Forecasting a Storm: Divining Optimal Configurations using Genetic Algorithms and Supervised Learning


Summarized by: Hengrun Zhang
Feb. 28th, 2020
Outline

• Motivation and Contribution
• Background
• System Design and Implementation
• Experiment
• Conclusion and Discussion
Motivation and Contribution

**Motivation**

- Maximize the utility of clusters’ resources → Optimal configuration.
- Vast search space, time-consuming.

**Contributions**

- A SVM-based classification method to optimize search space.
- A mapping from an arbitrary topology structure to that of training topologies.
- A retraining scheme to enable portability of the models.
Background

Apache Storm

- Originally developed to perform top-n analysis for the trending topics of tweets.
- Used to process real-time streams of data.
- A batch job in MapReduce → A topology as a directed acyclic graph (DAG) of control flow.
Background

Apache Storm

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workers</td>
<td>Number of processes distributed across the entire cluster assigned to processing a topology within its executor threads</td>
</tr>
<tr>
<td>Sentence Spout parallelism</td>
<td>Number of executor threads assigned to emitting sentences into the rest of the topology</td>
</tr>
<tr>
<td>Sentence Splitter bolt parallelism</td>
<td>Number of executor threads assigned to the execution of bolts which take in sentence and split them on whitespace</td>
</tr>
<tr>
<td>Counter bolt parallelism</td>
<td>Number of executor threads assigned to the execution of the terminal bolts which calculate the frequencies of individual words</td>
</tr>
<tr>
<td>Acker parallelism</td>
<td>Number of executor threads assigned to the acker (Required for all Storm topologies)</td>
</tr>
</tbody>
</table>

**TABLE I: Tunable parameters of the WordCount Topology**
Background

Apache Storm

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workers</td>
<td>Number of processes distributed across the entire cluster assigned to processing a topology within its executor threads</td>
</tr>
<tr>
<td>Sentence Spout parallelism</td>
<td>Number of executor threads assigned to emitting sentences into the rest of the topology</td>
</tr>
<tr>
<td>Sentence Splitter bolt parallelism</td>
<td>Number of executor threads assigned to the execution of bolts which take in sentence and split them on whitespace</td>
</tr>
<tr>
<td>Counter bolt parallelism</td>
<td>Number of executor threads assigned to the execution of the terminal bolts which calculate the frequencies of individual words</td>
</tr>
<tr>
<td>Acker parallelism</td>
<td>Number of executor threads assigned to the acker (Required for all Storm topologies)</td>
</tr>
</tbody>
</table>

TABLE I: Tunable parameters of the WordCount Topology
Background

Apache Storm

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workers</td>
<td>Number of processes distributed across the entire cluster assigned to processing a topology within its executor threads</td>
</tr>
<tr>
<td>Sentence Spout parallelism</td>
<td>Number of executor threads assigned to emitting sentences into the rest of the topology</td>
</tr>
<tr>
<td>Sentence Splitter bolt parallelism</td>
<td>Number of executor threads assigned to the execution of bolts which take in sentence and split them on whitespace</td>
</tr>
<tr>
<td>Counter bolt parallelism</td>
<td>Number of executor threads assigned to the execution of the terminal bolts which calculate the frequencies of individual words</td>
</tr>
<tr>
<td>Acker parallelism</td>
<td>Number of executor threads assigned to the aker (Required for all Storm topologies)</td>
</tr>
</tbody>
</table>

TABLE I: Tunable parameters of the WordCount Topology
Background

Apache Storm

![Diagram of Apache Storm topology]

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workers</td>
<td>Number of processes distributed across the entire cluster assigned to processing a topology within its executor threads</td>
</tr>
<tr>
<td>Sentence Spout parallelism</td>
<td>Number of executor threads assigned to emitting sentences into the rest of the topology</td>
</tr>
<tr>
<td>Sentence Splitter bolt parallelism</td>
<td>Number of executor threads assigned to the execution of bolts which take in sentence and split them on whitespace</td>
</tr>
<tr>
<td>Counter bolt parallelism</td>
<td>Number of executor threads assigned to the execution of the terminal bolts which calculate the frequencies of individual words</td>
</tr>
<tr>
<td>Acker parallelism</td>
<td>Number of executor threads assigned to the acker (Required for all Storm topologies)</td>
</tr>
</tbody>
</table>

**TABLE I**: Tunable parameters of the WordCount Topology
Background

Apache Storm

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workers</td>
<td>Number of processes distributed across the entire cluster assigned to processing a topology within its executor threads</td>
</tr>
<tr>
<td>Sentence Spout parallelism</td>
<td>Number of executor threads assigned to emitting sentences into the rest of the topology</td>
</tr>
<tr>
<td>Sentence Splitter bolt parallelism</td>
<td>Number of executor threads assigned to the execution of bolts which take in sentence and split them on whitespace</td>
</tr>
<tr>
<td>Counter bolt parallelism</td>
<td>Number of executor threads assigned to the execution of the terminal bolts which calculate the frequencies of individual words</td>
</tr>
<tr>
<td>Acker parallelism</td>
<td>Number of executor threads assigned to the acker (Required for all Storm topologies)</td>
</tr>
</tbody>
</table>

**TABLE I**: Tunable parameters of the WordCount Topology
Background

Hill climbing

Genetic Algorithm

Add noise: crossover and mutation.
Background

Genetic Algorithm

Algorithm 20 The Genetic Algorithm (GA)
1: \( psize \leftarrow \text{desired population size} \)
2: \( P \leftarrow \{\} \)
3: for \( psize \) times do
4: \( P \leftarrow P \cup \{\text{new random individual}\} \)
5: \( \text{Best} \leftarrow \square \)
6: repeat
7: for each individual \( P_i \in P \) do
8: \( \text{AssessFitness}(P_i) \)
9: if \( \text{Best} = \square \) or \( \text{Fitness}(P_i) > \text{Fitness}(\text{Best}) \) then
10: \( \text{Best} \leftarrow P_i \)
11: \( Q \leftarrow \{\} \)
12: for \( psize/2 \) times do
13: Parent \( P_a \leftarrow \text{SelectWithReplacement}(P) \)
14: Parent \( P_b \leftarrow \text{SelectWithReplacement}(P) \)
15: Children \( C_a, C_b \leftarrow \text{Crossover}(\text{Copy}(P_a), \text{Copy}(P_b)) \)
16: \( Q \leftarrow Q \cup \{\text{Mutate}(C_a), \text{Mutate}(C_b)\} \)
17: \( P \leftarrow Q \)
18: until \( \text{Best} \) is the ideal solution or we have run out of time
19: return \( \text{Best} \)
A non-linearly separable dataset may be linearly separable in a higher dimension.
System Design

System Architecture

- Nimbus: Scheduler. Schedule works and maintain application metrics for each topology.

- Supervisor: Forward the assignments to workers according to the scheduler and ensure assignments remain alive.

- Workers: Execute assigned works.

The issue of exploring the search space is an instance of the **NP-Hard Knapsack** optimization problem.
Genetic Algorithm (GA) Parameter Search

• Initial Population – Rules of Thumb Baseline
  ✓ An executor should not execute more than one task at a time.
  ✓ The sum of the number of threads for CPU-bound bolts and spouts should also not exceed the total number of cores across the cluster.
  ✓ The Acker thread needs only one thread per Worker process.

• Execution Monitoring – Fitness Calculation
  ✓ Deploys each configuration on the Storm cluster and observes it for a period of five minutes to evaluate its fitness (throughput).
  ✓ Regression model is prone to overfitting.
GA Parameter Search Continued

• Classifying Bad Parameters – Search Space Optimization
  ✓ Training corpus: Randomly generated configurations and observed throughput without violating Rules of Thumb, previously run results of GA.
  ✓ Regression discretized to classification: “Good” is defined as 80 percent or greater of the maximum throughput discovered in each topology training set.
  ✓ SVM-based ensemble techniques:

\[
\text{classification} = \begin{cases} 
0, & \text{if } \frac{\sum_{e \in G | e=0} P_e}{\sum_{e \in G} P_e} \geq \frac{\sum_{e \in G | e=1} P_e}{\sum_{e \in G} P_e} \\
1, & \text{otherwise}
\end{cases}
\]
GA Parameter Search Continued

• GA selection, mutation and crossover
  ✔ Selection: Tournament selection and Roulette wheel selection.
  ✔ Mutation and Crossover: No specified mutation. Single point crossover.

Algorithm 32  Tournament Selection
1:  $P \leftarrow$ population
2:  $t \leftarrow$ tournament size, $t \geq 1$
3:  $Best \leftarrow$ individual picked at random from $P$ with replacement
4:  for $i$ from 2 to $t$ do
5:     $Next \leftarrow$ individual picked at random from $P$ with replacement
6:     if Fitness($Next$) > Fitness($Best$) then
7:         $Best \leftarrow Next$
8:  return $Best$

Figure 9  One-Point Crossover.
GA Parameter Search Continued

• GA selection, mutation and crossover
  ✓ Selection: Tournament selection and Roulette wheel selection.
  ✓ Mutation and Crossover: No specified mutation. Single point crossover.

```plaintext
Algorithm 30 Fitness-Proportionate Selection
1: perform once per generation
2:   global $\vec{p}$ ← population copied into a vector of individuals $\langle p_1, p_2, ..., p_l \rangle$
3:   global $\vec{f} ← \langle f_1, f_2, ..., f_l \rangle$ fitnesses of individuals in $\vec{p}$ in the same order as $\vec{p}$ ▷ Must all be $\geq 0$
4:   if $\vec{f}$ is all 0.0s then ▷ Deal with all 0 fitnesses gracefully
5:     Convert $\vec{f}$ to all 1.0s
6:   for $i$ from 2 to $l$ do ▷ Convert $\vec{f}$ to a CDF. This will also cause $f_i = s$, the sum of fitnesses
7:     $f_i ← f_i + f_{i-1}$
8:   perform each time
9:     $n ←$ random number from 0 to $f_i$ inclusive
10:    for $i$ from 2 to $l$ do ▷ This could be done more efficiently with binary search
11:        if $f_{i-1} < n ≤ f_i$ then
12:            return $p_i$
13:    return $p_1$
```
System Implementation

- **Classifier service**: scikit-learn, Tensorflow, Keras.
- **Apache HTTP client**: Java and Python interoperability.

- Parameter sets mapping (Preliminary, future work):
  - Topology mapped to vector in the order: spout → bolt → sink.
  - Zero for a spout and a sink without a bolt in between.
  - Average parallelism of each of these elements’ thread counts for topologies with more than one spouts, bolts or sink bolts.

- Transfer learning for new clusters and topologies.
Evaluation

• Assess both the **quality and timeliness** of the derived configuration.

• Evaluate the **quality of the individual classifiers** and the impact of the training corpus’ size on them - F1 score.

• Evaluate the **effect of retraining the classifier** to a new cluster so as to analyze the **portability** of the system and ultimately demonstrate the full utility of our system.

• Testbed: Intel Cluster and Arm Cluster.
Evaluation

- Assess both the **quality and timeliness** of the derived configuration.
- Evaluate the quality of the individual classifiers and the impact of the training corpus' size on them - F1 score.
- Evaluate the effect of retraining the classifier to a new cluster so as to analyze the portability of the system and ultimately demonstrate the full utility of our system.

Testbed: Intel Cluster and Arm Cluster.
Spearmint-Derived Tuner vs. GA Tuner

- Spearmint-derived tuner:
  - Bayesian optimization.
  - Kernel density estimators and the posterior probability model to find promising new candidate configurations.
  - Bayesian optimization relies on two assumptions to work, which does **not work** in this paper: a lack of covariance among the parameters of a configuration and those parameters are each of a continuous range of values to set.
Spearmint-Derived Tuner vs. GA Tuner Continued

- Spearmint-derived tuner: Bayesian optimization. Kernel density estimators and the posterior probability model to find promising new candidate configurations.
- Bayesian optimization relies on two assumptions which do not work in this paper: a lack of covariance among the parameters of a configuration and those parameters are each of a continuous range of values to set.
Spearmint-Derived Tuner vs. GA Tuner Continued

• Spearmint-derived tuner: Bayesian optimization, kernel density estimators and the posterior probability model to find promising new candidate configurations.

Bayesian optimization relies on two assumptions to work, which does not work in this paper: a lack of covariance among the parameters of a configuration and those parameters are each of a continuous range of values to set.
Classifier Quality Comparison


Fig. 8: Classifier F1 Scores vs Corpus Size for WordCount.
Comprehensive GA Tuner Evaluation

Fig. 9: Throughput for Tuners on the Intel Cluster.

Fig. 10: Execution Time for Tuners on the Intel Cluster.
Comprehensive GA Tuner Evaluation

Fig. 9: Throughput for Tuners on the Intel Cluster.

Fig. 11: GA Exploration with or without a classifier.
Portability Evaluation

Fig. 12: Throughput of Tuners on the ARM Cluster.

Fig. 13: Training from Scratch vs Transfer Learning.

Fig. 14: STT Evolution with Nimbus and Retrained Classifiers.
Potential Reasons for not Behaving Well

- Underfitting, too small training corpus.
- F1 is just a statistical metric, and still cannot justify a specific sample.

\[
F_1 = \left( \frac{2}{\text{recall}^{-1} + \text{precision}^{-1}} \right) = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}
\]
Conclusion and Discussion

• Conclusion
  ✓ The stock GA is able to improve upon the standard “rules of thumb” tuning strategy’s throughput by an average of 4.6x, with between 1.1x and 2.2x improvement over Spearmint.
  ✓ Employing classifiers to optimize the search process reduces the execution time of our GA tuner by over 40x in certain cases.

• Discussion
  ✓ Old problem, old method. Application innovation. Experiments are comprehensive and real.
  ✓ Preliminary case – homogeneous tasks. No workload variation.
  ✓ Throughput calculation is quite important.
Thank you!

Q & A