

DataStax

Hunter

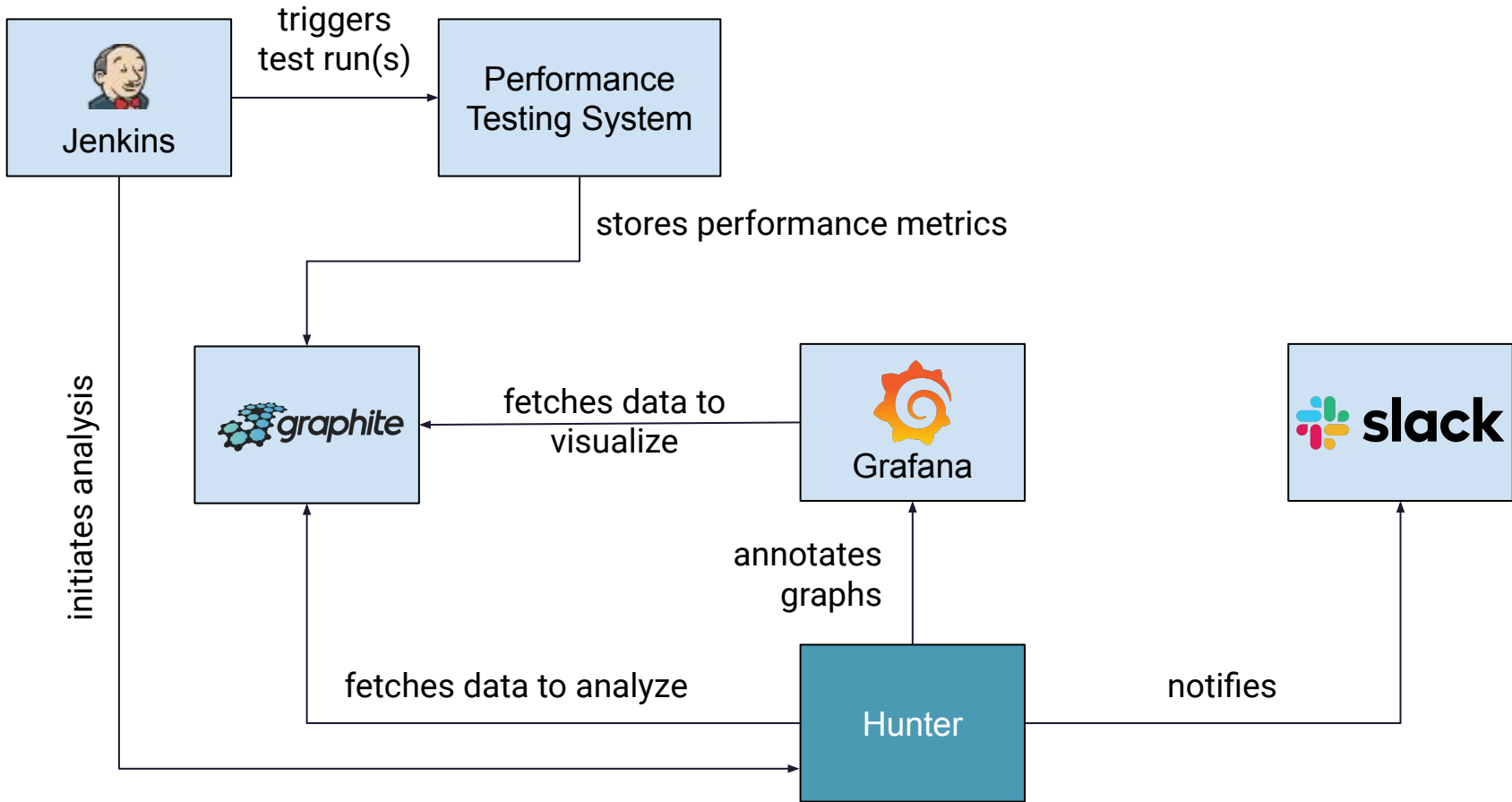
Using Change Point Detection to Hunt for Performance Regressions

Matt Fleming, **Piotr Kołaczkowski**, **Ishita Kumar**, Shaunak Das, Sean McCarthy,
Pushkala Pattabhiraman and Henrik Ingo

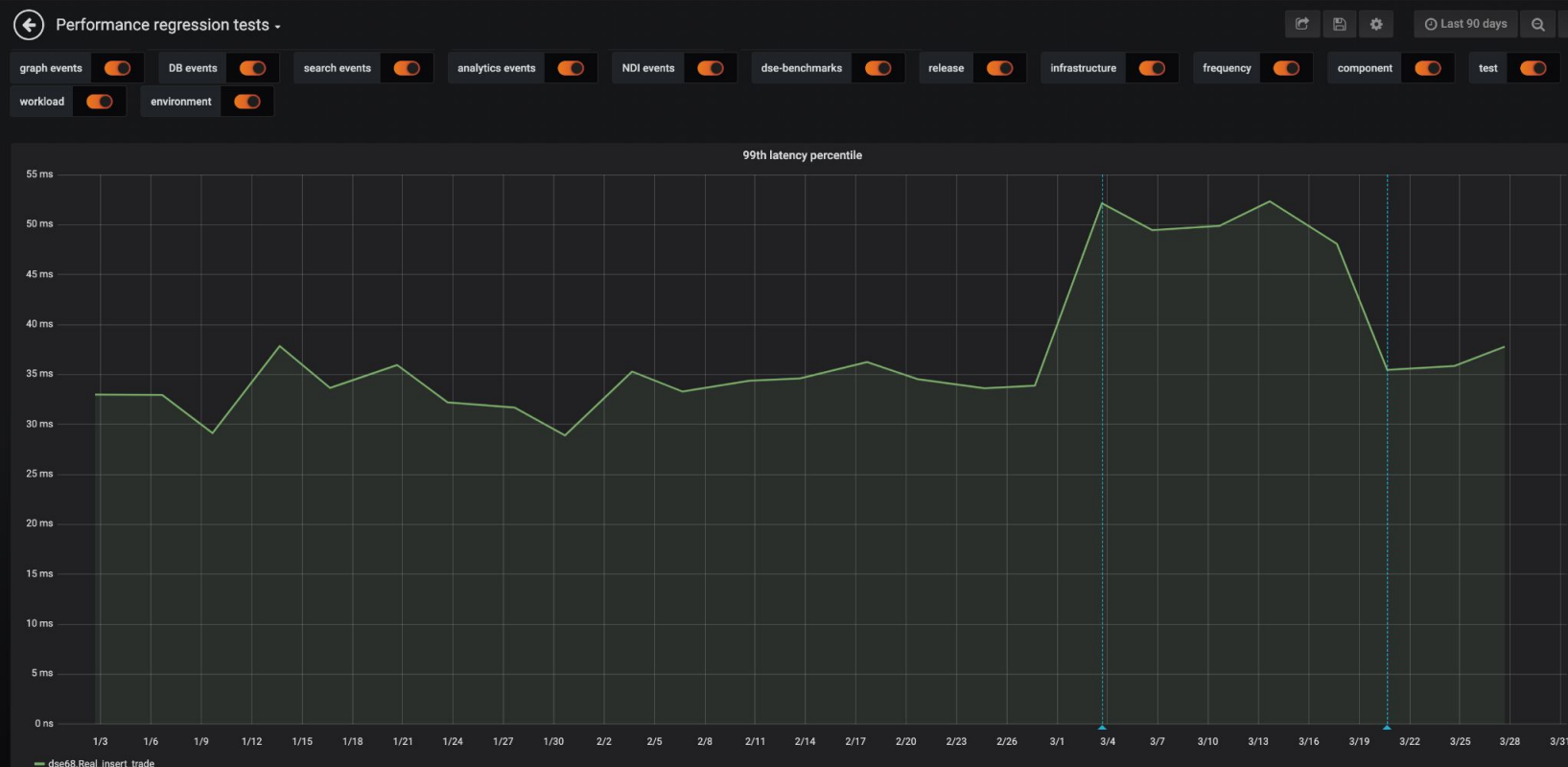
The Problems Hunter Solves

- Has performance of a software product changed?
 - If so, which specific commit caused it?
 - How big is the change?
-
- Can we make a new release?
 - Can we merge the topic branch?

Hunter does that by analyzing performance test results we collect in our history server (currently graphite).



How Perf Regression Looks Like



Automatic Regression Detection on Many Tests

```
$ hunter regressions dse-68-smoke --since '6 weeks'  
INFO: Fetching data from Graphite...  
INFO: Computing change points for test dse68.read.inmem...  
dse68.read.inmem:  
    p90          : 1.67e+07 --> 1.94e+07 ( +16.2%)  
    p95          : 1.92e+07 --> 2.09e+07 (  +8.9%)  
    p99          : 3.17e+07 --> 3.94e+07 ( +24.1%)  
INFO: Fetching data from Graphite...  
INFO: Computing change points for test dse68.write.rf1...  
dse68.write.rf1: OK  
INFO: Fetching data from Graphite...  
INFO: Computing change points for test dse68.write.rf3...  
dse68.write.rf3: OK
```

Analyzing Runs of a Single Test

```
$ hunter analyze dse68.read.inmem --since '6 weeks'
```

```
INFO: Fetching data from Graphite...
```


```
INFO: Computing change points for test dse68.read.inmem...
```

```
dse68.read.inmem:
```

time	run	branch	version	commit	throughput	p50	p95	p99	p999
2021-06-21 00:00:00 +0000	0f9feaf0	6.8-dev	6.8.14	994d6b13d27966e295e455fa69fabd9f543348d7	729620	1.1854e+07	1.9431e+07	3.1982e+07	2.9465e+08
2021-06-23 00:00:00 +0000	4708838a	6.8-dev	6.8.14	f6ac4b3df6d6886980dbb565684804cbf5ecfaea	735225	1.1747e+07	1.9382e+07	3.1769e+07	3.00941e+08
2021-06-25 00:00:00 +0000	17cc6af0	6.8-dev	6.8.14	f6ac4b3df6d6886980dbb565684804cbf5ecfaea	718087	1.2157e+07	1.9612e+07	3.1441e+07	2.96747e+08
2021-06-28 00:00:00 +0000	f90509d8	6.8-dev	6.8.14	f6ac4b3df6d6886980dbb565684804cbf5ecfaea	734331	1.1788e+07	1.9186e+07	3.2637e+07	2.95698e+08
2021-07-02 00:00:00 +0000	80ba41c7	6.8-dev	6.8.15	d26bb81b11d7d574769d9244794b7b916131fa7b	712336	1.2182e+07	1.9841e+07	3.3358e+07	2.98058e+08
2021-07-05 00:00:00 +0000	71be7e61	6.8-dev	6.8.15	d26bb81b11d7d574769d9244794b7b916131fa7b	741089	1.1723e+07	1.912e+07	3.2375e+07	2.98058e+08
2021-07-07 00:00:00 +0000	05052811	6.8-dev	6.8.15	db4fe601dd487d8ca984e9b4d7a00d34bbd03c1	747922	1.151e+07	1.8727e+07	3.1506e+07	2.99893e+08
2021-07-09 00:00:00 +0000	64acc6e3	6.8-dev	6.8.15	bf99ebe8693172397afe48024e2de5ce67de1ceb	733456	1.187e+07	1.9202e+07	3.2014e+07	2.98582e+08
2021-07-12 00:00:00 +0000	5a2e5a31	6.8-dev	6.8.15	bf99ebe8693172397afe48024e2de5ce67de1ceb	711107	1.2198e+07	1.9874e+07	3.2915e+07	2.89669e+08
2021-07-14 00:00:00 +0000	7c1a02d8	6.8-dev	6.8.15	5c64641cbeeda8a5af88c30d0e92a9eb057d9fec	725253	1.1944e+07	1.9562e+07	3.3047e+07	2.97533e+08
2021-07-16 00:00:00 +0000	c30165d5	6.8-dev	6.8.15	5c64641cbeeda8a5af88c30d0e92a9eb057d9fec	736175	1.1747e+07	1.9104e+07	3.2571e+07	3.0933e+08
2021-07-19 00:00:00 +0000	a60027ff	6.8-dev	6.8.15	5c64641cbeeda8a5af88c30d0e92a9eb057d9fec	745808	1.1682e+07	1.8874e+07	2.8475e+07	3.00155e+08
2021-07-21 00:00:00 +0000	5e082428	6.8-dev	6.8.15	00e292f0ffff12512835b281077750aa60b2c047c	713324	1.2009e+07	1.9759e+07	3.3423e+07	2.99893e+08
							+8.3%	+23.1%	
2021-07-23 00:00:00 +0000	15ce8e36	6.8-dev	6.8.15	00e292f0ffff12512835b281077750aa60b2c047c	696331	1.2149e+07	2.0922e+07	3.9354e+07	3.08543e+08

regression found

Slack Notifications

 **Hunter - Change Point Detection** APP 1:04 PM

Hunter has detected change points

Data Selection
last_n_points: 9223372036854775807
since_time: 2021-06-26 11:04:22.626069+00:00
until_time: 2021-07-26 11:04:22.568955+00:00

2021-07-02 00:00:00

stargazer.dse-db.write.throughput-batch-10

- throughput: 18%
- p50: -16%
- p75: -15%
- p90: -16%
- p95: -16%
- p99: -16%
- p999: -15%

stargazer.dse-db.lwt-rated.1000-partitions

- p95: -15%
- p99: -16%
- p999: -16%

2021-07-03 00:00:00

stargazer.dse-db.write.throughput-batch-50

- p999: -12%

stargazer.dse-db.lwt-rated.100-partitions

- p999: -22%

2021-07-04 00:00:00

stargazer.dse-db.write-throughput

- p75: -10%
- p90: -11%
- p999: -9%

Prior Work

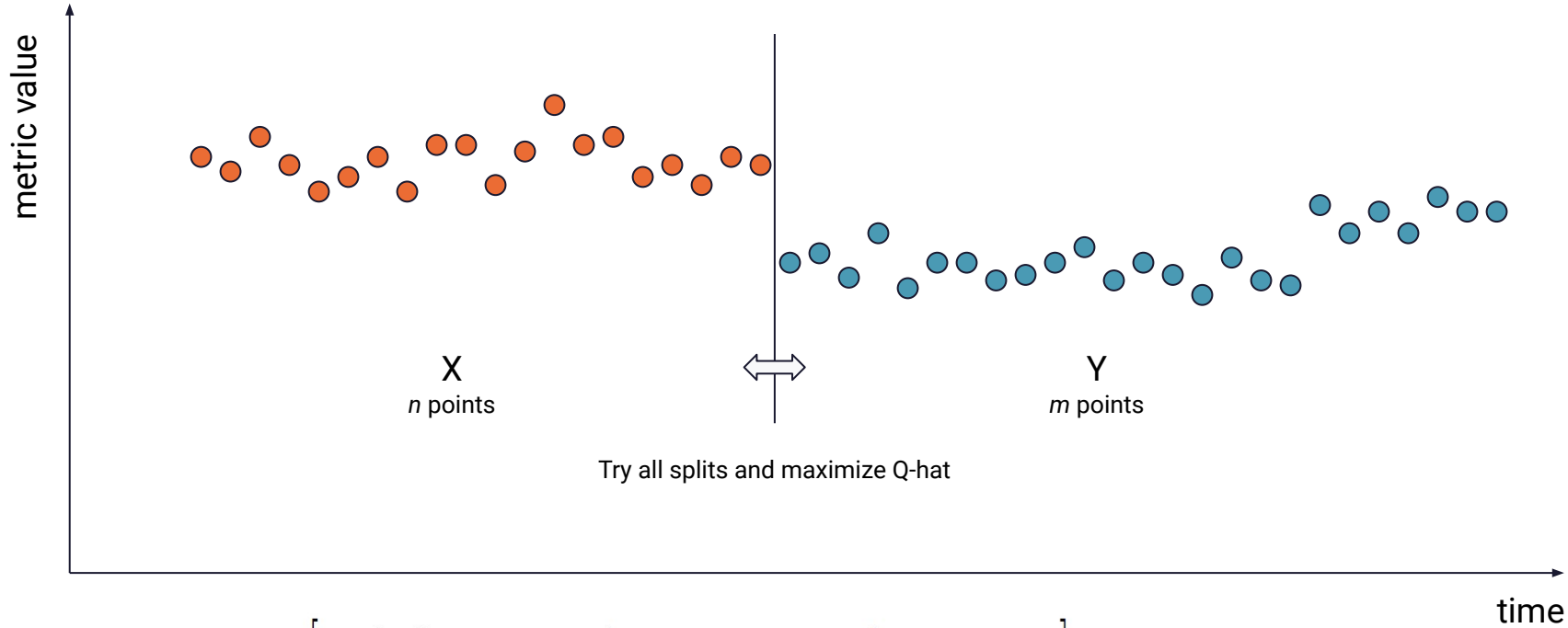
The Use of Change Point Detection to Identify Software Performance Regressions in a Continuous Integration System

David Daly, William Brown, Henrik Ingo, Jim O’Leary, and David Bradford.

In Proceedings of the 2020 ACM/SPEC International Conference on Performance Engineering (ICPE ’20)

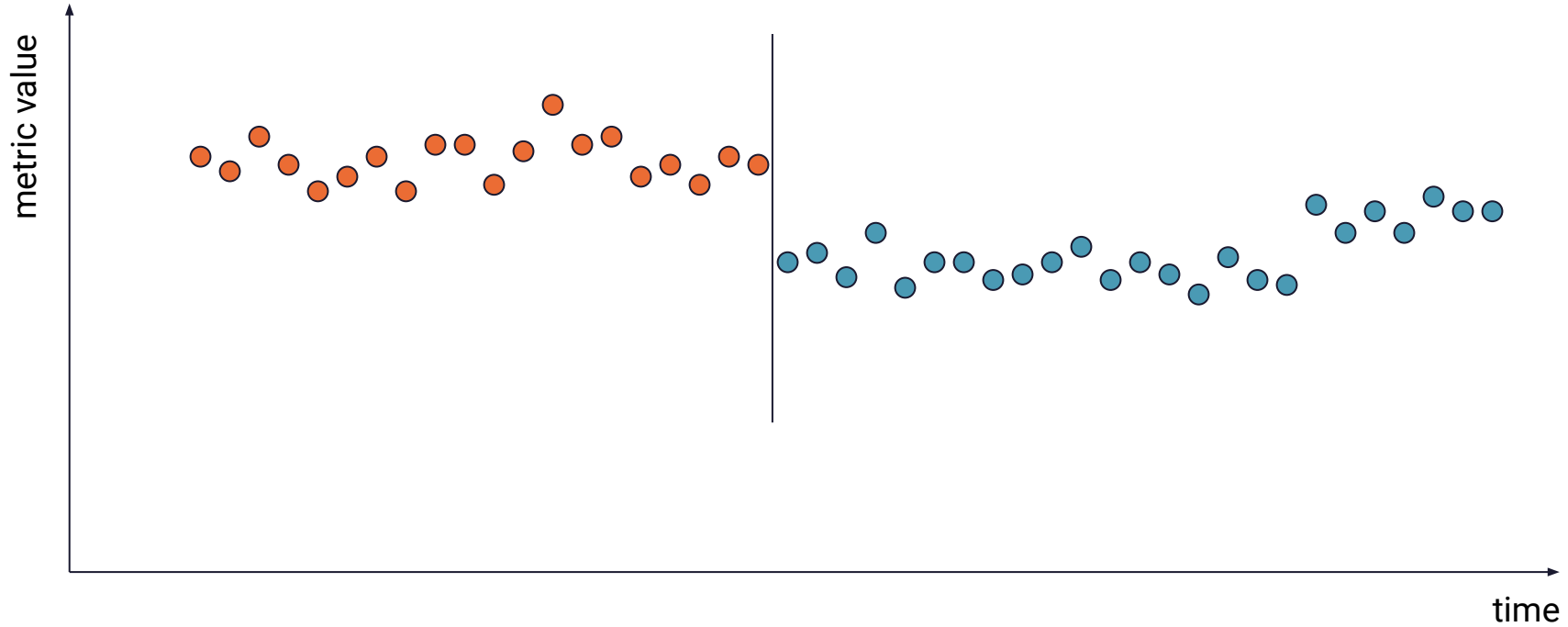
<https://github.com/mongodb/signal-processing-algorithms>

E Divisive with Means – Basic Idea

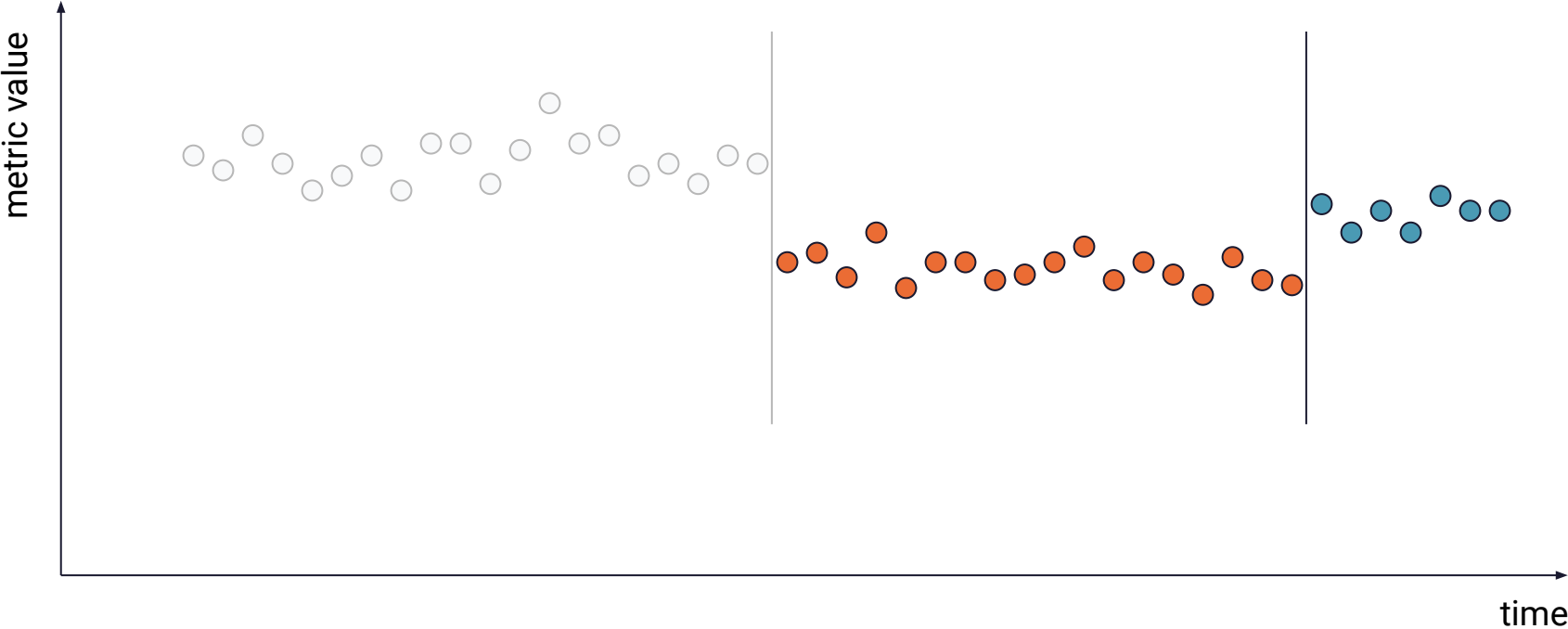


$$\hat{Q}(X_n, Y_m) = \frac{mn}{m+n} \left[\frac{2}{mn} \sum_{i=1}^n \sum_{j=1}^m |X_i - Y_j| - \binom{n}{2}^{-1} \sum_{1 \leq i < k \leq n} |X_i - X_k| - \binom{m}{2}^{-1} \sum_{1 \leq j < k \leq m} |Y_j - Y_k| \right]$$

E Divisive with Means – Apply Recursively Top Down

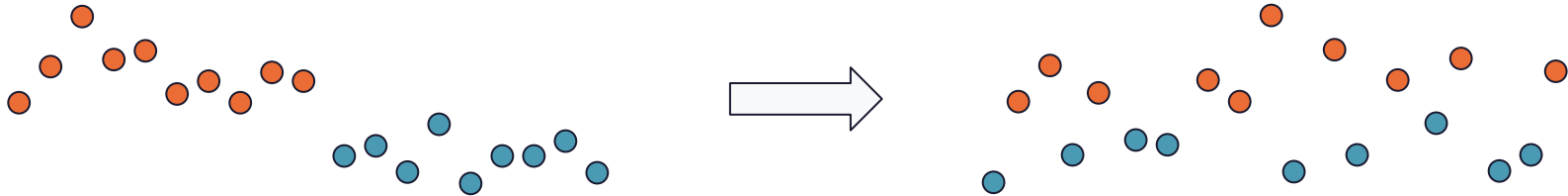


E Divisive with Means – Apply Recursively Top Down



Stop Criterion

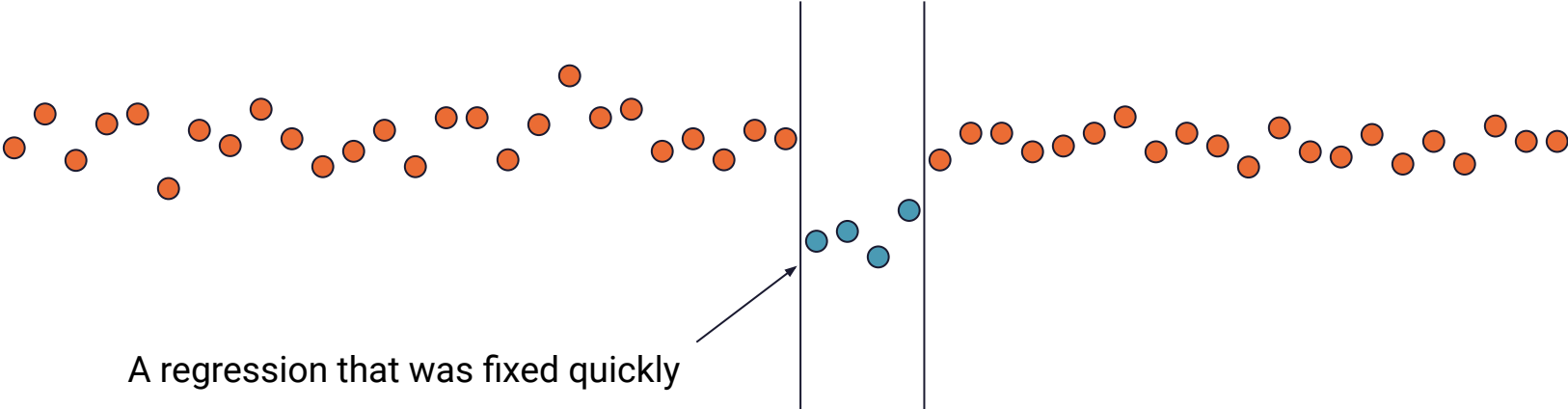
1. Permute the data points randomly N times
2. Compute max Q-hat for each permuted data set
3. If values computed in point 2 are not statistically different from Q-hat for the candidate split, then reject the split and stop



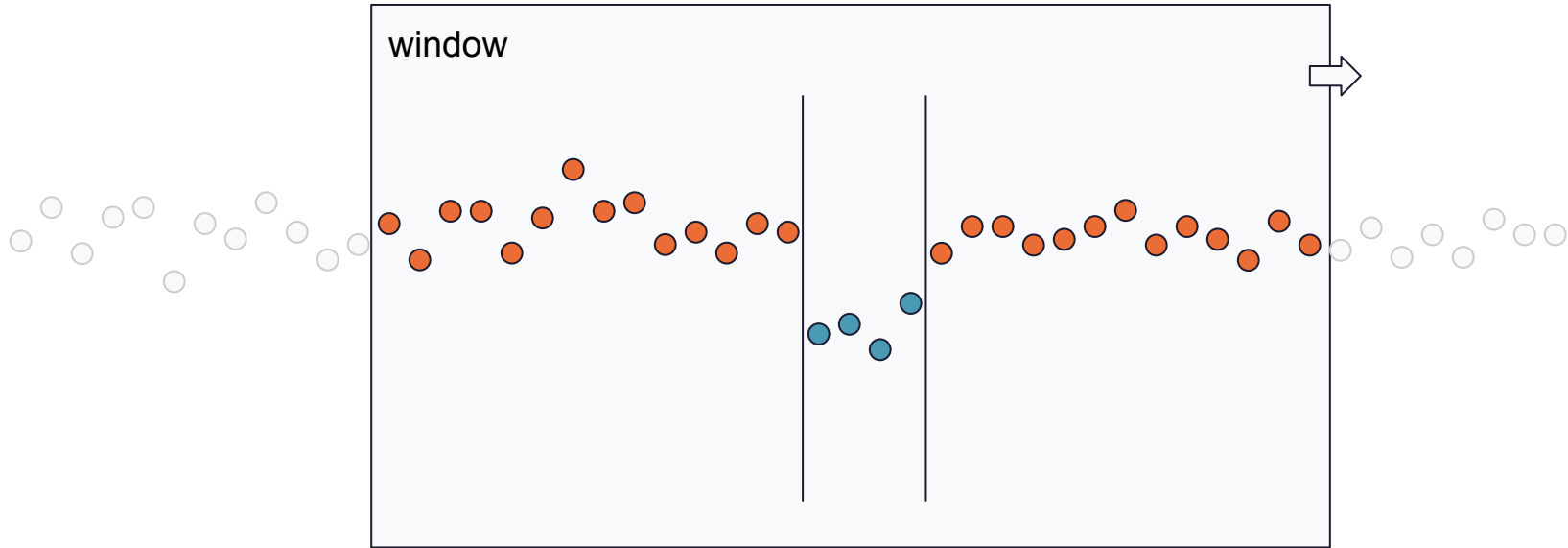
Problems With The Original E-divisive

1. The algorithm fails to find change points in the middle of longer runs of data, although it found them in shorter runs
2. The stop criterion is based on randomness – results not always repeatable
3. The stop criterion is costly to compute

Failure to Identify Some Change Points



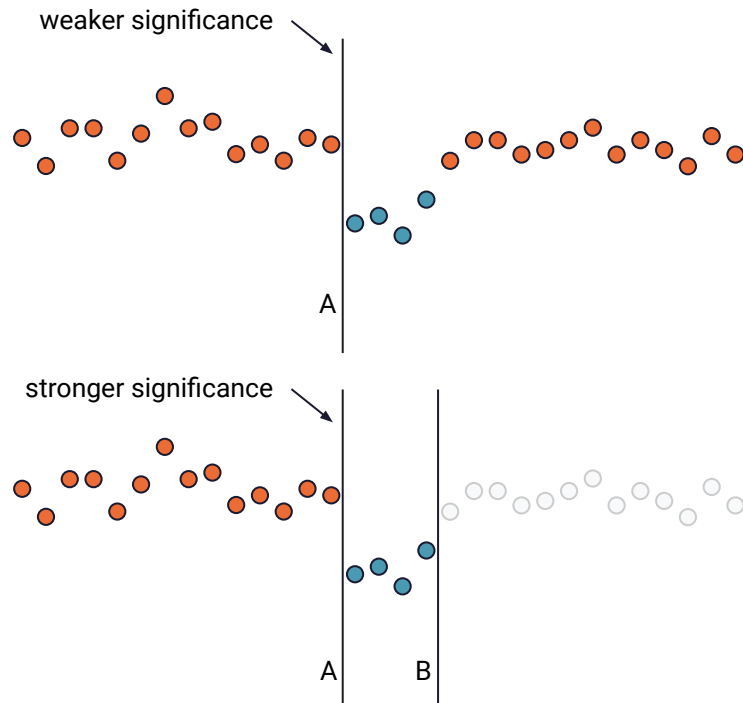
Improvement: Moving Window



Limiting the amount of data processed at once reduces the risk that valid change-points are considered noise

Improvement: Relax the Stop Condition

- Problem: The future splits may affect statistical significance of the earlier splits
- Use a weaker significance level to accept more change-points and terminate the recursion later
- Reevaluate change-points going bottom-up
 - Remove change-points if they turn out to be not statistically significant



Improvement: Welch's T-Test To Test Significance

- The stop criterion based on random permutations is:
 - Non-deterministic
 - Computationally intensive
- Solution: Use Welch's T-test (a variation of Student's T-test)
 - Deterministic
 - Very fast to compute
 - Works with small sample sizes
 - Not-robust, but distribution of data *between* change-points usually normal enough
- We also tried Mann-Whitney, but apparently the number of data points was sometimes too low

Evaluation

Creation of artificial Dataset:

- We employed Chaos Mesh to artificially generate network latency in the system under test to obtain a real data set.
- We artificially injected real changes, at known points in time, into a real benchmark to produce realistic results.
- We created 9 different scenarios by altering the values of variables such as the number of changepoints, magnitude of change of variance between groups, magnitude of change between groups, and the length of groups.
- The scenarios can be grouped into three categories: change in mean, change in variance, and change in both mean and variance.
- Each scenario contains 5 test series, each with minor variation.

Evaluation

Evaluation Metrics:

1. True Positives : $TP(X, X^*) := x \in X \mid \exists x^* \in X^* \text{ s.t. } |x - x^*| \leq M$

2. F1 : $2 \times \frac{\textit{precision} * \textit{recall}}{\textit{precision} + \textit{recall}}$

a. Precision : $\frac{|TP(X, X^*)|}{|X|^*}$

b. Recall : $\frac{|TP(X, X^*)|}{|X|}$

3. Rand Index : $RandIndex(X, X^*) = \frac{TP + TN}{TP + TN + FP + FN}$

a. TP : correctly predicts the positive class : True change points calculated

b. TN : correctly predicts the negative class : None in this case

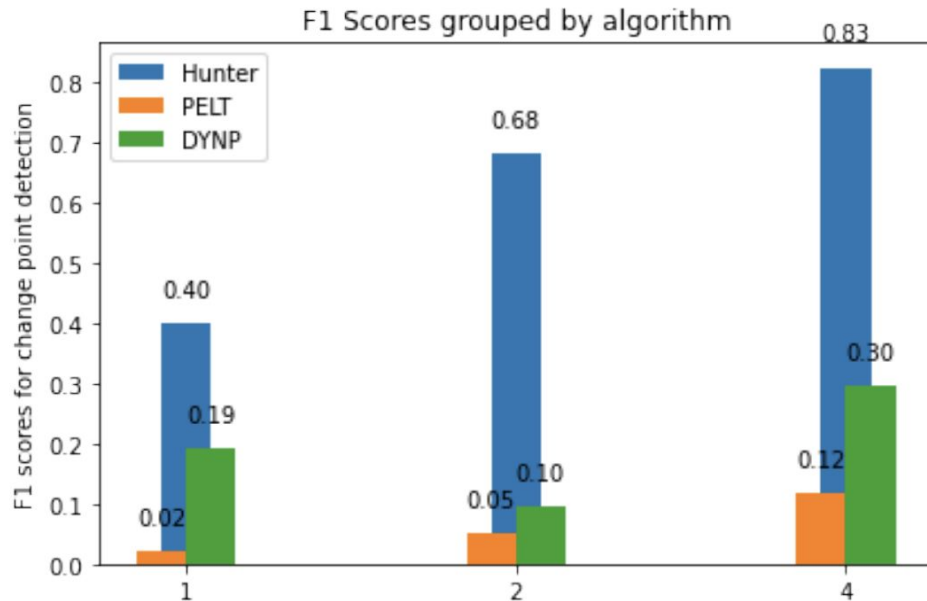
c. FP : model incorrectly predicts positive class : $|X^*| - |TP(X, X^*)|$

d. FN : :model incorrectly predicts negative class: $|X| - |TP(X, X^*)|$

Results

Correlation to the number of points:

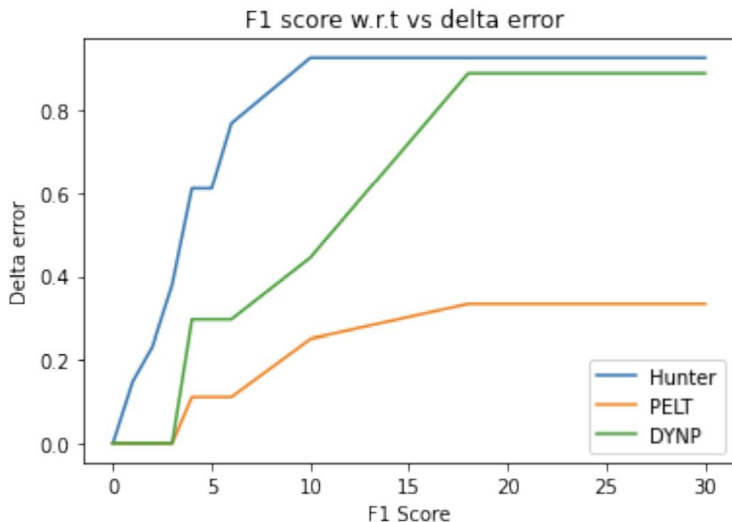
There is a positive correlation between the number of points and accuracy.



Results

Correlation between delta error and algorithms:

- Hunter is able to get an F1 score of 0.1481 with an delta error of one second, where PELT and DYNP need a margin of error of at least 3 seconds to get a non-zero score.
- With a margin of error of 4 seconds we see that the performance increases to 0.612 for Hunter. DYNP starts to catch up with Hunters accuracy at 15 seconds.



DS



Thank You

<https://github.com/datastax-labs/hunter>
<https://github.com/datastax/fallout>