Large-scale social-media analytics on stratosphere

Conference Item

How to cite:


For guidance on citations see FAQs.

© International World Wide Web Conference Committee (IW3C2).
Version: Version of Record
Link(s) to article on publisher’s website:
http://www2013.org/

Copyright and Moral Rights for the articles on this site are retained by the individual authors and/or other copyright owners. For more information on Open Research Online’s data policy on reuse of materials please consult the policies page.

oro.open.ac.uk
Large-Scale Social-Media Analytics on Stratosphere

Christoph Boden, Volker Markl
TU Berlin
Germany
{first.last}@tu-berlin.de

Marcel Karnstedt
DERI, NUI Galway
Ireland
marcel.karnstedt@deri.org

Miriam Fernandez
Knowledge Media Institute
Milton Keynes, UK
m.fernandez@open.ac.uk

ABSTRACT
The importance of social-media platforms and online communities – in business as well as public context – is more and more acknowledged and appreciated by industry and researchers alike. Consequently, a wide range of analytics has been proposed to understand, steer, and exploit the mechanics and laws driving their functionality and creating the resulting benefits. However, analysts usually face significant problems in scaling existing and novel approaches to match the data volume and size of modern online communities. In this work, we propose and demonstrate the usage of the massively parallel data processing system Stratosphere, based on second order functions as an extended notion of the MapReduce paradigm, to provide a new level of scalability to such social-media analytics. Based on the popular example of role analysis, we present and illustrate how this massively parallel approach can be leveraged to scale out complex data-mining tasks, while providing a programming approach that eases the formulation of complete analytical workflows.

Categories and Subject Descriptors
H.2.4 [Database Management]: Systems — Parallel Databases;
H.4.3 [Communications Applications]: Bulletin boards

General Terms
Algorithms, Performance, Measurement

Keywords
Role Analysis, Behaviour Analysis, Online Communities, Scalability, Stratosphere, Community Analysis, Boards.ie

1. OVERVIEW
Online communities are now an integral part of the World Wide Web. They provide the users with an environment in which they can interact and discuss topics of interest and seek answers to questions and support requests. Similar, companies identified the utility of such online communities and started to offer social-media platforms to foster intra- and intra-enterprise cooperation and communication. A recent report of McKinsey estimates that between 9,000 billion to 1.3 trillion dollars in annual value could be obtained if the knowledge created by social-media based collaboration is fully exploited.

Exploiting this knowledge requires the development of analysis tools that can help to understand and manage these communities as well as the social and economic objectives of its users, providers, and operators. A popular, and particularly relevant example, is the understanding of the users’ actions and interactions with other community members, i.e., the understanding of the users’ online behaviors. In this context, communities are often assessed by their role composition – i.e., the distribution of users assuming different roles like experts, popular participants, ignored users, etc. This involves content and structural analysis of the extracted interaction graphs and can provide important indicators of the community's health and evolution, its functioning and return of investment, as well as the key contributors that community managers and owners may want to maintain and engage more closely.

However, the development of such community analyses does not come without major challenges. Individual communities can easily exceed a million users with hundreds of thousands of online interactions each day. Content generation can be many Gigabytes per day and orders of magnitude more data are derived from observing interactions of the users with the system. Existing solutions struggle to handle such scale and cannot cope at all with the imminent growth predicted for online communities.

The massively parallel data processing system Stratosphere is particularly designed to flexibly support such complex big-data analysis tasks. It is positioned between systems supporting fixed parameterizable pipelines, such as MapReduce, and systems that provide the required non-relational data and operations on top of a relational database core. Extending the MapReduce paradigm, it offers first-class support for non relational data types and flexible user-defined operations. We propose and demonstrate how to formulate complete analytics pipelines, involving all steps from data pre-processing over feature extraction to machine learning and post-processing, in the form of so-called PACTs (Parallelization Contracts) plans. The Stratosphere system fully exploits the massively parallel execution of such PACT plans.

This is a crucial point particularly, but not only, for the data-mining tasks in the focus of this work, where analytical workflows usually involve a range of different complex steps.

http://www.mckinsey.com/insights/mgi/research/technology_and_innovation/the_social_economy
In our demonstration, we highlight the flexible and convenient support of PACTs to achieve this objective. We show how the Stratosphere system optimizes and executes the resulting programs in a massively parallel way and we emphasize the performance gains this provides. We exemplify the potential of this approach by offering an explorative tool for analyzing the dynamics of the role composition of an online community in a scale for which alternative solutions emerge as infeasible. As such, our demonstration will act as a showcase and practical guideline for other researchers that aim for huge-scale analytics of social media and online communities – and can achieve this by relying on the flexible support of PACTs offered by the Stratosphere system.

2. PACTS IN STRATOSPHERE

Stratosphere is an open-source system that enables the massively parallel execution of data analytics tasks. This demo is based on the publicly available release version 0.2 and focuses on how to specify a complete use case as an analytics pipeline in Stratosphere. Other Stratosphere related demos focused on the efficient execution of incremental iterations and the reordering of operators. Both system capabilities are not part of the current open source version of Stratosphere and might become part of future releases. The application of Stratosphere’s declarative query language Meteor to complex analytics tasks was demonstrated in.

The overall architecture of Stratosphere is depicted in Figure 1. The basic layer is provided by the parallel execution engine Nephele. It executes data flow programs modeled as directed acyclic graphs (DAGs) in a parallel and fault-tolerant way. Nephele keeps track of task scheduling and setting up the required communication channels. In addition, it dynamically allocates the compute resources during program execution.

A user of the Stratosphere system formulates algorithms in Stratosphere in one of multiple programming interfaces, which support different levels of abstraction. The optimizer chooses the physical execution strategies for the individual functions with the objective of minimizing the overall execution costs.

In order to leverage Stratosphere for truly large-scale community analytics, we propose that analysts should have to focus on only creating appropriate plans of Parallelization Contracts (PACTs). PACTs are essentially a similar level of abstraction as the popular MapReduce paradigm. The user implements functions that the system evaluates on partitions of the data, where special second-order functions define how these partitions are actually formed. In addition to Map and Reduce, PACT offers binary second-order functions, such as Match, CoGroup, and Cross. These contracts can be freely assembled into directed acyclic graphs which allows for the creation of complete analytics pipelines including feature extraction, model training and post-processing that can be implemented as UDFs inside the second order functions.

3. DEMO SCENARIO: ROLE ANALYSIS

For the actual demo scenario, we have selected role analysis as one of the key analytics that can provide valuable insights into the evolution and state of online communities and their members. We have implemented the behavior analysis approach proposed by and refined by, which analyses the percentages of different roles that are assumed by community users over time. To do this, the behavior that users exhibited are measured by extracting numeric features from the data, each intended to capture one particular dimension of a user’s behavior (e.g., focus dispersion, popularity, engagement, initiation or contribution). This feature extraction is a data-intensive task, since the features need to be computed for each particular individual over time from the raw data. To achieve a fine granular analysis over time, this is usually done on a daily or weekly basis for each subcommunity (e.g., all different forums) separately. In each step of time, the activities and interactions of each individual in the recent past are considered for the feature computation, where both of the before mentioned works apply a sliding window of 6 months.

Based on the computed features the approach clusters users to deduce the role composition for each subcommunity at a given point in time. The role mining and inference process requires the iterative application of clustering methods over time to extract the behavior roles that emerge from the data. Additionally, once the roles are identified, an inference process is put in place to derive role labels for each particular user over time. Note that this behavior analysis approach is not static and assumes that the role of a user is contextual (i.e., it depends on her actions and interactions observed at particular time steps). This means, for example, that a user can adopt the role of “novice user” the first time she registers with a particular online community, and achieve the role of “expert” months after. The dynamics of the role mining and inference process, involving continuous clustering of the whole user base, constitute a major bottleneck of the approach when applied to large-scale datasets.

For example, this approach has been previously applied to a community dataset with 95,200 threads, 421,098 posts and 32,942 users. After introducing intensive manual code and database optimizations (e.g., multithreading, database load on demand, indices, etc.) the execution of the whole workflow on an Intel(R) Xeon(R) CPU machine with 2 processors at 2.27GHz and 24 GB of memory took 4.5 hours for the feature extraction and more than 16 hours for the role mining and inference process. These numbers illustrate that this approach is limited by the significant computational cost.

A delay of hours of processing does not permit the extraction of rapid insights that community managers may need to maintain the health of their communities, neither is it feasible to support an explorative analysis for the involved researcher. Exploiting the massively parallel Stratosphere system for this task allows us to run this sort of analysis on data sets that were simply not feasible before. As such,
This is illustrated by Algorithm 2, which lists the UDF
tractions contains user code to actually carry out a certain sub-
cusDispersion feature as suggested in [8]. Each of these con-
eral different parallelization contracts which extract the Fo-
Algorithm 1 shows an exemplary sub plan consisting of sev-
eres. The contracts needed for preprocessing the data and
cluster consisting of 26 machines
Table 1: Average execution times on our research
we are able to now process the whole data from the most
popular Irish discussion site boards.ie This encompasses 8.26 million posts in 680,000 threads created by 138,000 dif-
ent users over 10 years of complete data, a significantly
larger data set than in all our experiments before. Table 1
shows the achieved runtimes for pre-processing, feature ex-	raction and clustering required for the role composition for
each user and each week. These were computed for each fo-
rum separately and in addition over all forum borders taking
all global user activity into account. This addition provides
further valuable insights into the evolution and state of the
boards.ie community as a whole.
It is important to note that this approach for community
analysis resembles close similarities to a wide range of other
analytical tasks. As such, it is an example tailored to show-
case the general applicability and the gained benefit from
processing such tasks in the Stratosphere system.

Algorithm 1: Simplified PACT example for extracting
one feature
1. MatchContract joinThreadsPosts(threads, posts, postID)
2. ReduceContract countPosts(joinThreadsPosts, keys: {forumId, users, week})
3. ReduceContract calculateEntropy(countPosts, keys: {week, user})
4. output FileDataSink FileDataSink(calculateEntropy)

Algorithms 1 and 2 illustrate how convenient and easy it
is to formulate the analytics pipelines consisting of PACTS.
Algorithm 1 shows an exemplary sub plan consisting of sev-
der different parallelization contracts which extract the Fo-
cusDispersion feature as suggested in [8]. Each of these con-
tracts contains user code to actually carry out a certain sub-
task. This is illustrated by Algorithm 2 which lists the UDF
implemented inside the calculateEntropy() Reduce contract
from line 3 in Algorithm 1. A plan is assembled by in-
stantiating the contracts and chaining the inputs such that
subsequent contracts consume the output of their predeces-
sors. The contracts needed for preprocessing the data and
clustering the users can be added to the PACT plan just as
easily so that the complete workflow can be expressed as one
PACT plan.

Algorithm 2: calculateEntropy(record, collector)
1. int sum = 0
2. List<Integer> grouped = Lists.newArrayList()
3. while records.hasNext() do
4. | PactRecord record = records.next()
5. | userId = record[0]
6. | week = record[1]
7. | int count = record[2]
8. | sum += count
9. | grouped.add(count)
10. double h = 0.0
11. for all the Integer group : grouped do
12. | double ratio = ((double) group) / sum
13. | h += ratio * Math.log(ratio)
14. entropy.setValue(-h)
15. outputRecord[0] = userId
16. outputRecord[1] = week
17. outputRecord[2] = entropy
18. collector.collect(outputRecord)

Figure 2 shows the FocusDispersion plan outlined in Al-
gorithm 1. A PACT plan and illustrates the execution strate-
gies chosen by the optimizer. Each vertex corresponds to
a PACT, containing the user-defined function and a strat-
ey to apply it to the data, according to the second-order
function. Edges represent the result of one PACT being for-
warded to the next. They are labeled with the data-shipping
strategy they execute (e.g., in-memory pipe or network hash
partitioning) and local operations they apply (such as sort-
ing). In addition, the plan shows which data structure is
used to store and update the solution set. A click on a vert-
ox shows inferred properties of the data, such as parti-
tioning and order properties, as well as basic cost estimates
for disk and network usage.

4. THE DEMO ITSELF
The demonstration comprises three main parts:
1. Formulation of complex analytical tasks by means of
understanding and writing PACT execution plans.
2. Running and monitoring the optimization and execu-
tion of the resulting data flows.
3. Exploration of the results gathered with the imple-
mented large-scale role analysis.

We believe that this variety of aspects will offer at least one
interesting facet for each conference attendee.
The objective of the first part, the formulation of PACT
plans, is to illustrate the intuitive and convenient approach
for expressing complex data-analysis workflows using PACTs.
We will explain the different parts of the role analysis code
and allow interested attendees to change parameters and
even code. As such, they can learn how different data sets
can be read and processed, how different features can be
computed, and how other mining tasks can be expressed.
This could even involve data provided by interested atten-
dees – given that they are willed to share them and that
an adaption of the import procedure is achievable in reason-
able time (i.e., the data is structured similar to the boards.ie
data). In limits naturally imposed by the demonstration
setup, this can involve the execution of modified code to
Finally, the third part of the demonstration will highlight the benefits of processing an analytical task as complex as the implemented role analysis on a massively parallelized system. We will provide a visualization tool that allows to interactively explore the pre-computed results over the whole boards.ie data set over all 10 years. Attendees will be able to explore the dynamics of role compositions over the complete lifetime of the boards.ie site, with different parameter settings, over different time spans, different forums or the global boards.ie data, etc. Figure 4 shows a screenshot of this explorative interface, which supports the interactive visualization of different (normalized) features as well as role compositions over arbitrary time intervals and forums. An analysis like this was not feasible before due to the immense computational requirements it imposes. Thus, it is tailored to showcase the benefits that the proposed approach based on Stratosphere brings for the field of social-media and online-community analytics.

Acknowledgements

We thank Christoph Nagel and Stephan Pieper (now with http://www.surpreso.com) for their implementation support while at TU Berlin and the Stratosphere team. The research leading to this result was funded by the European Commission under FP7 Project No. 257859 - ‘ROBUST’, the German Research Foundation under grant FOR 1036, the German Federal Ministry of Education and Research (BMBF) under grant number 01IS12033 - ‘RADAR’ and the European Institute of Innovation and Technology (EIT).

5. REFERENCES