name	What	Min	Default	Max (hardware dependent)
1	effective_cache_size	Memory (RAM)	8 KB	4 GB	80% of 8GB
2	maintenance_work_mem	Memory	1 MB	64 MB	80% of 8GB
3	max_wal_size	Storage (log)	32 MB	1 GB	80% of **5GB**
4	max_worker_processes	CPU	0	4	8
5	shared_buffers	Memory	128 KB	8 MB	80% of 8GB
6	temp_buffers	Memory	128 KB	8 MB	80% of 8GB
7	wal_buffers	Memory	64 KB	64 KB (same as min)	80% of 8GB
8	work_mem	Memory	64 KB	4 MB	80% of 8GB

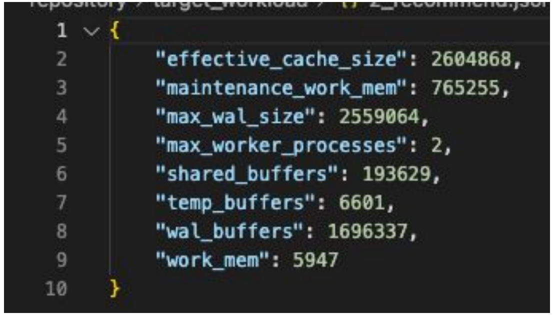
Domain for sampling algorithm

Algorithm returns n features



```
{
    "effective_cache_size": 1426659,
    "maintenance_work_mem": 4509998,
    "max_wal_size": 1878597,
    "max_worker_processes": 5,
    "shared_buffers": 5149266,
    "temp_buffers": 42646,
    "wal_buffers": 126151,
    "work_mem": 7625
}
```

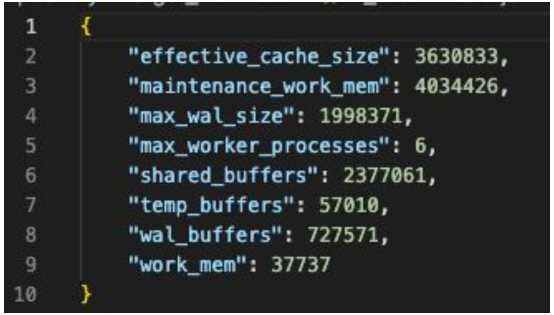
Feature 1



```
{
    "effective_cache_size": 2604868,
    "maintenance_work_mem": 765255,
    "max_wal_size": 2559064,
    "max_worker_processes": 2,
    "shared_buffers": 193629,
    "temp_buffers": 6601,
    "wal_buffers": 1696337,
    "work_mem": 5947
}
```

Feature 2

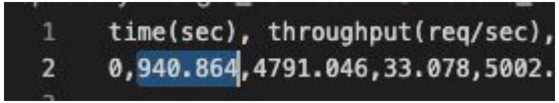
...



```
{
    "effective_cache_size": 3630833,
    "maintenance_work_mem": 4034426,
    "max_wal_size": 1998371,
    "max_worker_processes": 6,
    "shared_buffers": 2377061,
    "temp_buffers": 57010,
    "wal_buffers": 727571,
    "work_mem": 37737
}
```

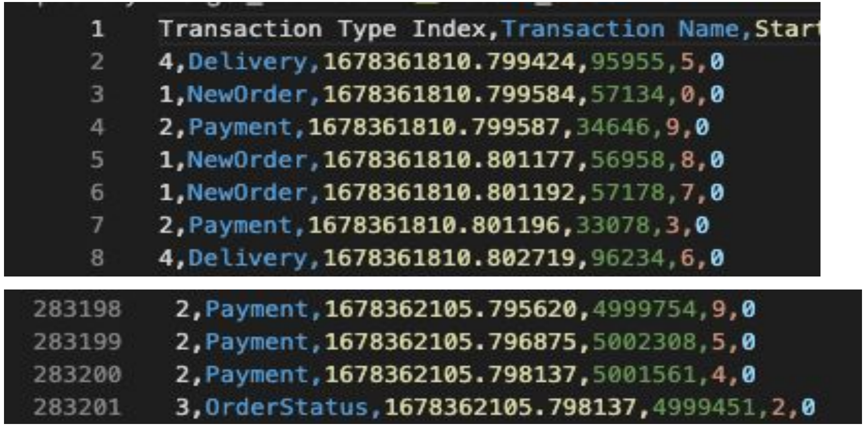
Feature n

Sampling



```
time(sec), throughput(req/sec),
0,940.864,4791.046,33.078,5002.
```

Average throughput (5min)

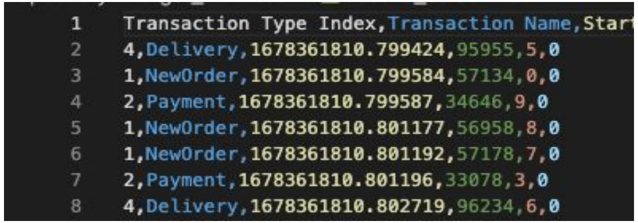


```
Transaction Type Index,Transaction Name,Star
4,Delivery,1678361810.799424,95955,5,0
1,NewOrder,1678361810.799584,57134,0,0
2,Payment,1678361810.799587,34646,9,0
1,NewOrder,1678361810.801177,56958,8,0
1,NewOrder,1678361810.801192,57178,7,0
2,Payment,1678361810.801196,33078,3,0
4,Delivery,1678361810.802719,96234,6,0
```

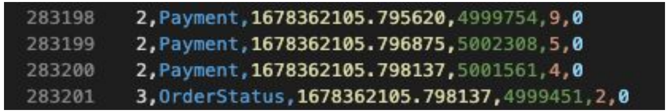
```
2,Payment,1678362105.795620,4999754,9,0
2,Payment,1678362105.796875,5002308,5,0
2,Payment,1678362105.798137,5001561,4,0
3,OrderStatus,1678362105.798137,4999451,2,0
```

Log of all transactions and their latency (5min)

Pre-processing

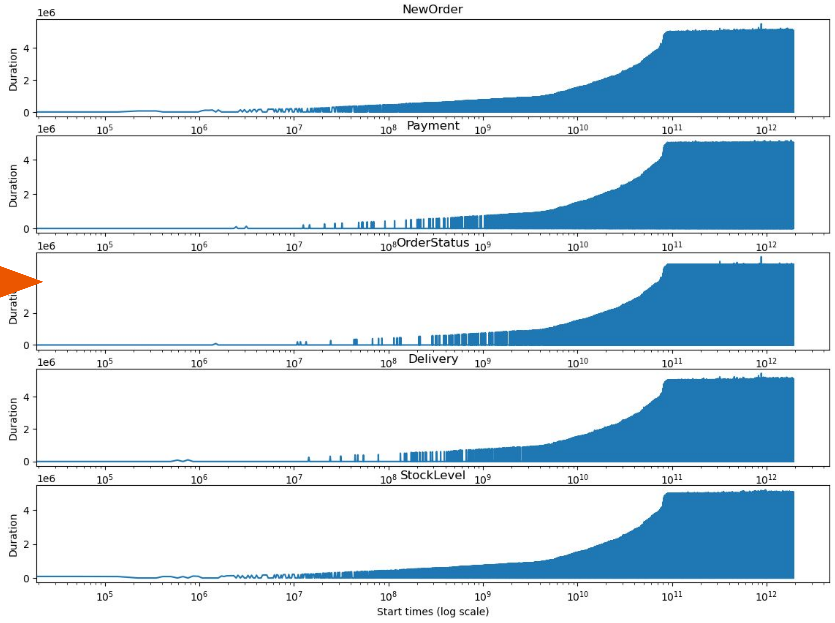


```
Transaction Type Index,Transaction Name,Start
4,Delivery,1678361810.799424,95955,5,0
1,NewOrder,1678361810.799584,57134,0,0
2,Payment,1678361810.799587,34646,9,0
1,NewOrder,1678361810.801177,56958,8,0
1,NewOrder,1678361810.801192,57178,7,0
2,Payment,1678361810.801196,33078,3,0
4,Delivery,1678361810.802719,96234,6,0
```



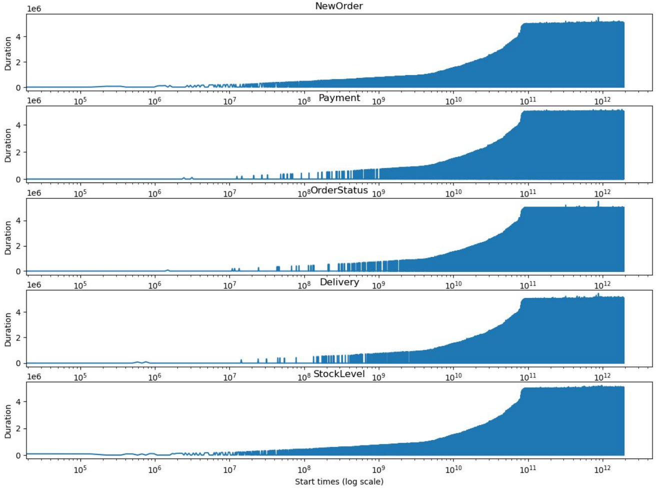
```
2,Payment,1678362105.795620,4999754,9,0
2,Payment,1678362105.796875,5002308,5,0
2,Payment,1678362105.798137,5001561,4,0
3,OrderStatus,1678362105.798137,4999451,2,0
```

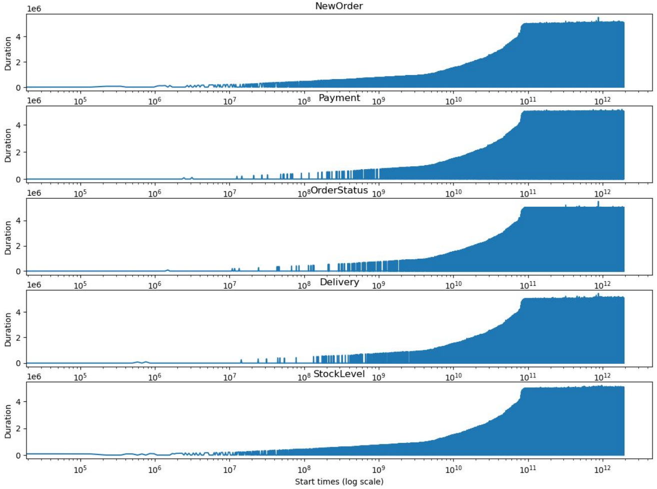
Log of all transactions and their latency (5min)



Pre-processing

$$JSD(A_{default}||B_{default})$$





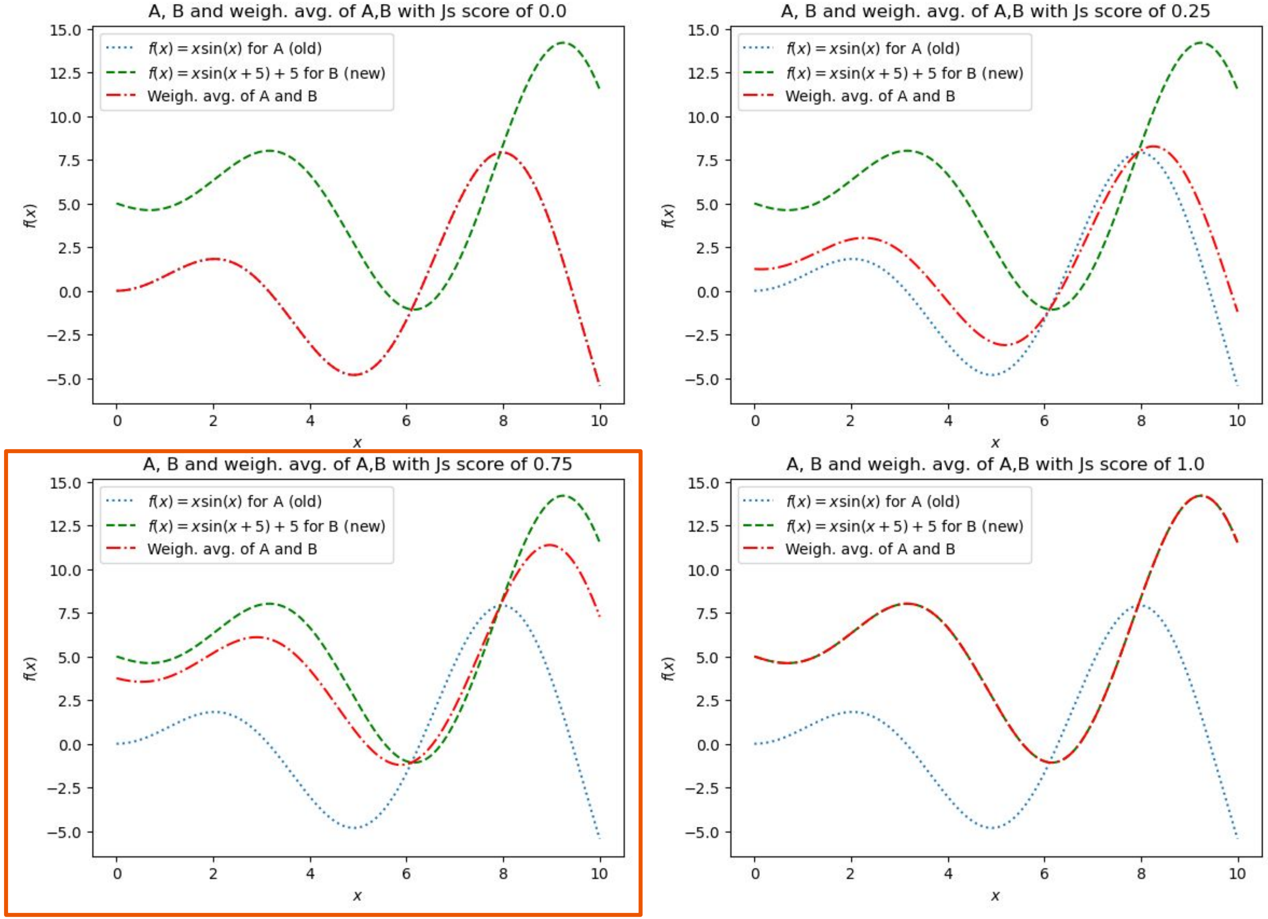
5 JSD scores

New ensemble model

$$n_{new} = f(n_{old}, \theta) = \theta * n_{old}$$

$$\hat{y} = \begin{cases} M_1(x_{old}) & ; \theta = 0 \\ M_2(x_{new}) & ; \theta = 1 \\ M_1(x_{old})(1 - \theta) + M_2(x_{new})\theta & ; 0 \leq \theta \leq 1 \end{cases}$$

New ensemble model



Results

Default value

A

1811 request/sec

B

1522 request/sec

JSD = 0.33

Sampling

A

100 samples

9 hours

B

33 samples

3 hours

JSD = 0.33

Ensemble prediction for B

Default: 1522 req/sec

Best config: 1915 req/sec

JSD = 0.33

25% faster

In only ⅓ of the tuning time

Evaluation

Contributions

Better than guessing

Faster than default

Reduced tuning bottleneck!

Limitations

Inconsistent results

Not enough data for hypothesis test yet

Proof of concept but data too inconsistent

Project management

Well conceived project

- All necessary research, analysis and design work completed
- Agile development approach

Could have received code base earlier from (Barbulescu, Triantafillou; 2022)

- Progress no always consistent with specification report

Unforeseen problems well detected and overcome

- DB Caching issues
- Legacy version of DB - CLI problems

Q & A