Exploring Patterns and Correlations in CMS Computing Operations Data with Big Data Analytics Techniques

V. Kuznetsov, T. Wildish, D. Bonacorsi, L. Giommi

March 17th, 2015
CMS has launched an **Analytics project**

- **Long-term goal** is building **data-driven models** of CMS data management (DM) and workflow management (WM) activities
- **Shorter-term goal** is learning how to improve efficiency in our DM/WM through small, independent analytics pilot projects based on measuring past performances and trying to predict future behaviours

This comes via a **deeper understanding of CMS monitoring (meta-)data**

- it may just be a by-product of the overall activity, but it has a huge value in itself

Use such data as input to **adaptive modelling** of CMS Computing

- models of the past aren’t going to apply to the future for long.
- adaptive modelling will give us confidence and predictive power in the long term

**Improved use of computing resources** in Run-2 and beyond

- aim for incremental improvements

**Bottom-up approach**

- start from specific things we want to learn about how the system works
- launch **pilot projects** about each - quite independent one from the other
- learn as we go
Motivation and approach

LHC Run-2 is around the corner
- LHC Run-1 is over. A huge success for LHC. The first Long Shutdown (LS1) is also almost over.

In retrospective, we built a model that worked: it is time to understand it.
- solid improvements come from a deep understanding of what the systems did in Run-1 and in LS1
- e.g. population vs pollution of Tier-2 storage disks, models of T2-T2 PhEDEx traffic, etc

CMS collisions data vs computing ops meta-data: “published” vs “unexplored”
- every single bit of collisions data has been read multiple times, and analysed by hundreds of physicists
- the same did not happen for the computing operations meta-data

This data is all archived, but rarely (or never) accessed by anyone
- e.g. transfers, job submissions, site performances, releases logs, analysis performances, etc
- we basically monitor to debug in near-time, not to analyse what happened in the past
- we never fixed holes in monitoring data, we never validated (most of) them with a decent care
- not polished and not complete/coherent ⇒ not explorable ⇒ not exploitable in its current form

Variety (and veracity) are the Big Data V’s that matter most here
- Volume: not negligible in itself, but definitely manageable
- Velocity: partially relevant, i.e. quick availability for analysis is a requirement, but real-time is not a must
- Variety: very irregular data set: structured, semi-structure and unstructured data
- Veracity: data integrity and the ability to trust them to make decisions is important
Structured info on a variety of CMS Computing activities are stored across multiple data services

✦ all info available via CMS data service APIs
Data sources for structured info

DBS
✦ physics meta-data

PhEDEx
✦ data transfer service and data replica catalog functionalities

PopDB
✦ dataset user access information (e.g. frequency, which replicas, CPU used)

SiteDB (/ REBUS)
✦ site pledges and deployed resources information, plus people

Dashboard
✦ massive repository of Grid jobs details (and beyond)

More may be added…
Unstructured data

Plenty of unstructured information in the CMS Computing ecosystem

- hard to process but very diverse and potentially very rich!
Unstructured info

A non-exhaustive list. Many of them are potentially useful to mine migration of interests and group activities evolution

- hard task, but would be a sensitive predictor of user activities within a large collaboration

HyperNews

- today, 400 different fora represent several years of user activities
- 'social data mining' aspects of collaboration-level activities
- announcements, info on user activities, change of focus in the physics interests of individuals/groups over time, hot topics, study of the HN membership, etc

Tickets and beyond: infrastructure issues reporting/tracking

- ticketing systems (Savannah, GGUS), activity-based ELOGs, topical e-groups, ..

Twikies (and more)

- their content is a knowledge graph that could be mapped to user activities and physics interests
- their evolution over time, depending on the content, could help to track hot periods of the year in given topics

Calendars of events (also non CMS), sub-projects planning info

- CMS events calendar, activity planning documents, list of major conferences and workshops, etc
- could identify diverse seasonal cycles within different physics communities

CMSWEB cluster front-end logs [ semi-structured data ]

- serves all data sources to users, thus its logs may be mined to extract valuable info on user activities

And more…
Several well formulated “problems” may find insights towards a solution with pilot sub-projects in the CMS Analytics project:

**Data popularity predictions**
Block latency analysis in PhEDEx data transfers
Asynchronous Stage Out (ASO) and FTS3
Co-scheduling CPU/Disk/Network
Seeking cost-effective solutions for CMS workflows
Modelling the CMS Computing Model
Content Centric Network architecture for CMS
… and more …

Take this one as an example in the following
One example:

Predict popularity of new datasets

Problem formulation:

✦ CMS has info on datasets “popularity”, i.e. most frequently accessed replicas
✦ The CMS Dynamic Data Placement team uses historical popularity info to add (remove) replicas of existing datasets that appear to be most (least) popular
✦ A looking-forward goal would be to predict which new datasets will become popular once they become available on Grid for distributed analysis

DCAF Pilot: a pilot project to understand the metrics, the analysis workflow, the necessary functionalities (and possible technology choices) of the machinery needed to attack this problem:

✦ DCAF stands for Data and Computing Analysis Framework
✦ Valentin Kuznetsov and Tony Wildish as main contributors
✦ See next
DCAFPilot: data collection flow diagram

Dataframe generator toolkit: collects/transfers data from CMS data services (structured data only, here) and extract necessary bits for datasets under study.

MongoDB used for internal cache

ML algorithms (python / R code) for data analysis

[ courtesy of V.Kuznetsov ]
The DCAFPilot workflow

Data collection flow:
- collect all datasets from DBS into internal cache
- query popular datasets from PopDB with a weekly granularity
- on those, get more info also from DBS/PhEDEx/SiteDB/Dashboard
- complement with random set of unpopular datasets

All such information is stored in different data-frame files
- these can be fed to any ML library for whatever purpose one may have
  - all data from 2014 have already been processed and are available for analysis

A prediction of which dataset(s) may become popular is hence given
- in the form of the probability vs. each dataset name
- that’s what you want! you can feed back this info to the infrastructure and use it

Some figures from a dry run of the machinery:
- queried 5 data services (4 DBS instances used), 10 APIs used; internal cache fed with ~220k datasets, ~900 release names, 500+ SiteDB entries, 5k people’s DNs; ~800k queries placed overall. Anonymization and factorisation via internal cache.
- final data-frame constructed out of 78 variables, made of 52 data-frame files, ~600k rows; each file is worth 1 week of CMS meta-data (~600kB gzipped) and has ~1k popular datasets with a ~1:10 ratio of popular vs unpopular randomly mixed)
Visualize the data-frame

Live data

Correlations

Different dataset popularity metrics

A. by # users accessing it
B. by total CPU used to process it
From data-frame to prediction

1. Data collection: see previous slides
2. Data transformation into suitable format for ML
3. The ML model:
   - use **classification** or **regression** techniques
     - the former allows to predict real values of metrics (e.g. # accesses)
     - the latter allows only to classify into categories (e.g. popular or unpopular)
   - train and validate your ML model
     - split data into train and validation sets
     - ~600K rows in the 2014 dataset: Jan-Nov used as a train set, Dec used as the validation set
     - estimate your predictive power on the validation set
4. Generate new data and transform it (similar to step #2)
5. Apply your best model to new data to make prediction
6. Verify prediction with Pop-DB once metrics become available
### Preliminary observations

#### Preliminary results from the ML model

- **rows tagged with “all”** = data from model training and validation
- **rows tagged with “new”** = new data not present in the training dataset

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Data</th>
<th>accu prec reca f1</th>
<th>accu prec reca f1</th>
<th>accu prec reca f1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Forest</td>
<td>all</td>
<td>0.97 0.85 0.89 0.87</td>
<td>0.89 0.00 0.00 0.00</td>
<td>0.92 0.70 0.89 0.78</td>
</tr>
<tr>
<td></td>
<td>new</td>
<td>0.83 1.00 0.83 0.91</td>
<td>0.00 0.00 0.00 0.00</td>
<td>0.92 1.00 0.92 0.96</td>
</tr>
<tr>
<td>SGDClassifier</td>
<td>all</td>
<td>0.97 0.88 0.68 0.77</td>
<td>0.95 0.86 0.70 0.77</td>
<td>0.95 0.70 0.72 0.71</td>
</tr>
<tr>
<td></td>
<td>new</td>
<td>0.45 1.00 0.45 0.62</td>
<td>0.71 1.00 0.71 0.83</td>
<td>0.60 1.00 0.60 0.75</td>
</tr>
<tr>
<td>Linear SVC</td>
<td>all</td>
<td>0.94 0.53 0.99 0.69</td>
<td>0.97 0.83 0.97 0.89</td>
<td>0.95 0.62 0.92 0.74</td>
</tr>
<tr>
<td></td>
<td>new</td>
<td>0.98 1.00 0.98 0.99</td>
<td>0.97 1.00 0.97 0.98</td>
<td>0.90 1.00 0.90 0.95</td>
</tr>
<tr>
<td>Vowpal Wabbit</td>
<td>all</td>
<td>0.95 0.61 0.69 0.65</td>
<td>0.99 0.91 1.00 0.95</td>
<td>0.94 0.65 0.61 0.63</td>
</tr>
<tr>
<td></td>
<td>new</td>
<td>0.54 1.00 0.54 0.70</td>
<td>1.00 1.00 1.00 1.00</td>
<td>0.49 1.00 0.49 0.65</td>
</tr>
<tr>
<td>xgboost</td>
<td>all</td>
<td>0.98 0.88 0.95 0.92</td>
<td>0.99 0.97 1.00 0.98</td>
<td>0.96 0.71 0.97 0.82</td>
</tr>
<tr>
<td></td>
<td>new</td>
<td>0.92 1.00 0.92 0.95</td>
<td>1.00 1.00 1.00 1.00</td>
<td>0.98 1.00 0.98 0.99</td>
</tr>
</tbody>
</table>

### Statistical variables: accuracy, precision, recall and F1 scorers
A good start!
Just an example of one of the projects (data popularity forecast)

✦ More are being worked on right now, others will be added later on.

DCAFPilot is already a good proof-of-concept

✦ Caution in drawing conclusion is a must, of course, but it gives interesting insights already.
✦ We had nothing only few months ago.

Plenty of work to do

✦ no ad-hoc work to avoid main and known ML obstacles has been made (yet)
✦ more data types needed to be added for meaningful conclusions to be drawn
✦ all work done on a single node with existing APIs, need to pursue other approaches
✦ explore more and diverse ML algorithms
✦ try out different popularity metrics
✦ … and more …
Summary

CMS has launched an **Analytics project**
- build a **data-driven adaptive model of CMS Computing**
- great value in **understanding the past to try to predict the future**
- precious by-product in understanding our (meta-)data from Run-1/LS1 ops

First **proof-of-concept** is out
- feeding enthusiasm into the team, and encourage to continue exploring!

**Methodology** is helping us along the road:
- **bottom-up** approach: start from simple and well formulated problems
  - add complexity as we progress
- several **pilot projects** running **in parallel** and contributing to a **common goal**
  - different people involved in different sub-projects which may exploit different technologies
- **learn by doing**, and adapt to what we learn!